International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

SBRidAPI – DETECT DECIMATE AND PREVENT SOCIALBOT IN OSN USING DEEP LEARNING TECHNIQUES

Ayesha Shereen A K[1], Christopher P[2]

[1]PG –Computer Science &Engineering, M.I.E.T Engineering College, Trichy, Tamilnadu
[2]Assistant Professor, Computer Science & Engineering, M.I.E.T Engineering College, Trichy, Tamilnadu

ABSTRACT

WEB 2.0-BASED Online Social Networks (OSNs) are one of the high impacting human innovations of the 21st century that facilitate their users to express views and thoughts on current affairs and personal life, connect with friends and celebrities, and get updated with the breaking news. OSNs facilitate their users in terms of connectivity, information sharing, knowledge acquisition, and entertainment, but these are not without any repercussions. The real-time message broadcasting and anonymity have exposed. The malicious and anti-social elements generally perform such activities using fake profiles in the form of bots, human-assisted cyborgs, Sybil, and compromised accounts. Recently, OSN platforms have witnessed emerging threats, having serious repercussions that are much more sophisticated in comparison to the classical cyber threats like spamming, DDoS attack, and identity theft. Among OSN-specific threats, automated profiles (aka social bots) are one of the major enablers of advanced illicit activities like political astroturfing. Social bots are very deceptive; they mimic human behavior to gain trust in an OSN and then exploit it for illicit activities. As a result, researchers are analyzing different malicious aspects of social bots. This project presents SBRidAPI, to profile users for detecting social bots on OSNs. To the best of our knowledge, this is the first deep learning-based approach that jointly models a comprehensive set of profile, temporal, activity, and content information for user behavior representation, which is fed to a two-layer stacked BiLSTM, whereas content information is fed to a deep CNN.

Keywords—OSN, SocialBots, BiLSTM, CNN

1. Introduction

Social media platforms like Facebook, Instagram, or Twitter have become an established part of the modern media diet (Newman et al., 2017) and communication nowadays is more often than ever mediated through online technologies.

Bot: Overview

A bot (shortened form of 'robot') is an automated program that is programmed for certain actions and executes them either regularly or reactively. The bot does this without needing human activation. It analyzes the environment and 'decides' which actions to take depending on the situation. Depending on the function, a bot isn't usually recognized by humans and performs its tasks unnoticed in the background, or appears as a human being (i.e., imitates human behavior). Some types of bots are

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

described below in more detail:

- **Web crawler:** They belong to the category of bots that work completely unnoticed. Usually, they are used by search engines to browse the web, analyze websites, and then enter this information into search directories. They usually act as 'innocent' within generally accepted standards (such as the Robots Exclusion Standard). Other web crawlers operate beyond these standards and collect unauthorized data on the net.
- **Chat bots:** In contrast to web crawlers, chat bots are reactive i.e., they react to human activities and specialize in responding to other chat participants in a meaningful way. In everyday life, chat bots are primarily encountered as digital assistants. For example, a website assistant can guide visitors through the website or answer questions about the website's topic or what it offers. Language assistants such as Siri or OK Google or external language assistants such as Amazon Echo or Google Home are also based on this chat bot technology.
- **Virtual computer players:** Virtual computer games also require virtual actors who react flexibly to human beings. These operations are performed by bots. They are called non-player characters (role-playing games), aim bots (action games), poker bots (online poker) and so on. These bots are reactive and work increasingly with artificial intelligent technology. A good example is Google's artificial intelligence, which is called Alpha Zero. It excelled in chess as well as the board game, Go. Intelligent technologies like these are also used in computer games – in the form of bots.
- **Social bots:** These are the bots that are used secretly in social networks. They are both repetitive and reactive: They like, comment, retweet, and try to provoke or involve others in conversations and discussions. They fake a human identity so that users react naturally to them.

