Sensors & Transducers, Vol. 244, Issue 5, September 2020, pp. 37-43

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2020
http://www.sensorsportal.com

Bot Detection in Social Networks Based on Multilayered Deep Learning Approach

[1] Sandeep Singh Sengar, [2] Sanjay Kumar, [2] Pradyot Raina and [2] Mukul Mahaliyan

[1] Department of Computer Science and Engineering, SRM University-AP, Andhra Pradesh, India-522502

[2] Department of Computer Science and Engineering, Delhi Technological University, Shahbad Daulatpur, Main Bawana Road, New Delhi, India-110042

E-mail: sandeep.iitdhanbad@gmail.com, sanjay.kumar@dtu.ac.in, pradyotrainaout@gmail.com, mukulmahaliyan@gmail.com

Received: 18 July 2020 /Accepted: 28 August 2020 /Published: 30 September 2020

Abstract: With the swift rise of social networking sites, they have now come to hold tremendous influence in the daily lives of millions around the globe. The value of one's social media profile and its reach has soared highly. This has invited the use of fake accounts, spammers and bots to spread content favourable to those who control them. Thus, in this project we propose using a machine learning approach to identify bots and distinguish them from genuine users. This is achieved by compiling activity and profile information of users on Twitter and subsequently using natural language processing and supervised machine learning to achieve the objective classification. Finally, we compare and analyse the efficiency and accuracy of different learning models in order to ascertain the best performing bot detection system.

Keywords: Bot detection, Machine learning, Natural Language Processing, Social network, Text classification.

1. Introduction

Today, social networking sites have acquired an integral position in the daily lives of almost every person with access to the internet around the globe. The amount of time the average user spends on social media apps has also been increasing at an almost alarming rate for some time. Nowadays, users don't just use social networks like Facebook, Twitter, Instagram, etc. for interacting within their social circles but also for getting news updates, multimedia entertainment, political discussions, business and even shopping. All this points to the fact that social media has a massive hold on a significant part of the world's population and can thus be used, or misused, for influencing them. With people increasingly resorting to the social media profiles to judge and make views of the people around them, it has become increasingly important to have an impressive social media following. This is particularly a cause of concern for celebrities, politicians and others with similar public careers. Moreover, social media has the tendency to broadcast content to millions of people in a matter of minutes and thus all prominent organizations, from political parties to corporations, have started using it to push their own narratives and sway public opinion one way or another. And so, the use of fake accounts and bots in an organized manner on a massive scale has recently become a short cut to becoming 'viral' and inflating one's social media presence. 'Bots' are simply computer programs that automatically produce or repost content and interact with humans on social

networks. When used on a large enough scale, such bots have had significant impact on the real world – from spreading fake news during elections, influencing stock prices, swaying opinion about a company or a product, hacking and cybercrimes, data stealing and also promoting and distorting the popularity of individuals and groups. Thus, with our project we aim to identify and distinguish bots from genuine humans in order to curb the menace caused by them. To achieve this, we have proposed a layered machine learning approach. Starting with a labelled dataset, we perform feature extraction based on established tests and correlation analysis. Then, text attributes are processed using feature engineering followed by initial classification that produces a vector of predictions for each text attribute. We further provide a comparative analysis of the performance of various well-known classification algorithms based on their prediction accuracy on the testing dataset. This evaluation also includes a comparison of two popular feature engineering techniques for text attributes i.e. Bag-of-Words and n-gram model.

1.1. Dataset

Our project has been implemented on a Twitter dataset which contains information about more than 5000 different Twitter users and nearly 200,000 tweets. The user information is across attributes including their *handle, name, location, followers count, profile description or bio, favourites count, tweets count, sample tweets,* etc., which we compiled together to build our model. Out of these 20 attributes, 16 attributes are either numerical or categorical while the remaining 4 i.e. *description, status, name* and *screen_name,* have text values. Also, the dataset includes labels for each of its instances, classifying a user as a bot (1) or a genuine user (0) which enabled us to take a supervised learning approach and train classifier algorithms over it.

