Iberian Journals

Iberian Journal of Applied Sciences and Innovations

2021 vol 1, Issue 2

Challenges in Integrity of E-voting Systems: Important Properties, Threats, and Solutions

Mohsen Borousan
Asia Pacific University of Technology & Innovation
Kuala Lumpur, Malaysia
Mohsen.boroosan@gmail.com

Mostafa Kateb
Department of Electical Engineering, Science and research Branch, Islamic Azad Univerity, Tehran ,Iran
mostafakateb@gmail.com

***Abstract*— Today, developed and developing countries are moving more and more towards e-government systems to deliver integrated, fast, and cheaper services to their citizens. Electronic voting is one of the crucial domains in this area, as the results of the elections profoundly affect the future of the nation and even other countries. Confidentiality, integrity, and availability are the three sides of the CIA triangle that are the principal measurements for evaluating the security of the employed e-voting systems. Since system and data integrity are crucial factors for preserving the security of the designed and developed systems, this study explores the properties, threats, solutions, and unresolved challenges in integrity of e-voting systems, to help researchers, designers, and developers evaluate their systems in term of integrity.**

***Keywords*— voting integrity; e-voting; e-government; voting security**

I. Introduction

In the traditional practices of paper balloting and hand counting, not only the whole process is observable for the public, but also it is simply understandable to the average voters. In the beginning, the empty ballot box is sealed by the polling staff, and after the election, the seal can be broken and the votes are counted in front of observers [1]. This simplicity and transparency make it easy for observers to identify likely errors. At the same time, candidate agents, political parties, and the media can perform a monitoring function [2].

This simplicity and transparency are lacking in the e-voting systems, as the complexity of the systems is only understandable for the field experts. E-voting systems utilize black-box technology that receives input from voters and then generates an output that is not simply verifiable by observers and even the election administrators [3,4]. This is the point where the integrity, transparency and trust problems arise. As a result, in the e-voting systems, complementary measurements are required to serve the same level of assurance as traditional practices [5]. These measurements may include the followings:

Transparency: is a way to satisfy the integrity problem in e-voting and vote counting technologies [4,6]. While this feature alone does not guarantee the accuracy of the results, it provides the ground to achieve this goal. Transparency in e-voting lets the electoral management bodies (EMB) and stakeholders to supervise the critical elements of the process, and avoid intentional and accidental errors [6].

Testing and certification: due to the lack of transparency in e-voting systems and counting process, compared to traditional paper balloting practices, it is critical that election administrators test and verify the voting machines to build trust and confidence before they are used [7]. Testing and verification are needed to guarantee that the machines meet the criteria defined by the EMB. The test results should be reviewed by observers and electoral contestants to ensure public confidence [8].

Additionally, some countries only accept certified e-voting and counting technologies. These certifications serve the same as testing procedures. However, the issuance of certifications should be independent of political parties, EMB, suppliers and government [9,10]. Ideally, the certification process must happen by a widely accepted source and through a transparent and open procedure.

Authentication: is the process of digitally signing the tested and verified software [11]. The signature can be verified by those which observe the election. Moreover, the validity of data in transition stages - like sending votes for the tabulation process – need to be verified as well; otherwise, the votes could be simply manipulated [11].

To prevent alteration of the votes, only the data with authentic digital signature are acceptable to be passed into the tabulation system. Transmission of the results requires safeguards that are monitored by candidate/party agents [11].

Audit: is verifying the operations and auditing the results of an e-voting or counting system. The most practiced way is using a voter-verified paper audit trail (VVPAT) that delivers the paper trail of the casted vote to the voter [12].

The audit trail is a critical factor for verifying the accuracy of the e-voting machines or counting process [12]. A randomly selected audit trail should be verifiable against the e-voting results to prove the consistency of the electronic and audit trails. Such a verification, if made for the public, has a great influence on the public trust [12].

II. Literature Review

E-voting integrity deals with system trustworthiness, including both provided function and data. In other words, it is to implement safeguards to protect e-voting data and software against changes in unauthorized ways. A solution to resolve the integrity issues of stored data is to utilize cryptographic protocols and techniques like public-key, homomorphic cryptography, Secure Socket Layer (SSL), and transport layer security (TLS) [13]. E-voting schemes utilize various techniques to enhance the preservation of their integrity. Some of the prominent schemes are as follows.

Since the date of introducing Votegrity [14] – the first end-to-end (E2E) verifiable e-voting protocol - various e-voting protocols have been introduced. In E2E, the voters can verify if their votes are cast and counted correctly in the final tally. Additionally, public members are able to verify the election externally. Some of the prominent E2E-based e-voting schemes include STAR-Vote [15], Helios [16], Scantegrity [17], Prêt à Voter [18], and Neff's Markpledge [19].

Some types of E2E-based protocols employ the public web bulletin board (WBB) to show the total casted ballots for the public. WBB is a broadcasting channel which displays the casted ballots in encrypted form, once the voters cast their votes and received the receipt of their encrypted votes [20,21,22]. Vote receipt is an important feature of the e-voting protocols, as a way to prove the vote in case of a dispute.

Apollo [23] is a developed version of Helios protocol which resolves some of the Helios' security drawbacks. Voting assistants is an added feature that helps in verifying, locking and auditing the votes. The assistants are external devices to the voting protocol that are designed for checking the bulletin board and displaying the value of the vote in plaintext format, after casting it [23].

Mixing is another technique that shuffles the votes' data in a random sequence before transmitting it to the next destination [24]. Zeus [25] is a sample protocol designed based on mixing technology. It runs the mixing procedure to remove the links between the encrypted ballots and the voters, in multiple rounds.

Homomorphic tally is a widely applied technique that involves modifications like addition and multiplication to the ciphertext during the decryption process. E-voting schemes like STAR-Vote [26] and Helios 2.0 [27] utilize homomorphic cryptography for tallying the votes, because of its simplicity in both application and verification by the public.

A number of protocols like Apollo [28] and Zeus [25] are designed based on the Helios system while trying to mitigate some of its security drawbacks. For example, clickjacking, cross-site forgery, cross-site scripting, and clash attacks are resolved in Apollo by utilizing the voting assistants feature.

III. Challenges in Data and Software Integrity of E-voting Systems

The integrity properties could be fallen into two categories of software and data integrity. Data integrity is protecting the integrity of audit records and election records (especially votes) [5,39]. Software integrity is to ensure that only genuine and unchanged software will be run on the electronic components [11,38].

A. Important propertiesof data integrity

Collected data during running an electronic election is the most important asset of the system. This asset includes stored data, transmitted data, and system recovery/traceability data. The following definitions are the criteria for preserving the safety and integrity of this asset [11,29].