2. Experimental Methods or Methodology

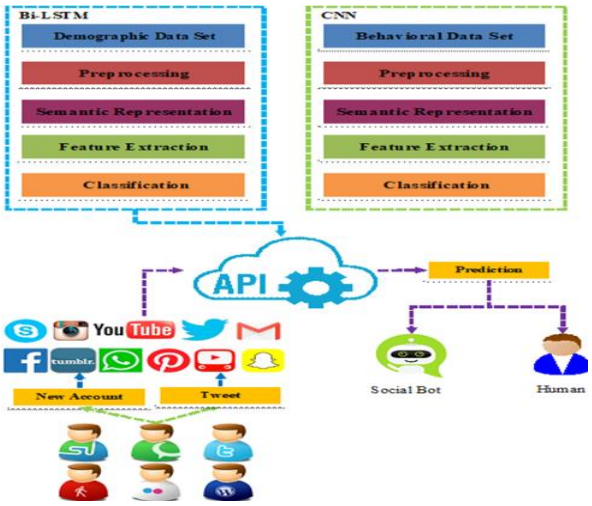

Fig 1. Social Bot based on BiLSTM and CNN

This article presents SBRidAPI, to profile users for detecting social bots on OSNs. To the best of our knowledge, this is the first deep learning-based approach that jointly models a comprehensive set of profile, temporal, activity, and content information for user behavior representation. It models profile, temporal, and activity information as sequences, which are fed to a two-layers stacked

DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

BiLSTM, whereas content information is fed to a deep CNN.

BiLSTM

These are like an upgrade over LSTMs. In bidirectional LSTMs, each training sequence is presented forward and backward to separate recurrent nets. Both sequences are connected to the same output layer. Bidirectional LSTMs have complete information about every point in each sequence, everything before and after it. The human brain uses its senses to pick up information from words, sounds, or from whole sentences that might, at first, make no sense but mean something in a future context. Conventional recurrent neural networks are only capable of using the previous context to get information. Whereas, in bidirectional LSTMs, the information is obtained by processing the data in both directions within two hidden layers, pushed toward the same output layer. This helps bidirectional LSTMs access long-range context in both directions.

CNN

CNN is a kind of feed-forward neural network used in deep learning, which was originally used in computer vision and included a convolutional layer to create the local features and a pooling layer for summarizing the representative features. Convolution layers in the artificial neural network play the role of a feature extractor that extracts the local features. This means that CNN establishes the specific local communication signals using a local connection pattern between neurons in the adjacent layer. Such a feature is useful for classifying in NLP, as it is expected that strong local clues should be found for the class, but these clues may appear in different places at the input. The convolutional and pooling layers allow CNNs to find local indicators, regardless of their location.

3. Results and Discussion

3.1 Preprocessing

Prior to training the Bi LSTM and CNN on the dataset, pre-process the data by forming a string of tokens from each tweet.Pre-processing data includes transforming raw data into a more explanatory form in the field of machine learning. The pre-processing helps to eradicate data noise, since the mix is made reliable by mixed with the real data. The process involves text insertion, selection of functions and normalization of the cleaning process. In addition, pre-processing can achieve better results in ml models. Different data set noise can be removed via Text Cleaning, such as hyperlinks, whitespace, punctuation and numbers. The standard processes here include conversion of lowercases, eradication of white and dotted spaces and numbers. In addition, there are also word lemmatization and word streaming. Standardization is essentially a process in which text documents are prepared for NLP events. Two major normalization methods, such as lemmatization and stemming, can be used to identify the word root forms.

Word lemmatization: This minimizes inflection, also by means of morphology and vocabulary analysis of these words the basic(rooted) form of the word is calculated. This is done by using a specific language dictionary and the words are converted in their primary format. Substantial thinking and pre-planning are an obstacle to the appropriate application of these algorithms. However, the process can be carried out with the help of NLTK library easily.

Word stemming is the process of reducing a word to their base form also known as written form. The stem is the morphological core of the word called as lemma. in order to map a word, it is

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

necessary to cut off its suffix or prefix. This heuristic technique is expected not to be always perfect as some tomes over-fitting or under-fitting also happen.

