

DeepHTTP: Semantics-Structure Model with Attention for Anomalous HTTP Traffic Detection and Pattern Mining

Yuqi Yu

Beihang University
Haidian Qu, Beijing Shi, China
yuyuqi_bh@buaa.edu.cn

Hanbin Yan

Beihang University, National Computer Network
Emergency Response Technical Team/Coordination
Center of China
Chaoyang Qu, Beijing Shi, China
yhb@cert.org.cn

Hongchao Guan

Beijing University of Posts and Telecommunications
Haidian Qu, Beijing Shi, China
ghc@bupt.edu.cn

Hao Zhou

National Computer Network Emergency Response
Technical Team/Coordination Center of China
Chaoyang Qu, Beijing Shi, China
zhz@cert.org.cn

ABSTRACT

In the Internet age, cyber-attacks occur frequently with complex types. Traffic generated by access activities can record website status and user request information, which brings a great opportunity for network attack detection. Among diverse network protocols, Hypertext Transfer Protocol (HTTP) is widely used in government, organizations and enterprises. In this work, we propose DeepHTTP, a semantics structure integration model utilizing Bidirectional Long Short-Term Memory (Bi-LSTM) with attention mechanism to model HTTP traffic as a natural language sequence. In addition to extracting traffic content information, we integrate structural information to enhance the generalization capabilities of the model. Moreover, the application of attention mechanism can assist in discovering critical parts of anomalous traffic and further mining attack patterns. Additionally, we demonstrate how to incrementally update the data set and retrain model so that it can be adapted to new anomalous traffic. Extensive experimental evaluations over large traffic data have illustrated that DeepHTTP has outstanding performance in traffic detection and pattern discovery.

CCS CONCEPTS

• Security and privacy → Intrusion detection systems; • Information systems → Data mining;

KEYWORDS

Anomalous HTTP Traffic Detection, Malicious Pattern Mining, Semantics Structure Integration, Attention Mechanism

ACM Reference Format:

Yuqi Yu, Hanbin Yan, Hongchao Guan, and Hao Zhou. 2018. DeepHTTP: Semantics-Structure Model with Attention for Anomalous HTTP Traffic

Detection and Pattern Mining. In *Proceedings of ACSAC 2018*. ACM, New York, NY, USA, 11 pages.

1 INTRODUCTION

As the development of computer technology, network systems and applications get increasingly more complex than ever before. More and more bugs and vulnerabilities appear constantly, which poses a great threat to network security. Attacks initiated by exploiting vulnerabilities are getting increasingly more sophisticated. As a result, many traditional anomaly detection methods based on standard mining methodologies are no longer effective. Thus, how to discover various hidden web attacks is still a topic of great concern in the field of network security.

HTTP, a representative of network protocol, occupies a considerable proportion of the application layer traffic of the Internet. Since HTTP traffic can record website access states and request content, it provides an excellent source of information for anomaly detection [16, 18, 38]. According to the characteristics, existing approaches for anomalous HTTP traffic detection can be roughly divided into two categories: feature distribution based methods [23, 35] and content-based detection technologies [36]. Additionally, since malicious traffic detection is essentially an imbalanced classification problem, many researches propose anomaly-based detection approaches that generate models merely from the benign network data[5]. With the rapid development of artificial intelligence, deep learning has been widely used in various fields and has a remarkable effect in natural language processing. Recently, deep learning has been successfully applied in anomaly detection[19, 32].

Even though these methods are successful in certain scenarios, they are not universal. The main difficulties and challenges of malicious traffic detection are as follows. First of all, it is difficult to automatically detect hidden anomalous traffic from massive network traffic with lots of noise. How to enhance the generalization ability and robustness of the model is still a critical issue. Secondly, in practical applications, anomaly-based detection model usually has a high false positive rate. Since the model is trained based on normal samples, traffic not present in the training set is likely to be labeled as malicious. This problem undoubtedly increases the

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

ACSAC 2018, December 3-7, 2018, San Juan, Puerto Rico, USA

© Copyright held by the owner/author(s).

workload of manual verification. Last but not least, grasping the characteristics and disciplines of traffic helps to improve defense rules, which is crucial in practical application scenarios. However, the study of pattern mining for malicious traffic is not yet mature. It remains to be a challenging task.

To resolve the above challenges, in this work, we propose DeepHTTP, a data-driven method leveraging the large volumes of traffic data. We assume that the new malicious traffic is similar to the known anomalous traffic in terms of structure or content. Indeed, each HTTP request follows strict architectural standards and language logic, which is similar to natural language. Hence, we treat elements in traffic content as vocabulary in natural language processing and establish models to learn the semantic relationships between them. Meanwhile, we propose a method for extracting traffic structure information. Specifically, we segment the content of the traffic entries and convert content clips to structural features. The integration of semantics and structural features helps to improve the generalization of the model.

The model we proposed is a combined semantics structure model utilizing Bidirectional Long Short-Term Memory (Bi-LSTM) with attention mechanism. Bi-LSTM can obtain information from past and future states, which allows the model to learn the text characteristics better. Moreover, neural network model with attention mechanism can automatically dig out key information to distinguish different traffic. To discover the patterns, we cluster samples that have been labeled as malicious and perform pattern mining for each cluster. Since it is a learning-driven approach, we set up a process that can verify and update data efficiently. The model is updated periodically so that it can adapt to new malicious traffic that appears over time. DeepHTTP is a complete framework that can automatically distinguish malicious traffic and mine patterns without prior knowledge.

In summary, we make the following contributions.

We have an in-depth analysis of the types and encoding forms of content of HTTP traffic, then propose an effective structure extraction method.

We adopt an integration framework that combines two attention-based Bi-LSTM models training upon semantics and structure features, separately. Vectors that combine semantic and structural information are ultimately used for classification. The proposed model has superior learning ability and generalization ability.

We propose a novel approach for mining traffic patterns. We use the attentional hidden states of traffic learned by the model as input for clustering. For each cluster, we perform pattern mining based on weight vector obtained from attention model. Experiments prove the effectiveness of our approach.

The rest of this paper is organized as follows. Section 2 gives a summary of the relevant research. Section 3 briefly introduces the system framework and data preprocessing methods, especially the extraction method of the structural characteristics of traffic. The proposed model is introduced in detail in section 4, including the malicious traffic detection model and pattern mining method. We launched a comprehensive experiment to demonstrate the effectiveness of the model. The experimental results are shown in section 5. Section 6 gives the conclusions and future works.

2 RELATED WORK

In recent years, there are quite a few researches aiming for detecting anomaly traffic and hidden attacks.

Communication traffic contains lots of information which can be used to mine anomaly behaviors. Lakhina et al.[22] perform a method that fuses information from flow measurements taken throughout a network. Content features are also valuable for detecting anomaly behaviors. Wang et al.[40] present Anagram, a content anomaly detector that models a mixture high-order n-grams designed to detect anomalous and "suspicious" network packet payloads. To select the important features from huge feature spaces, Zseby et al.[17] propose a multi-stage feature selection method using filters and stepwise regression wrappers to deal with feature selection problem for anomaly detection. The methods mentioned above care less about the structure features of communication payloads which are important for distinguishing anomaly attacking behaviors and mining anomaly patterns. In this paper, we put forward structure extraction approach, which can help enhance the ability of detecting anomaly traffic. The structure feature also makes an important role in pattern mining.