Accuracy: the results of elections are only figured based on votes of participated voters.

Auditability: during running the election and after it the system behavior is traceable.

Verifiability: auditors will be able to verify election results based on the shreds of evidence provided by the system.

Public verifiability: normal people independently are able to verify election results.

Traceability: every needed information will be recorded to let officials trace the cause of any problem.

Recoverability: every needed information will be stored to let recover in case of breaching integrity.

Preventing data alteration: any unauthorized modification, insertion, or deletion of data is prevented.

Data alteration logging: logging component of the e-voting system, records any data modification which may affect the results.

Data authenticity: the system must present enough evidence for auditors to show which record is generated by which entity.

B. Important properties of software integrity

Since the servers store sensitive votes' information, voters, and technical data for system recovery and traceability are very important to ensure they only run authorized software, and their programs have no important security defect [30,41]. The following definitions and criteria explain the integrity features that an e-voting software must meet [31,40].

Server software integrity: to ensure front-end and back-end components will run only the authorized software.

Server software authenticity: the authenticity of the installed software must be evaluable by auditors and administrators (to prevent the installation of malware).

Application of proper software engineering model: the chosen software development model must be one of the best software engineering practices.

IV. Integrity Threats and Solutions of E-voting Systems

A. Threats of e-voting systems

E-voting systems, the same as other electronic systems, are subject to attacks or having bugs [31,37]. This may result in integrity loss and modification of election results. Particularly, if the chosen platforms are either public or private computers, it would be more vulnerable [28,29].

Software bugs: software bugs, the same as malicious codes, are one of the most important roots of integrity loss. Statistically, every 1000 lines of codes would have 15 to 50 errors [28,36]. Considering the fact that e-voting systems are constituted from thousands of lines, the likeliness of the existence of bugs is highly considerable.

Server malicious codes: the malicious codes which aim to change election results could be installed on e-voting systems, even by their IT staff or administrators, to affect the election results [28,35].

Data and records modification: attackers, which potentially also could be the administrators, due to integrity or vulnerability issues may modify the records to affect the results [29,34].

Client malicious codes: as far as normally non-expert users operate client machines, these systems are more prone to be compromised by attackers via running malicious codes, worms, Trojans, or viruses, to take control of systems, collect the important information, or even abusing it as stepping stone to penetrate other systems [30].

B. Solutions of integrity threats

In this part, the important techniques for solving or mitigating integrity threats of e-voting systems are counted and described.

Integrity preservation through cryptography techniques: some cryptographic techniques are designed for protection of the integrity of transmitted data over insecure networks like Transport Layer Security (TLS) or Secure Socket Layer (SSL). In addition, data alteration examination techniques like Message Authentication Codes (MAC) or digital signature also can verify the integrity of the stored data [32,42].

Modern cryptographic techniques: end-to-end cryptographic voting techniques are the algorithms which are able to detect attacks if the final result is not aggregated on casted votes. Moreover, these protocols let people verify whether their votes are correctly counted [43,44].

Using voter side trusted hardware components: if the chosen platform is public or personal computers, the voting platforms are not trustable. Therefore, to overcome the insecure platform issues, trusted hardware could be designed and distributed among voters. Even though the implementation of this method is not economic, but it could be used as a multipurpose platform for e-voting, e-commerce, and other similar applications [45,46].

Malware detection and prevention systems: by heuristic methods or based on the signature of malicious codes, anti-malware programs are able to detect the presence of malicious codes. Though these programs are useful, they are able to detect only known signatures and even in some cases, they fail to remove the recognized malware. Using an up-to-date anti-malware distribution is a useful idea, but only for the mitigation of threats of malicious codes and not to solve this problem [26,33].

Remote software verification: end-point scanning software helps in scanning the computers in virtual private networks for security protection. These programs can scan the computers remotely for ensuring that they will only run authorized software [24,58].

Formal software verification: is a mathematical technique to prove the correctness of the written codes. In this type of verification, the codes must be accurately described as an algorithm. Performing this type of verification is very expensive and hard, and only for particular applications like military software or avionic programs is reasonable [49,50].

Bootable DVDs or CDs: bootable DVDs or CDs that contain needed software and applications for secure vote casting over public or private computers could be distributed among all of the voters, to help them boot up and use their computers in a safe manner. Running this process is expensive, hard, and insecure as the users may not recognize genuine DVDs or CDs from the fake ones. They may not run on all computers, and also the voters' mailing addresses may not be up to date. Accordingly, many of the voters may not receive DVDs or CDs [26,27].

Virtual machines: virtual machines could be used to provide a secure environment as a solution to bypass some difficulties and problems of distribution of bootable DVDs or CDs. These types of virtual machines do not require any configuration or any driver and use resources of the host computer. The main defects of this idea are the danger of distribution of fraudulent images infected by malicious codes and logistical difficulties of distribution of virtual machines for the images [47,48].

Second channel: as the computers might be infected by malicious codes or viruses, for verification of casted votes the voters can use a secondary channel like SMS or telephone to ensure that their votes are cast precisely. This e-voting model has outstanding usability problems [51,52].

Unintelligible contents for malware: easy and helpful techniques like CAPTCHA could be employed to prevent the modification of votes by malware. Since still no malware kit is designed which can support passing the CAPTCHAs, this technique could be utilized to prevent malware to vote on behalf of the people [53,54].

C. *Major unresolved integrity issues of e-voting systems*

Despite all developments of security techniques, still, there are some unsolved serious defects. The most current major integrity issues are:

Security of personal computers: still many important security threats like botnets, malware, or viruses exist that endanger the security of personal computers for casting secure votes [55,56].

Software security problem: despite many techniques are developed for discovering software security bugs, still, there is no guaranty that all of the bugs get discovered. After deployment, the attackers can exploit software bugs to modify election results [30,57].

Problems of advanced cryptographic techniques: despite the advanced cryptographic techniques that can dramatically enhance security, but only certain types of attacks can be detected and still there is no way to recover the original votes [30,31].

V. Conclusion

E-government is a growing field, especially in developing countries. E-voting is one of the most important aspects of e-government as it has a great influence on people’s life. Every developed system, especially those involved in the government area, must be secured against attackers to ban abuse of the system. CIA triangle defines the principal criteria which a secure system must meet. Since these criteria' details depend on the applied system, the relevant concepts and concerns must be clearly distinguished. This study reviews the concepts, threats, and solutions involved in the integrity of e-voting systems. In the last section, the remained and unresolved challenges are discussed.

References

[1] Zamani M, Manaf AA, Ahmad R, Jaryani F, Taherdoost H, Shojae Chaeikar S, Zeidanloo HR. A novel approach for genetic audio watermarking. Journal of Information Assurance and Security. 2010;5:102-11.