- Replace occurrences of hashtags, URLs, numbers, and user mentions with the tags "<hashtag>", "<url>", "<number>", or '<user>".
- Similarly, most common emojis are replaced with "<smile>", "<heart>", "<lolface>", "<neutralface>" or "<angryface>",
- depending on the specific emoji.
- For words written in upper case letters or for words containing more than 2 repeated letters, a tag denoting that is placed after the occurrence of the word. For example, the word "HAPPY" would be replaced by two tokens, "happy" and "<allcaps>".
- All tokens are converted to lower case.

	created_at	default_profile	default_profile_image	description	favourites_c
0	2016-10-15 21:32:11	0	0	Blame @xaiax, Inspired by @MakingInvisible, us...	4
1	2016-11-09 05:01:30	0	0	Photographing the American West since 1980. I ...	536
2	2017-06-17 05:34:27	0	0	Scruffy looking nerf herder and @twitch broadc...	3307
3	2016-07-21 13:32:25	1	0	Wife.Godmother.Friend.Feline Fanatic! Assistan...	8433
4	2012-01-15 16:32:35	0	0	Loan coach at @mancity & Aspiring DJ	88

Fig. 2. Data Set Preparation

3.2Feature Extraction

For building a machine learning classifier we require a matrix of feature values for training and later - classifying unseen test data. So, in a first step we extract a variety of features from the datasets. These features are discussed in the next subsections.It is critical to get text data ready for machine learning. To eliminate words from text data, special consideration is required. The word Tokenization is referred to for such a technique. The text must be transposed into numbers because we cannot handle the text directly in machine learning. Therefore science-study tokenization and additional functional extraction clears. A remarkable Bag of Words (BoW-ECM) technology was based in this regard on the number of word incident. Algorithms typically accept numeric values (int or float), so extraction layers convert words to "int." This is accomplished using popular methods such as words embedding, Tfdif vectorizer, and count vectorizer.

Account Based Features

Account Based Features The first group of features is derived from account metadata. Our simple user profile features directly reflect values the Twitter API provides about users. We additionally derive features with some processing from the screen and usernames. Some of the features are self-explanatory or explained by the Twitter API documentation. Nevertheless, some of these features

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

require additional discussion.

Simple user profile features: We hypothesize that metadata from the user profile provides valuable information about the user account. Some of this data is generated by Twitter itself and sometimes difficult to control directly by users. This data contains characteristics about a user's account we can exploit for machine learning.

These account features include:

- **default profile**: Has the user altered the profile?
- **geo enabled**: This feature reflects if users enable adding geographic information if they publish a tweet.
- **protected**: When true, indicates that this user has chosen to protect their Tweets.
- **is verified**: This is quality marker provided by Twitter: Accounts that are run by people of public interest can be verified as being authentic by Twitter itself.
- **friends count**: The number of users this account is following.
- **followers count:** The number of followers this account has.
- **favorites count**: The number of tweets this user has liked.
- **listed count**: The number of public lists this account is a member of.
- **statuses count**: The number of tweets issued by this account.
- **profile use background image**: Has the user provided a background image?

User profile name features: User and screen names are very much subject to a user's choice. Therefore, we hypothesize that it provides valuable information that helps to distinguish bots from humans. These features include:

- **screen name length**: The length of the screen name provided by a user.
- **username length**: The length of the account name provided by a user.
- **screen name digits**: Number of digits in the screen name.
- **username Unicode group**: See below.
- **screen name Unicode group**: See below.
- **Lowensteinusername screen name**: See below.

Then use some features that are closely related to the screen and usernames of accounts. Then determine which of the 105 Unicode groups an account uses in the screen and usernames. This is reflected in the categorical features username Unicode group and screen name Unicode group where it has one feature for every Unicode code group. The rationale behind this feature is that humans tend to be quite creative in their choice of names and sometimes tend to pick characters completely unrelated to the alphabet of their own language. By comparing occurrences of characters in various Unicode code groups it takes this behaviour into account. Furthermore, to make use of possible differences between an account's screen name and username. Designed this by calculating the respective Lowenstein distance. Then observed that bot accounts tend to choose usernames and screen names that are similar, while humans can be more creative in this respect.