There are various methods and models applied in anomaly detection. Some works have been done based on dimensionality reduction [4, 10, 12, 20]. Juvonen and Sipola [20] propose a framework to find abnormal behaviors from HTTP server logs based on dimensionality reduction. Researchers compare random projection, principal component analysis and diffusion map for anomaly detection. Ringberg et al. [34] propose a nonparametric hidden Markov model with explicit state duration, which is applied to cluster and scout the HTTP-session processes. This approach analyses the HTTP traffic by session scale, not the specific traffic entries. Additionally, there are also many researches based on traditional methods such as IDS(intrusion detection system and other rule based system) [13, 16, 18, 29, 42]. However, these methods rely on feature engineering, which cannot detect unknown types of traffic. In order to solve this problem, researchers began to apply machine learning and deep learning methods to the anomaly detection field[2, 3, 11, 14]. Among them, solution[2] is a new supervised hybrid machine-learning approach for ubiquitous traffic classification based on fuzzy decision trees with attribute selection. There are also quite a few works based on deep learning methods. Erfani et al.[14] present a hybrid model where an unsupervised DBN is trained to extract generic underlying features, and a one-class SVM is trained from the features learned by the DBN. LSTM model is used for anomaly detection and diagnosis from System Logs[11]. In this paper, we present Bi-LSTM based on attention mechanism, which has not been applied yet. This model works well in experiments and has good generalization ability in real traffic.

In addition to detecting malicious traffic and attack behaviors, some researches focus on pattern mining of cluster traffic. Most existing methods for traffic pattern recognition and mining are based on clustering algorithms [6, 25]. Le et al. [25] propose a framework for collective anomaly detection using a partition clustering technique to detect anomalies based on an empirical analysis of an attack's characteristics. Since information theoretic co-clustering algorithm is advantageous over regular clustering for creating more fine-grained representation of the data, Mohiuddin Ahmed and

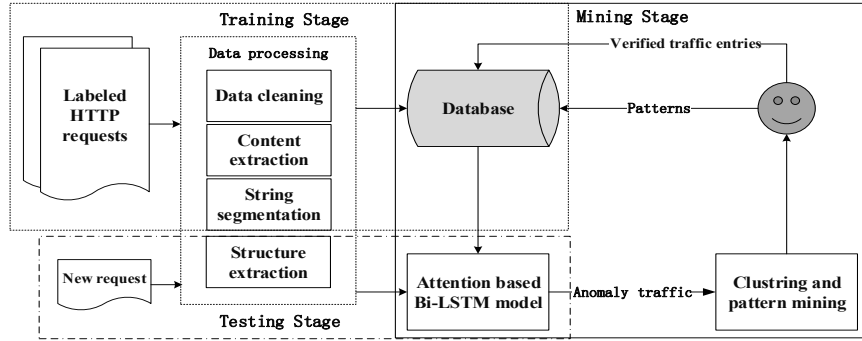


Figure 1: DeepHTTP architecture.

Abdun Naser Mahmood[1] extend the co-clustering algorithm by incorporating the ability to handle categorical attributes which augments the detection accuracy of DoS attacks. In addition to clustering algorithm, JT Ren [37] conducts research on network-level traffic pattern recognition, and uses PCA and SVM for feature extraction and classification. I. Paredes-Oliva et al. [31] build a system based on an elegant combination of frequent item-set mining with decision tree learning to detect anomalies.

In recent years, the attention-based neural network model has become a research hotspot in deep learning, which is widely used in image processing [43], speech recognition [7] and healthcare [28]. The application of attention mechanism in the field of natural language processing has also proved to be extremely effective. Luong et al.[27] first design two novel types of attention-based models for machine translation. Since the attention mechanism can automatically extract important features from raw data, it has been applied to topic classification[46], relation Classification[45], and abstract extraction[33]. Nevertheless, to the best of our knowledge, there are few works on detecting anomaly HTTP traffic using sequence model and mining patterns based on attention mechanism. Hence, in this paper, we build a model based on attention mechanism, which can get rid of the dependency of artificial extraction features and do well in traffic pattern mining.

3 PRELIMINARIES

3.1 DeepHTTP Architecture and Overview

The architecture of DeepHTTP shown in Figure 1 consists of three phases and three components. Specifically, it is divided into training stage, test stage, and verification stage. Structure extraction, anomaly traffic detection model and malicious pattern mining module are main parts that run through the entire framework. These parts will be described in detail in later sections.

3.1.1 Training Stage. In the training stage, the core task is the construction of the detection model. Specifically, it includes feature engineering and the training of models. We extract valuable information of the traffic after data processing. To characterize traffic, we fuse the content and structural features of traffic as input to the model. Subsequently, the neural network model with attention mechanism will be trained and iterated on a regular basis. To

enhance the robustness of the model, we build data sets containing positive and negative samples in different proportions and use cross-validation to train the model. The best model will be saved for actual traffic detection.

3.1.2 Testing Stage. In this stage, the trained model is used for anomaly traffic detection. For each new HTTP request, we apply data processing to deal with it. The content and structural features are feed into the trained model for prediction. The intermediate outputs of the model will be reserved and used in the mining phases.

3.1.3 Mining Stage. The main works of this phase are to verify the anomalous traffic labeled by the model and to mine malicious patterns. We obtain attentional hidden state from the proposed model as the input of clustering. For each cluster, we extract a small number of samples for manual verification, and then tag all members in the cluster. Simultaneously, we analyze the critical part of each request according to attention weight vector, and then sum up malicious patterns of each cluster. All results are updated to the database.

3.2 Data Preprocessing

We focus on HTTP traffic mainly for three reasons: 1) Hypertext Transfer Protocol is an application layer protocol that is widely used for communication between web browsers and web servers. It is used by most web servers. 2) A large majority of web attacks use HTTP, such as Cross-site scripting attack (XSS), SQL injection and so on. These attacks can be well submerged in massive amounts of normal traffic data. 3) Even though payload of traffic can be modified and confused, it is still extremely helpful for relevant research works.

Selecting valuable feature and performing valid data preprocessing are the foundation of data mining. This section introduces the data processing and feature extraction methods in detail.

3.2.1 Data Cleaning. To optimize the follow-up detection and analysis, we perform data processing on original HTTP traffic packages captured by monitor software. We parse packages and extract valuable information including request headers and request bodies. The rest of the data processing includes decoding, deleting erroneous and duplicate data, and filling in missing values.

Table 1: We replace substring with RS(Replacement String) according to its ET(Encoding Type). For strings not belonging to any specially encoded, we substitute each character with RC(Replacement Character) according to its CT(Character Type).

ET	RS	CT	RC
MD5 hash	MD5_HASH	Arabic Numerals	D
SHA hash	SHA_HASH	English Alphabet	W
Base64	BASE64_ENCODE	Garbage Character	G
Hexadecimal	HEXADECIMAL	Chinese Character	C
Binary	BINARY	Invisible Character	I

3.2.2 Content Extraction. Malicious information of web attacks is usually contained in the parameter value of the request path and the request body (if a POST request is sent). Hence, we extract Uniform Resource Locator (URL) and payload from processed data.