The approach proposed in this paper generates predictions based entirely on a user's public data and information which can be easily extracted from Twitter itself in real-time.

Table 1. Comparative analysis of the related work.

Author, Year	Dataset	Approach	Description	Advantages	Disadvantages
Wei F., *et al*., 2020	Cresci, 2017	Model	Classification using BiLSTM with word embeddings	Comprehensive tweet level analysis using RNN	Does not take into account user level data
Luo L., *et al*., 2020	PAN Bots and Gender Profiling	Model	Deepbot consisting of a classifier and a Web interface	Uses GloVE embedding + BiLSTM for better tweet analysis	Does not take into account user level data
Kudugunta S., *et al*., 2018	Cresci, 2017	Layered Model	Two-layered model for classification	Combines tweet level and tweet text analysis for more reliable result	Does not account for both textual analysis and account level analysis
Zhao C., *et al*., 2020	Chen, *et al*., 6 million spam tweets	Algorithm	Uses a layered ensemble framework	Better performance than normal LSTM models on tweet level data	Only takes into account tweets, not user data

2. Literature Review

The literature of twitter bot detection is vast, but the problem is still a very challenging task with bots becoming smarter with time. It's relatively simple to detect bots which show unusual behaviour such as retweeting very frequently. But smarter bots curb to show unusual behaviour and act more humanly. It is harder for bots to show similar tweeting patterns as humans thus deep learning models are quite successful in detecting bot by textual analysis.

The existing state of research in this field consists of approaches such as "Twitter Bot Detection Using Bidirectional Long Short-term Memory Neural Networks and Word Embeddings. 2019" and "Deepbot: A Deep Neural Network based approach for Detecting Twitter Bots" employ latest deep learning techniques like Bi-LSTM based approach for textual analyses for bot detection. A similar approach "Kudugunta S., Ferrara E.: Deep Neural Networks for Bot Detection" models the user classification task differently at user and tweet level data and comparisons based on performance of each level are made.

Our work combines the two levels and attempts to find unusual behaviour in engagement of the user and also in tweets of the user. Bi-LSTMs are slightly better than LSTMs in the sense that each state has memory of not just previously occurring series but also the memory of future occurring series. LSTMs perform good on time series data where sequence of the sentence is important. CNN has performed equally good or even better than LSTMs in text classification tasks "Yin W., Kann K., Yu M., Schütze H.: Comparative Study of CNN and RNN for Natural Language Processing. arXiv". when feature detection is extremely important, and certain sequence to sequence learning tasks "Gehring J., Auli M., Grangier D., Yarats D., Dauphin Y.N.: Convolutional sequence to sequence learning. 34th Int. Conf. Mach. Learn.".

In the context of this problem, identification of features such as abuses, angry terms, and hate speech

is very useful in understanding the context of the tweet. Thus, we employed CNN for textual analyses of tweets. Our proposed approach achieves performance as good as or better when compared to existing work "Twitter Bot Detection Using Bidirectional Long Short-term Memory Neural Networks and Word Embeddings. 2019", "Deepbot: A Deep Neural Network based approach for Detecting Twitter Bots", "Kudugunta, S., Ferrara, E.: Deep Neural Networks for Bot Detection" and "Zhao, C., Xin, Y., Li, X., Yang, Y., Chen, Y.: A Heterogeneous Ensemble Learning Framework for Spam Detection in Social Networks with Imbalanced Data. Appl. Sci. 10, 936 (2020)".

3. Methodology

Before performing the actual learning on our data, we analyse our dataset in a comprehensive manner in order to better identify strongly correlated attributes, highlight weak correlations and determine which features are most significant in the prediction of the target variable i.e. *bot* status. Due to the variance in attribute types we proceed in steps towards our aim of generating a smaller set of strong features to train our learning algorithm on.