[2] Karamizadeh, S., Abdullah, S. M., Manaf, A. A., Zamani, M., & Hooman, A. (2013). An overview of principal component analysis. Journal of Signal and Information Processing, 4(3B), 173.

[3] Manaf AB, Zamani M, Ahmad RB, Jaryani F, Taherdoost H, Shojae Chaeikar S, Zeidanloo HR. Genetic Audio Steganography. International J. of Recent Trends in Engineering and Technology. 2010 May;3(2).

[4] Karamizadeh, S., Abdullah, S. M., Halimi, M., Shayan, J., & javad Rajabi, M. (2014, September). Advantage and drawback of support vector machine functionality. In 2014 International conference on computer, communications, and control technology (I4CT) (pp. 63-65). IEEE.

[5] Shojae Chaeikar S, Jafari M, Taherdoost H, Kar NS. Definitions and criteria of CIA security triangle in electronic voting system. International Journal of Advanced Computer Science and Information Technology. 2012 Oct;1(1):14-24.

[6] Azarnik, A., & Shayan, J. (2012). Associated risks of cloud computing for SMEs. Open International Journal of Informatics (OIJI), 1(1), 37-45.

[7] Zamani M, Abdul Manaf AB, Zeidanloo HR, Shojae Chaeikar S. Genetic substitution-based audio steganography for high capacity applications. International Journal of Internet Technology and Secured Transactions. 2011 Jan 1;3(1):97-110.

[8] Alizadeh, M., Salleh, M., Zamani, M., Shayan, J., & Karamizadeh, S. (2012). Security and performance evaluation of lightweight cryptographic algorithms in RFID. Kos Island, Greece.

[9] Shojae Chaeikar S, Zamani M, Manaf AB, Zeki AM. PSW statistical LSB image steganalysis. Multimedia Tools and Applications. 2018:77(1):805-835.

[10] Shayan, J., Azarnik, A., Chuprat, S., Karamizadeh, S., & Alizadeh, M. (2014). Identifying Benefits and risks associated with utilizing cloud computing. arXiv preprint arXiv:1401.5155.

[11] Zeidanloo HR, Manaf AB, Ahmad RB, Zamani M, Shojae Chaeikar S. A proposed framework for P2P Botnet detection. International Journal of Engineering and Technology. 2010 Apr 1;2(2):161.

[12] Hooman, A., Marthandan, G., Yusoff, W. F. W., Omid, M., & Karamizadeh, S. (2016). Statistical and data mining methods in credit scoring. The Journal of Developing Areas, 50(5), 371-381.

[13] Taherdoost H, Sahibuddin S, Namayandeh M, Jalaliyoon N, Kalantari A, Shojae Chaeikar S. Smart card adoption model: Social and ethical perspectives. Science. 2012 Aug;3(4).

[14] Mollaie, F., Alizadeh, M., Dadsetan, S., & Rashno, A. (2013). Implementation and evaluation of lightweight encryption algorithms suitable for RFID. Journal of Next Generation Information Technology, 4, 65-77.

[15] Yazdanpanah S, Shojae Chaeikar S. IKM-based Security Usability Enhancement Model. IRACST-International Journal of Computer Science and Information Technology & Security (IJCSITS). 2012 Aug(4).

[16] Dehzangi, A., & Karamizadeh, S. (2011). Solving protein fold prediction problem using fusion of heterogeneous classifiers. INFORMATION, An International Interdisciplinary Journal, 14(11), 3611-3622.

[17] Mazdak Z, Azizah BA, Shahidan MA, Shojae Chaeikar S. Mazdak technique for PSNR estimation in audio steganography. Applied Mechanics and Materials. 2012:1(229): 2798-2803.

[18] Fard, M. A. K., Karamizadeh, S., & Aflaki, M. (2011). Enhancing congestion control to address link failure loss over mobile ad-hoc network. arXiv preprint arXiv:1110.2289.

[19] Shojae Chaeikar S, Zamani M, Chukwuekezie CS, Alizadeh M. Electronic Voting Systems for European Union Countries. Journal of Next Generation Information Technology. 2013 Jul 1;4(5):16.

[20] Karamizadeh, S., Abdullah, S. M., Zamani, M., & Kherikhah, A. (2015). Pattern recognition techniques: studies on appropriate classifications. In Advanced Computer and Communication Engineering Technology (pp. 791-799). Springer, Cham.

[21] Alizadeh M, Hassan WH, Zamani M, Khodadadi T, Shojae Chaeikar S. A prospective study of mobile cloud computing. International Journal of Advancements in Computing Technology. 2013;5(11):198-210.

[22] Alizadeh, M., Hassan, W. H., Behboodian, N., & Karamizadeh, S. (2013). A brief review of mobile cloud computing opportunities. Research Notes in Information Science, 12, 155-160.

[23] Yazdanpanah S, Shojae Chaeikar S. Secure SMS Method Based on Social Networks. International Journal of Scientific Research in Science, Engineering and Technology. 2016: 2(6): 368-376.

[24] Karamizadeh, S., Abdullah, S. M., & Zamani, M. (2013). An overview of holistic face recognition. IJRCCT, 2(9), 738-741.

[25] Shojae Chaeikar S, Ahmadi A. Ensemble SW image steganalysis: a low dimension method for LSBR detection. Signal Processing: Image Communication. 2019:70: 233-245.

[26] Karamizadeh, F. (2015). Face Recognition by Implying Illumination Techniques–A Review Paper. Journal of Science and Engineering, 6(01), 001-007.

[27] Shojae Chaeikar S, Manaf AA, Alarood AA, Zamani M. PFW: polygonal fuzzy weighted - an SVM kernel for the classification of overlapping data groups. Electronics. 2020: 9, 615.

[28] Karamizadeh, S., & Arabsorkhi, A. (2018, January). Methods of pornography detection. In Proceedings of the 10th International Conference on Computer Modeling and Simulation (pp. 33-38).

[29] Shojae Chaeikar S, Manaf AB, Zamani M. Comparative analysis of Master-key and Interpretative Key Management (IKM) frameworks. In Cryptography and Security in Computing 2012. InTech.

[30] Karamizadeh, S., Abdullah, S. M., Zamani, M., Shayan, J., & Nooralishahi, P. (2017). Face recognition via taxonomy of illumination normalization. In Multimedia Forensics and Security (pp. 139-160). Springer, Cham.

[31] Shojae Chaeikar S, Moghaddam HS, Zeidanloo HR. Node Based Interpretative Key Management Framework. In Security and Management 2010 (pp. 204-210).