3.2.3 String Segmentation. Text vectorization is the key to text mining. Numerous studies use n-grams [9] to extract the feature of payloads [21, 41, 47]. This method can effectively capture the byte frequency distribution and sequence information, but it is easy to cause dimension disaster. To extract crucial parts of a HTTP request, we split the string with special characters as delimiters instead of n-grams. Here is an instance. Suppose the decoded content is: `/tienda1/publico/vaciar.jsp <EOS> B2=Vaciar carrito; DROP TABLE usuarios; SELECT * FROM datos WHERE nombre LIKE`. The data after string segmentation is denoted as: `/ tienda1 / publico / vaciar . jsp <EOS> B2 = Vaciar carrito ; DROP TABLE usuarios ; SELECT * FROM datos WHERE nombre LIKE`. Words in string are connected by spaces. The identifier `<EOS>` indicates the end of the URL and the start of the request body. The purpose of using this word segmentation method is to preserve words with semantic information as much as possible (like `SELECT`). Another benefit is that it makes the result of pattern mining more interpretable. In this case, the pattern we want to obtain is `{SELECT, FROM, WHERE}`. However, if we use n-grams(n=3), the result may be denoted as `{SEL, ELE, ..., ERE}`, which is not intuitive.

3.2.4 Structure Extraction. Research observed that in many cases there are "stable" path components that are specific to a particular malicious family or operation [30]. In other words, there is similarity in the content or structure of the same type of traffic. Inspired by this, the paper extracts structure information based on a series of string substitution rules. In order to cover all the string types that appear in the request data, we have performed frequency statistics. The following Table 1 gives a summary of pivotal rules.

Unlike general natural language, there are plentiful special characters in HTTP request. These special characters usually reflect the traffic characteristics and have high practical value. Therefore, for a substring with length 1, we will not replace it if it is a special character. We traverse other substrings in the segmented string to determine their encoding types (such as md5 or hexadecimal) and convert it to the replacement string according to rules shown in Table 1. If it does not belong to any special encoding form, we

traverse each character in the substring, judge what type it is and replace it.

```

/mobile ? verifycontent = 68247 & tenantid = 3c5fee3560000218bf9c5d7b5d3524e
/WWWWW ? WW WWWWWW WWW WWW W = DDDDD & WW WWWWWW WW = MD5_HASH

/mobile ? articlecontent = 486975 & password = 8ef04d797dad53d5c43d21a0d320eab
/WWWWW ? WW WWWWWW WWW WWW W = DDDDD & WW WWWWWW WW = MD5_HASH

```

Figure 2: An example of structure extraction.

An example of structure extraction is shown in Figure 2. Since the encoding type of the substring `"3c5fee3560000218bf9c5d7b5d3524e"` is MD5, we replace it with `"MD5_HASH"`. For the substring `"68247"`, we replace each number in it with `"D"`, then we can obtain its structure feature, which is denoted as `"DDDDD"`. Obviously, by extracting structural features, we can easily find requests with different content but almost the same structure. Combined content and structural features are beneficial to improve the generalization ability of the model. It is crucial for identifying malicious traffic and discovering malicious patterns.

4 OUR APPROACH

4.1 Anomaly HTTP Traffic Detection

The goal of the proposed algorithm is to identify anomaly HTTP traffic based on semantics and structure of traffic entries. Figure 3 shows the high-level overview of the proposed model. The model based on Bi-LSTM and attention mechanism contains five components: input layer, word embedding layer, Bi-LSTM layer, attention layer and output layer. Before output layer, we train content sequence and structure sequence respectively. Then, we adopt dropout strategy to avoid overfitting and simply fuse the feature vector of content and structure. In the output layer, we perform classification using a softmax function.

4.1.1 Problem definition. Let $R = \{R_1, R_2, \dots, R_N\}$ be the set of HTTP traffic entries after data processing. For each traffic entry $R_i (i = 1, 2, \dots, N)$, there are two sequences $S_i^1 = \{c_{11}, c_{12}, \dots, c_{1n}\}$ and $S_i^2 = \{c_{21}, c_{22}, \dots, c_{2n}\}$, which respectively represent content sequence and structure sequence. Because structure sequence is derived from content sequence, the length of both sequence is equal to n. As a result, we obtained two feature sets to characterize the semantics and structure of traffic.

4.1.2 Input Layer. One-hot representation is a widely used and relatively simple word vector generation method in the field of natural language processing. However, the length of the word vector generated by this text preprocessing method is up to vocabulary size. Usually, vectors are usually quite sparse.

In this paper, we use the content and structure sequence after word segmentation as a corpus, and select words that are common in the corpus to build a vocabulary according to term frequency-inverse document frequency (TFIDF). Then, the unique index is generated for each word in the vocabulary. The final input vector with fixed length is composed of indexes. The length of input vector is denoted as z, which is a hyper-parameter(the fixed length in this paper is set to 300 because the proportion of sequence length within

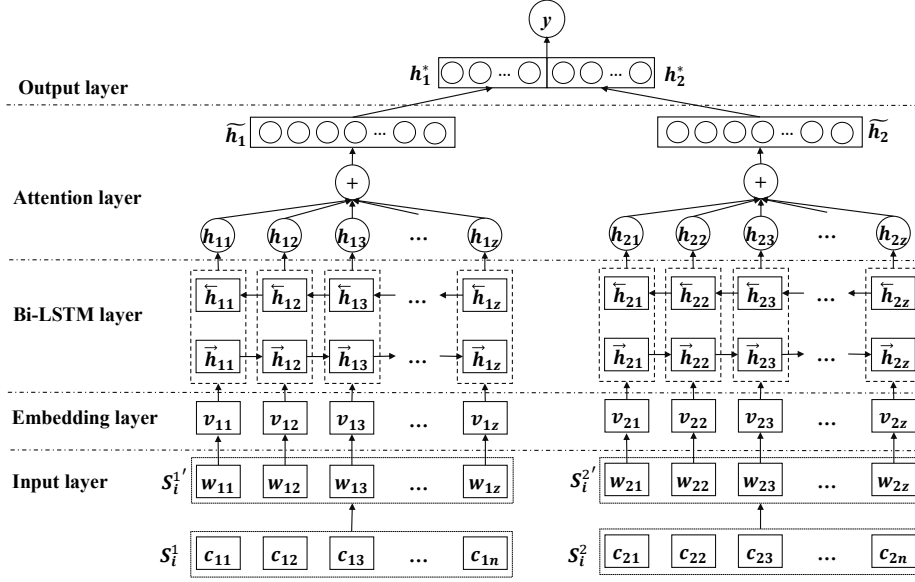


Figure 3: Model Architecture.

300 is 0.8484). The excess part of input sequence is truncated, and the insufficient part is filled with zero. Formally, the sequence of content can be converted to $S_i^1 = \{w_{11}, w_{12}, \dots, w_{1z}\}$ and the sequence of structure can be expressed as $S_i^2 = \{w_{21}, w_{22}, \dots, w_{2z}\}$. We use a simple example to illustrate the process. Given a sequence of content: {'/', 'admin', '/', 'caches', '/', 'error_ches', '/', 'php'}. The input vector with fix length can be denoted as $[23, 3, 23, 56, 23, 66, 0, 0, \dots, 0]$, the index of 'admin' in vocabulary is 3. Since the length of this sequence is less than fixed length, the rest of the vector is filled with zeros.

