IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 1, 2021, accepted December 11, 2021, date of publication December 20, 2021,
date of current version January 10, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3137078

Blockchain-Enabled Deep Recurrent Neural Network Model for Clickbait Detection

ABDUL RAZAQUE[1], (Senior Member, IEEE), BANDAR ALOTAIBI[2,3], (Member, IEEE), MUNIF ALOTAIBI[4], (Member, IEEE), FATHI AMSAAD[5], (Member, IEEE), ANSAGAN MANASOV[6], (Member, IEEE), SALIM HARIRI[7], (Senior Member, IEEE), BANU B. YERGALIYEVA[8], (Member, IEEE), AND AZIZ ALOTAIBI[9], (Member, IEEE)

[1]Department of Computer Engineering and Information Security, IITU, Almaty 050000, Kazakhstan
[2]Sensor Networks and Cellular Systems Research Center, University of Tabuk, Tabuk 47512, Saudi Arabia
[3]Department of Information Technology, University of Tabuk, Tabuk 47512, Saudi Arabia
[4]Department of Computer Science, Shaqra University, Shaqra 11961, Saudi Arabia
[5]School of Information Security and Applied Computing, Eastern Michigan University (EMU), Ypsilanti, MI 48197, USA
[6]Department of Information Security, Kazakh National University, Almaty 050040, Kazakhstan
[7]Department of Electrical Engineering, University of Arizona, Tucson, AZ 85721, USA
[8]Department of Information Security, Gumilyov Eurasian National University, Nur-Sultan, Astana 010000, Kazakhstan
[9]Department of Computer Science, College of Computers and Information Technology, Taif University, Taif 21944, Saudi Arabia

Corresponding author: Abdul Razaque (a.razaque@iitu.edu.kz)

Taif University Researchers Supporting Project number (TURSP-2020/302), Taif University, Taif, Saudi Arabia; and partially supported by the Sensor Networks and Cellular System (SNCS) Research Center under grant 1442-002.

ABSTRACT When people use social networks, they often fall prey to a clickbait scam. The scammer attempts to create a striking headline that attracts the majority of users and attaches a link. The user follows the link and can be redirected to a fraudulent resource where the user easily loses personal data. To solve this problem, a Blockchain-enabled deep recurrent neural network (BDRNN) is proposed to detect the nature safe and malicious clickbait from the contents. The proposed BDRNN consists of three phases: analysis of clickbait and source rating, clickbait search process and multi-layered clickbait detection. The analysis of clickbait and source rating phase helps to analyze different sources to detect the clickbait and also rating the content-sources. To achieve the clickbait analysis and source rating, the detection of blocklisted/allowlisted source and source rating check algorithms are introduced. The clickbait search process is accomplished by incorporating the binary search features for a faster and more efficient search process for malicious content-detection. The multi-layered clickbait detection is main phase of the proposed BDRNN that consists of three models: content-to-vector model (layer-1), deep neural network model(layer-2), and Blockchain-enabled malicious content detection model (layer-3). These models collectively detect the malicious and safe clickbait from the contents. The extensive experiments are conducted to determine the effectiveness of the proposed BDRNN model and compared with the existing state-of-the-art neural network models designed for clickbait detection, and the result demonstrates that the proposed BDRNN model outperforms the counterparts from the, accuracy, link detection, memory usage, analogous perspectives, and attacker's successful content capturing rate.

INDEX TERMS Clickbait, fraudulent resources, scam, security.

I. INTRODUCTION

Recently, people have been highly engaged in conducting different activities on social networks [1]. According to the latest research, almost half the world's population uses social networks [2]. Advertising headlines were found in newspapers and banners before the existence of the Internet [3]. With the emergence of different utilities on the Internet, clickbait has gained popularity [4]. Clickbait is a good source, particularly for e-commerce [5]. If people often use clickbait, then a company realizes that their product is popular and in great demand. Moreover, fewer uses of

The associate editor coordinating the review of this manuscript and approving it for publication was Chun-Wei Tsai.

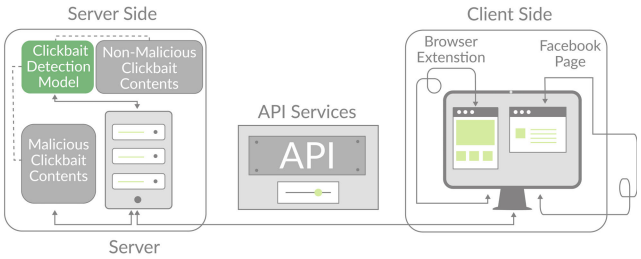


FIGURE 1. Simple model of clickbait detection.

clickbait prove that people are indifferent regarding the product. As a result, the company may lose profits [6]. However, in recent years, clickbait has begun to be frequently used by cybercriminals to lure people to harmful Internet resources to steal their personal information [7]. This is a very large problem since in addition to the various viruses contained in the links [8], they can also contain executable code that can harm the user [9]. The user can become the target of phishing in the future [10]. A malicious link can also redirect the user to prohibited web resources [11]. Some sites remotely use the power of the computer for mining [12]. The average user cannot distinguish whether a link is harmful because people do not see a threat in clickbait; rather, they see ordinary advertisements [13]. Therefore, the clickbait problem is severe and should be addressed. Figure 1 depicts simple model of Clickbait detection.

We propose a BDRNN that helps users recognize malicious links in clickbait to solve this issue. There are already several clickbait detection models, either using word-to-vector or neural network models [14]. However, in most cases, existing models block almost all ads coming from different sources [15]. Therefore, clickbait is blocked, which might be helpful to users. Artificial intelligence (AI) can be used to fight against clickbait. The AI systems are trained to make signals for clickbait. AI can also recognize new types of malware and shield confidential data for organizations. Antivirus software with the use of AI can also detect the trend of malicious activities and clickbait. On the other hand, the attacker can also use AI to crack the defenses by developing mutating malware [16]. With our proposed BDRNN, it is possible to analyze and distinguish malicious and non-malicious clickbait, providing users with all possible information and offering recommendations for further improvement and action.

A. RESEARCH CONTRIBUTIONS

Motivated by this problem, the contributions of this paper are summarized as follows:

- The features to protect and validate the authenticity of the sources of the advertisements, news, and other online multimedia content are leveraged by the blockchain technology layer to thwart the possibility of clickbait.
- The content allocation layer is proposed based on the content-to-vector representation in which connotation similarities, which help identify the different characteristics of contents, are calculated.
- The deep recurrent neural network layer serves as an auto-encoder that gets the input contents or links from the input gated recurrent unit (GRU) to detect the probability of the malicious contents from the content-to-vector layer.
- Extensive experiments are conducted to determine the effectiveness of the proposed BDRNN model and compared with the existing state-of-the-art neural network models designed for clickbait detection. It is demonstrated by the results that the proposed BDRNN model outperforms its counterparts.

B. PAPER ORGANIZATION

The remaining article is organized as follows: Section II discusses the problem identification and significance. Section III presents a literature survey of related work. Section IV explains the system model for multi-layered clickbait detection. Section V introduces the formal security analysis. Section VI extensively explains the proposed Blockchain-enabled deep recurrent neural network model. Section VII shows implementations and results. Section VIII discusses the advantages and possible shortcomings of the proposed BDRNN. Finally, the entire article is concluded in section IX.

Section II discusses the problem identification and significance. Section III presents a literature survey of related work. Section IV explains the system model of the browser extension. Section VI describes the proposed URL analysis process. Section VII shows implementations and results. Section VIII discusses the results, including advantages and possible shortcomings. Finally, the paper concludes in section IX.

II. PROBLEM IDENTIFICATION AND SIGNIFICANCE

Clickbait is a major threat to the confidentiality of users. The clickbait lures the users; as a result, the users become the victim of severe threats. With the emergence of the Internet, cybercriminals use its resources to severely affect users. Corporations can suffer significant losses if an attacker infiltrates performs illegal actions. The problem is relevant at the current time. Approximately 50 percent of cyberattacks since 2016 have been launched in Internet browsers. Additionally, more than 77 percent of attacks were carried out using malicious links. This occurs because most users do not have enough knowledge to analyze and understand Internet resources. Additionally, every day, there are new malicious Internet resources and methods. Based on the above, the following suggestions can be made:

- Create more services to educate ordinary users about the dangers of the Internet.
- Publish lists of the most popular fraudulent Internet resources.
- Create tools that can protect users from malicious links in ads.

The most useful solution creates a tool that protects the user from destructive actions. Even with the necessary information and experience, a person can still make a mistake and switch to a malicious resource.

III. PREVIOUS RELATED CONTRIBUTION

In this section, the salient features of related work are explained. Filtering social media posts can help reduce the number of targeted ads. A solution is proposed to browse and analyze the domains. Analyzing process occurs by setting up crawler to automatically determine the malicious domains [17]. The method determines the concealing behaviors of the websites and is also effective to protect personal data. It reduces the number of malicious attempts. However, the method is unable to support all types of blocklisted websites. An automated allow-list method for phishing attack detection is introduced [18]. A detailed analysis is conducted between actual and visual links to determine the allowlists. The similarity ratio of the trusted websites has been calculated to determine the domain for the contents of the allowlist websites and later matched the allowlisted websites with the IP address before making a decision. The proposed method is only limited to allowlisted websites and blocklisted websites have not been considered.

A basic set of rules will help you avoid falling for popular Internet tricks. This solution proposes taking simple steps to avoid switching to a malicious resource [19]. An example of this is to check a contents for a suspicious title and disable the execution of scripts on unverified sources. This method helps to give the user basic rules of conduct on Internet resources, but these rules cannot protect the user from most Internet fraud.

The use of artificial intelligence can protect against cyberattacks. The systemic risks are identified in the domains. The emerging risks are identified and discussed to highlight the limitations of the recent governance mechanisms that address the artificial intelligence sustainability risks. Artificial intelligence can help solve a problem, but not everyone can afford to use it because of its financial and technical costs.

Currently, there is no universal protection against phishing, but compliance with some rules allows users to protect their personal data from theft. The vulnerabilities have been identified that lead to phishing attack [20]. Furthermore, the negative effect of personal values and the values related to the alleged sender on the authenticity of phishing messages are examined. This method attempts to reduce the insight of phishing messages. However, Trust propensity cannot envisage authenticity of messages.

Convolution neural network Long Short-Term Memory (CNN-LSTM) is introduced for accuracy improvement [21]. Based on the results, it has been observed that CNN-LSTM detected more Clickbait in the social media. Furthermore, it supports to detect the several classification of clickbait headlines for social media platforms. However, the proposed method does not provide higher accuracy. Lure and Similarity for Adaptive Clickbait Detection (LSACD) method is proposed for ClickBait detection [22]. The proposed LSACD model combines the similarity and lure features. This model helps in utilizing the adaptive prediction. To confirm the validity of the proposed LSACD, a novel Chinese clickbait dataset is created that involves nearly 5000 media news. Based on the results, effectiveness and reliability have been proved. However, the proposed method is limited with media and news and also takes longer time to detect the legitimacy/illegitimacy of Clickbait. The convolutional neural network for Clickbait (CNNC) is introduced [23]. The proposed method CNNC identifies not only entire features, but specifies the particular features from the articles. The experiment is conducted to determine the recall, precision, and accuracy. The experimental results demonstrate higher accuracy. The proposed CNNC method just focused on articles. The recurrent neural network-based bidirectional long short-term memory model is introduced to determine which word contributes to the advertisement's clickbait score [24]. Furthermore, a Siamese net is employed to determine the similarity information between source and target. Image embedding has been learned from the huge amount of data using convolutional neural networks (CNN). As CNN provides another layer of complexity to the proposed model. The experiments were conducted using 19538 social media posts for determining accuracy. However, the proposed model is restricted only to headlines classification and attributes' similarity scoring and failed to detect malicious clickbait. Development of a browser-based extension to block the collection of personal data is another method to protect users. A browser extension for Google Chrome was developed that blocks the taking of personal data from websites [25]. An installed extension finds and blocks executable scripts on sites that interact with the user's data. A standard browser extension blocks all ads on the site. This solution is poor because the extension, in this case, does not allow the user to decide for himself whether to block ads. As a result, useful ads can be blocked. All of the existing methods focus on accuracy improvement, but our proposed BDRNN not only improves the accuracy, but provides the faster link detection, better analogous and less memory usage.

