ICIC Express Letters
Volume **15**, Number **5**, May **2021**

 ISSN 1881-803X
pp. **481–491**

NEW MODEL FOR TOR NETWORK TRAFFIC IDENTIFICATION BASED ON LIGHT GRADIENT BOOSTING MACHINE

Hussein Yousef AbuMansour

Forensic Sciences Department
College of Criminal Justice
Naif Arab University for Security Sciences
KSA P.O.Box 6830, Riyadh 11452, Saudi Arabia
hmansour@nauss.edu.sa

Received October 2020; accepted January 2021

Abstract. *The Onion Router network (Tor) is a reliable anonymous communication system over the Internet and tracking its users is considered a difficult and challenging activity. Many Internet users are keen to remain anonymous on the Internet due to many reasons such as freedom, privacy protection, and confidential surfing. Cyber-perpetrators are often using Tor in order to cover their illegitimate activities through hiding their real identities, which are notably increased recently and become a challenge to cyber investigators and scholars as well. This research work proposes a new approach that uses LightGBM's algorithm for training the classification model in order to detect Tor-related traffic. The proposed method's performance was compared with classical machine learning algorithms on the same dataset, and experimental results revealed promising results. Particularly, the proposed approach achieves a detection rate of 98.6% with a false positive rate of 0.9%.*

Keywords: Cybersecurity, Machine learning, Classification, Tor network

1. **Introductions.** The Onion Router network or what so-called by "Tor" is a freeware used by Internet users to conceal their identities during their different activities on the Internet and enable anonymous severing of the Internet. Tor network is a free open sources software by a group of volunteer-operated servers that offer secure Internet surfing and privacy protection for its users. Tor users from both categories, individuals as well as organizations use this network through connecting to series of virtual secured tunnels instead of one direct connection when browsing, communicating and information sharing over the Internet and public networks without breaching their privacy [1]. On the other hand, Tor is an effective tool for circumventing control measures which enable its users to reach otherwise blocked entities or content. Tor can also be used as a building block to create new communication tools with built-in privacy protection features for the software developers.

Routing process in Tor networking is applied through encryption in the application layer of a communication protocol stack, nested like the onion layers. Tor network encrypts the flow of data that comprises different chunks including the IP address of the next node's destination, multiple times those sent through a virtual circuit comprising randomly and successive chosen Tor relays [2]. Each relay does decrypt a one layer of encryption to retrieve only the next relay in the route aiming to pass the rest of the encrypted data on to it. The innermost layer of the encrypted data is decrypted by the last relay and then sends the original data to the designated destination without disclosing its content and remains anonymous, i.e., source IP address remains unknown [3]. The routing process of the data is partially covered at every hop in the Tor circuit, and this process eliminates

DOI: 10.24507/icicel.15.05.481

any single point at which the communicating peers can be specified through network surveillance facilities that required knowing its source IP, destination IP as well.

Recently, Tor has been used increasingly for committing illegal online activities such as accessing censored data, consolidating political activities such as hacktivism's different activities [4], or avoid laws against criticism of political and ley persons. Tor has, for instance, been used by criminal enterprises, hacktivism groups, and cross-purpose law enforcement agencies, sometimes; similarly, U.S. government agencies fund Tor [5]. Tor has been called by famous economist as "dark web cornerstone" in relation to Silk Road and crypto currencies [6-9]. In Figure 1, we show a Tor circuit example that illustrates the communication among two points, a node and a destination server which is plain in this case (unencrypted) outside Tor network. That is, Tor provides encryption to traffic data within the Tor network only [10]. When data messages went out of the Tor network, it is the user's decision to whether encrypt the traffic or leave it plain. Consider for an instant a user who attempts to access a website that uses SSL security protocol, in this case not only the communication channel is encrypted in Tor connection, it is also encrypted outside of Tor network, i.e., SSL protocol is used. However, this is not the case a user attempts to access website uses HTTP only, the communication, in this case, is encrypted within the Tor network only and plain outside of the Tor network.