[32] Karamizadeha, S., Mabdullahb, S., Randjbaranc, E., & Rajabid, M. J. (2015). A review on techniques of illumination in face recognition. Technology, 3(02), 79-83.

[33] Shojae Chaeikar S, Razak SA, Honarbakhsh S, Zeidanloo HR, Zamani M, Jaryani F. Interpretative key management (IKM), a novel framework. In 2010 Second International Conference on Computer Research and Development, 2010 May 7 (pp. 265-269). IEEE.

[34] Karamizadeh, S., Cheraghi, S. M., & MazdakZamani, M. (2015). Filtering based illumination normalization techniques for face recognition. Indonesian Journal of Electrical Engineering and Computer Science, 13(2), 314-320.

[35] Zamani M, Manaf AB, Ahmad RB, Jaryani F, Shojae Chaeikar S, Zeidanloo HR. Genetic audio watermarking. In International Conference on Business Administration and Information Processing, 2010 Mar 26 (pp. 514-517). Springer, Berlin, Heidelberg.

[36] Fard, M. A. K., Karamizadeh, S., & Aflaki, M. (2011). Enhancing congestion control to address link failure loss over mobile ad-hoc network. arXiv preprint arXiv:1110.2289.

[37] Yazdanpanah S, Shojae Chaeikar S, Zamani M, Kourdi R. Security features comparison of master key and IKM cryptographic key management for researchers and developers. In International Conference on Software Technology and Engineering, 3rd(ICSTE 2011) 2011. ASME Press.

[38] Shayan, J., Abdullah, S. M., & Karamizadeh, S. (2015, August). An overview of objectionable image detection. In 2015 International Symposium on Technology Management and Emerging Technologies (ISTMET) (pp. 396-400). IEEE.

[39] Sen J, editor. Cryptography and Security in Computing. BoD–Books on Demand; 2012 Mar 7.

[40] Fard, M. A. K., Karamizadeh, S., & Aflaki, M. (2011, May). Packet loss differentiation of TCP over mobile ad hoc network using queue usage estimation. In 2011 IEEE 3rd International Conference on Communication Software and Networks (pp. 81-85). IEEE.

[41] Honarbakhsh S, Masrom M, Zamani M, Chaeikar SS, Honarbakhsh R. A Trust Based Clustering Model for Dynamic Monitoring in Ad hoc Network. InInternational Conference on Computer and Computational Intelligence (ICCCI 2010) 2010 Dec 25.

[42] Karamizadeh, S., Shayan, J., Alizadeh, M., & Kheirkhah, A. (2013). Information Security Awareness Behavior: A Conceptual Model for Cloud. International Journal Of Computers & Technology, 10(1), 1186-1191.

[43] Shojae Chaeikar S. Pixel Similarity Weight for Statistical Image Steganalysis [dissertation]. Universiti Teknologi Malaysia; 2016.

[44] Karamizadeh, S., & Abdullah, S. M. (2018). Race classification using gaussian-based weight K-nn algorithm for face recognition. Journal of Engineering Research, 6(2), 103-121.

[45] Zamani M, Manaf AB, Abdullah SM, Shojae Chaeikar S. Correlation between PSNR and bit per sample rate in audio steganography. In11thInternational Conference on Signal Processing 2012 Apr 2 (pp. 163-8).

[46] Karamizadeh, S., Abdullah, S. M., Shayan, J., Nooralishahi, P., & Bagherian, B. (2017). Threshold Based Skin Color Classification. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 9(2-3), 131-134.

[47] Shojae Chaeikar S, Ahmadi A. SW: a blind LSBR image steganalysis technique. In the 10thInternational Conference on Computer Modeling and Simulation2018 Jan 8 (pp. 14-18). ACM.

[48] Fard, M. A. K., Bakar, K. A., Karamizadeh, S., & Foladizadeh, R. H. (2011, May). Improve TCP performance over mobile ad hoc network by retransmission timeout adjustment. In 2011 IEEE 3rd International Conference on Communication Software and Networks (pp. 437-441). IEEE.

[49] Shojae Chaeikar S. Interpretative Key Management Framework (IKM) [dissertation]. Universiti Teknologi Malaysia; 2010.

[50] Karamizadeh, S., Abdullah, S. M., Shayan, J., Zamani, M., & Nooralishahi, P. (2017). Taxonomy of Filtering Based Illumination Normalization for Face Recognition. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 9(1-5), 135-139.

[51] Azarnik, A., SHAYAN, J., ZADEH, S. K., & PASHANG, A. (2013, February). Lightweight authentication for user access to Wireless Sensor networks. In Proceedings of the 12th WSEAS Int. Conf. on Electronics, Hardware, Wireless and Optical Communications (EHAC'13), Cambridge, UK (pp. 35-39).

[52] Duan, W., Nasiri, R., & Karamizadeh, S. (2019, December). Smart City Concepts and Dimensions. In Proceedings of the 2019 7th International Conference on Information Technology: IoT and Smart City (pp. 488-492).

[53] Karamizadeh, S., & Arabsorkhi, A. (2017). Enhancement of Illumination scheme for Adult Image Recognition. International Journal of Information and Communication Technology Research, 9(4), 50-56.

[54] Dehzangi, A., Foladizadeh, R. H., Aflaki, M., & Karamizadeh, S. (2011, April). The application of fusion of heterogeneous meta classifiers to enhance protein fold prediction accuracy. In Asian Conference on Intelligent Information and Database Systems (pp. 538-547). Springer, Berlin, Heidelberg.

[55] Karamizadeh, S., & Arabsorkhi, A. (2018). Skin Classification for Adult Image Recognition Based on Combination of Gaussian and Weight-KNN. International Journal of Information and Communication Technology Research, 10(2), 56-62.

[56] Zadeh, S. K. (2012). Information Security Behaviours in Enhancing Awareness (Doctoral dissertation, Universiti Teknologi Malaysia).

[57] Fard, M. A. K., Karamizadeh, S., & Aflaki, M. (2011). Enhancing congestion control to address link failure loss over mobile ad-hoc network. arXiv preprint arXiv:1110.2289.