IV. SYSTEM MODEL

The clickbait detection system model consists of three phases. The first phase is to analyze the clickbait and provide a rating to the source. In this phase, all the contents coming from different sources are analyzed to detect clickbait. Furthermore, the rating of the different sources was done to determine the lower-to-higher rated sources. The contents coming from lower-rated sources are not allowed. The second phase is to evaluate the clickbait search process. In this phase, binary search features are used to process the clickbait efficiently. The third phase is a multi-layered clickbait detection process to detect the malicious contents that lead to clickbait. This phase is of paramount prominence and consists of three models: content-to-vector model (layer 1), deep neural network model (layer 2), and blockchain-enabled

TABLE 1. Summary of the contributions of existing surveys/reviews.

Works	Clickbait detection protocol	Features / Characteristics	Vulnerabilities
[17]	Monitoring the behavior of malicious websites	The crawler is built to automatically analyze and browse the contents from the malicious domains	Unable to support all types of blocklisted websites
[18]	Allowlist website detection method	Designing and implementing the automated allowlist method for allowlisted domains that helps to block harmful websites	limited to allowlisted websites and blocklisted websites have not been considered
[19]	Employing simple steps to handle malicious resources	Usage of the basic rules to avoid Internet tricks by checking the domain contents for a suspicious title. Additionally, it disables the execution of scripts on unverified websites sources.	The rules cannot protect the user from most Internet frauds.
[26]	Mechanism to identify the risk of the domains	Identifying emerging risks of the domains. Additionally, the limitations of the recent governance mechanisms are addressed to handle the artificial intelligence sustainability risks	It is only restricted to governance mechanisms and unable to identify the malicious domains
[20]	Vulnerability detection mechanism to avoid phishing attack	Examining the negative effect of personal values and the values related to the alleged sender on the authenticity of phishing messages	Trust propensity cannot envisage authenticity of messages
[25]	Development of the browser-based extension for clickbait detection	Implementation of the standard browser extension that blocks executable scripts and clickbait on sites that interact with the user's personal data	There is possibility of blocking the useful clickbait. Additionally, the properties of the extension are poor
[21]	Convolution neural network Long Short-Term memory model for clickbait detection	Deployment of the convolution neural network to detect the clickbait accurately. Additionally, it supports to detect the several classification of clickbait headlines for social media platforms	Failed to provide higher accuracy and limited only to social media platforms
[22]	Neural Network-based ClickBait detection method	combines the similarity and lure features based on the neural network to detect the clickbait. The proposed model uses the novel Chinese clickbait dataset to show effectiveness and reliability	limited with media and news and also takes longer time to detect the legitimacy/illegitimacy of Clickbait
[23]	Convolutional neural network model for the clickbait detection	Employing convolutional neural network model that identifies not only entire features, but specifies the particular features from the articles to determine any potential clickbait. The focus of the model is to show higher accuracy.	Limited to detecting the clickbait only from the articles. The proposed model consumes more CUP and memory
[24]	Multi-strategy approach for the clickbait detection on the Internet	Combines the recurrent neural network-based bidirectional long short-term memory model and Siamese net to determine the clickbait. The proposed approach provides multi-strategy support.	It is restricted to headlines classification and attributes' similarity scoring and failed to detect malicious clickbait.
This work	Blockchain-enabled deep recurrent neural network model for clickbait detection	The proposed BDRNN consists of three phases: analysis of clickbait and source rating, clickbait search process and multilayered clickbait detection. The multilayered clickbait detection is main phase of the proposed BDRNN that consists of three models: content-to-vector model (layer-1), deep neural network model(layer-2), and Blockchain-enabled malicious content detection model (layer-3). The proposed model outperforms the counterparts from the, accuracy, link detection, CPU consumption, memory usage and analogous perspectives	Leverages the features from three layers: Blockchain, deep recurrent neural network and vector-to-content models. However, it inherits the negative features of Blockchain technology, but these features have marginal negative impact

malicious content detection model (layer 3). In the content-to-vector model, content information is processed, where stops words and punctuation are removed to make the sentence in the correct order and conduct a content lemmatization process. Clean inputs are transformed as the content vectors. In the deep neural network model, the GRU is used in the RNN. The GRU is identical to long short-term memory and is supported by the forget gate by using a gating tool to trail the nature of the contents of the sequences without using distinct memory cells. The auto-encoder is used to obtain input from the GRU to determine the possibility of malicious content. In the blockchain-enabled malicious content detection model, the detected malicious or safe content is forwarded to the smart contract for the content in the blockchain. The smart contract consists of the following components: content source, content status, valid identity, time stamp, and content broadcaster. These components help include the information in the block and broadcast it to the peer-to-peer (P2P) network. The reliable minor nodes in the P2P network are responsible for including valid content blocks in the blockchain and declining malicious content blocks. Once the valid content blocks become part of the blockchain, content authenticity and truthfulness are verified using semantic similarity. The multi-layered clickbait is depicted in Figure 2.

V. FORMAL SECURITY ANALYSIS

The proposed BDRNN framework consists of three phases to detect clickbait. The phase-1 is responsible to characterize and distinguish the Allow/Block-list links and rating the sources of coming links. The phase-2 analyzes and sorts out the contents using the binary search features. This phase decides the nature of the contents whether they are safe or malicious. The phase-3 involves a multi-layered clickbait detection process to detect malicious and safe contents. This phase consists of three models: content-to-vector, deep neural network), and Blockchain-enabled malicious content detection. In the content-to-vector, the content information is processed, where punctuations and stop words are detached to bring the sentence in the precise order and conduct a content lemmatization process on them. The deep neural network can detect most of the safe and malicious contents. Additionally, this module has the support of important feature semantic similarity content that forwards the malicious and safe contents successfully to Blockchain technology. The inclusion of Blockchain technology augments detection accuracy and provides a good level of protection, However, Blockchain technology consists of four major components: a node application, consensus algorithm, shared ledger, and virtual machine, but we do not deal with components in this article. Our focus is on the smart contract, Proof-of-truthfulness, peer-to-peer network, and reliable miners. Smart contracts are of paramount importance that confirms the possibility of external attacks. Therefore, it is important to consider the following properties before building robust smart contracts.

- Blockchain architecture modeling
- Security characteristics evaluation
- Security attack analysis
- Blockchain modeling behavior

A. BLOCKCHAIN ARCHITECTURE MODELING

The model applies Blockchain technology to determine the presence of malicious content in the links. The analysis is based on Blockchain-based software that syndicates model examination methods and ontology reasoning as depicted in Figure 3. First, the architecture design of the Blockchain is properly established as the deployment view and component with connector (C&C) view. Second, the ontology detector is employed to determine the security vulnerabilities of the model using security features and basic patterns. These basic patterns are demarcated as the classes of the ontology library. Third, contents are included in the behavior model that is obtained through identified security vulnerabilities. Finally, the model organizer processes the contents to determine threat scenarios.

B. SECURITY CHARACTERISTICS EVALUATION

Several security characteristics are measured in the architectural/model design. In this article, we focus only on the security characteristics of the Blockchain-enabled software architecture.

1) ATTACKING ENVIRONMENT

This characteristic determines the vulnerabilities in the model that the attacker can launch to attack the model. This characteristic is mostly applied to the model that is open for the outside environment where data are publically accessible to everyone. If the lower attacking environment is determined, then the model is considered more secure. In our case, public Blockchain is an attacking environment that permits an entity to join and run the node in the Blockchain technology network environment. The internet-of-things web application programming interface is developed on the local servers. In addition, a Blockchain integrator is deployed on the public cloud. Thus, these components pose security threats to the public Blockchain implemented with Ethereum and Private Blockchain instigated with Quorum. As a result, there is a possibility of a threat.

2) LEAST ENTITLEMENT

This characteristic guarantees that the users should be provided minimal access to confidential data or operations in the system. Therefore, the number of components should be restricted from an architectural perspective. The on-chain components are considered as the critical components in the Blockchain-based software that should be retrieved by only necessary off-chain components. In our case, the Blockchain integrator is the only component that has direct access to the on-chain component (Ledgers, content-blocks, and reliable miner nodes).

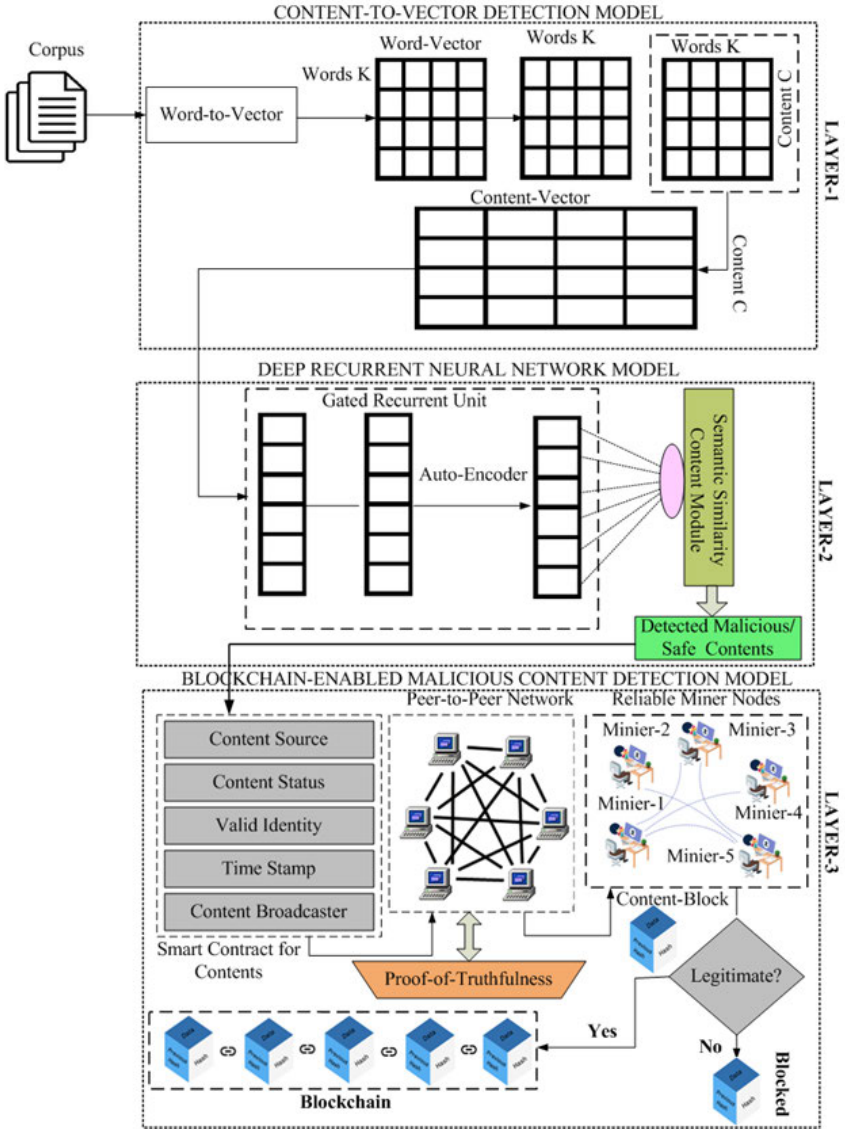


FIGURE 2. Multi-layered clickbait detection process.

3) DEFENSE IN DEPTH

This characteristic determines how the security countermeasures are used at different points of the system. The Blockchain-accessing components should apply security countermeasures at the user, peer-to-peer network, and components to protect the data in Blockchain. The ratio of off-chain components is measured that is used for accessing the on-chain components. If the calculated ratio is higher, then the system is considered more secure. Blockchain integrators should use authentication and authorization countermeasures.

C. SECURITY ATTACK ANALYSIS

As mentioned earlier that the on-chain components are highly vulnerable to being attacked because they process the request and are activated through the off-chain components. The known attack scenarios have been discussed that have a massive impact on Blockchain.