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

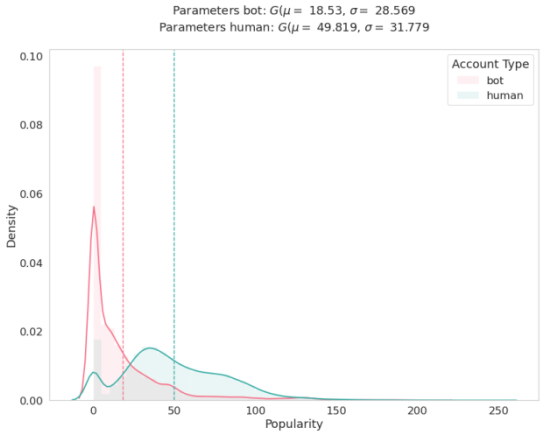

Fig. 3. Account Based Feature Extraction

Content Based Features

Then derive additional features from the content a user provides. For that purpose, this step tokenizes the tweets. Tokenize by breaking the tweet text at a variety of spaces and punctuation such as commas, colons, exclamation marks, brackets and similar characters that are commonly used in sentences. This tokenization respects characters occurring in emojis as well. While tokenization is not entirely language agnostic, these heuristics should nevertheless work for a large set of (English script) languages. Then it can extract features based on these tokens. Other features are directly derived from the metadata of tweets. The following features are mainly used by tweet-based bot detection techniques to distinguish between tweet-based bots and humans accounts:

- **ID**: It represents the unique identifier of the tweet.
- **User**: It represents the user who posted the tweet.
- **Created at**: It indicates the UTC time when the tweet is created.
- **Text Tweet**: It refers to the body of the tweet.
- **Length of Tweet**: It gives the number of characters in the tweet.
- **#Hashtags**: It indicates the number of hashtags in the tweet.
- **#URLs**: It indicates the number of URLs in the tweet
- **in_reply_to_status_id**: If the tweet is a reply, this feature represents the original tweet's ID.
- **in_reply_to_user_id**: If the tweet is a reply, this feature represents the author of the original tweet. Coordinates: It represents the geographic location of the tweet.
- **Favorite Count**: It indicates how many times the tweet has been liked by Twitter users.
- **Retweet Count**: It is the number of times the tweet has been retweeted
- **Reply Count**: It is the number of times the tweet has been replied to.
- **Favorited:** A Boolean feature, which holds true when the tweet is liked by the authenticating user.
- **Retweeted**: A Boolean feature, which holds true when the tweet is retweeted by the authenticating user. Possibly sensitive: a Boolean feature, which holds true when the tweet contains a link.

Behavioral features: In this module it hypothesizes that the tweeting behavior of bots and humans should exhibit differences. Then model this behavior by calculating statistical properties from the

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

data such as minimum, maximum, average, mean, median, standard deviation, skewness, kurtosis, and others.

- **time between retweets (distributional feature):** This set of features models time between retweeting activities of an account.
- **time between tweets (distributional):** This set of features models time between tweeting activities.
- **tweet rate (average tweet rate):** The average number of tweets per day

Core content features: This step hypothesize that some aspects of intention and emotions can be derived from a tweet's content. The following features honor this in a language independent way.

- **emojis classic** (distributional feature): See below.
- **emojis kaomoji faces** (distributional): See below.
- **emojis line art** (distributional): See below.
- **emojis other** (distributional): See below.
- **number of tokens** (distributional): This feature models the size of a tweet to some extent.
- **number of hashtags** (distributional): The number of hashtag-based references contained in tweets.
- **n of tokens wo hashtags URLs symbols** (distributional): A distribution based on the number of plaintext tokens in the tweets excluding hashtags, URLs and non-alphanumeric tokens.
- **number of URLs** (distributional): The number of URL based references contained in tweets.
- **special char repeats rate**: This feature detects sequences of question marks and exclamation marks. These are also often used for affirmation to give special weight to what has been expressed in a tweet. Then to model this aspect.

Assume that a precise sentiment analysis of the twitter text is not available for all languages tweets could be written in. Nevertheless, want to extract and use emotional aspects from the content. To model some basic aspects of emotion to detect various sets of emoticons here. The extraction is pattern based and derived from public emoticon collections as provided by Wikipedia. The four emoji features above are distributions derived on individual occurrences of emojis in single tweets.