[58] arabsorkhi A, karamizadeh S. Method to improve the illumination normalization in adult images based on fuzzy neural network. فصلنامه 11 ;2020 .اطلاعات فناوری (41 and 42) :1-12 URL: http://jor.iranaict.ir/article-1-1503 en.html

Iberian Journals

Iberian Journal of Applied Sciences and Innovations

2021 vol 1, Issue 2

Constructing New Features for Spam Detection on Twitter

Arash Erami
Department of Computer
Engineering, Shiraz Branch,
Islamic Azad University,
Shiraz, Iran
arsherami@gmail.com

*Elham Parvinnia**
Department of Computer
Engineering, Shiraz Branch, Islamic
Azad University, Shiraz, Iran
parvinnia@iaushiraz.ac.ir

Abstract—Nowadays, by the growth of the internet, social networks are attracting unprecedented attention to themselves. Most people are at least active in one social network. Users in social networks follow their favorite people and topics to discover the latest news about them. This rising number of users has made social networks fertile grounds for advertising and finding the bait. Social networks also become celebrities' popularity criterion. The problem is that some accounts created to spread malicious links, steal user's information, and display advertising. These accounts are mainly controlled and supervised by an automatic program. Not only the increase in fake accounts has costs for social networks companies, but it also influences network quality. In this paper, we offer some new and low-cost features to distinguish spam accounts on Twitter. This paper offers some low cost and a new feature to distinguish spam accounts of Twitter. We apply machine learning algorithms to predestined datasets, and by looking at the characteristics of the accounts, then we anticipate class of users by the accuracy of 99.18%.

KeyWords— Spam Detection • Machine Learning • Twitter Spam Bots • Feature Extraction.

I. INTRODUCTION

Online Social Networks (OSNs) have spread at a remarkable speed over the past decade. They have become one of the main ways for people to keep track of events and communicate with one another. Websites such as Facebook, Twitter, and LinkedIn are consistently on the top 20 most-visited websites. Twitter is the fastest growing social networking web site among all the social networking websites [1]. The increase in the use of social networking websites is gaining a great deal of recognition because they play a double role of online social networking and micro-blogging, but these websites have constraints, i.e., the spammers.

Twitter is a popular online social networking and microblogging tool, which allows users to share content limited to 140 characters. These small messages (tweets) create substantial information dissemination in the network and make Twitter a successful social network for content share. There are about 500 million tweets published every day [expanded ramblings, 2015].

Spam is becoming a significant problem with Twitter as well as with other online social networking websites. Spammers can use Twitter as a tool to send unsolicited messages to legitimate users, post malicious links, and hijack trending topics. Spammers could be phishers, malware propagators, marketers, and adult content propagators. Fake followers are Twitter accounts specifically created to inflate the number of followers of a target account, in order to increase its popularity and influence.

With more than 500 million users on Twitter, it is almost impossible to manually verify the identity of every user who signs up on Twitter, and it is even more challenging to keep track of users who tend to spread information of questionable authenticity, unknowingly or deliberately. Therefore, we need some tools to identify these spammers automatically.

More than 19% of all tweets are about organizations or product brands, less than 20% of which are shown to have significant sentiment [13].

Since spam bots amend their behaviors to remain undetected, we need some new features to detect them. In this work, we combine features and make new rules to distinguish a spam account from a legitimate one.

a) Spam Bots and Sybil Accounts

There are several ways to take advantage of free online advertisement, and many agencies and companies rely on Spam Bots or Sybil accounts. These fake accounts pretend to be legitimate distinct users, and their behavior seems to be similar [12]. Some of these accounts might seem surprisingly akin to being legitimate. They cause the social network platform millions of dollars in revenue loss each year.

In a social network such as Twitter, users can access all public information, including usernames, tweets, etc. Spammers need to parse the public content to get all the information they need for both sending the malicious content (usernames) and for making it appealing to the victim (relationships, interests, and content of previous messages). By using this information, it is possible to semantic to analyze victim accounts.

b) Roadmap

The remainder of this paper is structured as follows. In Section 2, we consider related work in Twitter spams and bot detection. In Section 3, we describe the outlines of our baseline dataset. In Section 4, we extract and examine some new features of our baseline dataset. In Section 5, we present our results and compare them to previous works. In Section 6, we present possible methods for future work.

II. RELATED WORK

[8] used a machine learning approach to distinguish spam bots from normal ones. He suggested three graph-based features and three content-based features. They used graph-based features such as a number of friends, a number of followers, and a follower ratio. He also extracted the number of duplicate tweets, the number of HTTP links and the number of replies/mentions from the user's 20 most recent tweets. His best overall performance was .917 by Naïve Bayesian algorithm.

Some research focused on analyzing the behaviors of social spammers and detecting these spammers [5] [4]; [11]. LEE, K, and others conducted a long-term study of content polluters, analyzed their behaviors, and detected them. [6] used a machine learning approach to detect social spammers.

In other work [10], researchers suggested some new features and used the profile-based feature, content-based feature, graph-based features to distinguish spammers.

[7] analyze how spammers operate in social network sites operate. They created a large and diverse set of "honey-profiles" on three large social network sites and logged the messages and friends request they received. They analyzed the collected data and identified the anomalous behaviors of users who contacted predestine profiles. Based on the analysis of their behaviors, they showed that it is possible to automatically identify the accounts used by spammers and correctly detected 15,857 spam profiles.

III. DATASET

In this section, we describe the datasets of Twitter accounts that we used to conduct our study throughout the paper. We use "The Fake Project Dataset" provided by MIB Datasets, which is publicly available to the scientific community. This dataset contains five different sources of Twitter user's data in two classes: Human and Fake. Gathering data from multiple sources make the data more reliable because each source contains different kinds of behaviors and information. More information on the dataset can be found in [2] as we use the same dataset in order to compare the result.

Each source contains four separate files named: followers, friends, tweets, users. Here are the details of each file:

Followers: The list of user IDs and the IDs of their followers

Friends: The list of user IDs and the IDs of their friends

Tweets: Contains 19 attributes of a tweet like a tweet text, time of creation, number of hashtags, user ID, tweet ID, the source of the tweet, etc.

Users: Contains user 's information from such as name, number of friends, location, description, language, profile text color, class, etc.

In each file, we have a user`s ID so that we can join files together by a unique User ID.

Since we only had 1950 genuine users and many machine learning algorithms are affected by the imbalance [9] of natural distributions of the minority and majority classes, we selected all human users and randomly selected 1950 accounts out of 3351 bots accounts, to balance out both categories. Table 1 provides the details of this dataset.

Table 1: Brief Description of Our Dataset Sources

Dataset	Accounts
TFP(@TheFakeProject)	**469**
E13 (#elezzioni2013)	**1481**
INT(intertwitter)	**1337**
TWT(twittertechnology)	**845**
FSF(fasstfolliwerz)	**1169**
HUM(total human dataset)	**1950**
FAKE(total fake dataset)	**3351**
BAS(baseline dataset Hum Union with random Fake dataset)	**3900**

a) Preprocessing

In this section, we reduced features in order to make our dataset lighter. First, we removed useless features and features with lots of missing values. So, how can we tell if a feature is useful? Attributes that have lots of different values for each user such as name, description, URL, and ID are not practical for machine learning algorithms. Since in machine learning algorithms such as decision tree, unique attributes, and attributes that have massive variety in value have a GINI index close to zero, which means it cannot be a good separator for detecting different classes.