4.1.3 Embedding Layer. Take a content sequence of i -th traffic entry as an example. Given $S_i^1 = \{w_{11}, w_{12}, \dots, w_{1z}\}$, we can obtain vector representation $v_{1k} \in R^m$ of each word $w_{1k} \in R^1$ ($k = 1, 2, \dots, z$) as follows:

$$v_{1k} = \text{ReLU}(W_e w_{1k} + b_e) \quad (1)$$

where m is the size of embedding dimension, $W_e \in R^{m \times 1}$ is the weight matrix, and $b_e \in R^m$ is the bias vector. ReLU is the rectified linear unit defined as $\text{ReLU}(v) = \max(v, 0)$, where $\max()$ applies element-wise to vector.

4.1.4 Bidirectional Long Short-Term Memory. We employ Bidirectional Long Short-Term Memory (Bi-LSTM), which can exploit information both from the past and the future to improve the prediction performance and learn the complex patterns in HTTP requests better. A Bi-LSTM consists of a forward and backward LSTM. Given embedding vector $\{v_{11}, v_{12}, \dots, v_{1z}\}$ of content sequence of i -th traffic entry R_i , the forward LSTM f reads the input sequence from v_{11} to v_{1z} , and calculates a sequence of forward hidden states $\vec{h}_{11}, \vec{h}_{12}, \dots, \vec{h}_{1z}$ ($h_{1i} \in R^p$ and p is the dimensionality of hidden states). The backward LSTM \bar{f} reads the input sequence in the

reverse order and product a sequence of backward hidden states $\overleftarrow{h}_{11}, \overleftarrow{h}_{12}, \dots, \overleftarrow{h}_{1z}$ ($h_{1i} \in R^p$). The final latent vector representation $h_{1i} = [h_{1i}; \overleftarrow{h}_{1i}]^T$ ($h_{1i} \in R^{2p}$) can be obtained by concatenating the forward hidden state \vec{h}_{1i} and the backward one \overleftarrow{h}_{1i} . We deal with the embedding vector of structure sequence in the same way.

4.1.5 Attention Layer. In this layer, we apply attention mechanism to capture significant information, which is critical for prediction. The function we use to capture the relationship between h_t and h_i ($1 \leq i < t$) is called general attention:

$$\alpha_{ti} = h_t^T W_\alpha h_i \quad (2)$$

$$\alpha_t = \text{softmax}([\alpha_{t1}, \alpha_{t2}, \dots, \alpha_{t(t-1)}]) \quad (3)$$

where $W_\alpha \in R^{2p \times 2p}$ is the matrix learned by model, α_t is the attention weight vector calculated by softmax function. Then, the context vector $c_t \in R^{2p}$ can be calculated based on the weights obtained from Eq.(3) and the hidden states from h_1 to h_{t-1} as follows:

$$c_t = \sum_i^{\alpha_t} h_i \quad (4)$$

We combine current hidden state h_t and context vector c_t to generate the attentional hidden state as follows:

$$\tilde{h}_t = \tanh(W_c [c_t; h_t]) \quad (5)$$

where $W_c \in R^{r \times 4p}$ is the weight matrix in attention layer, and r is the dimensionality of attention state. \tilde{h}_1 and \tilde{h}_2 can be obtained using Eq.(2) to Eq.(5), which denote the attention vector of content and structure sequence learned by the model.

4.1.6 Output Layer. Before feeding the attention vector into softmax function, the paper apply dropout regularization [16] randomly disables some portion of attention state. It is worth noting that we concatenate vector of content and structure to generate output vector. Vectors incorporating semantic and structural information are eventually used for prediction. The classification probability is calculated as follows:

$$p = \text{softmax}(w_s[h_1^*; h_2^*] + b_s) \quad (6)$$

where h_1^* is the output of \tilde{h}_1 after dropout strategy, h_2^* is the output of \tilde{h}_2 . $w_s \in R^{q \times r}$ and $b_s \in R^q$ are the parameters to be learned.

$$\hat{y} = \text{argmax}(p) \quad (7)$$

where \hat{y} is the label predicted by the attention model.

4.1.7 Objective Function. The paper calculate the loss for all HTTP traffic entries using the cross-entropy between the ground truth $y_i \in (0, 1)$ and the predicted $p_i (i = 1, 2, \dots, N)$:

$$L = -\frac{1}{N} \sum_{i=1}^N y_i \log(p_{i1}) + (1 - y_i) \log(1 - p_{i1}) \quad (8)$$

where N is the number of traffic entries, p_{i1} denotes the probability that the i -th sample is predicted to be malicious.

4.2 Malicious Pattern Mining

This section provides in-depth analysis of model results, including malicious pattern mining and verification of model results. Figure 4 shows the architecture of mining stage. The process is explained as follows.

4.2.1 Clustering. We cluster traffic entries labeled as malicious by model, which is the basis for validation and pattern mining in our work. Specifically, we use the attentional hidden state extracted from the model as input vector for clustering. The clustering method we apply is DBSCAN [15], a density-based clustering algorithm, which does not require prior declaring the number of clusters. After clustering, we obtain several clusters. Traffic entries in each cluster are similar in content or structure.

4.2.2 Tag Verification. In practical applications, massive HTTP traffic requests are generated every day. There is no doubt that manual verification requires a lot of time and efforts. In this paper, we combine clustering and sampling to reduce the workload. After clustering, we extract some samples from each cluster for verification. If the predicted labels of these samples are consistent with the ground truth, then all the prediction results in this cluster are considered correct.

4.2.3 Malicious Patterns Discovery. Pattern mining of malicious traffic can help discovering commonalities and characteristics of malicious traffic and generating corresponding rules. Especially for unknown web attacks, analyzing their attack patterns is particularly significant.

As mentioned in section 3.1, the attention weight vector obtained in attention layer can reflect the crucial parts where the model is concerned. Therefore, for each malicious traffic entry, we dig out the key parts according to the corresponding attention weight vector. The greater the weight is, the more important the word is. Then,

Table 2: Distribution of malicious traffic entries.

Data Type	Number
Deserialization	6014
CMS	5836
File Inclusion	46438
SQL Injection	463776
Webshell	288050
XSS	127750
Sensitive Data Exposure	16656
Middleware Vulnerability	47614
Struts2 Vulnerability	42477
Botnet	19901
Total	1064512

given a series of HTTP requests in the same cluster, malicious pattern can be summed up by comprehensively analyzing the key parts. The specific steps are as follows:

Get Top_N words as candidate set. Given a cluster with N traffic requests $T = \{t_1, t_2, \dots, t_N\}$, we first calculate TFIDF for each word in segmented sequence set to obtain top n words in the entire cluster. The candidate set consisting of these words will be expressed as $C = \{c_1, c_2, \dots, c_n\}$.

Get Top_m words of each traffic. Then, we select top m key words $K_i = \{k_1, k_2, \dots, k_m\}$ for each traffic entry $t_i (i = 1, 2, \dots, N)$ according to the corresponding weight. As a result, we can obtain a set of key words identified by the model, which can be denoted as $K = \{K_1, K_2, \dots, K_N\}$.