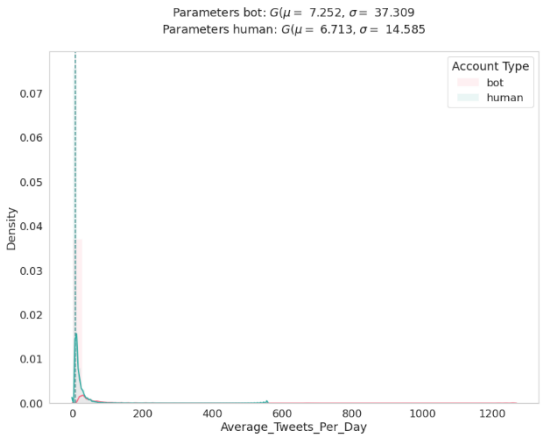

Fig. 4. Content Based Feature Extraction

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

3.3 Classification

This module uses two classification tasks: account-level bot detection and tweet-level bot detection. when OSN data is pre-processed and Feature Extracted, BiLSTM-CNN model is used in the classification of users into either legitimate users or bots.

Account-Level Classification

BiLSTM uses two LSTMs to learn each token of the sequence based on both the past and the future context of the token. one LSTM processes the sequence from left to right; the other one from right to left.The model mainly includes three layers. In this phase, it uses social user tweet metadata as temporal information. Using Bi LSTM as a training model for this temporal information, the temporal information for a period is input to the BiLSTM, and the output of the BILSTM is extracted as the input of the fully connected layer. Like the joint content feature detection part, the full connection layer is followed by the SoftMax layer to directly output whether the user is a genuine user or bot. The first layer is the joint content feature extraction layer, which focuses on the feature extraction of the tweets content and the relationship between them. The second layer is the tweet metadata temporal feature extraction layer, which regards the tweet metadata as temporal information and uses this temporal information as the input of the BiLSTM to extract the user social activity temporal feature. The third layer is the feature fusing layer, which fuses the extracted joint content features with the temporal features to detect social bots.

Tweet Level Classification

This module uses a CNN. The CNN consists of two one-dimensional convolutional neural networks layers, followed by Max Pooling layers, with a dense neural networks layer processing the output of the second CNN layer. The model is completed by a final output layer that uses the sigmoid activation function to return a binary output (namely, the classification into human or bot).

Input layer. Considering the article's words were embedded into d-dimensional vectors, the final matrix used as input to the model can be written as I = l x d where l is the chosen sequence length (i.e., the length of the post). Recall that l = 11000 in our setting.

First convolutional layer. The first transformation this embedded input goes through is a convolutional layer with f 1-dimensional filters of length k. Thus, the layer weights can be considered a matrix of shape W_c E 2 R^{fxk}. Then used a filter size f of 64 and a filter length k of 4. In our context, having 1-dimensional filters means that for each word in an article, its three adjacent words are considered as the context of the word. The output of the convolutional layer then is C = conv (I, Wc) where conv is the convolutional operation applied to input, I using the weights matrix W_c. This operation includes applying the ReLu activation function to complete weights calculations. Also, a dropout function is used to prevent overfitting. Then used a dropout rate of 0:5, which means 50% of the weights (randomly chosen) during each training epoch are set to 0.

First max pooling layer. The above output C is then considered the input to a 1- dimensional Max Pooling layer. The purpose of the layer is to extract only the most important features of the convolution outputs. This is done by keeping only the max value from a pool size p. As we chose p = 4, the output of this layer can be written as M = max pool (C; p). This operation reduces the number of weights by four times.

Second convolutional layer and max pooling layer. The output M of the max pooling layer is the

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

input of the second convolutional layer, and the whole process described above is applied to this input, to get a final output M2.

Fully connected layer. Given the two-dimensional matrix of weights from the last step, the next layer in the network is a fully connected one with a size of 256 hidden neurons. But for the convolutional output to serve as an input to this layer, its dimensionality needs to be reduced. This was done using the flatten method, which keeps all the values but flatten them in a long vector. A ReLU activation function and a dropout layer with a rate of 0:5 was used here.

Output layer. The last layer is a fully connected layer with one neuron. The sigmoid activation function is used to provide a binary output.