IV. PROPOSED FEATURES

We have extracted features from two sources of information: the feature of tweets and the user's information. Obviously, good features should be informative and have discriminative power. Some features such as followers count and friends count have some correlation with each other so we decided to make a ratio.

a) Extracting New Features

We previously mentioned that programmers of these bots try to find ways to evade from spam detection algorithms, therefore it is necessary to continuously need new features and algorithms to correctly distinguish bots.

Based on [2] we categorized attributes into three categories by their crawling costs.

A) Profile: Features that use information in a profile account.

B) Timeline: Features that use information in tweets.

C) Relationship: Features that uses information about the accounts that are in a relationship with followers of the target account

As [2] mentioned features of profile have the least crawling time and relationship features have the most time needed. Our suggested features are only in profile and timeline category.

b) Profile Attributes

Attributes extracted from timeline need more time to collect than attributes which can be found in profile information. In user information, we had "created_time" so we can calculate new attribute named "Howlong_day" that shows how long does this account exist and no need for curling timeline. Another attribute that we amended to be more accurate is the number of tweets. Since the number of tweets in users’ info had slightly different from tweet files, we calculate the exact number called that feature "num_of_tweets."

Dou to following reasons, the number of tweets foregoing in users’ files and the number of tweets existed in tweets file are not the same. One of the possible reasons for this is that user data is cached by Twitter and thus, it is not always updated on the current number of followers, friends, statuses (tweets), etc. Another reason might be that we were not able to collect all the tweets produced by a user because the user deleted some of his/her original tweets. Alternatively, because he/she "protected" his/her timeline. The third possible reason is that the crawling of user data and tweets have been carried out at two slightly different times, and this may have caused inconsistencies.

We made new ratios by combining profile attributes. These ratios demonstrate the popularity of the account. We did this because using only one attributes could deceive the machine learning algorithm. For example, number of followers without considering the number of friends could lead us to wrong predictions.

Twitter saves each user's signup date and the time of each tweet. We figured out how long each user has been active on Twitter by an attribute called "Howlong_day." Spambots probably have a shorter lifespan than real users, because spam bots may be deactivated by Twitter spam detection system or deactivated by programmer after some time. We assume that the more significant this number is, the higher possibility of being a legitimate user.

Follower counts have long provided a decent indicator of Twitter accounts' popularity. Twitter provides a feature to make a list of accounts that a user follows. By this feature, users can make any list they want (brands, news, entertainment, ...), so users can categorize each account to a list. We think this feature can be another indicator of popularity. The more list you are on, the more popular you probably are. So we create an attribute called "listed_count" that shows the number of public lists a specific user is a member of. Obviously, if the number of followers is relatively small compared to the number of people you are following, the follower ratio is relatively small and close to zero. At the same time, the probability that the associated account is spam is high.

We know a number of followers and friends are two major attributes, and using ratio can be helpful. To consider other attributes, we add "listed_count" attribute to this ratio and create "Ratio1" feature. Because "listed_count" most of the time is much smaller than the number of followers and friends, we rose it to a power of two to infect ratio. In the ratio2 feature, we emphasize on a number of friends by rose friends count to the power of two. To see how tweets can attract followers, we created "populaty_by_tweet" ratio. Growth rate considers how fast an account absorbs followers.

$$\text{Ratio1} = \frac{\text{followers\_count} + \text{num\_of\_tweets}}{\text{friends\_count} + \text{listed\_count}^2} \qquad (1)$$

$$\text{Ratio2} = \frac{\text{friends\_count}^2 + \text{num\_of\_tweets}}{\text{followers\_count}} \qquad (2)$$

$$\text{Popularity\_by\_tweets} = \frac{\text{friends\_count} + \text{number\_of\_tweets}}{\text{followers\_count}} \qquad (3)$$

$$\text{Growth\_rate} = \frac{\text{followers\_count}}{\text{howlong\_day}} \qquad (4)$$

c) Timeline Attributes

Besides profile attributes, we select six timeline attributes that exist in every single tweet. Since these attributes are for every tweet, not every user, we calculate the average, variance, and maximum of these attributes for all accounts in our dataset.

Favorite count: Shows how many users have set the tweet as a favorite Retweet count: Demonstrates the importance of the tweet. Retweets build on the authority of another user and are used to increase the volume of followers to see a tweet.

Replay count: Represents the reaction of users to a certain tweet

Num hashtags: Number of hashtags in every single tweet.

Num URLs: Number of URL in every single tweet Num mentions: Number of mentions in every single tweet These attributes are important for us for the following reasons. Since spam tweets are seldom retweeted, set to favorite, or replied to, we selected these features. Spam bots are very likely to share URLs to reach their goal like phishing, advertising, or spreading malware. Spammers also use trending hashtags and mention other users to be indexed in search results and more people be able to see their tweets. For these reasons, we choose the attributes "num_hashtags," "num_URLs" and "num_mentions." We also calculated tweet lengths for each tweet and then computed the average tweet length and created a new attribute called "average character."

Mention: To address a particular user in order to reference the user directly. Mentions may be used by spammers to personalize the message in an attempt to increase the likelihood a victim follow spam links. Mentions can be used to communicate with users that do not follow a spammer.

When a tweet is sent, Twitter keeps the source of the tweet. This is embodied in the device type and application or API, which is used to publish the tweet. We put all sources into the three following categories:

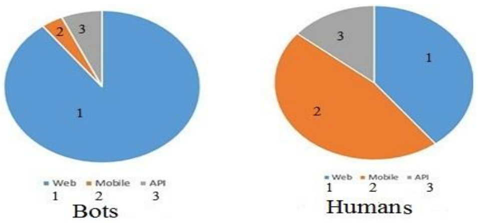
Web: Uses Twitter web site for sending tweets.

Mobile or Tablet: Tweets sent from portable devices like phones and tablets.

3rd Party and API: Tweets sent from applications and API requests.

We categorized tweets for each user, then we took the most repeated sources and made a new attribute called "Mode Source." Figure 1 shows the dispersion of legitimate users against spam bots.

Figure 1 Source dispersion of legitimate users versus spam bots



So far we have created 21 features from our base dataset, which is listed in table 2. We should test the attributes to see if they are suitable for detecting spammer. Our new features are tested in the following section.