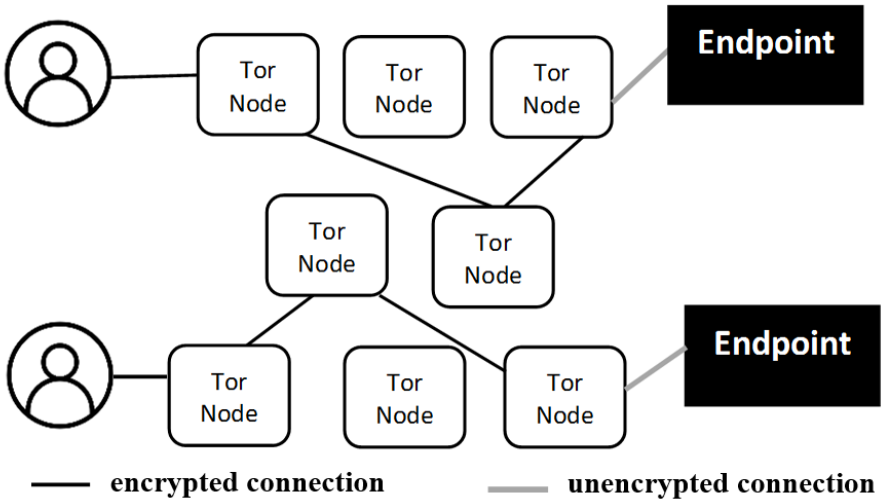

FIGURE 1. Tor network example

The rest of the article is structured as follows. In Section 2, an introduction about the Tor network with current identification approach is presented. In Section 3, the selected approach is explained in detail. Experimental results and dataset are discussed in Section 4. Finally, conclusion and future research direction are presented in Section 5.

2. **Literature Review.** Last decades have witnessed increasing research effort in the domain dark web investigations, where scholars have presented several solutions aiming to achieve accurate identification for generated traffic through Tor networks. Lashkari et al. [11], for instance, considered the time analysis on Tor traffic flows, which were seized between two points. They achieved accuracy rate of 95% when using "RandomForest" classification method. Particularly, they proposed a "Hidden Markov Models" (HMM) based method in order to identify Tor traffic in four different categories, i.e., Peer to Peer communication, File Transfer, chatting and browsing, and features they employ were extracted from Tor traffic flows. They employed HMM approach in building entry and exit models ("ingress and egress") of the different applications as described above. As a result, their model archived an error rate value of 8.0%.

Chakravarty et al. [12] presented work that relies on committing an attack on a Tor network, in an attempt to reveal the identity of the clients, i.e., their IP addresses. In their work, they proposed a real-time traffic analysis attack based on intentionally perturbing the characteristics of user traffic at the server side which colludes server in their case. Similarly, they observed a similar perturbation via statistical correlation at the client's side as well. In their offline lab experiments, the above-mentioned methodology achieved an error rate of 0 percent, and nearly 19 percent error rate when testing real traffic results.

Chaabane et al. [3] employed Open DPI (Deep Packet Inspection) in their work for analyzing the traffic flow extracted through six exit nodes which were dedicated for this task. The achieved results revealed that more than 50% of the traffic was through "BitTorrent applications" traffic. Although Open DPI was incapable to deal with encrypted connections, nearly 30% of the whole traffic belonged to Peer to Peer connection after usage of encryption analysis in BitTorrent based connections.

In 2014, Ling et al. [13] have presented traffic data analysis generated by Tor connection by using the "Intrusion Detection System". Their work presented the results on an analysis that was done using "Suricata", and a "commercial IDS rule-set (ETPro)". Results revealed that 10% of the traffic were malicious as it triggers alerts. On the other hand, a 70%+ of those alerts were triggered by Peer to Peer traffic from within the former 10%.

He et al. [14] did another research work that proposed a method based on "Hidden Markov Models" (HMM) approach in order to classify encrypted Tor traffic in four different traffic categories: Peer to Peer, File Transfer, IM chat and browsing (an unknown category for anything else is). Big volumes and directions were extracted from Tor traffic and used as classification features. "HMM" based model was used to form ingress and egress models of the aforementioned applications. The achieved result reached overall accuracy rate of 92%.

Further, Serjantov and Sewell in [15] have discussed the anonymity in the connection-oriented system through delineation different attack scenarios against anonymous web browsing. Basically, they formed a threat model for a passive attacker in an attempt to identify the different browsing activities in user browsing activities and this is done on the clients' side by running a small additional latency (without adding dummy traffic to minimize bandwidth requirement). The number of simultaneous connections per second was measured to be initiated in order to have anonymous browsing. Data shows that a hundred users with network links ranged from 2-4 have provide 92% poor anonymity (almost disclosed identities). In contrast, another scenario with 20 users with 200 connections ends up with only 2.5% poor anonymous browsing, and this indicates a very high anonymous system. However, the researchers ignored active attacks related to connection-based anonymity systems, specifically those attacks related to tracking established connection of source and destinations.