- Data leakage
- Data modification

1) DATA LEAKAGE

When designing the Blockchain-enabled software, then it is mandatory to decide whether data should be kept visible or put off-chain. Additionally, the copy of the data is shared among the miners, then there is the possibility that the minor can be compromised, and shared data can be exploited by the attacker. The data is encrypted on the Blockchain, but it can be leaked if the secret key is compromised. Safeguarding the components and connections that access the Blockchain is of paramount significance. The smart contract for contents or its connection to peer-to-peer can be compromised. But, in our case, the smart contract for contents only processes

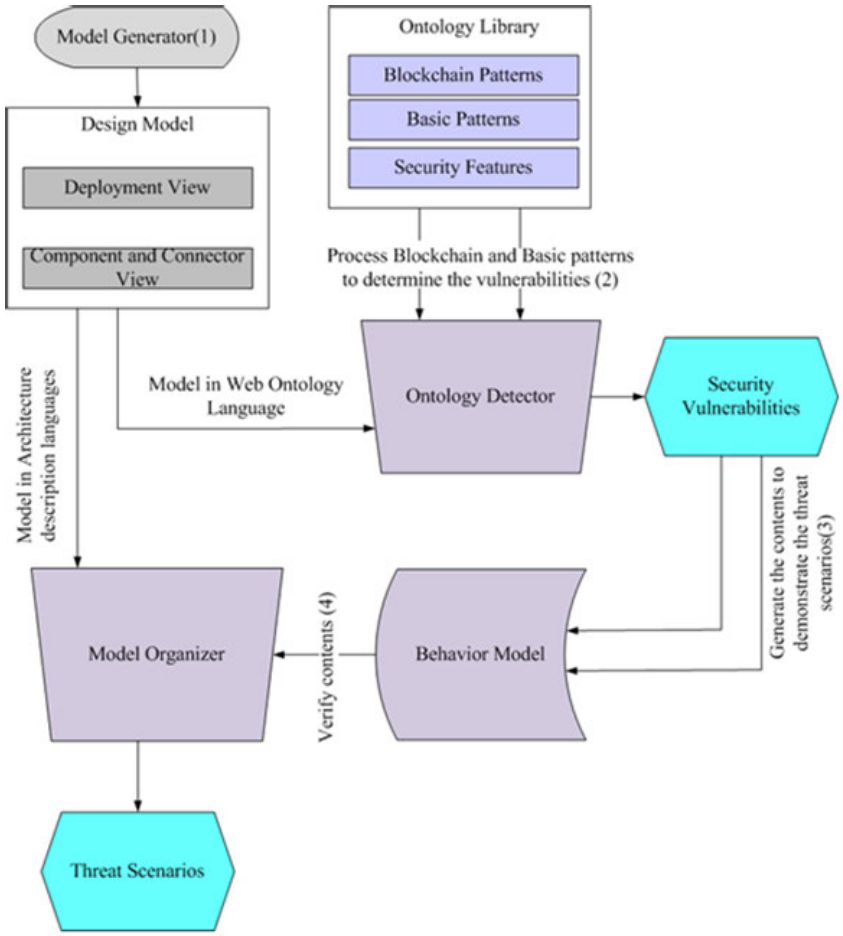


FIGURE 3. Vulnerabilities and threat detection model for Blockchain/Basic patterns.

the contents for dual authentication of the safe and malicious contents.

2) DATA MODIFICATION

Data in Blockchain is almost considered as unchallengeable and collision-resistant. However, when the oracle module inserts data into the Blockchain, then inserted data are presumed to be authenticated by all miners. If the oracle module is compromised, then the attackers can tamper data before it is inserted into the Blockchain. Therefore, there is a need of protecting the oracle by providing countermeasures. the Blockchain integrator functions as an oracle. If the components of Blockchain integrator inserts data to internet-of-things web application programming interface that could be compromised. Thus, there is the possibility that the data on public and private Blockchain can be altered. In our case, the malicious and safe contents are already decided at layer 2. If the oracle module inserts data into the Blockchain that does not affect the decision made on layer 2.

D. BLOCKCHAIN MODELING BEHAVIOR

To model the behavior of the Blockchain, the following components are required.

- Transaction component
- Mining component
- Block component
- Register contract component

1) TRANSACTION COMPONENT

It deals with external transactions and stores them in the list of pending transactions until they are randomly retrieved to initiate the mining process.

2) MINING COMPONENT

It models the entire mining process and starts mining the pending transactions. Additionally, the execution command is sent to the related smart contract. The types of consensus and the number of the miners are not considered for simplification purposes.

3) BLOCK COMPONENT

It encapsulates the execution results of the contracts into blocks. The execution results of the contracts are exported through the port Blocks.

4) REGISTER CONTRACT COMPONENT

It is responsible to execute the function calls of the contract register and forwarding the execution results to the corresponding Block component.

The attacker model is generated to evaluate the security of the smart contract register that is used in the Blockchain. The attacker aims to compromise the identity of the legitimate source of the contents by registering an alias using its own address. Therefore, there is the possibility to have three situations be used by the attacker for attacking purposes on the Blockchain.

- **Situation-1:** The attacker can obtain the name of the legitimate source of the contents when coming from the second-layer through mined blocks.
- **Situation-2:** The attacker can get the name of the legitimate source of the contents through pending contents that are still not mined.
- **Situation-3:** The attacker can obtain the name of the legitimate source of the contents from the peer-to-peer network that is directly connected with the Blockchain.

We have analyzed each situation to determine the success of the attacker that intends to alter the contents. In situation 1, the attacker cannot be successful to obtain the details of the legitimate source of the contents because the smart contract rejects the attempt of the attacker. In situation 2, the attacker can get a very small chance to hack a smart contract. In situation 3, there could be the possibility of modifying the contents if the contents remain longer time in the network. The smart contract execution with blockchain and hacker behaviors is depicted in Figure 4.

VI. PROPOSED BLOCKCHAIN-ENABLED DEEP RECURRENT NEURAL NETWORK MODEL

The proposed method is capable of detecting the clickbait in an efficient fashion. This solution is designed to ensure that the links that the user follows are safer to use. The process consists of three phases:

- Analysis of Clickbait and Source Rating
- Clickbait Search Process
- Multi-layered Clickbait Detection

A. ANALYSIS OF CLICKBAIT AND SOURCE RATING

Analyzing clickbait is the critical phase that analyzes the sources of clickbait. This process enables the user to know whether the content resource is safe or not. The clickbait analysis process is depicted in Figure 5.

The process focuses on determining the blocklist/allowlist sources to safeguard the user from malicious attempts. The detection process of blocklisted/allowlisted sources is shown in algorithm 1. The function of algorithm 1 is to classify the block-listed/allow-listed sources. When the sources are classified, then the second process is to rate identified sources shown in algorithm 2. Based on the rating, the maliciously identified resources are blocked not to be accessed in the future.

In algorithm 1, the determination process of block- or allow-list sources of advertisements, news, and multimedia content is explained. In the first step, the variables are initialized for the determining process. The input and output processes are defined at the beginning of the algorithm,

Algorithm 1 Detection of Blocklisted/Allowlisted Sources

Input: $\{S_U\}$ in
Output: $\{B_L, W_L, S_{vn}\}$ out
1: **Initialization:** S_U*: Source;* W_M*: Warning message;* O_M*: Okay message;* B_L*: Blocklisted;* W_L*: Allowlisted;*
2: **if** $S_U \in B_L$ **then**
3: **Show** W_M
4: **end if**
5: **if** $S_U \in W_L$ **then**
6: **Show** O_M
7: **end if**
8: **if** S_U ! $\in W_L$ and S_U ! $\in B_L$ **then**
9: **Show** S_{vn}
10: **end if**

respectively. Steps 2-4 check the source as to whether it is block-listed. If the source is block-listed, then the source-aware server sends the warning message. In steps 5-7, the source is checked to see whether it is allow-listed. If the source is classified as allow-listed, then the contents coming from that source are permitted. In steps 8-10, source-aware server notifies that source is new. The time complexity of the algorithm 1 is O(log n) in the best-case, and O(n log n) is in the worst-case for detecting the Blocklisted/Allowlisted sources.

Algorithm 2 Source Rating Process

Input: $\{S_U, D_L, D_R\}$ in
Output: $\{W_M or O_M\}$ out
1: **Initialization:** S_U*: Site URL;* W_M*: Warning message;* O_M*: Okay message;* D_R*: Domain rating*
2: **if** $D_R < 1$ **then**
3: **Show** W_M
4: **end if**
5: **if** $D_R > 1$ **then**
6: **Show** O_M
7: **end if**

In algorithm 2, the source rating process is explained. In step 1, the variables are initialized for the determining process. Steps 2–3 are the input and output processes, respectively.

In steps 4–6, the source rating is checked. If the rating is less than or equal to 1, then a warning message is shown. In steps 7-9, if the source rating is higher than 1, then the user receives a message informing that everything is fine. The time complexity of the second algorithm is O(log n) in the best case and O(n log n) in the worst case for the source rating check. Whenever the nature of the source is determined, time is required to check and announce it. Therefore, time for allow- or block-listed sources can be calculated as follows:

$$T = \sum_{i=1}^{S_{vt}} (T_W + T_B) \times T_s \tag{1}$$

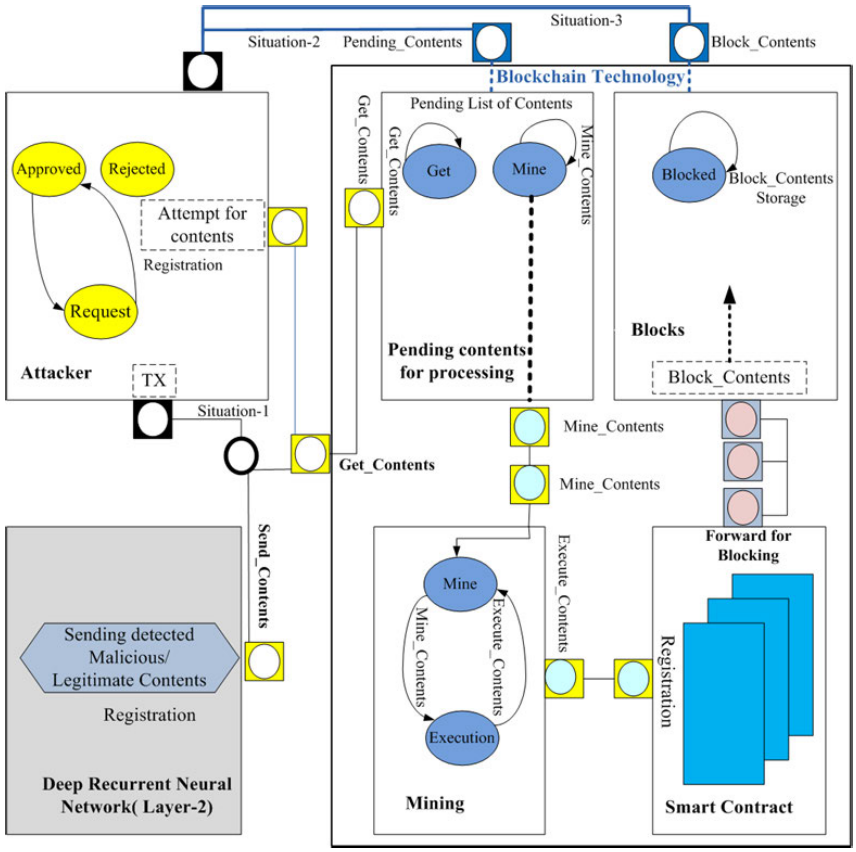


FIGURE 4. Smart contract execution with blockchain and attacker-behavior.