Table 2: Extracted Features

#	Name	Description
1.	Average_character	Average number of tweet character
2.	Average_favorite_count	Average number of tweets favorite
3.	Average_num_hashtags	Average number of hashtags in tweets
4.	Average_num_mentions	Average number of mention in tweets
5.	Average_num_urls	Average number of URL in tweets
6.	Average_reply_count	Average number of replay for tweets
7.	Average_retweet_count	Average number of retweet of their tweets
8.	Variance_favorite_count	Variance of favorite count
9.	Variance_num_hashtags	Variance of number of hashtags in tweets
10.	Variance_num_mentions	Variance of number of mentions in tweets
11.	Variance_num_urls	Variance of number of URL in tweets
12.	Variance_reply_count	Variance of number of replay to tweets
13.	Variance_retweet_count	Variance of number of retweeted tweets
14.	Mode_of_source	Source that users mostly used for tweeting
15.	Maximum_ favorite_count	Maximum number of

		favorite tweet
16.	Maximum_num_hashtags	Maximum number of hashtags in one tweet
17.	Maximum_num_urls	Maximum number of URLs in one tweet
18.	Maximum_reply_count	Maximum number of replay to one tweet
19.	Maximum_num_mentions	Maximum number of mentions in one tweet
20.	Maximum_retweet_count	Maximum number of retweeted tweet
21.	Howlong_day	Number of days that users register up to now

d) Evaluation Methodology

We have two classes in our dataset: humans and bots (spammers).

- True Positive (TP): the number of those bots recognized as bots;
- True Negative (TN): the number of those humans followers recognized as human;
- False Positive (FP): the number of those humans recognized as bots;
- False Negative (FN): the number of those bots recognized as human.

In order to evaluate the application of every single rule to the accounts in the baseline dataset, we have to consider the following standard evaluation metrics:

- Accuracy: the proportion of predicted true results (both true positives and true negatives) in the population, that is $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision: the ratio of predicted positive cases that are indeed real positive, which is $\frac{TP}{TP+FP}$
- Recall: the ratio of real positive cases that are indeed predicted positive, which is $\frac{TP}{TP+FN}$
- F-Measure: the harmonic mean of precision and recall, namely $\frac{2*\text{precision}*\text{recall}}{\text{precision}+\text{recall}}$
- The area under the curve (AUC): that relates the hit rate to the false alarm rate has become a standard measure in testing the accuracy of predictive modeling.

e) Testing Our Features

To obtain the optimal classifier, this is crucial to combine the features effectively. We test our new features using k-fold Cross-validation decision tree algorithms. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged (or otherwise combined) to produce a single estimation. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

We assayed each attribute alone in 10-fold validation using a decision tree algorithm. The result is shown in Table 3.

Table 3: Testing new attributes by 10 k fold validation decision tree

#	Attribute	Accuracy	Precision	Recall	F-Measure
1.	Average_character	80.03	79.94	79.92	79.91
2.	Average_favorite_count	89.85	89.19	90.66	89.90
3.	Average_num_hashtags	82.69	82.32	83.23	82.70
4.	Average_num_mentions	91.33	95.79	86.40	90.81
5.	Average_num_urls	85.59	97.20	73.18	83.48
6.	Average_reply_count	86.00	96.71	74.54	84.15
7.	Average_retweet_count	79.95	78.21	86.38	80.99
8.	Variance_favorite_count	89.44	86.93	92.81	89.74
9.	Variance_num_hashtags	84.67	89.74	78.25	83.58
10.	Variance_num_mentions	90.26	96.06	83.86	89.51
11.	Variance_num_urls	85.64	97.33	73.19	83.52

12.	Variance_reply_count	88.28	92.54	83.49	87.66
13.	Variance_retweet_count	80.95	94.43	65.67	77.37
14.	Mode_of_source	75.23	69.44	90.14	78.40
15.	Maximum_favorite_count	89.05	88.65	89.52	89.05
16.	Maximum_num_hashtags	90.15	91.37	88.55	89.93
17.	Maximum_num_urls	84.97	97.91	71.40	82.56
18.	Maximum_reply_count	69.85	81.95	50.93	62.79
19.	Maximum_num_mentions	91.38	95.80	86.44	90.86
20.	Maximum_retweet_count	88.56	96.82	79.69	87.39
21.	Howlong_day	86.46	96.98	75.33	84.74
22.	Ratio1	96.77	97.56	95.93	96.72
23.	Ratio2	94.31	92.1	96.86	94.42
24.	Popularity_by_tweets	83.90	78.98	92.42	85.11
25.	Groth_rate	85.79	80.31	94.80	86.94

The result of our features average compares to previous work which is mentions in [2] average is shown in table 4. The result shows an observable increase in accuracy, precision, recall, and F-measure. The accuracy of 86.04% for average singe feature confirm these features are good.

Table 4: Average of Single feature comparison

Average of Single Feature	Accuracy	Precision	Recall	F-Measure
Camisani-Calzolari	64.4	71.3	58.18	54.02
Van Den Beld	41.36	41.36	68.12	18.65
Social Bakers	50.2	66.26	4.9	69.2
Our features	86.04	89.77	82.54	85.28

We test our 4 ratios together by 10-fold validation and different machine learning algorithms. We did this because our 4 ratios are extracted from profile information and need less time to acquire.

Table 5: Result of all rules together for 10 fold validation

#	Algorithm	Accuracy	Precision	Recall	F-Measure
1.	Decision Tree	98.31	98.92	97.69	98.30
2.	Random Forest	97.87	99.11	96.59	97.83
3.	AdaBoost	98.38	98.97	97.82	98.39
4.	Naïve Bayes	95.18	97.98	92.34	95.05

Although we only use profile information in our ratio, the results are remarkable. We achieve the accuracy of 98.38% by using profile information and AdaBoost algorithm. We want to go farther and add our extracting timeline features to this ratio to make it even better. In the next section, we describe it more.

f) Finding Best Feature Set

After testing attributes one by one now it is time to find the best feature set for getting the best result. We used “Forward Selection” algorithms in RapidMiner software to find the best-combined features.

The Forward Selection operator starts with an empty selection of attributes and, in each round, it adds each unused attribute of the given Example Set. For each added attribute, the performance is estimated using the inner operators, in this case, cross-validation. Only the attribute giving the highest increase in performance is added to the selection. Then a new round is started with the modified selection. This implementation avoids any additional memory consumption besides the memory used originally for storing the data and the memory which might be needed for applying the inner operators. The stopping behavior parameter specifies when the iteration should be aborted. There are three different options:

Without increase: The iteration runs as long as there is an increase in performance.

Without an increase of at least: The iteration runs as long as the increase is at least as high as specified, either relative or absolute. The minimal relative increase parameter is used for specifying the minimal relative increase if the use relative increase parameter is set to true. Otherwise, the minimal absolute increase parameter is used for specifying the minimal absolute increase.