AlSabah et al. [16] presented and evaluated a new ML based classification model "Diff-Tor" model which classifies real-time Tor flows. This model aims to enhance the performance of the Tor network traffic classification, and this model works by assigning different classes of services on traffic data instances generated by the Tor connection. According to their observation, different applications have diverse throughput and time requirements. Consequently, the selected features were circuit lifetime, data transferred volume, cell inter-arrival times and number of recent cells sent are chosen to identify the Tor traffic flows. Experimentation aims to classify Tor circuit that was generated in real-time Tor traffic, and results show extremely promising accuracy ratios.

Soleimani et al. [17] focused on identifying Tor pluggable transports by deploying different machine learning models. Tor pluggable transport basically represents a bridge from public networks into Tor ones which considered an effective technique to bypass the worldwide Tor controlled activities. In their experiment, they worked on three plugin techniques which are "Obfs3", "Obfs4" and "ScrambleSuit" techniques. Experimentation

was employing supervised learning; the process of identifying those plugins could be executed with only first ten to fifty packets inspection in real time. The research work uses statistical flow features including flow size, sent packet's mean size and size's standard deviation, all packets in both directions.

3. **The Proposed Approach.** The main goals of the proposed method are represented by the following points: the detection of malicious activities that are insensible and the detection of malicious activities without having to carry out a deep packet inspection. The Gradient Boosting Decision Tree (G3) is the primary elements of the proposed detection scheme displayed in Figure 2.

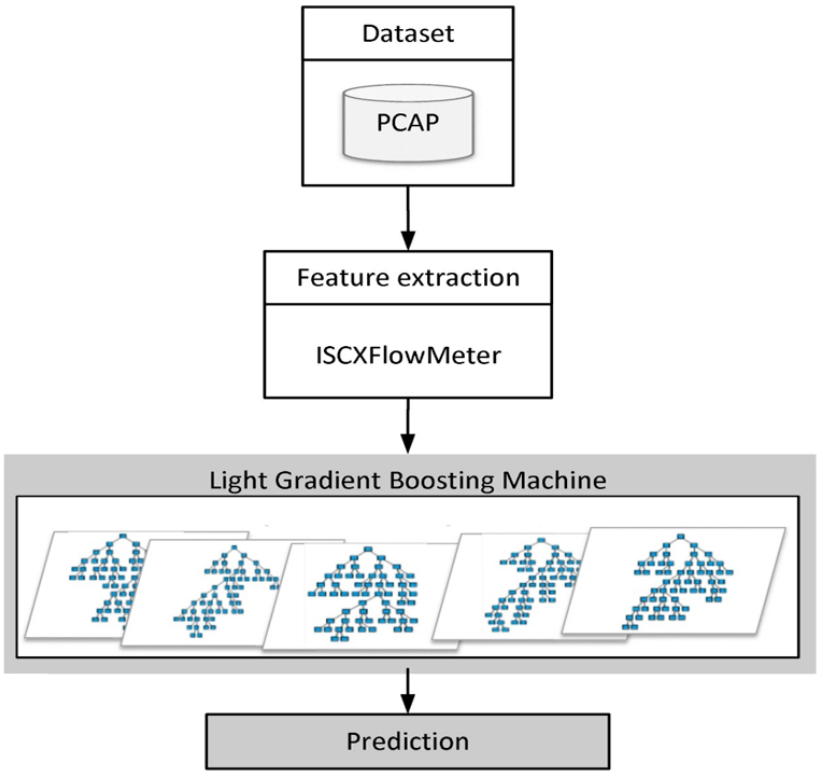

FIGURE 2. The proposed intrusion detection approach

BDT [18] showed much success in many applications, the GBDT which is an ensemble model based on decision trees as well. Within the iteration in the learning phase, GBDT learns the trees of decision that is done by fitting the negative gradients. Despite this, the development of data which is considered relatively big, the efficiency and accuracy ratios of GBDT is facing a number of difficulties and challenges. For instance, the computational complexities of GBDT are considered as proportionate to the number of instances and features. That leads to having several calculations that are time-consuming.

In an attempt to solve those challenges, the method of LightGBM was proposed [19]. This method is a framework for boosting the gradient, which is distributed and based on a tree of decisions in the implementation of the GBM.

In comparison with other GBMs, the LightGBM has made some optimization on its successor; it is based on a histogram-based tree of the decision and uses the subtraction of histogram for the purpose of acceleration. Particularly, it contributed to the optimization of sparse features through using the method that is based on the histogram. Leaf-wise leaf growth applies strategy with a depth constraint that tends to minimize the number of errors and shall increase the level of accuracy with ensuring a high level of performance. Further, it is capable of preventing the over-fitting at the same time: the rate of the cache hit was optimized, and the multi-threaded was optimized. LightGBM has added the rules

of decision to the features of category. That is done to avoid additional computational and memory overhead. It is done through the conversion of features into a one-hot multi-dimensional feature.