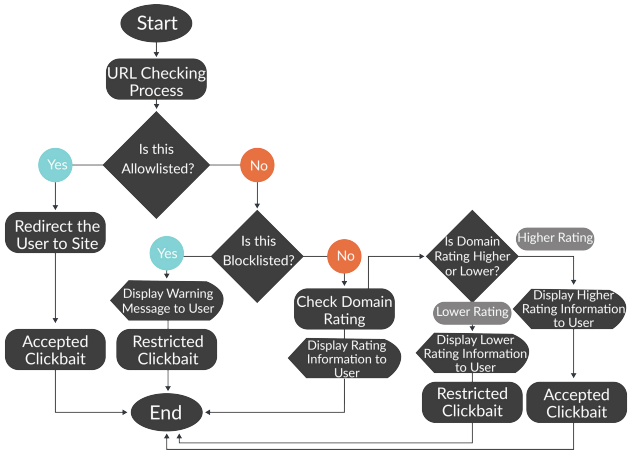


FIGURE 5. Process of analyzing clickbait.

where T is the total time taken by the source-aware server; T_w is the time taken for checking the allow-listed sources; S_{vt} is the total number of the sources to be checked; T_B is the time taken for detecting the blocklisted sources, and T_s is the time taken for forwarding the sources to the user. When the allow-listed and blocklisted sources are confirmed, then the process of rating the sources is significant that further checks the validity of the sources. Thus, the total time for the detection of the sources and rating is calculated as:

$$T = \sum_{i=1}^{S_{vt}} \{(T_W + T_B)_I \times T_s\} + T_r \tag{2}$$

If an additional analysis is required in the worst case, that is calculated as follows:

$$T = \sum_{i=1}^{S_{vt}} [\{(T_W + T_B)_i \times T_s\} + T_r] + T_a \tag{3}$$

where T_r is the time required for rating the sources, and T_a is the time taken for conducting an additional analysis in the worst case.

B. CLICKBAIT SEARCH PROCESS

In our extension and contents, we use a binary search features. Because we use sorted lists of URLs and contents, the binary search works on sorted arrays. In the worst case, the binary search takes many iterations of the comparison loop, as shown in the following equation:

$$A_m = \log_2^n + 1 \tag{4}$$

where A_m: worst case number of iterations; n: number of elements in array.

On average, assuming that each element is equally likely to be searched, a binary search calculates the number of iterations that can be demonstrated as

$$A_m = \left(\log_2 n + 1 - (x^{\log_2 n+1} + 1) - 2 \times \log_2 \frac{n}{n} \right) \quad (5)$$

Let us also analyze the best and worst cases of the binary search.

In the binary tree representation, a successful search can be represented by a path from the root to the target node. The initial iteration is given by

$$I_N = (l + 1) \quad (6)$$

where l: length of path; I_N: initial iteration.

Then, the average number of iterations for a successful search can be calculated by

$$T_N = 1 + \frac{I_N}{n} \quad (7)$$

where T_N: Average number of iterations for a successful search.

Since a binary search features provide the optimal process for searching with comparisons, this problem is reduced to calculating the minimum internal path length of all binary search with n sources, which can be demonstrated by

$$M_l = \sum_{k=1}^{n} \log_2 k \quad (8)$$

where M_l: minimum internal path length.

Theorem 1: In a 7-element array, the program will have a minimum internal path that equals 254 for comparing all elements.

Proof: Equation 8 is used to calculate the minimum internal path to obtain the solution.

$$\sum_{k=1}^{7} \log_2 k = 2 + 4 + 8 + 16 + 32 + 64 + 128 = 254$$

Unsuccessful searches can be represented by augmenting the tree with external sources. The average number of iterations for an unsuccessful search is represented by

$$T'_N = \frac{E_N}{n + 1} \quad (9)$$

where T'_N: average number of iterations for an unsuccessful search; E_N: path length.

Substituting the equation for E_N into the equation for T'_N, the worst case for unsuccessful searches can be determined by

$$T'_N = (\log_2 n + 2 - 2^{\log_2 n+1}) - \frac{(n + 1)}{n} \quad (10)$$

This algorithm has an average search speed, which makes it an acceptable solution for a content and URLs search.

After the source rating check and search algorithms have been executed, additional analysis of the content resource begins. The additional analysis consists of calculating various indicators. To determine whether a link is harmful in additional analysis, we check the depth of the legal address. We find the depth indicator, which is calculated by

$$h = \frac{1}{S_D} \quad (11)$$

where h: depth indicator; S_D: number of subdomains.

When checking for security, the link is divided into main and domain parts. The importance of the main part is 75% (0.75), and the importance of the domain part is 25% (0.25). After that, the percentage of matches is calculated by

$$C = 0.75 \times C_M + 0.25 \times \frac{C_D}{100} \quad (12)$$

where C_M: the coincidence of the main part; C_D: the coincidence of the domain part; C: percentage of matches.

Based on the coincidence of the source rating, where the importance of two parts equals 50% (0.5), the initial security check score is calculated as

$$B_I = 0.5 \times S_C + 0.5 \times C_D \quad (13)$$

where B_I: Base security indicator; S_C: availability of security connection (if site has security connection, then parameter is set to 1, else to 0).

Another parameter that is considered in the analysis is site visibility. This parameter can be calculated as

$$V = D \times \sqrt{B} \quad (14)$$

where V: the visibility parameter; B: the number of backlinks that lead to the site, and is the number of source where these back-links are located.

Theorem 2: Even for a secure source content, the visibility parameter will be low if the source content are new.

Proof: Suppose, we get the content from the new link edo.prgapp.kz. In addition, we have only 2 sources that contain a total of 14 links which are calculated by equation 14

$$V = 2 \times \sqrt{14} = 7.48$$

Additional analysis also includes checking the traffic coming from the source, which is calculated as:

$$A = \frac{I}{R_{lt}} \quad (15)$$

where A: attendance parameter; I: the number of incoming users; $R_l t$: resource lifetime.

Based on the source attendance, the source rating is calculated as:

$$S_R = \frac{A}{P} \quad (16)$$

where S_R: source rating; P: number of pages in Google search.

To store information, a dictionary collection is used, which performs well in adding, taking and deleting operations that are calculated by

$$O_B = \log_2 D_n \quad (17)$$

where O_B: best time complexity; D_n: number of elements in dictionary.

$$O_A = \log_2 D_n \tag{18}$$

where O_A: average time complexity.

$$O_w = \log_2 D_n + 1 \tag{19}$$

where O_W: worst time complexity.

Additionally, the certain contents associated with different sources are checked that are glued to the clickbait. The executable links are counted which are obtained from the URL and contents given by:

$$E_C = \sum_{k=1}^{n} J_k \tag{20}$$

where E_C total amount of executed code; J: found JavaScript.

In the next step, we calculate the maliciousness index of the executable JavaScript code, which can be calculated by

$$E_I = \sum_{k=1}^{n} J_k \times R \tag{21}$$

where E_I index of executed code; R: malicious link redirect parameter (if code has redirected to malicious resource, then $R = 1$. If code has suspicious logic, then $R = 0.5$, else $R = 0$).

The extension uses an external sandbox to execute JavaScript code and determine what the external program will produce. The time taken for this check is calculated by

$$T_S = A_E \times T_E \tag{22}$$

where T_S: time for sandbox checking; A_E: number of rows in the single code; T_E: time for checking single row.

Sandbox returns a sandbox indicator, which is calculated by

$$S_I = \frac{(W_M + M_M)}{C_A} \tag{23}$$

where S_I: the sandbox indicator; W_M: the number of warning messages; M_M: the number of malicious code messages; and C_A: the total amount of checked code.

Finally, we calculate the complete indicator of malicious code, where the importance of two parts equals 50% (0.5), which is demonstrated by

$$C_I = E_I \times 0.5 + W_I \times 0.5 \tag{24}$$

where E_I: index of executed code; C_I: complete indicator of executed code.

Hypothesis 1: If the source of the content is attached to a clickbait is popular, then the algorithm will be completed in 1 step and will not take much time.

Proof: For example, we will consider a google.com site. It is in the first 100 elements of the allowlist. Thus, we can calculate the time to receive a reply message using equation 1. $T = 0.9$ *second*.

We have 0.9 *seconds* because this time, we have to find this URL in the first 100 elements. In addition, as we can see, this time is very short.

Corollary 1: Based on the proof of the first hypothesis, we conclude that the most popular sources, such as Google, have a very short time for analysis of clickbait detection since they are at the first stage of testing.

Hypothesis 2: The visibility parameter in most cases will be very small for new sources.

Proof: We have website edo.base.kz. We have only 1 domain that contains a total of 2 backlinks to this resource. The visibility parameter for this resource is equal to $V = 1 \times \sqrt{2} = 1, 4$.

This web resource is safe, but because it is new, it has a low visibility parameter.

Corollary 2: Based on the proof of the second hypothesis, we observe that the visibility parameter is a weaker indicator for a new content resource when analyzing it. This is because this indicator is based on the number of domains on which links to this content resource are located. Since the new resource has few domains on which links to it are posted, the visibility indicator for new resources will always be small.

C. MULTI-LAYERED CLICKBAIT DETECTION MODEL

This model consists of three layers. Each layer supports for the clickbait detection process. Three layers are given as:

- Content-to-Vector Detection layer
- Deep Recurrent Neural Network for Malicious Content Detection layer
- Blockchain-Enabled Malicious Content Detection layer

1) CONTENT-TO-VECTOR DETECTION LAYER

The content-to-vector model gets the total number of K words $W = (w_1), (w_2), \ldots, (w_k)$ from the different sources that involve various links, headlines, and social media. The model is employed to obtain general clickbait features that can considerably increase detection quality by overwhelming the shortcomings of existing probabilistic models for clickbait. Therefore, the features obtained from the different sources S_o with the total number of words, can be shown through the content-to-vector CV representation given by:

$$CV = v(w_1) \oplus v(w_2) \oplus, \ldots, v(w_K) \tag{25}$$

The entire content-to-vector process consists of three processes, which are given below:

- Each word should be vectorized as K-dimensional and to be written as the word-vector $w_i \in R^k$.
- Each content-specific vector $\gamma_j, j = 1, \ldots, C$, denoting the features of the contents that are generated by applying the matrix multiplication between w_i and $K \times C$. The content-to-vector building process is given in equation 26, as shown at the bottom of the next page.

The similarity between w_i and γ_j should be calculated by equation 26. The final similarity between the words and content-specific vector is identified with the mean value of the cosine similarity Cos_S. If the mean value of the cosine

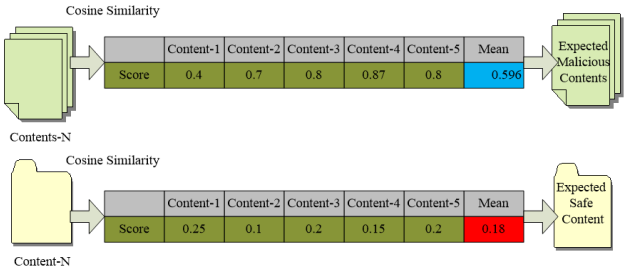


FIGURE 6. Expected malicious/safe content detection.

similarity contents is $Cos_S \geq 0.5$ then content is considered as expected malicious content. If the mean value of the $Cos_S < 0.5$ is calculated then, the expected contents are deemed to be expected safe content as depicted in Figure 6.

$$Cos(w_i, \gamma_j) = \frac{w_i \cdot \gamma_j}{||w_i|| \times ||\gamma_j||} \tag{27}$$

The content-to-vector is updated as follows:

$$v^{fs+1}(c) \leftarrow v^{fs}(c) + \frac{\rho}{2} \forall \delta r(c) \tag{28}$$

where $v^{fs}(c)$: content vector of the content c in the final step of the iteration; $\forall \delta r(c)$: Error rate of the content-to-vector layer; and ρ: processing rate.

2) DEEP RECURRENT NEURAL NETWORK FOR MALICIOUS CONTENT DETECTION LAYER

We need to confirm that the malicious contents in the links should be extracted. Therefore, auto-encoder is used to detect the malicious contents in the links. RNN autoencoder is a typical type of RNN based encoder-decoder model. The autoencoder helps learn the compressed illustration of input data and supports text, audio, video, and time-series data. In our case, we deal with malicious content coming from different sources in the form of text, audio, video, and time-series data. The RNN autoencoder compresses the input data because the massive amount of data comes from different sources. Thus, it is of paramount importance to save the hardware storage using data compression. Additionally, communication bandwidth can also be improved that could have a positive impact on the performance of the system. Furthermore, the decoder regenerates the input data from the compresse. Let us assume that the auto-encoder β be used to extract ω the malicious contents from the link l to be written as: $\{l = l^1, l^2, \ldots, l^m\}$.