First, we separate the 4 text valued attributes and perform feature engineering on each of them in a subsequent step. Of the remaining non-text attributes, by elementary analysis tools and intuition, we are able to eliminate some like *location, created_at,* etc. as these contain either majority NaN, have a single repeated value, are completely random or strongly correlated with another attribute in the dataset. Thus, on the remaining numerical/categorical attributes and 4 text attributes, we perform feature engineering. For the non-text attributes, we obtain the following correlation matrix, as shown in Fig. 1 that provides an approximate idea about the relation between the attributes and the target variable i.e. *bot.* However, feature selection based purely on correlation is not always reliable. Thus, we apply the chi^2 test to determine the 5 best or strongest features and the resulting dataset with reduced dimensionality includes the attributes *id, followers_count, friends_count, listed_count* and *statuses_count*.

Fig. 1. Correlation matrix for user-level non text attributes.

Having previously separated the 4 text attributes, we proceed with feature engineering for these by observing the correlation between them and the target variable one by one. To achieve this, we employ two feature extraction methods i.e. Bag-of-Words model and the n-gram model and then compare results for the

two. Once we derive the 'features' or 'vectors', which are simply the frequency of words appearing in the text after the appropriate transformations, we train a classifier on each of these 4 vectorized attributes and use it to make predictions for the training samples which are then recorded in a column. The resulting 4 prediction columns, one for each text attribute, are then added to the selected features.

In the final step, we perform the final classification using the extracted feature set and the *bot* labels by training different classification models and then comparing their accuracy scores. This step is performed for two feature sets i.e. one obtained from Bag-of-Words extraction and the other from n-gram extraction technique.

As stated earlier, we follow a supervised learning approach to learn how a bot and human differ in the behaviour on social networks. First, we load the data into a *pandas* dataframe to make it usable in the Python environment. By analysis we find that there are some missing values throughout the dataset so we proceed by filling these values in order to prevent errors. This leads us to the feature selection and extraction step where we separate the text attributes beforehand. On the remaining dataset consisting only of numerical and categorical attributes, we select a few best performing attributes based on the chi^2 test. Now, we turn our focus to extracting useful features from the text attributes. The text attributes to be analysed are *screen_name*, *name*, *description* and *status*. The *status* attribute contains nearly 200k tweets grouped by the user id of the user Now, we employ two different feature engineering techniques i.e. Bag-of-Words and n-gram.

These methods work by analysing how frequently each word occurs and how this relates to the target attribute. Once this 'vectorization' is achieved, we perform term frequency-inverse document frequency transformation in order to normalize the 'vectors' and pre-empt bias towards higher frequency words. The result after this is a sparse matrix of transformed word frequencies which we then convert to pandas dataframes for all the four attributes and then concatenate them together to perform classification for all four together. Also, the feature sets resulting from the Bag-of-Words model and the one resulting from the n-gram model are maintained separately in order to allow comparisons at a later stage.

The flowchart of the methodology followed in the implementation of this research paper is elaborated in Fig. 2.

Once the two step feature extraction process is finished and the final training dataset is generated, we perform the actual learning using several different established machine learning algorithms in order to compare and analyze their performance and determine the best suited model for our problem. The machine learning algorithms thus used and analyzed for performing the classification in this paper are

- K Nearest Neighbours Classifier
- Decision Tree Classifier
- Random Forest Classifier
- Ada Boost Classifier
- Gradient Boosting Classifier
- Gaussian NB Classifier
- Multinomial Naïve Bayes Classifier, and
- Multilayer Perceptron