Calculate the co-occurrence matrix. To discover pattern existed in the cluster, we calculate the co-occurrence matrix of words in candidate set C . The goal is that we want to unearth words that not only frequently occur in this cluster but also recognized by the model as key parts. If we can discovery several words with high attention weight that appear together constantly, then the combination of these words can represent the pattern of such traffic.

5 EVALUATION

5.1 Data Set

Due to the insufficient amount of data in the standard data set, we use real traffic data accumulated over time to validate our approach. Monitoring software is used to capture HTTP traffic packets via the gateway of an international university. The collected data is highly sensitive because it contains most of the network activities during work hours of teachers and students. For these data, we perform manual verification and tagging. In addition, other malicious traffic is collected by vulnerability scans under experimental environment. The total number of labeled data is 2095222, half of them are malicious traffic entries. The types and quantities of malicious samples are shown in the Table 2. Moreover, there is five million unmarked HTTP traffic prepared for model testing.

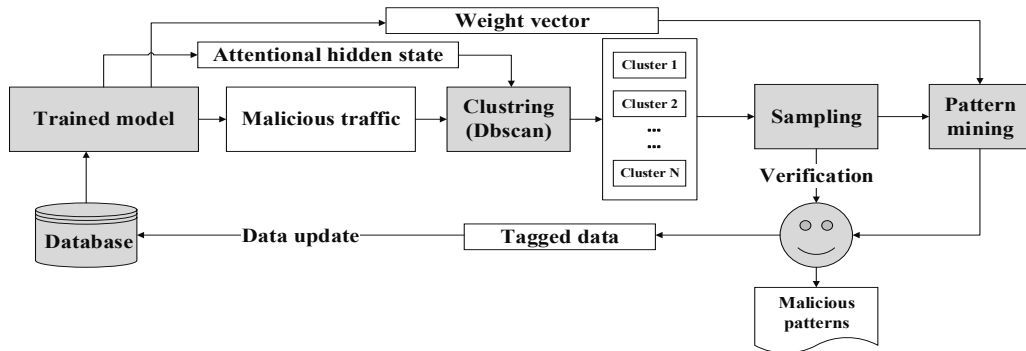


Figure 4: The architecture of mining stage.

5.2 Model Comparison

5.2.1 Detection in the Labeled Dataset. Three baseline methods are used for comparison experiments. Convolutional neural networks (CNNs)[24] is a class of deep feedforward artificial neural networks, most commonly applied to analyzing visual imagery. So far, it is also widely used in video recognition, recommender systems [39] and natural language processing [8]. Recurrent neural networks (RNNs), the collective name for a series of sequence models, which are usually applied to solve sequence problems, such as time series prediction[44] and speech recognition [26]. Long short-term memory (LSTM), variants of recurrent neural network (RNN), which can avoid the vanishing gradient problem. The proposed model is based on Bi-LSTM, which combines LSTM and bidirectional strategy. It can predict or label each element of the sequence based on the element’s past and future contexts. To illustrate the

Table 3: Model results in the labeled dataset.

Model	Sample Ratio	Precision	Recall	F1-score	AUC
CNN	1:100	0.9637	0.3556	0.5196	0.6778
LSTM		0.9408	0.6778	0.7879	0.8387
Bi-LSTM		0.9175	0.7448	0.8222	0.8721
Our Model		0.9561	0.9609	0.9585	0.9795
CNN	1:10	0.8772	0.8239	0.8496	0.9067
LSTM		0.9249	0.9759	0.9497	0.9844
Bi-LSTM		0.9866	0.9586	0.9724	0.9787
Our Model		0.9905	0.9747	0.9825	0.9869
CNN	1:1	0.9452	0.9867	0.9656	0.9679
LSTM		0.9954	0.9947	0.9951	0.9953
Bi-LSTM		0.9961	0.9948	0.9955	0.9957
Our Model		0.9979	0.9963	0.9971	0.9973

superiority of the proposed model, we extract positive and negative samples from the data set according to different proportions. The evaluation metrics consist of precision, recall, F1-score and AUC (the area under the receiver operating characteristic curve). Table 3 shows the experimental results of mentioned models. Overall,

Table 4: Model results in the unlabeled dataset.

Model	MT_MODEL	MT_RULE	MT_NEW	FP
CNN	795692	208917	60973	525802
LSTM	238689	139302	14527	84860
Bi-LSTM	295621	191267	28753	75601
Our Model	428270	216809	150974	60487

sequence model is more suitable for traffic detection than convolutional neural network model, especially Bi-LSTM. It is worth mentioning that our model is superior to all baseline models in almost all metrics. In unbalanced data sets, the superiority of the proposed model is even more pronounced.

5.2.2 Detection in the Unlabeled Dataset. We conduct comparative experiments in five million unlabeled traffic entries. Traditional filtering rules are utilized to assist model validation. The explanation of the assessment indicators we constructed is as follows:

MT_MODEL. The number of traffic labeled as malicious by the model.

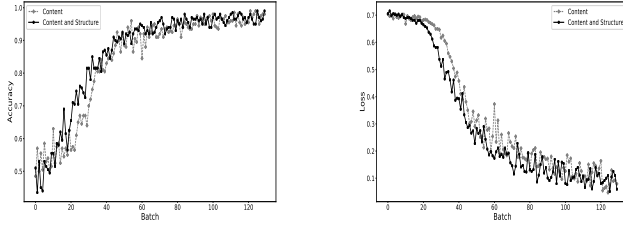
MT_RULE. The number of traffic that is labeled as malicious by the model and matches the rules.

MT_NEW. The number of traffic that is labeled as malicious by the model but does not match the rules.

FP. The number of false positive. ($FP = MT_MODEL - MT_RULE - MT_NEW$)

The result of model evaluation in real traffic dataset without label is shown in Table 4. According to MT_RULE, it can be seen that almost all malicious traffic discovered by rules can be detected by CNN and our model (the number of malicious traffic matching the filtering rules is 217100). Moreover, these models have the ability to discover new malicious traffic. However, MT_MODEL of CNN is extremely large, which means there are quite a few false positives. Additionally, compared with LSTM and Bi-LSTM, the proposed model is able to disclose more anomalous traffic entries. The FP of our model is the lowest. The result further proves the superiority of our model.

5.2.3 Model Performance with Different Features. We record the loss and accuracy of each iteration of the model and draw loss



(a) Accuracy curve.

(b) Loss curve.

Figure 5: Accuracy curve and loss curve.

curve and accuracy curve(Figure 5). To balance the memory usage and model training efficiency, the best batch size is set to 200. As we observe from the figure, the model trained based on content and structural features converges faster. In other word, after fusing structural features, the learning rate has been enhanced, and it can reach the convergence state faster.

In order to verify the validity and rationality of the structural features, we first construct three labeled data sets composed of different proportions of positive and negative samples. Then, in these data sets, we use different input features to train the models (including content features, structural features, combined content and structural features). After that, statistical measures of binary classification are used to evaluate the effectiveness of these models. As shown in Figure 6, these models perform better on balanced data sets. The model trained with structure feature is less effective than the other two models in three data sets. After integrating the structural features, the effect of the model has been significantly improved, especially in unbalanced data sets.

5.2.4 Malicious pattern mining. In this part, we first cluster anomaly traffic entries to find traffic families. Due to the combination of structural features, the model is able to detect anomalous HTTP requests that are different in content but with similar structure. Some examples of malicious traffic families discovered by the model are shown in Figure 7. It is worth noting that these malicious traffic entries has not appeared in the training set.