Without significant increase: The iteration stops as soon as the increase is not significant to the level specified by the alpha parameter.

We gave all our extracted features to the forward selection algorithm and choose stopping behavior to be iteration without a significant increase. The output of this operator will be a list of all attributes with weights 0 or 1, which 1 means good to select and 0 means not have a significant effect on the validation result and not selected. Table 6, shows a list of the most effective attributes for the detection model.

Table 6: Selected Attributes Using a Forward Selection Algorithm.

#	Attributes	Weights
1	Average_reply_count	1
2	Ratio1	1
3	Ratio2	1
4	Popularity_by_tweets	1
5	Variance_favorite_count	1

V. RESULT

In Table 7, we can see the result of our feature set on the base dataset.

Table 7: Result of our feature set by 10 fold validation

#	Algorithm	Accuracy	precision	Recall	F-Measure	AUC
1.	Decision Tree	99.18	99.32	99.02	99.17	0.993
2.	Random Forest	98.87	99.11	98.59	98.85	0.998
3.	AdaBoost	98.54	98.52	98.56	98.53	0.796
4.	Naïve Bayes	72.66	99.06	45.23	62.08	0.989
5.	K Nearest Neighbors	92.05	92.23	91.85	92.02	0.951

In comparison to [10], we got the same result with fewer features. We only used profile information and timeline features and we do not imply relationship features to our feature set. It means that the time of crawling for features is much less than [10] research.

Table 8 is comparing the result of two previous research [10], [7] that have been mentioned on [2] by the same dataset that we used in our work. Our detection system has higher accuracy than [7].

Table 8: Compare the result to previous works

Algorithms	Suggested by	Accuracy	Precision	Recall	F-Measure	AUC
Random Forest	Stringhini	.981	.983	.979	.981	.995
	Yang	.991	.991	.991	.991	.998
	Our	.9887	.9911	.9859	.9885	.998
Decision Tree	Stringhini	.979	.984	.974	.979	.985
	Yang	.990	.991	.989	.990	.997
	Our	.9918	.9932	.9902	.9917	.993
Adaptive Boost	Stringhini	.968	.965	.970	.968	.995
	Yang	.988	.989	.937	.988	.999
	Our	.9854	.9852	.9856	.6208	.989
K-NN	Stringhini	.954	.961	.946	.953	.974
	Yang	.966	.966	.966	.966	.983
	Our	.9205	.9223	.9185	.9202	.951

In [2], they used features from the profile category proposed by others which contains 23 features after that they used all features without considering the timing category which contains 49 features. Table 9 reports the result of k fold validation on our suggested five features and compare it to 23 features of profile category and 49 features from all categories presented in [2].

Table 9: Result of Our 5 Features Compare to 23 Features and 49 Features

Algorithms	Features	Accuracy	Precision	Recall	F-Measure	AUC
Random Forest	49 features	.994	.997	.990	.994	.999
	23 Features	.987	.993	.980	.987	.995
	Our5 Features	.9887	.9911	.9859	.9885	.998
Decision Tree	49 features	.992	.991	.992	.992	.993
	23 Features	.983	.987	.979	.983	.983
	Our5 Features	.9918	.9932	.9902	.9917	.993

Adaptive Boost	49 features	.987	.988	.987	.987	.999
	23 Feature	.972	.975	.969	.972	.995
	Our5 Features	.9854	.9852	.9856	.6208	.989
K-NN	49 features	.971	.963	.979	.971	.990
	23 Features	.957	.961	.953	.957	.978
	Our5 Features	.9205	.9223	.9185	.9202	.951

We realize from table 9 that our suggested features have more accuracy than the 23 profile features suggested before. Although our features come with slightly less accuracy by comparing them to 49 features, we should consider the time for gathering relationship features is much more from features from profile and timeline.

VI. FUTURE WORKS

In the future, we can aim for new features based on text mining and analyzing tweets, and if tweets meaning are related to the hashtag, they are making or not. Also, some new behavioral features, such as the speed of replying to tweets and making trending hashtags can be considered.

REFERENCES

[1]'Compete Site Comparison' <http://siteanalytics.compete.com/facebook.com+myspace.com+twitter.com> accessed 11 June 2016.

[2] Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M. (2015). Fame for sale: Efficient detection of fake Twitter followers. Decision Support Systems, 80, 56-71. http://dx.doi.org/10.1016/j.dss.2015.09.003

[3] '388 Amazing Twitter Statistics And Facts' (expandedramblings, 2015) <http://siteanalytics.compete.com/facebook.com+myspace.com+twitter.com> accessed 10 May 2016.

[4] Ghosh, S., Korlam, G., Ganguly, N. (2011). Spammers' networks within online social networks. Proceedings Of The 20Th International Conference Companion On World Wide Web - WWW '11. http://dx.doi.org/10.1145/1963192.1963214

[5] Lee, K., Caverlee, J., Webb, S. (2010). Uncovering social spammers. Proceeding Of The 33Rd International ACM SIGIR Conference On Research And Development In Information Retrieval - SIGIR '10. http://dx.doi.org/10.1145/1835449.1835522

[6] McCord, M., Chuah, M. (2011). Spam Detection on Twitter Using Traditional Classifiers. Lecture Notes In Computer Science, 175-186. http://dx.doi.org/10.1007/978-3-642-23496-5_13

[7] Stringhini, G., Kruegel, C., Vigna, G. (2010). Detecting spammers on social networks. Proceedings Of The 26Th Annual Computer Security Applications Conference On - ACSAC '10. http://dx.doi.org/10.1145/1920261.1920263

[8] Wang, A. (2010). Detecting Spam Bots in Online Social Networking Sites: A Machine Learning Approach. Lecture Notes In Computer Science, 335-342. http://dx.doi.org/10.1007/978-3-642-13739-6_25

[9] Weiss, G., Provost, F. (2003). Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction.

[10] Yang, C., Harkreader, R., Gu, G. (2013). Empirical Evaluation and New Design for Fighting Evolving Twitter Spammers. IEEE Transactions On Information Forensics And Security, 8(8), 1280-1293. http://dx.doi.org/10.1109/tifs.2013.2267732

[11] Yardi, S., Romero, D., Schoenebeck, G., Boyd, D. (2009). Detecting spam in a Twitter network. First Monday, 15(1). http://dx.doi.org/10.5210/fm.v15i1.2793

[12] Yu, H., Kaminsky, M., Gibbons, P., Flaxman, A. (2006). SybilGuard. ACM SIGCOMM Computer Communication Review, 36(4), 267. http://dx.doi.org/10.1145/1151659.1159945

[13] Yu, S., Kak, S. (2012). A Survey of Prediction Using Social Media.long.