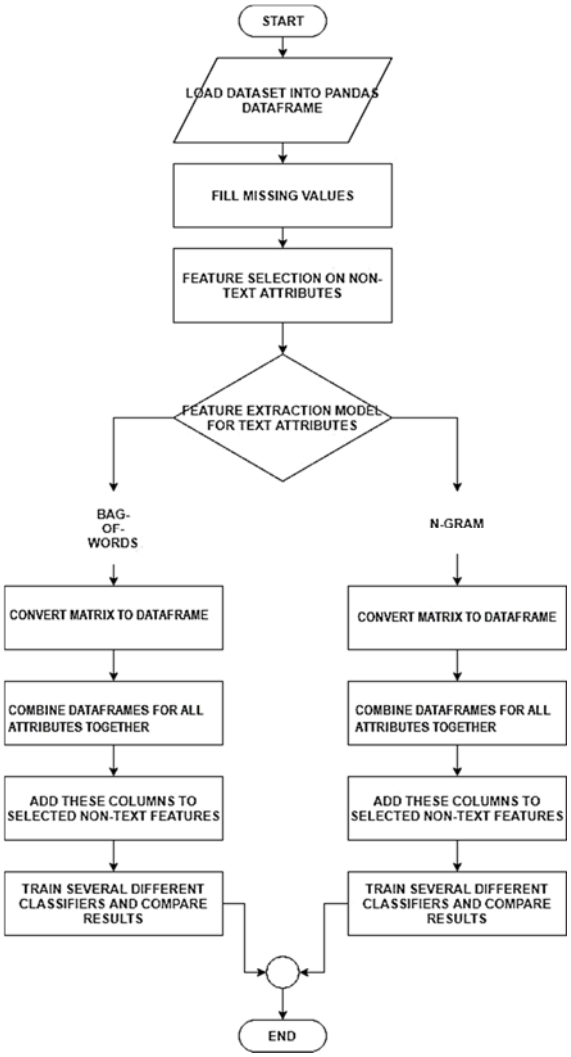

Fig. 2. Flowchart representing the methodology of the proposed method.

4. Findings

The accuracy scores for different classifiers on both the textual feature extraction models are detailed in Table 2 and Table 3 below.

As seen, the multilayer perceptron and multinomial Bayes classifiers show the best results when the bag-of-words model is used to extract features from the textual data.

As is evident from the table of accuracy scores above, the n-gram model of feature extraction offers significantly better accuracy on average over different classifiers with 89.7 % average accuracy than the Bag-of-Words model with average accuracy score of 87.3 %.

Table 4 below details an analysis of the performance of each classifier.

Table 2. Accuracy Scores for bag-of-words model.

S. No.	Algorithm	Accuracy Score
1.	K Neighbors Classifier	57.1019%
2.	Decision Tree Classifier	92.9699%
3.	Random Forest Classifier	86.8006%
4.	Ada Boost Classifier	91.1047%
5.	Gradient Boosting Classifier	91.6069%
6.	Gaussian Naïve Bayes	89.3113%
7.	Multinomial Naïve Bayes	94.7633%
8.	Multilayer Perceptron	94.7633%
	Average	**87.3027%**

Table 3. Accuracy scores for n-gram model (n=2).

S. No.	Algorithm	Accuracy Score
1.	K Neighbors Classifier	79.5552%
2.	Decision Tree Classifier	92.8264%
3.	Random Forest Classifier	92.6112%
4.	Ada Boost Classifier	90.6026%
5.	Gradient Boosting Classifier	90.0287%
6.	Gaussian Naive Bayes	87.7331%
7.	Multinomial Naïve Bayes	89.3831%
8.	Multilayer Perceptron	95.4089%
	Average	**89.7685%**

Table 4. Accuracy scores(%) of each classifier model.

S.No	Classifier	Bag-of-Words accuracy	n-gram accuracy	Average Accuracy
1.	K Neighbors	57.10%	79.55%	**68.32%**
2.	Decision Tree	92.96%	92.82%	**92.89%**
3.	Random Forest	86.80%	92.61%	**89.70%**
4.	Ada Boost	91.10%	90.60%	**90.85%**
5.	Gradient Boosting	91.60%	90.02%	**90.81%**
6.	Gaussian Naïve Bayes	89.31%	87.73%	**88.52%**
7.	Multinomial Naïve Bayes	94.76%	89.38%	**92.07%**
8.	Multilayer Perceptron	94.76%	95.40%	**95.08%**
	Overall Average			***88.53%***

5. New Findings

There are many deep learning algorithms available that could be applied to text classification. LSTMs are highly useful in case of time series data. LSTMs evaluate the relationship between the consecutive time step data. LSTMs as opposed to RNNs keep a memory of previously occurring series (which in current case is the sequence of words) which it uses while feed forwarding from cell to cell. However, LSTM has a few demerits in the context of the problem. LSTMs are not the best model where feature detection is paramount. For example searching for abuses, angry terms and named entities, ConvNet works better. LSTMs work better in tasks where sequence of the sentence is important, which is seen in problems of translation using computers or automated answering to queries. Studies in "Yin W., Kann K., Yu M., Schütze H.: Comparative Study of CNN and RNN for Natural Language Processing. arXiv" show clear comparison. Thus, we select ConvNet as the text classification model.