Machine learning and deep learning algorithms are powerful models that first consider the learning styles that an algorithm can adopt. These approaches can be divided into 3 broad categories (i) supervised learning, (ii) unsupervised learning, and (iii) reinforcement learning. Supervised learning is useful in cases where a property (label) is available for a certain dataset (training set) but is missing and needs to be predicted for other instances. Unsupervised learning is useful in cases where the challenge is to discover implicit relationships in each unlabelled dataset (items are not pre-assigned). Reinforcement learning falls between these extremes; there is some form of feedback available for each predictive step or action, but no precise label or error message. In this project, it focused on using several supervised learning models as classification-based approaches to detect bots from twitter accounts. The scikit-learn and Kera's libraries have been used to implement the following classification methods.

Decision Model

In this module it combines all the features and embeddings together and create SBRid Model for final prediction. Obtain the classified normal user set and social bots set: the normal user set, and social bots set can be finally obtained by detecting with live OSN user data.

Performance Evaluation

Traditional evaluation metrics for the two classification problems are adopted, namely, accuracy rate, accuracy, and recall, as follows

$$\boldsymbol{Accuracy = TP + TN / TP + TN + FP + FN}$$

$$\boldsymbol{Precision = TP / TP + FP}$$

$$\boldsymbol{Recall = TP / TP + FN}$$

The F1-score is the harmonic mean of precision and recall. In imbalanced data sets, the F1-score is much more effective than accuracy in determining the performance of the model. The F1-score is defined as follows.

$$\boldsymbol{F1 - score = 2 * Precision * Recall / Precision + Recall}$$

The true positive rate (TPR) is the ratio of the number of positive predictions that are classified as positive to the number of all positive samples, that is, the recall rate. The TPR is defined as follows.

$$\boldsymbol{TPR = TP / TP + FN}$$

The false positive rate (FPR) is the ratio of the number of negative predictions that are classified as negative to the number of all negative samples. The FPR is defined as follows.

$$\boldsymbol{FPR = TN / TN + FP}$$

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 4 Volume No.6 July – 2022
DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

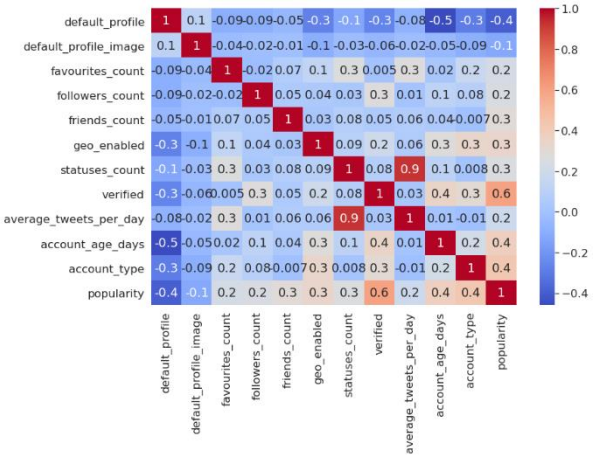

Fig. 5. Correlation of Features

CONCLUSION

Online social media platforms that allows connecting people and helps organizations reaching out to customers. Tweet-based botnet can compromise OSN accounts and create bot accounts to launch large-scale attacks and manipulation campaigns. This paper presents a deep neural network-based model, SBRidAPI, for detecting socialbots on OSN. This article proposed an BiLSTM and CNN model to increase the detection accuracy of malicious social bots. SBRidAPI jointly models' users' behavior using CNN and BiLSTM architectures. It models the profile, temporal, and activity information as sequences, which are fed to a two-layers stacked BiLSTM, whereas content information is given to a deep CNN. Our model requires no prior knowledge or assumption about users' profiles, friendship networks, or historical behavior on the target account. To the best of our knowledge, our work is the first that develops a CNN model with word embeddings to detect bots that relies only on tweets and does not require heavy feature engineering. This advantage allows for faster and easier implementation and deployment of the bot detection scheme. In addition, our proposed bidirectional convolutional neural architecture can be relatively easily adapted to a new problem, for example, using BiLSTM with word embeddings to detect phishing email, webpages, or SMS.