Where l^m: total number of malicious contents on the each link.

The auto-encoder βe deduces the malicious contents C_m from the link l to formulate as: $\beta e : l \rightarrow \omega$, and decoder βd that rebuilds the link after removing the concealed malicious contents to be written as $\beta d : \omega \rightarrow l$. Where ω: content-illustration for the link when using the auto-encoder and auto-decoder.

The RNN autoencoder βe that learns the probability of detecting the malicious links by being trained. The RNN involves the hidden state S_h and possible output to be activated on the total malicious contents on the link. Therefore, the input and output hidden states of the RNN are updated by each time step t that can be calculated by equations 29-30 and depicted in Figure 5.

$$S_h(t) = f\Big(\beta e\Big)\Big(W_i S_h \times S_h(t) - 1, l^t\Big) \tag{29}$$

$$f(t) = f\Big(o\Big)\Big(W_o S_h \times S_h(t), l^t\Big) \tag{30}$$

After accessing the link, the final hidden state of the RNN is applied as a content illustration ω for the link. The GRU is used with RNN that reduces the computational complexity. The GRU improves the memory capacity of the RNN and provides the ease to successfully train the model. Additionally, GRU can provide early observation, if encountered any situation that requires early observation for future predictions.

Figure 8 depicts that an automatic-encoder βe gets the input link $I(t)$ from the GRU to determine the probability of the malicious contents. The sigmoid function is used to process the input to the next hidden state $S_h(t) - 1$. This process continues until the output decision $\beta e(t)O$ is obtained regarding the malicious contents from the given link. The encoder is reset for detecting the malicious contents in the next link. The encoder possesses the updated features $\varepsilon \beta e(t)$ which are the important to update the next layer. Additionally, the encoder is a capable of activating $A\beta e(t)$ the process.

$$\begin{array}{cccc} Word-to-Vector & ContentMatrix & & Content-to-Vector \\ w_1, w_2, \ldots, w_K & C_1, C_2, \ldots, C_N & & C_1, C_2, \ldots, C_N \\ a_{11}, a_{12}, \ldots, a_{1K} & b_{11}, b_{12}, \ldots, b_{1C} & & \sum_{i=1}^{K} a_{1i}b_{i1}, \sum_{i=1}^{K} a_{1i}b_{i2}, \ldots, \sum_{i=1}^{K} a_{1i}b_{iC} \\ a_{21}, a_{22}, \ldots, a_{2K} & \times b_{21}, b_{22}, \ldots, b_{2C} & = & \sum_{i=1}^{K} a_{2i}b_{i1}, \sum_{i=1}^{K} a_{2i}b_{i2}, \ldots, \sum_{i=1}^{K} a_{2i}b_{iC} \\ \vdots & \vdots & & \vdots \\ a_{Y1}, a_{Y2}, \ldots, a_{YK} & b_{Z1}, b_{Z2}, \ldots, b_{ZC} & & \sum_{i=1}^{K} a_{Yi}b_{i1}, \sum_{i=1}^{K} a_{Yi}b_{i2}, \ldots, \sum_{i=1}^{K} a_{Yi}b_{iC} \end{array} \tag{26}$$

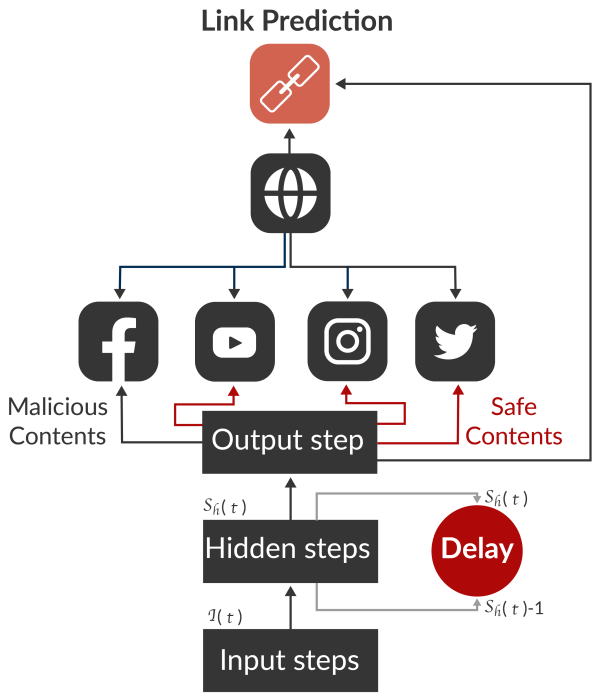


FIGURE 7. Recurrent neural network for malcious/safe content detection from the links.

The formulation of the content-detection through the automatic-encoder using the GRU is derived as: Here, initially for $t = 0$; and $\beta e(t)O = 0$ Thus, it can be determined as:

$$\varepsilon\beta e(t) = \sigma_{sigmoid}\{(\forall\rho \cdot \varepsilon\beta e(t) \cdot l) + (\forall U \cdot \varepsilon\beta e(t) \cdot S_h(t) - 1) + \forall b \cdot \varepsilon\beta e(t)\} \quad (31)$$

$$R\beta e(t) = \sigma_{sigmoid}\{(\forall\rho \cdot R\beta e(t) \cdot l) + (\forall U \cdot R\beta e(t) \cdot S_h(t) - 1) + \forall b \cdot R\beta e(t)\} \quad (32)$$

$$A\beta e(t) = \varphi h\{(\beta e(t)O \cdot l) + \forall AU \cdot A\beta e(t) \times (R\beta e(t) \oplus S_h(t) - 1) + \forall b \cdot S_h\} \quad (33)$$

$$\beta e(t)O = \{(1 - \varepsilon\beta e(t)) \oplus (\beta e(t)O - 1) + (\varepsilon\beta e(t) \oplus A\beta e(t))\} \quad (34)$$

where $S_h(t)\varphi h$: hyperbolic tangent; and $\sigma_s igmoid$: sigmoid function.

$\forall\rho$, $\forall U$, and $\forall b$ are the parameters of the auto-encoder, which are used to help for detecting the malicious contents given by

$$\omega = \beta e(l, \forall\rho) \quad (35)$$

The decoder is an important that uses the ω as the input to begin building process. The decoder βd is built using another RNN to create the sequence of the malicious output contents $\{i = i^1, i^2, \ldots, i^m\}$. The hidden state of the decoder is activated with each time step t given by

$$S_h(t) = f(\beta e)(W_i S_h(t) - 1, i^t) \quad (36)$$

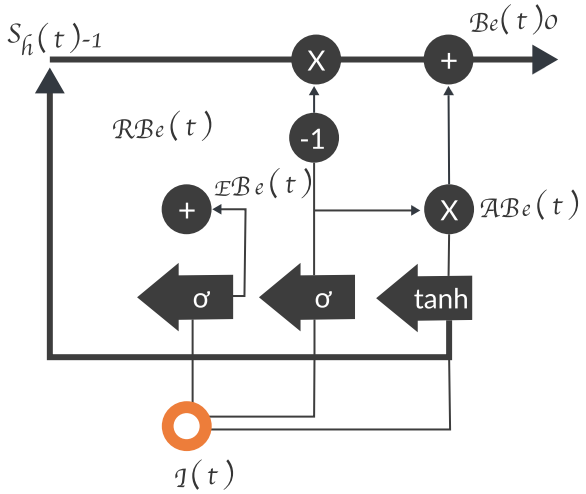


FIGURE 8. Content-detection through the automatic-encoder using gated recurrent unit.

The overall probability of building the malicious output sequence i producing the input link l is defined as:

$$p(i|l; \theta_d) = \prod_{i=1}^{m} p(i_t|i_t - 1, i_t - 2, \ldots, i_1, \omega, \theta_d) \quad (37)$$

where θ_d: parameter for the decoder.

The components of the auto-encoder are mutually trained to reduce the negative effect of the probability p for all of the malicious contents of the link.

$$l^m(\forall\rho, \forall U, \forall b, \theta_d) = -\sum_{t=1}^{m} \log p(i_t|l_t; \rho, \forall U, \forall b, \theta_d) \quad (38)$$

3) BLOCKCHAIN-ENABLED MALICIOUS CONTENT DETECTION LAYER

The Blockchain technology improves security and privacy of the system [27]. Thus, the Blockchain model is used to combat [28], [29]. Most centralized systems, such as banking systems, data management systems, and user management systems, suffer from malicious clickbait. The model consists of the following components:

- Smart Contract for Contents
- Content Authenticity
- Reliable Miner Nodes
- Proof-of-Truthfulness

a) Smart Contract for Contents: The smart contract is employed to broadcast the contents on the P2P network. Pertinent content information, such as content source, content status, valid identity, time stamp, and content broadcaster, may be stored by the smart contract. The content broadcasting C_br process is given by:

$$C_{br} = \sum_{i=0}^{C_t} (S_{ic})_i + \{(R_{bc}) \times C_{pc}\} + C_\mu \quad (39)$$

b) Content Authenticity: Ensuring authenticity of the contents coming from the different sources about a

clickbait is another challenge. A simple hash-based method is not appropriate for guaranteeing positive or negative clickbait credibility because of its lack of stoutness. Cosine similarity of the contents impending from two or more different sources to address this issue. Cosine similarity is used to measure the content authenticity of the blockchain, whether the content comes from authenticated or unauthenticated sources. The content authenticity C_{au} is determined as:

$$C_{au} = Sig(C_{so}) + V_{Id} + S_I \tag{40}$$

where C_{so}: content source; V_{Id}: valid identity; and S_I: Time stamp.

c) Reliable Miner Nodes: There is a possibility of inflowing malicious clickbait through the P2P network, which causes the spreading of fake information. Thus, the miners used the proof-of-authority protocol to sustain the blockchain while intending a new "content block." The majority of reliable miner nodes provide services to sustain the authenticity of the system. It is more probable that a reliable node enters a turn every time to suggest that a new content block be added to the blockchain. The reliable miner nodes should be deployed by reliable organizations.

d) Proof-of-truthfulness: Any participating miner node in the P2P network handles confirming whether the contents belong to the blockchain or not, employing the proof-of-truthfulness process. If the safe clickbait is derived from the contents, it is stored in blockchain; otherwise, it is discarded. Given the contents, anyone can authenticate their reliability in O (log n) time by using time complexity.

VII. EXPERIMENTS RESULTS AND SETUP

To validate the performance of our proposed BDRNN, realistic testing scenarios are conducted for malicious link detection and are compared with state-of-the-art methods: LSACD, CNN-LSTM and CNNC. These experiments are also performed to demonstrate the benefits of choosing this tool when calculating values such as the system load, detection accuracy of the malicious links, detection accuracy of safe links, analysis, and display of a message to the user. To obtain a practical result, a BDRNN was created that implements the algorithms proposed for identifying malicious links. The minimum system requirements for the components are described in Table 2. The program was created in the Java language using the IntelliJ IDE development environment. The Java language was chosen because it does not depend on a platform due to its virtual machine. The built-in library collection was used, which has a binary search implementation. Features of binary and domain rating check algorithms are used for faster and efficient search process.

This prototype was created to test the work of the program and algorithms with real links. Then, we determined how well they cope with the tasks.

TABLE 2. Components for conducting experiments.

Components	Version/ name of system
Operation system	Any
Processor	Intel(R) Core 2 Duo
RAM	2048 MB
HARD Disk free space	100 MB

For testing, contents were used, among which were malicious and safe types. Different contents were given to the proposed BDRNN, after which the BDRNN performed the analysis and displayed a message to the user based on the obtained data. The algorithm consists of several scenarios. Parameters used in the scenarios are described in Table 3.

TABLE 3. Parameters and descriptions.