For using ML or deep learning algorithms on texts the necessary step is to apply appropriate language modelling technique. Language modelling allows representation of textual data in a vectorized form which has a high imperative in machine learning. We used Glove "Pennington J., Socher R., Manning C. D.: Glove: Global Vectors for Word Representation. 1532–1543 (2014)" word embedding for language modelling which is a better technique than previously used language modelling techniques namely Bag-of-words and N-gram. Glove is a context independent open source embedding provided by Stanford University that captures semantic similarity between large vocabularies of words. It is also useful for handling out-of-dictionary words which were absent in the dataset during deployment of the model. ConvNets are usually applied to computer vision problems but also show promising results on NLP problems.

The input to the convolutional neural network model used during tweet level analysis to generate the first level of predictions based on the textual data can be mathematically represented as:

Input dimensions =

=(No of words in text) × (Embedding length)

Fig. 3 represents a dummy input matrix having an embedding length of 5 and text "I am a Bot #kill #lol !".

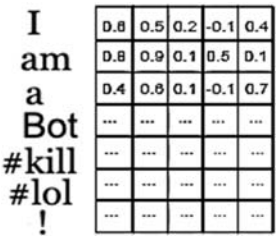

Fig. 3. A dummy input matrix to the CNN text classification model.

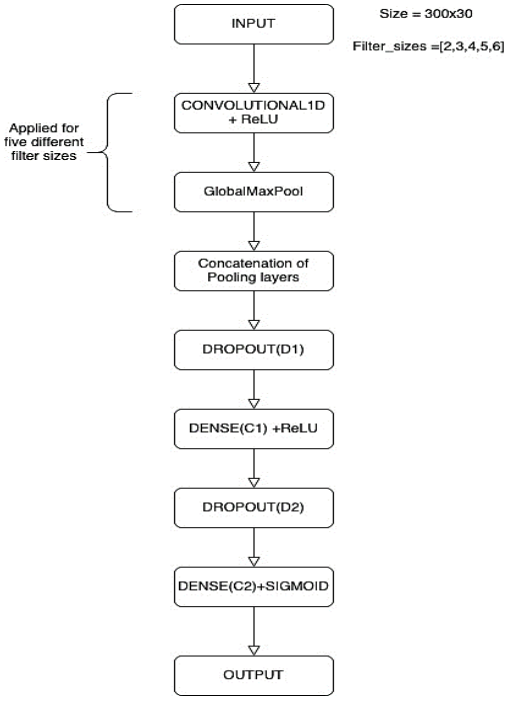

Fig. 4. Architecture of the proposed CNN used for tweet level analysis.

Row 1 corresponds to embedding weights of word number 1, in the input sentence which is 'I', as stored in Glove word embeddings. Similarly, Row 4 corresponds to embedding weights of word number 4 in the input sentence which is 'Bot' in this case and so on.

5.1. Algorithm

Let set U contain U1, U2 … Un Ui represents a user tuple with features uf(1), uf(2) … uf(n-1), uf(n). // total n features and set T contains T1, T2 … Tn, where Ti represent a tweet tuple with features tf(1), tf(2) … tf(m-1), tf(m) and tf(text). // total m+1 features.

Step 1:
Select an appropriate k.
Select a random Tk tweet from T and discard the all Ui in U where Ui.userid == Tj.userid (for all Tj int Tk).