LightGBM is a new gradient boosting decision tree algorithm, introduced in 2017 by Ke and colleagues, and it is used in many fields of data mining domain such as classification, regression and ordering [20]. Two new techniques are included in the LightGBM algorithm, which are one-sided gradient analysis and the exclusive features bundling. Given the supervised training set $X = \{(x_i, y_i)\}_i^n = 1$, the target of LightGBM is to find an approximation $f(x)$ to a specific function $\hat{f}(x)$ which reduces the expected loss function value $L(y, f(x))$ as follows:

$$\hat{f} = \arg\min Ey, xL(y, f(x)) \tag{1}$$

LightGBM integrates a number of T regression trees $\sum_{t=2}^{T} f_t(x)$ to approximate the final model, which is

$$f_T(X) \sum_{t=2}^{T} = f_t(x) \tag{2}$$

The regression tree is represented as $Wq(x)$, $q \in \{1, 2, \ldots, J\}$, where J denotes the leaves number, q represents the rule of the decision tree, and W is a vector of leaf nodes weights. Hence, LightGBM would be additively trained at step t as follows:

$$T_t = \sum_{i=1}^{n} L(y_i, f_{t-1}(x_i) + f_t(x_i)) \tag{3}$$

In LightGBM, Newton's method easily approximates the objective function. Here g_i and h_i indicate the first- and second-order gradient statistics of the loss function, and let I_j show the example set of leaf j.

$$T_t = \sum_{i=1}^{n} \left(\left(\sum_{i \in I_j} g_i \right) + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right) \tag{4}$$

For the tree structure $q(x)$, the optimum leaf weight score of each leaf node is w_j. Furthermore, the extreme value of T_t could be solved as follows:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} (h_i + \lambda) w_j^2} \tag{5}$$

LightGBM employs a one-sides-sampling (GOSS) approach to detect split value in data instances. At the same time, XGBoost utilizes pre-sorted algorithms & histogram-based algorithms to compute the best split point.

The "histogram-based" algorithm separates all data points in discrete cases for an element and uses them to identify the splitting point of the histogram. Although it is more efficient than the "pre sorted speed" algorithm, which numerates all possible split-points on the pre sorted feature value, in terms of speed, it remains behind GOSS. Figure 3 shows a comparison between the procedure work of XGBoost and LightGBM.

4. **Experiments and Analysis.** Scikit-learn, as an ML library for python, is used to run the experiments. The software environment is Jupyter Notebooks. All the experiments ran on a Dell OptiPlex 7020 with 8GB, Intel i5 3.5GHz processor. In our experiment, LightGBM algorithm has several parameters to tune the algorithm, such as type of boosting, max depth, learning rate, leaves number fraction of features, max depth and number of iterations. As a type of boosting, we selected the gradient boosting decision tree. Table 1 shows parameters values used tuning LightGBM.

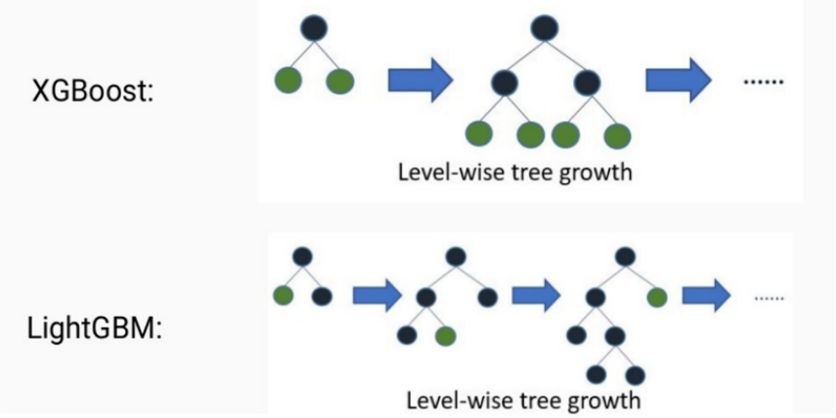

FIGURE 3. Leaf-wise tree growth in XGBoost and LightGBM

TABLE 1. LightGBM algorithm parameters

Parameter	Value
Boosting_type	gbdt
Objective	Binary
Learning_rate	0.1
Number_of_leaves	100
Feature_fraction	1.0
Maximum_depth	5
Num_of_boosting_iterations	100

4.1. **Dataset.** We use the data set [8] that has been collected in a real-world ToR network [11]. It comprises eight traffic forms (Web-browsing, IM, audio and video streaming, chat, voice mail, Peer to Peer communication, files transfer) as shown in Table 2. ISCXFlowMeter [21] was also used to produce the flows and to measure the necessary parameters. In forward and reverse directions, the statistical time-related features are measured separately.