Parameter	Description
B_L	Blocklist of URLs' (malicious contents)
W_L	Allowlist of URLs' (safe contents)
D_R	Content rating of sources

- **Scenario 1:** The BDRNN finds a link in a B_L or W_L and, based on this, gives information to the user.
- **Scenario 2:** The BDRNN does not find the content in B_L and W_L but finds its contents in the source and, based on this D_R, gives information to the user.
- **Scenario 3:** The content-to-vector detection layer counts the malicious words from the different sources (e.g. headlines, and social media) and informs the user.
- **Scenario 4:** The proposed BDRNN finds the malicious/ safe contents from the sources using deep recurrent neural network for malicious contents and inform the user about the nature of the contents.
- **Scenario 5:** The attacker attempts to alter the legitimate source of the contents arriving from the deep recurrent neural network into Blockchain technology.
- **Scenario 6:** The attacker attempts to get the legitimate source of the contents from the pending contents that are still not mined.
- **Scenario 7:** The attacker attempts to modify the legitimate source of the contents on the peer-to-peer network that is directly connected with the Blockchain.

Table 4 shows messages that are received from the program after it is executed with test data.

Based on the testing process, results of interest were obtained, and state-of-the-art metrics were measured.

- Link detection
- Memory and CPU usage
- Comparison with analogs
- Accuracy
- Successful content capturing rate

A. LINK DETECTION

The obtained results were expected based on three scenarios. The program was able to identify both malicious and safe sites. Additionally, good informational messages were

TABLE 4. Obtained results from test program.

Source of the contents/URL	Received Message
http://a1540.g.akamai.net	The source is secure
http://citrixreceiver493000.html	The source is secure
http://download.macromedia.com	The source is secure
http://download.videolan.org	The source is secure
02.2m1sdzs.co.cc	The source is malicious
04.64764sd.co.cc	The source is malicious
1tomohappy.com	The source is malicious
25-trafbvicimj.tf	The source is malicious
28843622.biz	The source is malicious
www.trinity.com	This source has good rating so that the contents coming for that source are safe
moneyhelp.br	The contents coming from that source are highly likely malicious
mmrlen.online	The contents coming from that source are highly likely malicious
dmr.top	The contents coming from that source are highly likely malicious
www.prg-app.com	This source has good rating so that the contents coming for that source are safe
www.transfer.ibnm	The contents coming from that source are ambitious so that nature is unknown

received on the analyzed links, which also indicates the successful operation of the BDRNN. The results of testing the link detection are depicted in figure 9a. Based on the testing results, it is observed that the proposed BDRNN is capable of detecting the more malicious links as compared to contending methods (LSACD, CNN-LSTM, and CNNC). Thus, computational complexity can be reduced using the proposed BDRNN.

The BDRNN detected 287 malicious links and 98 safe links; whereas competing methods detected 228, 242, and 269 malicious links for CNN-LSTM, LSACD, and CNNC respectively. On the other hand, the competing methods detected more safe links even though some of the links from those links were not safe. CNN-LSTM is declared a less reliable method due to its performance. Therefore, the number of the malicious and safe detected links $\Delta\forall\beta$ can be determined as:

$$\Delta\forall\beta = L_{da} \times \omega \times \sum_{i=0}^{L_t} (D_{cl} i + (1 - Lda) \times N_t) \quad (41)$$

where L_{da} is the link detection activity that represents the link detection time to conduct the tests for determining the nature of the links (either malicious or safe). ω denotes the probabilities of determining the trends of links. D_{cl} is the probability of the correctly detected-links. L_t is the total number of monitored links. N_t represents the normal trend of the traffic for links' monitoring.

B. MEMORY AND CPU USAGE

During testing, the hardware resources used by the BDRNN were also measured to analyze the possibility of use by a regular user. The memory load is shown in figure 9b. The CPU load is shown in figure 9c. The proposed BDRNN consumed a 211-Mb memory load that is lesser than competing methods. On the other hand, the competing methods consumed more memory load that is counted between 347-316 MB depicted in figure 9b. Based on the results, it is observed that the proposed BDRNN consumed 36-101 Mb less memory load. LSACD consumed a 312-Mb memory load that is higher than the competing methods.

The memory utilization m_u of the proposed BDRNN and contending methods is calculated as:

$$m_u = \left\{\Delta m_p - (mnu + m_{bc} + m_c)\right\} \quad (42)$$

where Δm_p is the total memory of the system that is used for link-detection. The m_{nu} denotes the memory of the system that is not utilized due to testing process for the link-detection done by each methods. m_{bc} represents the size of the system that provide the memory-buffering capacity when conducting the link-detection process, and m_c is the cache-memory of the system utilized by each method. Figure 9c depicts the CPU load. The results demonstrate that the proposed BDRNN average 5.26% CPU load whereas the contending methods consumed 6-8.13% CPU consumption maximum 36-minutes testing time. The CPU utilization C_{ut} can be calculated as:

$$\boldsymbol{C_{ut}} = \frac{\boldsymbol{\delta h} \times 100\%}{\forall \boldsymbol{\delta h}} \quad (43)$$

where $\forall \boldsymbol{\delta h}$ is the total capacity of each method to use the CPU resources, and $\boldsymbol{\delta h'}$ is the resources used by each method.

C. COMPARISON WITH ANALOGS

To compare the developed extension with analogs, two experiments were carried out.

In experiment 1, we checked how many ordinary ClickBait that do not contain malicious links were blocked. Interesting results were obtained: contending methods blocked almost all links that were not malicious and could benefit the user. Since they do not carry out analysis, they simply block all advertising. Comparison A comparison showed is shown in figure 10a. The proposed BDRNN blocked 5-links by marking them malicious links, but those are originally non-malicious links. As a result, this obstruction could restrict the number of users to get access to those links due to the result of false positive rate. On the other hand, CNN-LSTM, LSACD, and CNNC blocked 8,9, and 10 respectively. The Non-Malicious blocked link probability NM_{lp} can be calculated as:

$$NM_{lp} = \frac{NM_{ln}}{NM_{ln} + M_{ln}} \quad (44)$$

where NM_{ln} denotes the number of non-malicious blocked links, which have been considered mistakenly as the malicious links that happen due to the vulnerability of the methods. The M_{ln} denotes the number of malicious links that are originally malicious links but considered as non-malicious detected links.

In experiment 2, we checked how many malicious links BDRNN and contending methods could find. After the results were obtained, it became clear that although the extension

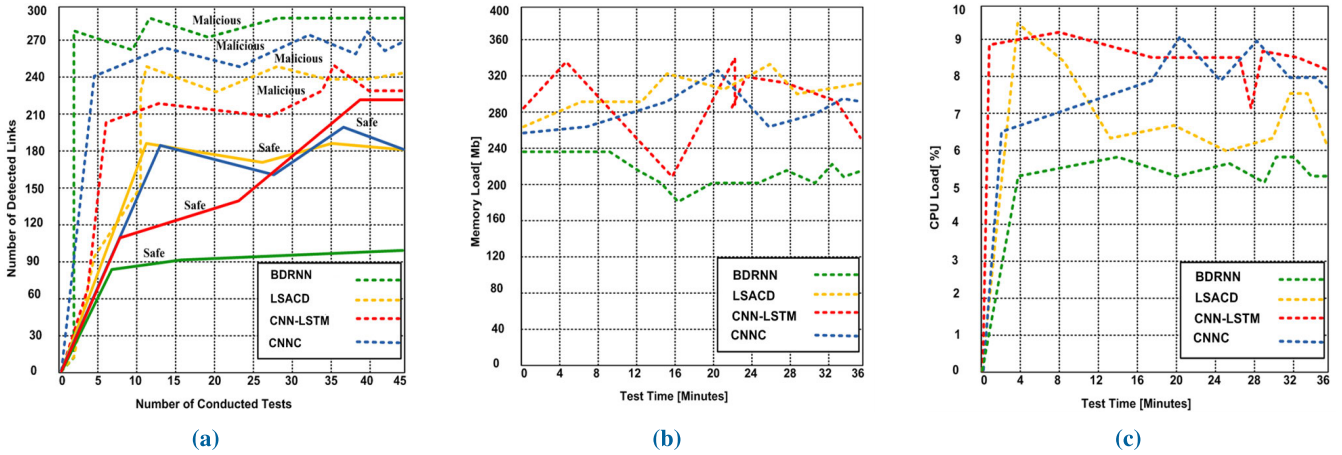


FIGURE 9. (a) The number of malicious and safe detected links for the proposed BDRNN and contending methods LSACD, CNNC, and CNN-LSTM with a maximum of 45 conducted tests. (b) Memory load for the proposed BDRNN and contending methods LSACD, CNNC, and CNN-LSTM with a maximum of 36-minutes test time. (c) Consumed CPU for the proposed BDRNN and contending methods LSACD, CNNC, and CNN-LSTM with a maximum of 36-minutes test time.

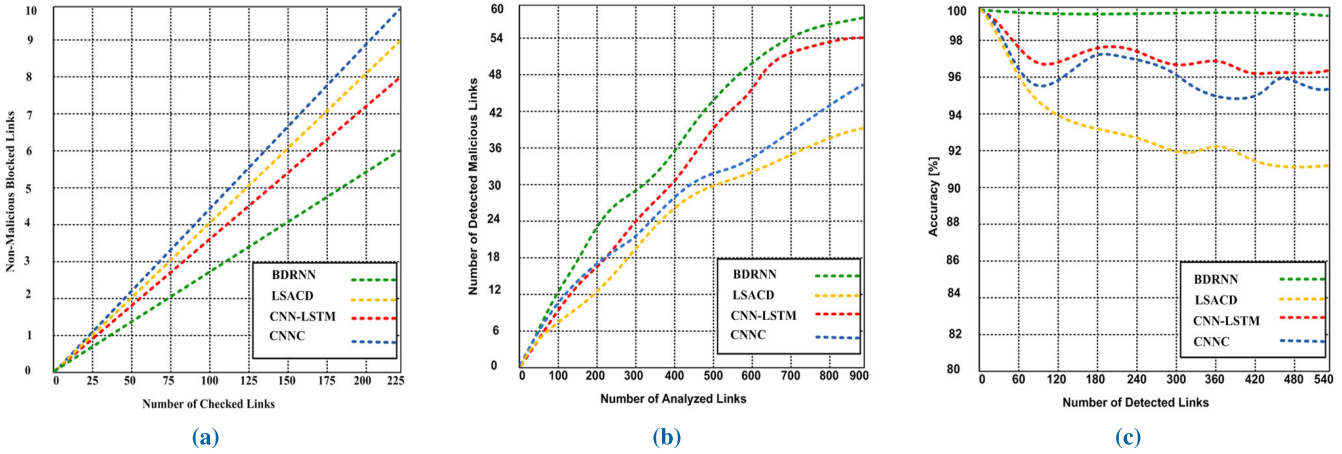


FIGURE 10. (a) The number of Non-Malicious detected links for the proposed BDRNN and contending methods LSACD, CNNC, and CNN-LSTM with a maximum of 225 checked links. (b) The number of the malicious detected links for the proposed BDRNN and contending methods LSACD, CNNC, and CNN-LSTM. (c) Accuracy of proposed BDRNN and contending methods with maximum 540 detected links.

was completely new, it detected malicious links quite well as compared to competing methods. The results are depicted in figure 10b. Based on the results, it has been observed that the proposed BDRNN detected 57 malicious links which are higher than competing methods. On the other hand, the competing methods: LSACD, CNNC, and CNN-LSTM have detected 39, 46, and 48 malicious links respectively. It is proved that the proposed BDRNN has a better capacity of detecting malicious links as compared to competing methods.

D. ACCURACY

It refers to the identification of conformism to the truth. Figure 10c demonstrates the accuracy of proposed BDRNN and contending methods (LSACD, CNNC, and CNN-LSTM). Based on the results, it has been proved that the accuracy of the proposed method is higher than the contending methods. As proposed BDRNN shows 99.43% accuracy, whereas the contending methods produce the accuracy of 91.2-96.2. It is proved that BDRNN has 3.23-8.23% higher accuracy for detecting the Clickbait links.

The accuracy of each method is calculated as:

$$\forall A_m = \frac{\forall M_{pl} + M_{ln}}{\forall M_{pl} + NM_{ln} + M_{ln} + M_{ld}} \tag{45}$$

where $\forall M_{pl}$ is the correct link prediction regarding malicious/non-malicious links, and M_{ld} represents the state, which demonstrates that there is no existing malicious link, but it exists in the Clickbait.