As mentioned before, attention weight is pivotal for pattern mining. We can summarize the critical parts of each traffic request according to its weight vector. The visualization of attention vector is shown in Figure 8. The HTTP request is extracted from a cluster after clustering. Color depth corresponds to the weight α_i (Eq.3). Obviously, the key words selected by model are $\{ 'submit', 'execute', 'wscript', 'action', 'shell' \}$, which are almost in line with the ground truth.

To assess the ability of mining patterns, we extract traffic entries by regex matching method to compose the test data set. We use five typical rules and extract one thousand entries for each rule. The specific rules and expressions are shown in Table 5.

We feed the traffic entries into the proposed model, then the prediction results and attention weights will be received. After that, for each traffic request, we extract top 5 words, top 10 words, and top 20 words ordered by the attention weights and calculate the count of key words appear in top k words ($k=5, 10, 20$). Just take

Table 5: Regex rules.

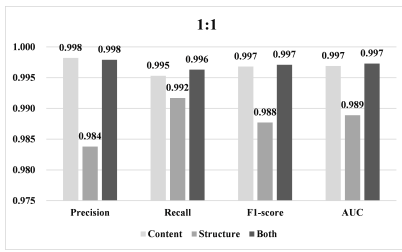
Regex Rule	Expression
memberaccess.allowstaticmethodaccess	Structs2
(select.+(from limit))((?:union(?:select))	Sql injection
<(iframe script body img input)	XSS
\$_(GET post GLOBALS SERVER)	Webshell
etc.{0,10}passwd	File include

Table 6: Pattern mining evaluation.

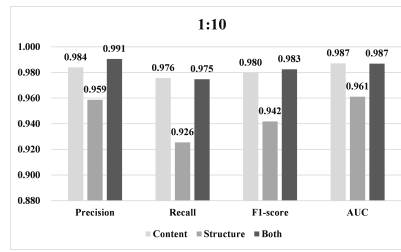
Rule	Key Words	Top 5	Top 10	Top 20
Rule1	memberaccess	0.822	0.893	0.964
	allowstaticmethodaccess	0.697	0.850	0.923
Rule2	select	0.744	0.832	0.871
	from	0.502	0.650	0.792
	limit	0.672	0.789	0.799
	union	0.482	0.573	0.630
Rule3	iframe	0.721	0.817	0.821
	script	0.789	0.820	0.854
	body	0.340	0.554	0.590
	img	0.470	0.334	0.652
	input	0.456	0.562	0.652
Rule4	get	0.523	0.585	0.789
	post	0.434	0.687	0.697
	globals	0.249	0.472	0.688
	server	0.209	0.426	0.532
Rule5	etc	0.790	0.893	0.948
	passwd	0.832	0.902	0.952

Structs2 exploitation of vulnerability as an example. According to the first rule, the key words are "memberaccess" and "allowstaticmethodaccess". Suppose there are five traffic entries, we collect top 5 words from each traffic according to their weights. If the top 5 words of each traffic contain the key word of malicious traffic (like "memberaccess"), the count plus one (we convert the count into a ratio). The experimental results are shown in Table 6. Obviously, for traffic with relatively simple attack patterns, the model can perform well (like "structs2" and "file include"). Moreover, for some sensitive words (like "scrip" and "limit"), the model can also well recognize.

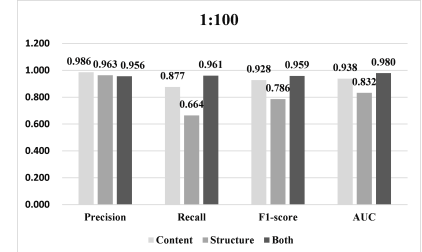
To further verify the performance of the proposed model on malicious pattern mining, we visualize the results generated by the method mentioned in section 3.2.3 in Figure 9. The darker the color of the square is, the more times the words appear together. The number of co-occurrences of these words can reflect the number of times that they were simultaneously concerned by the model. It is evident that the black area in the upper left corner of the figure is what the model considers important. Hence, the pattern of these traffic can be denoted as $\{ "<", "/", "textarea", "script", ">", "alert" \}$.



(a) The balanced dataset.



(b) The unbalanced dataset(1:10).



(c) The unbalanced dataset(1:100).

Figure 6: Model performance with different features in different datasets.

```

• /wso2.php<EOS>p2=view&p1=index.php&c=/home/vinograd/sanservice.by/&charset=wind
ows-1251&a=filetools&p3=
• /templates/bee3/index.php<EOS>p2=view&p1=index.php&c=/home/frioempresas/public_
html/&charset=utf-8&p3=&a=filetools
• /wp-content/uploads/2017/05/wp12.php<EOS>p2=view&p1=index.php&c=/home/clapt100/
public_html/&a=filetools&p3=&charset=windows-1251
-----
• /images/warmn.asp:<EOS>fname=c:\inetpub\wxhshaoyou1\payr.as\00&action=editfile
• /admin/jsinc/indax.asp<EOS>fname=c:\inetpub\wwwroot\wu\campc\11.html&action=delfile
• /assets/11.asp<EOS>fname=c:\inetpub\wwwroot\web.config&action=editfile
-----
• /lu/servlet/httpservletwrapper<EOS>echo+[e]&z1=/cmd&z2=cd+/d+"m:\$recycle.bin"&ec
ho+[s]&type+1.txt&z0=utf-8&023=m&cd
• /jssagdsagp.jsp<EOS>echo+[e]&z0=gb2312&echo+[s]&666=m&cd&z2=cd+/d+"e:\schoole
ard2.6.3.0\tomcat6.0\webapps\root"&z1=/cmd&whoami
• /fio/report/css.jsp<EOS>echo+[e]&echo+[s]&aasqzr11=m&z2=cd+/d+"c:\windows\system
32"&cd&z1=/cmd&z0=utf-8&net+start

```

Figure 7: Examples of malicious traffic families.

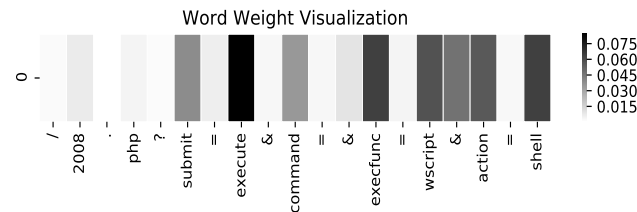


Figure 8: Visualization of attention. The depth of the color corresponds to the importance of the word in sentence given by model.