TABLE 2. Dataset distribution

	TOR	NO-TOR	Total
15s	5631	48123	53754
30s	3130	43892	47022
60s	1723	41376	43099
120s	969	38285	39254

Each of the datasets consists of 28 time-based attributes and a label. Time-based features have a high speed of calculation when compared to some other features that can be extracted from flows. Time-based features require simple arithmetic to calculate them and can be derived from the first few packets of a flow making efficient for near real-time use. The features included

- The Inter-Arrival Times (IAT) in the backward and forward direction (with the stander statistical measures), the time is the time between two packets sent forward or backwards.
- Flow IAT which is the time between two packets in either direction (with the stander statistical measures).
- An idle time which is the time flow was idle before becoming active (with the stander statistical measures).

- Active time, the time flow was active before becoming idle (with the stander statistical measures).
- Flow bytes/s, flow packets/s and flow duration.

4.2. **Performance evaluation.** N fold cross-validations are used to measure the error ratio of classifiers in order to test the reliable detection rates. The dataset is randomly divided into N samples with N-fold cross-validation, with tests performed for N iterations. The $N-1$ samples are selected for each iteration for training and the last sample is used to determine the accuracy of the classifier. The experiments have been selected for $N = 10$. A classifier can identify a network connection into one of four categories [22-26]: 1) True Positive (TP), indicates the number of connections that is correctly classified as Tor activities; 2) False Positive (FP), indicates the number of connections that is incorrectly classified as Tor activities; 3) True Negative (TN), indicates the number of connections that is correctly classified as legitimate activities; 4) False Negative (FN), indicates the number of connections that is incorrectly classified as legitimate activities.

1) The FPR indicates the rate of legitimate connections incorrectly classified as Tor connections:
$$\text{FPR} = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{6}$$
2) DR, also named recall, shows the rate of Bot connections that is successfully identified as Tor.
$$\text{DR(TPR)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$
3) ACC shows the accurate predictions rate for all cases (legitimate and Tor).
$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{8}$$
4) The F-score is a measure of a test's accuracy.
$$\text{F-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}$$
5) RMSE shows the differentiation between target label and actual values estimated by identification system
$$\text{RMSE} = \sqrt{\sum_{i=1}^{N} \frac{(y_i - t_i)^2}{N}} \tag{10}$$
where N is the sample size, y_i indicate outputs of the model and t_i indicate targets of samples. Root Mean Square Error (RMSE) is one of the primary measures that show variation between y_i (model outputs) and t_i (model targets), so, when RMSE = 0 it denotes that the model prediction precisely matches the targets [27].
6) The operating characteristic (ROC) of the receiver is a graph that defines the output of the classifier. ROC curves figure the TPR to the horizontal axis on the vertical axis versus the FPR. A region below the ROC curve (AUC) is approximately 1.0 for a sound classifier. The AUC refers to the output of the classifier [28]. Besides, it is known that the AUC is a much more robust classification performance estimator [29].

4.3. **Results and analysis.** The results show that the method proposed offers the highest accuracy and detection rate of about 98.6 and 98.7 percent with 120s, as shown in Table 3 respectively. The findings also show that for the proposed LightGBM with a 120 second window, the highest F-measuring rate was 98 percent, while for 15 second windows the lowest F-measuring rate was 96 percent. Besides, the method proposed gave approximately 0.75% for the lowest false positive rate.

Figure 4 gives a comparison between the size of the time window in terms of accuracy, true positive and true negative rates. Higher performance of accuracy, true positive and

TABLE 3. Proposed approach results

	15s	30s	60s	120s
Accuracy	0.96592	0.97665	0.97910	**0.98684**
True Negative Rate (TNR)	0.97042	0.97347	0.97987	0.98617
True Positive Rate (TPR)	0.96042	0.97247	0.97887	**0.98717**
False Negative Rate (FNR)	0.04653	0.04553	0.03313	0.01383
False Positive Rate (FPR)	0.01753	0.01553	0.01313	0.00383
Mean Square Error (MSE)	0.2131	0.1332	0.0192	0.0132
F1-scores	0.96092	0.97118	0.98296	0.98069
Area under the ROC (AUC)	0.962942	0.97447	0.97687	0.98817