Step 2:
Train CNN text classifier with all tf(text) in Tk. Let C(x) be the function that describes this classifier. Discard Tk from T. Generate a new features tf(CNN prediction) in T calculated as:
Ti.tf(CNN prediction) = C(Ti.tf(text))

Step 3:
Generate new features uf(n+1) … uf(n+m+1) in U which corresponds to tf(1) … tf(m) and tf(CNN prediction).

Step 4:
For all tuples Ti, Ti+1 ... Ti+j in T, where Ti.userid = Un.userid calculate the central tendency using A.M. for each feature tf(j). Assign this to uf(n+j).

Step 5:
Split U in train_set and test_set

Step 6:
Choose an appropriate machine learning classifier which is used to train on train_set. Let M(x) describe this classifier.

Step 7:
M makes predictions for each tuple in test_set and performance of the classifier is evaluated on the basis of disparity in prediction and original value.

5.2. Results

The model was trained and tested on very large data to avoid sampling bias and obtain more reliable results. The dataset was acquired from MIB Datasets (Cresci), The dataset consists of a total of 8,386 accounts and 62,96,505 tweets of the accounts. The new approach gave a staggering improvement of prediction result and gave an average 97.033 % prediction over an average of 88.5 % having used an extremely larger dataset than the latter.

Table 2. Results obtained by combining user data with tweet metadata and CNN predictions using the proposed new approach.

Classifier	Accuracy	Precision	Recall	F-score
K Neighbours	96.10	96.09	96.10	96.09
Support Vector	94.78	94.76	94.78	94.60
Decision Tree	99.48	99.48	99.48	99.48
Random Forest	**99.54**	99.54	99.54	99.54
Ada Boost	99.48	99.48	99.48	99.48
Gradient Boosting	**99.54**	99.54	99.54	99.54
Gaussian NB	90.31	90.76	90.31	89.24
Linear Discriminant	97.42	97.71	97.42	97.47
Multilayer Perceptron	96.67	96.87	96.67	96.73

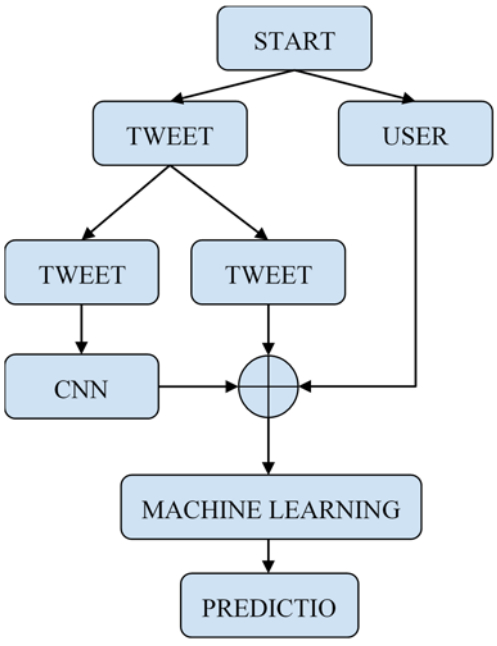

Fig. 5. An abstraction of the newly proposed model.

6. Conclusions

This research paper lays emphasis on the identification, detection and eradication of bots that populate social networks today. We focus on Twitter to determine which user is a bot and which is a human based on their activity and profile information. By taking a machine learning approach for such

distinction, we are not only able to detect bots with sufficient accuracy but this also enables us to define the behaviour and activity patterns exhibited by bots. Apart from the mechanism to detect bots, we also offer a comparative analysis of the performances of various established machine learning techniques and algorithms in our project. We evaluate the accuracy of several classification algorithms when applied after two different feature engineering techniques. The results show that neural networks based Multilayer Perceptron algorithm gives the most accurate predictions. We find that our proposed technique results in an average accuracy of 88.5 % across several established tools.

Later research observed that using convolutional neural networks for text level prediction of the stacked ensemble the results flung very high and showed an average of 97.03 %.

References

[1]. Oentaryo Richard J., Murdopo Arinto, Prasetyo Philips K., Lim Ee-Peng, On Profiling Bots in Social media, in *Proceedings of the International Conference on Social Informatics*, 2016.