6 CONCLUSION

This paper presents DeepHTTP, a general-purpose framework for HTTP traffic anomaly detection and pattern mining using a deep neural network based approach. We construct a deep neural networks model utilizing Bidirectional Long Short-Term Memory (Bi-LSTM). This enables effective anomaly diagnosis. We design a novel method that can extract the structural characteristics of traffic. DeepHTTP learns content feature and structure feature of traffic automatically, and unearths considerable section of input data. It performs anomaly detection at per traffic entry level, and then mines pattern at cluster level. The intermediate output including

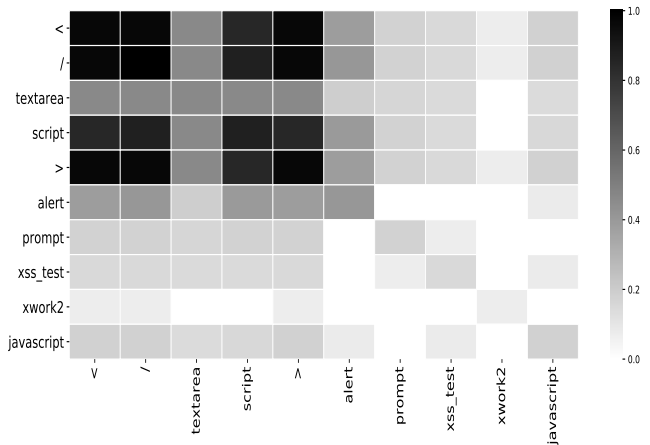


Figure 9: visualization of pattern mining.

attention hidden state and the attentional weight vector can be applied to clustering and pattern mining, respectively. By incorporating user feedback, DeepHTTP supports database update and model iteration. Hence it is able to incorporate and adapt to new traffic patterns. Experiments on large traffic datasets have clearly demonstrated the superior effectiveness of DeepHTTP compared with previous methods.

Future works include but are not limited to incorporating other types of deep neural networks into DeepHTTP to test their efficiency. Besides, improving the ability of the model to detect unknown malicious traffic is something we need to further study on in the future.

ACKNOWLEDGEMENT

This work is partially supported by the National Natural Science Foundation of China (U1736218).

REFERENCES

- [1] Mohiuddin Ahmed and Abdun Naser Mahmood. 2015. Network Traffic Pattern Analysis Using Improved Information Theoretic Co-clustering Based Collective Anomaly Detection. In *International Conference on Security and Privacy in Communication Networks*, Jin Tian, Jiwu Jing, and Mudhakar Srivatsa (Eds.), Springer International Publishing, Cham, 204–219.
- [2] F. Al-Obeidat and E. S. M. El-Alfy. 2017. Hybrid multicriteria fuzzy classification of network traffic patterns, anomalies, and protocols. *Personal & Ubiquitous Computing* 2 (2017), 1–15. <https://doi.org/10.1007/s00779-017-1096-z>