References

[1] M. Mohsin. (2020). 10 Social Media Statistics You Need to Know in 2021. [Online]. Available: https://www.oberlo.com/blog/social-mediamarketing-statistics

[2] I. Arghire. (2020). Twitter Hack:24 Hours From Phishing Employ- ees to Hijacking Accounts. https://www.securityweek.com/twitter-hack-24-hours-phishing-employees-hijacking accounts

[3] The Rise of Social Media Botnets. Accessed: Feb. 21, 2021. [Online]. Available: https://www.darkreading.com/attacks-breaches/the-rise-ofsocial- media-botnets/a/d-id/1321177

[4] M. Imran, M. H. Durad, F. A. Khan, and A. Derhab, ``Toward an optimal solution against denial of service attacks in software de_ned networks," *Future Gener. Comput. Syst.*, vol. 92, pp. 444_453, Mar. 2019.

[5] M. S. Savell. (2018). Protect Your Company's Reputation From Threats by Social Bots. [Online].

DOI:10.46647/ijetms.2022.v06i04.0049 ISSN: 2581-4621

Available: https://zignallabs.com/blog/protect-yourcompanys-reputation-from-threats-by-social-bots/
[6] S. Aslam. (2021). Twitter by the Numbers: Stats, Demographics & Fun Facts. [Online]. Available: https://www.omnicoreagency.com/twitterstatistics/
[7] A. Aldweesh, A. Derhab, and A. Z. Emam, ``Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues," Knowl.-Based Syst., vol. 189, Feb. 2020, Art. no. 105124.
[8] S. Mahdavifar and A. A. Ghorbani, ``Application of deep learning to cybersecurity: A survey," Neurocomputing, vol. 347, pp. 149_176, Jun. 2019.
[9] E. B. Karbab, M. Debbabi, A. Derhab, and D. Mouheb, ``MalDozer: Automatic framework for Android malware detection using deep learning," Digit. Invest., vol. 24, pp. S48_S59, Mar. 2018.
[10] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, ``A novel twostage deep learning model for ef_cient network intrusion detection," IEEE Access, vol. 7, pp. 30373_30385, 2019.
[11] A. Derhab, A. Aldweesh, A. Z. Emam, and F. A. Khan, ``Intrusion detection system for Internet of Things based on temporal convolution neural network and ef_cient feature engineering," Wireless Commun. Mobile Comput., vol. 2020, pp. 1_16, Dec. 2020.
[12] Mahesh, B. "An Active Approach for Load Balancing in Grid Computing."
[13] A. T. Kabakus and R. Kara, ``A survey of spam detection methods on Twitter," Int. J. Adv. Comput. Sci. Appl., vol. 8, no. 3, pp. 29_38, 2017.
[14] M. Chakraborty, S. Pal, R. Pramanik, and C. R. Chowdary, ``Recent developments in social spam detection and combating techniques: A survey," Inf. Process. Manage., vol. 52, no. 6, pp. 1053_1073, Nov. 2016.
[15] E. Alothali, N. Zaki, E. A. Mohamed, and H. Alashwal, ``Detecting social bots on Twitter: A literature review," in Proc. Int. Conf. Innov. Inf. Technol. (IIT), Nov. 2018, pp. 175_180.
[16] Naik, S. Bhaskara, B. Mahesh, and Kurnool Dist Koilakuntla. "Evaluating Malware Detection System using Machine Learning Algorithms." (2021).
[17] K. Yang, O. Varol, C. A. Davis, E. Ferrara, A. Flammini, and F. Menczer, ``Arming the public with arti_cial intelligence to counter social bots," Hum. Behav. Emerg. Technol., vol. 1, no. 1, pp. 48_61, Jan. 2019.
[18] Z. Guo, J.-H. Cho, I.-R. Chen, S. Sengupta, M. Hong, and T. Mitra, ``Online social deception and its countermeasures: A survey," IEEE Access, vol. 9, pp. 1770_1806, 2021.
[19] S. B. Abkenar, M. H. Kashani, M. Akbari, and E. Mahdipour, ``Twitter spam detection: A systematic review," 2020, arXiv:2011.14754. [Online]. Available: http://arxiv.org/abs/2011.14754
[20] W. Daffa, O. Bamasag, and A. AlMansour, ``A survey on spam URLs detection in Twitter," in Proc. 1st Int. Conf. Comput. Appl. Inf. Secur. (ICCAIS), Apr. 2018, pp. 1_6.