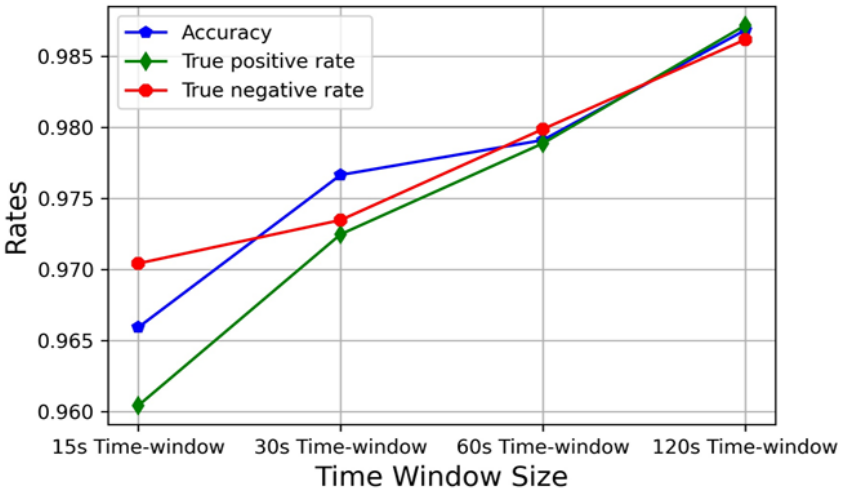

FIGURE 4. Accuracy, true positive and true negative rates

true negative rates indicated that the LightGBM with 120-seconds time window is better compared to the other time windows.

As shown in Figure 5, the proposed model gives the best false negative and false positive rates of around 1.38% and 0.383% respectively based on the 120-seconds time window, while, 15-seconds time window was used to achieve the lowest efficiency of the proposed model. Also, the proposed model results' quality measured based on the time window size was compared using the RMSE measure and the 120-seconds time window achieved the best RMSE of 0.0132 as shown in Figure 6.

In order to test the efficiency of our proposed approach for the detection of Tor network traffic, the Receiver Operating Characteristic (ROC) curve is plotted to show the trade-off between TPR and FPR. A perfect classifier would have a curve area (AUC) close to 1.0. The x-axis is an FPR, and the y-axis is a TPR. As shown in Figure 7, the Area under the Curve (AUC) for the detection of Tor network traffic is 0.988. The proposed approach has been found to perform well in classifying Tor traffic in a 120s time window size.

Our experiments are conducted using classic machine learning algorithms for traffic identification in the Tor network such as random forest [30], decision tree classifier [31], K-nearest neighbors [32]. As shown in Table 4 our proposed approach based on LightGBM performs better than classical machine learning algorithm. Compared to classical machine learning models, LightGBM is rising its speed by ten times; meanwhile, classification results would be improved. LightGBM does not depend on the form of data entered into the classification system. The entered data can be converted to a numerical form easily. After that, it can classify the data in proper speed levels and sufficient accuracy.