[2]. Sandeep Singh Sengar, Sanjay Kumar, Pradyot Raina, Mukul Mahaliyan, Bot Detection in Social Networks Based on Machine Learning Techniques, User Information and Activities, in *Proceedings of the 2nd International Conference on Advances in Signal Processing and Artificial Intelligence*, Berlin, Germany, 2020, pp. 121-124.

[3]. Fazil Mohd, Abulaish Muhammad, A Hybrid Approach for Detecting Automated Spammers in Twitter, *IEEE Transactions on Information Forensics and Security*, Vol. 13, Issue 11, 2018, pp. 1-4.

[4]. Aswani Reema, Kar Arpan, Ilavarasan Vigneswara, Detection of Spammers in Twitter marketing: A Hybrid Approach Using Social Media Analytics and Bio Inspired Computing, *Information Systems Frontiers*, Vol. 20, Issue 2, 2017, pp. 1-16.

[5]. Sengar S. S., Mukhopadhyay S., Moving object detection based on frame difference and W4, *Signal, Image and Video Processing*, Vol. 11, Issue 7, 2017, pp. 1357-1364.

[6]. Efthimion Phillip George, Scott Payne, Nicholas Proferes, Supervised machine learning bot detection techniques to identify social twitter bots, *SMU Data Science Review*, Vol. 1, Issue 2, 2018.

[7]. Sengar S. S., Mukhopadhyay S., Moving object area detection using normalized self adaptive optical flow, *Optik*, Vol. 127, Issue 16, 2016, pp. 6258-6267.

[8]. Wang A.H., Machine Learning for the Detection of Spam in Twitter Networks, in: Obaidat M. S., Tsihrintzis G. A., Filipe J. (Eds.), e-Business and Telecommunications. ICETE 2010. Communications in Computer and Information Science, *Springer*, Berlin, Heidelberg. Vol. 222, 2012, pp. 319-333.

[9]. A. Mostrous, M. Bridge, K. Gibbons, Russia used Twitter bots and trolls 'to disrupt' Brexit Vote, 2017. [Online]. Available: https://www. thetimes.co.uk/ edition/news/russia-used-web-posts-to-disrupt-brexitvote-h9nv5zg6c

[10]. A. Bessi, E. Ferrara, Social Bots Distort the 2016 U.S. Presidential Election Online Discussion, Nov. 2016. [Online]. Available: http://firstmonday.org/ojs/ index.php/fm/article/view/7090

[11]. Sengar S. S., Mukhopadhyay S., Motion detection using block based bi-directional optical flow method, *Journal of Visual Communication and Image Representation*, Vol. 49, 2017, pp. 89-103.

[12]. S. Kramer, Identifying Viral Bots and Cyborgs in Social Media, 2017. [Online]. Available: https://www.oreilly.com/ideas/identifying-viral-bots-and-cyborgs-in-social-media

[13]. Sengar S. S., Mukhopadhyay S, Moving object detection using statistical background subtraction in wavelet compressed domain, *Multimedia Tools and Applications*, 2019.

[14]. S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, M. Tesconi, The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race, in *Proceedings of the 26th International Conference on World Wide Web Companion (WWW)*, Geneva, Switzerland, 2017, pp. 963-972.

[15]. Sengar S. S., Mukhopadhyay S., Motion segmentation-based surveillance video compression using adaptive particle swarm optimization, *Neural Computing and Applications*, 2019, pp. 1-15.

[16]. P. Suárez-Serrato, M. E. Roberts, C. Davis, F. Menczer, On the influence of social bots in online protests, in *Proceedings of the International Conference on Social Informatics (SocInfo'2016)*, 2016, pp. 269-278.

[17]. Sengar S. S., Motion segmentation based on structure-texture decomposition and improved three frame differencing, in *Proceedings of the 15th IFIP International Conference on Artificial Intelligence Applications and Innovations (AIAI'19)*, Hersonissos, Crete, Greece, 2019, pp. 609-622.