E. SUCCESSFUL CONTENT CAPTURING RATE

The scenario-5 is generated to determine the successful content capturing rate of the attacker. In this scenario, the attacker attempts to capture the legitimate sources of the contents to include the malicious contents. The contents come from the deep recurrent neural network into Blockchain technology.

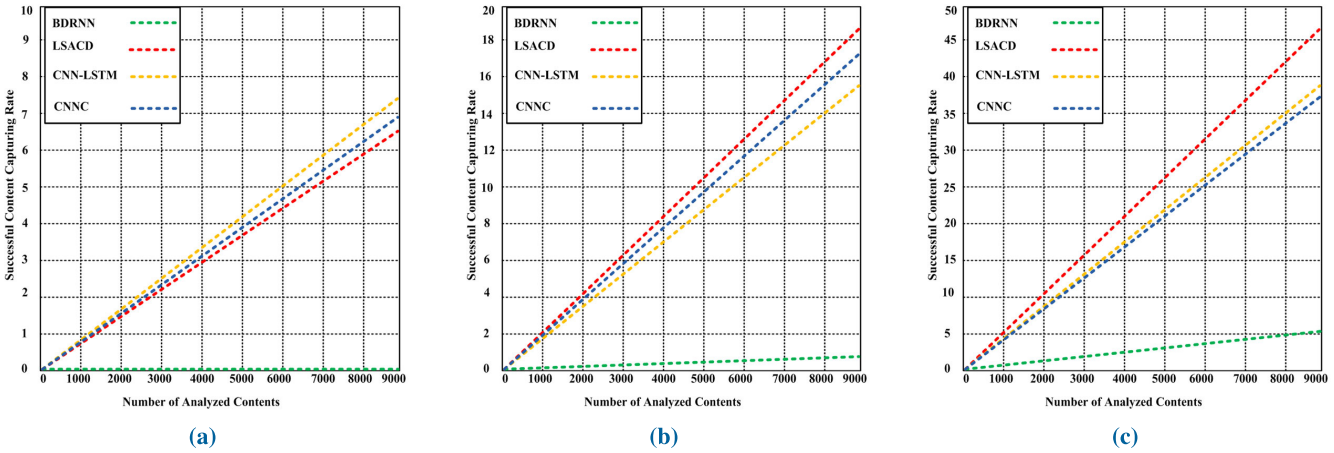


FIGURE 11. (a) Attacker's successful content capturing rate by attempting the maximum number of 9000 contents to maliciously change them and contents arriving from the deep recurrent neural network and entering into Blockchain. (b) Attacker's successful content capturing rate by attempting the maximum number of 9000 contents to maliciously modify them and the contents are forwarded by the smart contracts, but the status of the contents shows mined-pending. (c) Attacker's successful content capturing rate by attempting the maximum number of 9000 contents to maliciously modifying the contents that are on the peer-to-peer network, which is directly connected with the Blockchain.

As these contents have already been decided either malicious or safe contents. However, the dual authentication process is introduced using Blockchain technology. Blockchain technology provides dual security to process these contents, but there is the possibility that Blockchain technology could be compromised when processing the contents. Thus, the effectiveness of Blockchain technology has been the main focus in scenario-5. Based on the experimental result, it has been observed that the proposed BDRNN is not affected due to the attacker's malicious attempts depicted in Figure 11a. The success rate of the attacker is found 0%, while the attacker has been successful in capturing the contents when using the contending methods. The attacker can get successful content capturing rate of 6.48% with the use of the LSACD, 6.91% with CNNC, and 7.31% with CNN-LSTM. Thus, the overall rate of content capturing is even not higher with contending methods. We have also generated and tested scenario-6. In which, the attacker attempts to hack the pending contents that are still not mine. The result depicted in Figure 11b demonstrates that the attacker has little success to capture the contents. However, the content capturing rate is lower for the proposed BDRNN that is only 0.76%. On the other hand, the contending models are highly affected. The 15.52% content capturing rate is detected with CNN-LSTM, but this rate is higher for CNNC and LSACD that is 17.21% and 18.41% respectively. When the blocks are mined then they are transmitted on the peer-to-peer network for storage illustrated in scenario-7, then the possibility of capturing the contents is much higher as demonstrated in Figure 11c. Based on the results, it has been observed that the proposed BDRNN is slightly affected when sending the mined blocked on the network. The successful content capturing rate with BDRNN has been calculated to be 5.09%. This content capturing rate proves the vulnerability of Blockchain technology. However, the vulnerability is not only inherited from the Blockchain technology in the proposed BDRNN, but it could be the vulnerability of the peer-to-peer network that could affect the overall performance of the proposed BDRNN. The contending models have a massive negative effect, so the 37.23% successful content capturing rate is calculated with CNNC. Moreover, other contending models are highly affected and the successful content capturing rates are determined to be 38.82% and 46.12% for CNN-LSTM and LSACD respectively. Hence, it is concluded that the proposed BDRNN can be the victim of the attacker due to inherited features of the Blockchain technology, but the negative impact is trivial. While the contending models are highly affected and the content capturing rate has been calculated much higher in scenarios 5-7.

VIII. DISCUSSION OF RESULTS

The proposed framework BDRNN has an edge over counterparts. The reason for the edge is that the BDRNN consists of three phases to detect clickbait. All of these three phases are responsible for clickbait detection. The detection goes through different processes (e.g. Characterizations and distinguishing the allow/block-list links, rating the source of the links, and malicious or safe clickbait detection). The first phase is to rate the sources. In this phase, all of the sources are determined to rate as the lower-to-higher rated sources. The contents coming from the lower-rated sources are not allowed and only highly-rated sources are allowed. There is the possibility the contents coming from the lower-rates sources could be safe, but the proposed approach does not deal with those sources. This is one shortcoming of the proposed approach that could be addressed by using an additional layer named source registration layer that will provide an opportunity for all of the sources to be registered under certain criteria that could be incorporated in future work. There could be another possibility that the

received contents (clickbait) from the highly-rated sources may bring the malicious contents. To overcome this issue, all of the contents should be arranged and sorted out. Thus, the second phase of the proposed approach is responsible to evaluate and sort out the contents. In this phase, all of the clickbait is sorted efficiently using the binary search features. When all of the clickbait is sorted out then it should be checked for detecting either safe or malicious content. There is a massive probability that highly-rated sources could also be compromised by the attacker. Thus, this problem is addressed using multiple layers. Therefore, the third phase is a multi-layered clickbait detection to detect malicious and safe contents. This phase is of paramount prominence that consists of three models: content-to-vector model (layer-1), deep neural network model(layer-2), and Blockchain-enabled malicious content detection model (layer-3). In the content-to-vector model, content information is processed, where stop words and punctuations are removed to make the sentence in the correct order and conduct a content lemmatization process on it. And, clean inputs are transformed as the content vectors.

A few existing approaches use only the source rating process, but they do not use the ClickBait arranging and sorting processes. Additionally, contending approaches neither have source rating nor Clickbait sorting features. As a result, the contending approaches accept all of the contents coming from all of the sources that may take a longer time to distinguish and process the contents. This weakness of the existing as well as contending approaches causes higher CPU and memory consumption that is not a healthy sign particularly when the system requires faster clickbait detection. This weakness increases the complexity and slows down the process that is reflected in Figures 9b-c. The second module is to use of deep neural network model. As deep neural network consists of three types: Artificial Neural Network (ANN), Convolutional neural network (CNN), and RNN. Most of the existing approaches either use ANN or CNN, and only a competing model named CNNC uses the RNN. Our proposed approach BDRNN also leverages the features from RNN. The ANN and CNN models are not fully compatible with Clickbait detection. As ANN and CNN are only capable to process temporal information or data that is received in sequences that could be a problem when generating the next output in the series. Additionally, CCN can only extract the local and position-invariant characteristics that are not required for clickbait detection. If CNN is used for clickbait detection, then there should be a supporting model to sort out the contents into the form of sentences because CNN supports sentences well, but Clickbait is initially not found in the form of sentences. Hence, this is the weakness of the existing models including contending models LSACD, and CNN-LSTM. As a result, the malicious detection rate of these contending models' is lower and reflected in Figures 10a-b and attacker's successful content capturing rate is higher than the proposed method and reflected in Figures 11a-c. On the other hand, ANN supports tabular, image, and text data. In most cases, links are not available in tabular, text, and images forms. Thus, this method is not suitable for clickbait detection. The main disadvantage of ANN is not capturing sequential data in the form of input data that is highly important to deal with sequential data. Thus ANN is not a good candidate to be used for Clickbait detection. The contending model CNNC only uses the features of RNN. The RNN provides better outcomes when the classification is made by the extended range semantic dependency. The RNN supports two variants: long short-term memory (LSTM) and GRU. The contending CNNC uses LSTM that has more overhead as compared to GRU. Thus, contending CNNC takes more computation to complete tasks. On the other hand, the proposed BDRNN uses GRU. The advantage of using GRU is to reduce computational complexity. In addition, layer-2 has the support of another important feature semantic similarity content module that forwards the malicious and safe contents successfully to Blockchain technology (layer-3). The existing approaches including contending models do not have these features. This feature provides easy access to be used Blockchain technology. Most of the malicious and safe contents are detected on layer-2. However, the inclusion of Blockchain technology increases detection accuracy. Blockchain provides a good level of protection, as it always provides reliable information about malicious blocks of the contents. Most of the existing works do not use Blockchain technology and little work has been found that uses Blockchain technology for detecting fake news, but not clickbait. The proposed BDRNN integrates the additional component (content authenticity) with Blockchain technology. As Blockchain technology cannot detect the contents coming from different sources about clickbait. Blockchain technology uses a simple hash method that is not appropriate for guaranteeing positive or negative clickbait credibility because of its lack of stoutness. Cosine similarity is used to measure the content authenticity of the Blockchain, whether the content comes from authenticated or unauthenticated sources. Thus, the accuracy of the proposed framework is higher than the contending methods reflected in Figure 10c. From Table 5, it is observed that the proposed BDRNN successfully detects malicious and non-malicious links, whereas the contending methods falsely detect malicious links as safe links.

The proposed BDRNN also inherits the shortcomings of Blockchain technology for example adaptability, scalability, and energy efficiency. These Blockchain technology affecting factors do not have too much negative impact on the proposed approach because the proposed approach aims to detect the Clickbait successfully, then energy efficiency can be sacrificed. Another disadvantage of the proposed BDRNN is not to detect the malicious executable code is available inside the Clickbait. If the BDRNN cannot obtain enough information for some reason, then it cannot analyze and just provides the recommendations to the user. However, disadvantage number two is improbable, so users do not have to worry about it. In summary, we can claim that the proposed

TABLE 5. Comparison between proposed BDRNN and contending approaches: LSACD, CNN-LSTM and CNNC.

Metrics #	LSACD	CNN-LSTM	CNNC	**BDRNN**
Number of detected links	242 malicious & 128 safe links	228 malicious & 180 safe links	269 malicious & 180 safe links	287 malicious & 98 safe links
Memory usage	316-Mb	247-Mb	295-Mb	211-Mb
CPU usage	6%	8.13%	7.68%	5.26%
Non-malicious detected links	9 links	8 links	10 links	5 links
Malicious detected links	39 links	48 links	46 links	57 links
Detection accuracy	91.2%	96.2%	95.32%	99.43%
Attacker's successful content capturing rate with scenario-5	6.48%	7.31%	6.91%	0%
Attacker's successful content capturing rate with scenario-6	18.41%	15.52%	17.21%	0.76%
Attacker's successful content capturing rate with scenario-7	46.12%	38.82%	37.23%	5.09%

BDRNN has the advantage over the counterparts and existing approaches that provide a good level of protection.

IX. CONCLUSION

Research has been conducted that demonstrates that clickbait can pose a threat to users when using the Internet. The proposed BDRNN enables the evaluation of the security of a content resource. Additional information can be obtained using the existing sources of information about web resources.