FIGURE 5. False negative vs false positive

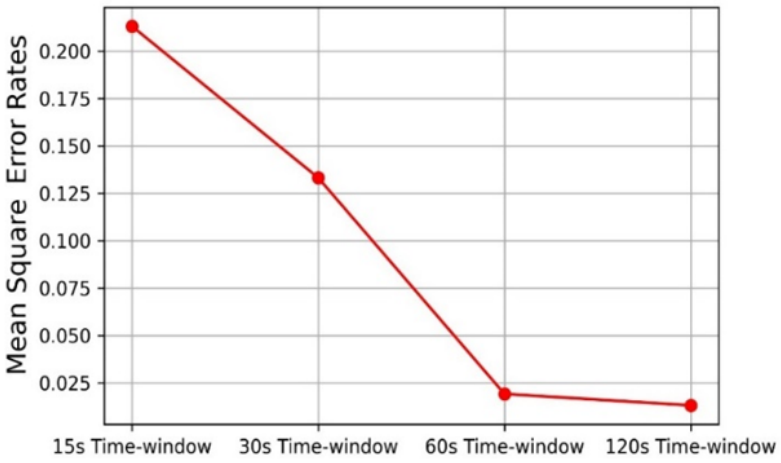

FIGURE 6. Mean square error

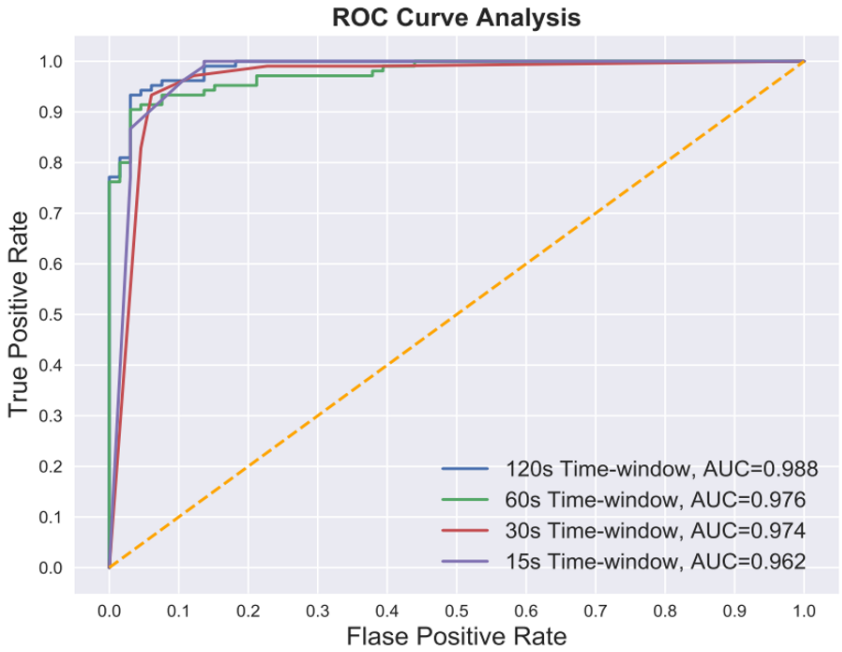

FIGURE 7. ROC curve

TABLE 4. Comparison with a classical machine learning algorithm

	120s		**60s**		**30s**		**15s**	
	Accuracy	F1-scores	Accuracy	F1-scores	Accuracy	F1-scores	Accuracy	F1-scores
Random forest classifier	0.871	0.932	0.851	0.912	0.841	0.914	0.82	0.904
Decision tree classifier	0.882	0.918	0.872	0.906	0.849	0.896	0.838	0.889
K-nearest neighbors	0.815	0.887	0.795	0.868	0.806	0.887	0.786	0.875
Proposed approach	**0.98684**	**0.98069**	**0.97910**	**0.98296**	**0.97665**	**0.97118**	**0.96592**	**0.96092**

5. **Conclusion.** This study suggests an approach that determines whether or not a person uses the Tor network using a lightGBM algorithm. Leistung and fast learning were developed for the LightGBM algorithm. This was based on decision-making algorithms driven by gradients. In addition, LightGBM can be used in various computing environments. The model proposed uses the LightGBM algorithm to achieve a high accuracy rate at 98.6%. Also, we make a comparison with the Tor traffic identification of many machine learning algorithms with the LightGBM classifier. In future, the performance of the algorithm proposed will be evaluated in the UNB-CIC Tor Network Traffic dataset, which includes eight different types of traffic.

REFERENCES

[1] D. McCoy, K. Bauer, D. Grunwald, T. Kohno and D. Sicker, *Shining Light in Dark Places: Understanding the Tor Network*, Springer, Berlin, Heidelberg, 2008.

[2] R. Snader and N. Borisov, A Tune-up for Tor: Improving security and performance in the Tor network, *NDSS*, vol.8, 2008.

[3] A. Chaabane, P. Manils and M. A. Kaafar, Digging into anonymous traffic: A deep analysis of the Tor anonymizing network, *The 4th International Conference on Network and System Security*, pp.167-174, 2010.

[4] P. Syverson, R. Dingledine and N. Mathewson, Tor: The second-generation Onion Router, *USENIX Security*, pp.303-320, 2004.

[5] J. Barker, P. Hannay and P. Szewczyk, Using traffic analysis to identify the second generation Onion Router, *IFIP the 9th International Conference on Embedded and Ubiquitous Computing*, pp.72-78, 2011.

[6] J. Zhai, H. Shi, M. Wang, Z. Sun and J. Xing, An encrypted traffic identification scheme based on the multilevel structure and variational automatic encoder, *Security and Communication Networks*, 2020.

[7] L. Wang, H. Mei and V. S. Sheng, Multilevel identification and classification analysis of Tor on mobile and PC platforms, *IEEE Trans. Industrial Informatics*, vol.17, no.2, pp.1079-1088, 2021.

[8] M. Alharbi and M. A. Albahar, Time and frequency components analysis of network traffic data using continuous wavelet transform to detect anomalies, *International Journal of Innovative Computing, Information and Control*, vol.15, no.4, pp.1323-1336, 2019.

[9] P. Syverson, G. Tsudik, M. Reed and C. Landwehr, Towards an analysis of onion routing security, *Designing Privacy Enhancing Technologies*, pp.96-114, 2001.