- [3] Tomasz Andrysiak, Andrzej Saganowski, Michał Choraś, and Rafał Kozik. 2014. Network Traffic Prediction and Anomaly Detection Based on ARFIMA Model. *Advances in Intelligent Systems & Computing* 299 (2014), 545–554. https://doi.org/10.1007/978-3-319-07995-0_54
- [4] Tahereh BABAIE, Sanjay Chawla, and Sebastien Ardon. 2014. Network Traffic Decomposition for Anomaly Detection. *CoRR abs/1403.0157* (2014). arXiv:1403.0157 <http://arxiv.org/abs/1403.0157>
- [5] Riccardo Bortolameotti, Thijs van Ede, Marco Caselli, Maarten H. Everts, Pieter Hartel, Rick Hofstede, Willem Jonker, and Andreas Peter. 2017. DECANter: Detection of Anomalous outbound HTTP Traffic by Passive Application Fingerprinting. In *Proceedings of the 33rd Annual Computer Security Applications Conference (ACSAC 2017)*. ACM, New York, NY, USA, 373–386. <https://doi.org/10.1145/3134600.3134605>
- [6] Tao-Wei Chiou, Shi-Chun Tsai, and Yi-Bing Lin. 2014. Network Security Management with Traffic Pattern Clustering. *Soft Comput.* 18, 9 (Sept. 2014), 1757–1770. <https://doi.org/10.1007/s00500-013-1218-0>
- [7] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, and Yoshua Bengio. 2015. Attention-Based Models for Speech Recognition. *CoRR abs/1506.07503* (2015). arXiv:1506.07503 <http://arxiv.org/abs/1506.07503>
- [8] Ronan Collobert and Jason Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, New York, NY, USA, 160–167. <https://doi.org/10.1145/1390156.1390177>
- [9] Marc Damashek. 1995. Gauging Similarity with n-Grams: Language-Independent Categorization of Text. 267, 5199 (1995), 843–848. <https://doi.org/10.1126/science.267.5199.843>
- [10] Meimei Ding and Hui Tian. 2016. PCA-based network Traffic anomaly detection. *Tsinghua Science & Technology* 21, 5 (Oct 2016), 500–509. <https://doi.org/10.1109/TST.2016.7590319>
- [11] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikanth. 2017. DeepLog: Anomaly Detection and Diagnosis from System Logs Through Deep Learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*. ACM, New York, NY, USA, 1285–1298. <https://doi.org/10.1145/3133956.3134015>
- [12] N. H. Duong and H. Dang Hai. 2015. A semi-supervised model for network traffic anomaly detection. In *2015 17th International Conference on Advanced Communication Technology (ICACT)*. 70–75. <https://doi.org/10.1109/ICACT.2015.7224759>
- [13] El-Sayed M. El-Alfy and Feras N. Al-Obeidat. 2014. A Multicriterion Fuzzy Classification Method with Greedy Attribute Selection for Anomaly-based Intrusion Detection. *Procedia Computer Science* 34 (2014), 55 – 62. <https://doi.org/10.1016/j.procs.2014.07.037> The 9th International Conference on Future Networks and Communications (FNC'14)/The 11th International Conference on Mobile Systems and Pervasive Computing (MobiSPC'14)/Affiliated Workshops
- [14] Sarah M. Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. 2016. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognition* 58 (2016), 121 – 134. <https://doi.org/10.1016/j.patcog.2016.03.028>
- [15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-based Algorithm for Discovering Clusters in a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, 226–231. <http://dl.acm.org/citation.cfm?id=3001460.3001507>
- [16] Juan M. Estévez-Tapiador, Pedro García-Teodoro, and Jesús E. Díaz-Verdejo. 2004. Measuring Normality in HTTP Traffic for Anomaly-based Intrusion Detection. *Comput. Netw.* 45, 2 (June 2004), 175–193. <https://doi.org/10.1016/j.comnet.2003.12.016>
- [17] Félix Iglesias and Tanja Zseby. 2015. Analysis of Network Traffic Features for Anomaly Detection. *Mach. Learn.* 101, 1-3 (Oct. 2015), 59–84. <https://doi.org/10.1007/s10994-014-5473-9>
- [18] Aruna Jamdagni, Zhiyuan Tan, Priyadarsi Nanda, Xiangjian He, and Ren Ping Liu. 2010. Intrusion Detection Using GSAD Model for HTTP Traffic on Web Services. In *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference (IWCMC '10)*. ACM, New York, NY, USA, 1193–1197. <https://doi.org/10.1145/1815396.1815669>
- [19] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. 2016. A Deep Learning Approach for Network Intrusion Detection System. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (Formerly BIONETICS) (BICT'15)*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 21–26. <https://doi.org/10.4108/eai.3-12-2015.2262516>
- [20] Antti Juvonen, Tuomo Sipola, and Timo Hämmäläinen. 2015. Online Anomaly Detection Using Dimensionality Reduction Techniques for HTTP Log Analysis. *Comput. Netw.* 91, C (Nov. 2015), 46–56. <https://doi.org/10.1016/j.comnet.2015.07.019>
- [21] Marius Kloft, Ulf Brefeld, Patrick Düessel, Christian Gehl, and Pavel Laskov. 2008. Automatic Feature Selection for Anomaly Detection. In *Proceedings of the 1st ACM Workshop on Workshop on AISeC (AISeC '08)*. ACM, New York, NY, USA, 71–76. <https://doi.org/10.1145/1456377.1456395>
- [22] Anukool Lakhina, Mark Crovella, and Christophe Diot. 2004. Characterization of Network-wide Anomalies in Traffic Flows. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement (IMC '04)*. ACM, New York, NY, USA, 201–206. <https://doi.org/10.1145/1028788.1028813>
- [23] Anukool Lakhina, Mark Crovella, and Christophe Diot. 2005. Mining Anomalies Using Traffic Feature Distributions. *SIGCOMM Comput. Commun. Rev.* 35, 4 (Aug. 2005), 217–228. <https://doi.org/10.1145/1090191.1080118>
- [24] S. Lawrence, C. L. Giles, Ah Chung Tsoi, and A. D. Back. 1997. Face recognition: a convolutional neural-network approach. *IEEE Transactions on Neural Networks* 8, 1 (Jan 1997), 98–113. <https://doi.org/10.1109/72.554195>
- [25] Tourneau T Le, A Millaire, P Asseman, Groote P De, C ThÃ©ry, and G Ducloux. 2015. Novel Approach for Network Traffic Pattern Analysis using Clustering-based Collective Anomaly Detection. *Annals of Data Science* 2, 1 (2015), 111–130. <https://doi.org/10.1007/s40745-015-0035-y>
- [26] Xiangang Li and Xihong Wu. 2014. Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition. *CoRR abs/1410.4281* (2014). arXiv:1410.4281 <http://arxiv.org/abs/1410.4281>
- [27] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. *CoRR abs/1508.04025* (2015). arXiv:1508.04025 <http://arxiv.org/abs/1508.04025>
- [28] Fenglong Ma, Radha Chitta, Jing Zhou, Quanzenq You, Tong Sun, and Jing Gao. 2017. Dipole: Diagnosis Prediction in Healthcare via Attention-based Bidirectional Recurrent Neural Networks. *CoRR abs/1706.05764* (2017). arXiv:1706.05764 <http://arxiv.org/abs/1706.05764>
- [29] M. V. Mahoney and P. K. Chan. 2003. Learning rules for anomaly detection of hostile network traffic. In *Third IEEE International Conference on Data Mining*. 601–604. <https://doi.org/10.1109/ICDM.2003.1250987>
- [30] Terry Nelms, Roberto Perdisci, and Mustaque Ahamad. 2013. ExecScent: Mining for New C&C Domains in Live Networks with Adaptive Control Protocol Templates. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*. USENIX, Washington, D.C., 589–604. <https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/nelms>
- [31] I. Paredes-Oliva, I. Castell-Uroz, P. Barlet-Ros, X. Dimitropoulos, and J. SolÃ- Pareta. 2012. Practical anomaly detection based on classifying frequent traffic patterns. In *2012 Proceedings IEEE INFOCOM Workshops*. 49–54. <https://doi.org/10.1109/INFOCOMW.2012.6193518>
- [32] Benjamin J. Radford, Leonardo M. Aponio, Antonio J. Trias, and Jim A. Simpson. 2018. Network Traffic Anomaly Detection Using Recurrent Neural Networks. *CoRR abs/1803.10769* (2018). arXiv:1803.10769 <http://arxiv.org/abs/1803.10769>
- [33] Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Jun Ma, and Maarten de Rijke. 2017. Leveraging Contextual Sentence Relations for Extractive Summarization Using a Neural Attention Model. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, New York, NY, USA, 95–104. <https://doi.org/10.1145/3077136.3080792>
- [34] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. 2007. Sensitivity of PCA for Traffic Anomaly Detection. *SIGMETRICS Perform. Eval. Rev.* 35, 1 (June 2007), 109–120. <https://doi.org/10.1145/1269899.1254895>
- [35] A. Samant and H. Adeli. 2010. Feature Extraction for Traffic Incident Detection Using Wavelet Transform and Linear Discriminant Analysis. *Computer-Aided Civil & Infrastructure Engineering* 15, 4 (2010), 241–250. <https://doi.org/10.1111/0885-9507.00188>
- [36] Mayank Swarnkar and Neminath Hubballi. 2016. OCPAD: One class Naive Bayes classifier for payload based anomaly detection. *Expert Systems with Applications* 64 (2016), 330–339. <https://doi.org/10.1016/j.eswa.2016.07.036>
- [37] Jiang tao Ren, Xiao ling Ou, Yi Zhang, and Dong cheng Hu. 2002. Research on network-level traffic pattern recognition. In *Proceedings. The IEEE 5th International Conference on Intelligent Transportation Systems*. 500–504. <https://doi.org/10.1109/ITSC.2002.1041268>
- [38] Elvis Tombini, Herve Debar, Ludovic Me, and Mireille Ducasse. 2004. A Serial Combination of Anomaly and Misuse IDSes Applied to HTTP Traffic. In *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04)*. IEEE Computer Society, Washington, DC, USA, 428–437. <https://doi.org/10.1109/CSAC.2004.4>
- [39] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems* 26, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2643–2651. <http://papers.nips.cc/paper/5004-deep-content-based-music-recommendation.pdf>
- [40] Ke Wang, Janak J. Parekh, and Salvatore J. Stolfo. 2006. Anagram: a content anomaly detector resistant to mimicry attack. In *International Conference on Recent Advances in Intrusion Detection*. Springer, Berlin, Heidelberg, 226–248. https://doi.org/10.1007/11856214_12
- [41] Ke Wang and Salvatore J. Stolfo. 2004. Anomalous Payload-Based Network Intrusion Detection. In *Recent Advances in Intrusion Detection*, Erlend Jonsson, Alfonso Valdes, and Magnus Almgren (Eds.). Springer Berlin Heidelberg, Berlin,

- Heidelberg, 203–222.
- [42] Yi Xie and Xiangnong Huang. 2010. HTTP-session Model and Its Application in Anomaly HTTP Traffic Detection. In *Sixth International Conference on Semantics Knowledge and Grid*. 141–148.
 - [43] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. 2048–2057.
 - [44] Jun Zhang and K. F. Man. 1998. Time series prediction using RNN in multi-dimension embedding phase space. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, Vol. 2. 1868–1873 vol.2. <https://doi.org/10.1109/ICSMC.1998.728168>
 - [45] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In *ACL*.
 - [46] Yujun Zhou, Changliang Li, Bo Xu, Jiaming Xu, Jie Cao, and Bo Xu. 2017. Hierarchical Hybrid Attention Networks for Chinese Conversation Topic Classification. In *Neural Information Processing*, Derong Liu, Shengli Xie, Yuanqing Li, Dongbin Zhao, and El-Sayed M. El-Alfy (Eds.), Springer International Publishing, Cham, 540–550.
 - [47] M. Zolotukhin, T. Härmäläinen, T. Kokkonen, and J. Siltanen. 2014. Analysis of HTTP Requests for Anomaly Detection of Web Attacks. In *2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing*. 406–411. <https://doi.org/10.1109/DASC.2014.79>