A BDRNN was developed to solve the security issues of clickbait. A comparison was conducted with state-of-the-art existing neural network methods (LSACD, CNN-LSTM, and CNNC) to determine the effectiveness of the presented solution. The BDRNN provides a good level of security for users because the results obtained during the testing process demonstrate that our proposed BDRNN for the Clickbait accurately identifies malicious links and indicates this to the user. The proposed BDRNN consists of multi-layered models that leverages the features from the content-to-vector, deep recurrent neural network and Blockchain models. Furthermore, the use of BDRNN will help to provide a greater level of security. The information technology industry has changed rapidly, and therefore there is a need to create a mechanism to protect against viruses and threats. Therefore, in the future, we will provide insights into additional metrics.

REFERENCES

[1] A. Pujahari and D. S. Sisodia, "Clickbait detection using multiple categorisation techniques," *J. Inf. Sci.*, vol. 47, no. 1, pp. 118–128, Feb. 2021.

[2] J. A. Nasir, O. S. Khan, and I. Varlamis, "Fake news detection: A hybrid CNN-RNN based deep learning approach," *Int. J. Inf. Manage. Data Insights*, vol. 1, no. 1, Apr. 2021, Art. no. 100007.

[3] W. T. Al-Sit and R. Al-Hamadin, "Real estate market data analysis and prediction based on minor advertisements data and locations' geo-codes," *Int. J.*, vol. 9, no. 3, pp. 4077–4089, 2020.

[4] L. Shang, D. Zhang, M. Wang, S. Lai, and D. Wang, "Towards reliable online clickbait video detection: A content-agnostic approach," *Knowl.-Based Syst.*, vol. 182, Oct. 2019, Art. no. 104851.

[5] N. Hurst, "To clickbait or not to clickbait?: An examination of clickbait headline effects on source credibility," Ph.D. dissertation, Univ. Missouri-Columbia, Columbia, MO, USA, 2016.

[6] A. S. Manideep, M. S. K. R. Reddy, and P. S. Reddy, "Impact of social network advertisements on brand equity of wellness firms and the mediating role of brand awareness: An empirical analysis ABSTRACT," *Int. J. Manage. Bus. Res.*, vol. 9, no. 2, pp. 46–53, 2019.

[7] B. Berendt, P. Burger, R. Hautekiet, J. Jagers, A. Pleijter, and P. Van Aelst, "FactRank: Developing automated claim detection for dutch-language fact-checkers," *Online Social Netw. Media*, vol. 22, Mar. 2021, Art. no. 100113.

[8] D. R. Patil and J. B. Patil, "Malicious URLs detection using decision tree classifiers and majority voting technique," *Cybern. Inf. Technol.*, vol. 18, no. 1, pp. 11–29, Mar. 2018.

[9] M. Mouzarani, B. Sadeghiyan, and M. Zolfaghari, "Detecting injection vulnerabilities in executable codes with concolic execution," in *Proc. 8th IEEE Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2017, pp. 50–57.

[10] K. L. Chiew, K. S. C. Yong, and C. L. Tan, "A survey of phishing attacks: Their types, vectors and technical approaches," *Expert Syst. Appl.*, vol. 106, pp. 1–20, Sep. 2018.

[11] E. Sidorova, I. Kononenko, and Y. A. Zagorulko, "An approach to filtering prohibited content on the web," in *Proc. CEUR Workshop*, 2017, pp. 64–71.

[12] P. Papadopoulos, P. Ilia, and E. P. Markatos, "Truth in web mining: Measuring the profitability and cost of cryptominers as a web monetization model," 2018, *arXiv:1806.01994*.

[13] J. Brands and J. van Wilsem, "Connected and fearful? Exploring fear of online financial crime, internet behaviour and their relationship," *Eur. J. Criminol.*, vol. 18, no. 2, pp. 213–234, Mar. 2021.

[14] C. Zhang and P. D. Clough, "Investigating clickbait in Chinese social media: A study of WeChat," *Online Social Netw. Media*, vol. 19, Sep. 2020, Art. no. 100095.

[15] J. Y. Khan, M. T. I. Khondaker, S. Afroz, G. Uddin, and A. Iqbal, "A benchmark study of machine learning models for online fake news detection," *Mach. Learn. with Appl.*, vol. 4, Jun. 2021, Art. no. 100032.

[16] M. Garnaeva, F. Sinitsyn, Y. Namestnikov, D. Makrushin, and A. Liskin, *Overall Statistics for 2016*. Irvine, CA, USA: Blue Book, 2016.

[17] V. Subramaniyaswamy, R. Logesh, V. Vijayakumar, and V. Indragandhi, "Automated message filtering system in online social network," *Proc. Comput. Sci.*, vol. 50, pp. 466–475, Jan. 2015.

[18] A. K. Jain and B. B. Gupta, "A novel approach to protect against phishing attacks at client side using auto-updated white-list," *EURASIP J. Inf. Secur.*, vol. 2016, no. 1, pp. 1–11, Dec. 2016.

[19] S. R. Sahoo and B. B. Gupta, "Multiple features based approach for automatic fake news detection on social networks using deep learning," *Appl. Soft Comput.*, vol. 100, Mar. 2021, Art. no. 106983.

[20] N. Asanka Gamagedara Arachchilage and M. Abdul Hameed, "Integrating self-efficacy into a gamified approach to thwart phishing attacks," 2017, *arXiv:1706.07748*.

[21] S. Kaur, P. Kumar, and P. Kumaraguru, "Detecting clickbaits using two-phase hybrid CNN-LSTM biterm model," *Expert Syst. Appl.*, vol. 151, Aug. 2020, Art. no. 113350.

[22] J. Zheng, K. Yu, and X. Wu, "A deep model based on lure and similarity for adaptive clickbait detection," *Knowl.-Based Syst.*, vol. 214, Feb. 2021, Art. no. 106714.

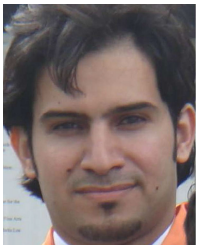
[23] H.-T. Zheng, J.-Y. Chen, X. Yao, A. Sangaiah, Y. Jiang, and C.-Z. Zhao, "Clickbait convolutional neural network," *Symmetry*, vol. 10, no. 5, p. 138, May 2018.

[24] J. Noh, S. Jeon, and S. Cho, "Distributed blockchain-based message authentication scheme for connected vehicles," *Electronics*, vol. 9, no. 1, p. 74, Jan. 2020.

[25] A. Mathur, J. Vitak, A. Narayanan, and M. Chetty, "Characterizing the use of browser-based blocking extensions to prevent online tracking," in *Proc. 14th Symp. Usable Privacy Secur. (SOUPS)*, 2018, pp. 103–116.

[26] P. Patil, "Artificial intelligence in cyber security," *Int. J. Res. Comput. Appl. Robot.*, vol. 4, no. 5, pp. 1–5, 2016.
[27] M. A. Ferrag and L. Shu, "The performance evaluation of blockchain-based security and privacy systems for the Internet of Things: A tutorial," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17236–17260, Dec. 2021.
[28] A. Razaque, A. Al Ajlan, N. Melaoune, M. Alotaibi, B. Alotaibi, I. Dias, A. Oad, S. Hariri, and C. Zhao, "Avoidance of cybersecurity threats with the deployment of a web-based blockchain-enabled cybersecurity awareness system," *Appl. Sci.*, vol. 11, no. 17, p. 7880, Aug. 2021.
[29] S. Karnouskos, "Artificial intelligence in digital media: The era of deepfakes," *IEEE Trans. Technol. Soc.*, vol. 1, no. 3, pp. 138–147, Sep. 2020.

ABDUL RAZAQUE (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from the University of Bridgeport, USA, in 2015. He is currently a Professor with the Department of Computer Engineering and Telecommunications, International Information Technology University. His research interests include wireless sensor networks, cyber security, cloud computing security, design and development of mobile learning environments, and ambient intelligence. He has authored over 170 international academic publications, including journals, conferences, book chapters, and four books. He is an editor, an associate editor, and a member of editorial board of several journals.



BANDAR ALOTAIBI (Member, IEEE) received the Bachelor of Science degree (Hons.) in computer science (information security and assurance) from the University of Findlay, USA, the Master of Science degree in information security and assurance from Robert Morris University, USA, and the Ph.D. degree in computer science and engineering from the University of Bridgeport, USA. He is currently an Associate Professor with the Information Technology Department, University of Tabuk. His research interests include computer vision, network security, mobile communications, computer forensics, wireless sensor networks, and quantum computing.

MUNIF ALOTAIBI (Member, IEEE) received the Bachelor of Science degree in computer science (information security and assurance) from the University of Findlay, USA, the Master of Science degree in information security and assurance from Robert Morris University, USA, and the Ph.D. degree in computer science and engineering from the University of Bridgeport, USA. He is currently an Assistant Professor with the College of Computing and Information Technology, Shaqra University. His research interests include biometric authentication, pattern recognition, information security, network security, and machine learning.

FATHI AMSAAD (Member, IEEE) received the Ph.D. degree in engineering science from Toledo (UToledo), OH, USA. He is currently an Assistant Professor with the School of Information Security and Applied Computing (SISAC), Eastern Michigan University (EMU), MI, USA. He is also the Founder and the Director of the Cyber Security Laboratory, and the Co-Director of the Advanced Computing Research Laboratory. His research interests include cyber security and cyber-physical systems with special interests in hardware-oriented security and trust for device and system authentication, secure embedded architectures, VLSI/FPGA systems testing, fault tolerance hardware, detection and prevention of hardware trojans, network and mobile wireless security, and the security of IoT applications and smart systems. He is also an active IEEE/ACM Member. He has served as a project advisor for several groups of senior undergraduate students and a reviewer of high-impact and peer-review conferences/journals. He was a recipient of the prestigious IEEE Best Graduate Student Award by IEEE Region 4 and the College of Engineering, UToledo. He was also the 2017 nominee for the Best Ph.D. Dissertation Award. He holds MCP, MCSA, MCTS, and MCSE Professional Certificates from Microsoft.

ANSAGAN MANASOV (Member, IEEE) received the M.Sc. degree from the Department of the Radio Engineering Electronics and Telecommunications, Al-Farabi Kazakh National University, Almaty, Kazakhstan, in 2020, where he is currently pursuing the Ph.D. degree in information security systems. He is currently working as an Engineer with a focus on the cybersecurity domain. His research interests include network design, virtualization, cybersecurity, and cloud computing. He is certified in CCNA RS, HCNA RS, CCNA Security, Palo Alto ACE, CCNP Security, and Splunk Enterprise Security.

SALIM HARIRI (Senior Member, IEEE) received the M.Sc. degree from The Ohio State University, Columbus, OH, USA, in 1982, and the Ph.D. degree in computer engineering from the University of Southern California, in 1986. He is currently a Full Professor and The University of Arizona Site Director of the NSF-Funded Center for Cloud and Autonomic Computing. He founded the IEEE/ACM International Symposium on High Performance Distributed Computing (HPDC). He is also the Co-Founder of the IEEE/ACM International Conference on Cloud and Autonomic Computing. He has coauthored three books on autonomic computing, parallel computing, and distributed computing, and edited *Active Middleware Services (Kluwer)*, a collection of articles from the second annual AMS Workshop in 2000. He serves as the Editor-in-Chief of the scientific journal *Cluster Computing*, which presents research and applications in parallel processing, distributed computing systems, and computer networks.

BANU B. YERGALIYEVA (Member, IEEE) is currently pursuing the Ph.D. degree in information security with Gumilyov Eurasian National University, Nur-Sultan, Kazakhstan. She is a Scientific Researcher with the Institute of Information Security and Cryptography. She has published 20 academic articles in peer-reviewed conferences and journals. Her research interests include information security, cryptography, cloud computing, secure outsourcing, and secure cloud storage.

AZIZ ALOTAIBI (Member, IEEE) received the Ph.D. degree in computer science and computer engineering from Bridgeport University, Bridgeport, CT, USA. He is currently an Associate Professor with the Computer Science Department, Taif University. His research interests include artificial intelligence, machine learning, deep learning, computer vision, and cybersecurity.