[10] T. B. Phobos, Plaintext over Tor is still plaintext, *Tor Blog*, https://blog.torproject.org/plaintext-over-tor-stillplaintext, 2010.

[11] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun and A. A. Ghorbani, Characterization of Tor traffic using time based features, *The International Conference on Information Systems Security and Privacy (ICISSP)*, pp.253-262, 2017.

[12] S. Chakravarty, M. V. Barbera, G. Portokalidis, M. Polychronakis and A. D. Keromytis, On the effectiveness of traffic analysis against anonymity networks using flow records, *International Conference on Passive and Active Network Measurement*, pp.247-257, 2014.

[13] Z. Ling, J. Luo, K. Wu, W. Yu and X. Fu, TorWard: Discovery of malicious traffic over Tor, *IEEE INFOCOM 2014 – IEEE Conference on Computer Communications*, pp.1402-1410, 2014.
[14] G. He, M. Yang, J. Luo and X. Gu, Inferring application type information from Tor encrypted traffic, *The 2nd International Conference on Advanced Cloud and Big Data*, pp.220-227, 2014.
[15] A. Serjantov and P. Sewell, Passive-attack analysis for connection-based anonymity systems, *Int. J. Inf. Secur.*, pp.172-180, DOI: 10.1007/s10207-004-0059-3, 2005.
[16] M. AlSabah, K. Bauer and I. Goldberg, Enhancing Tor's performance using real-time traffic classification, *Proc. of the 2012 ACM Conference on Computer and Communications Security*, pp.73-84, 2012.
[17] M. H. M. Soleimani, M. Mansoorizadeh and M. Nassiri, Real-time identification of three Tor pluggable transports using machine learning techniques, *The Journal of Supercomputing*, vol.74, no.10, pp.4910-4927, 2018.
[18] J. H. Friedman, Greedy function approximation: A gradient boosting machine, *The Annals of Statistics*, vol.29, no.5, pp.1189-1232, 2001.
[19] G. Ke et al., LightGBM: A highly efficient gradient boosting decision tree, *Proc. of the 31st International Conference on Neural Information Processing Systems*, Long Beach, CA, USA, 2017.
[20] G. Ke et al., LightGBM: A highly efficient gradient boosting decision tree, *Advances in Neural Information Processing Systems*, pp.3146-3154, 2017.
[21] A. H. Lashkari, G. D. Gil, M. Mamun and A. A. Ghorbani, Characterization of encrypted and VPN traffic using time-related features, *The International Conference on Information Systems Security and Privacy (ICISSP)*, Italy, 2016.
[22] M. Almseidin, M. Alzubi, S. Kovacs and M. Alkasassbeh, Evaluation of machine learning algorithms for intrusion detection system, *IEEE the 15th Symposium on Intelligent Systems and Informatics (SISY)*, pp.277-282, 2017.
[23] M. Alkasassbeh, A novel hybrid method for network anomaly detection based on traffic prediction and change point detection, *arXiv Preprint*, arXiv: 1801.05309, 2018.
[24] M. Alkasassbeh and M. Almseidin, Machine learning methods for network intrusion detection, *arXiv Preprint*, arXiv: 1809.02610, 2018.
[25] M. Alauthman, N. Aslam, M. Al-Kasassbeh, S. Khan, A. Al-Qerem and K.-K. R. Choo, An efficient reinforcement learning-based Botnet detection approach, *Journal of Network and Computer Applications*, vol.150, 2020.
[26] M. Alauthaman, N. Aslam, L. Zhang, R. Alasem and M. A. Hossain, A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks, *Neural Computing and Applications*, vol.29, no.11, pp.991-1004, 2018.
[27] A. Almomani, B. B. Gupta, S. Atawneh, A. Meulenberg and E. Almomani, A survey of phishing email filtering techniques, *IEEE Communications Surveys & Tutorials*, vol.15, no.4, pp.2070-2090, 2013.
[28] J. A. Swets, *Signal Detection Theory and ROC Analysis in Psychology and Diagnostics*, Psychology Press, 2014.
[29] T. Fawcett, An introduction to ROC analysis, *Pattern Recognition Letters*, vol.27, no.8, pp.861-874, 2006.
[30] L. Breiman, Random forests, *Machine Learning*, vol.45, no.1, pp.5-32, 2001.
[31] S. R. Safavian and D. Landgrebe, A survey of decision tree classifier methodology, *IEEE Trans. Systems, Man, and Cybernetics*, vol.21, no.3, pp.660-674, 1991.
[32] M. Schmidt, N. Le Roux and F. Bach, Minimizing finite sums with the stochastic average gradient, *Mathematical Programming*, vol.162, no.1, pp.83-112, 2017.