Visual Analytics for Anomaly Classification in LAN Based on Deep Convolutional Neural Network

Yuwei Sun
Graduate School of Information Science and Technology
The University of Tokyo
Tokyo, Japan
sywtokyo@hongo.wide.ad.jp

Hiroshi Esaki
Graduate School of Information Science and Technology
The University of Tokyo
Tokyo, Japan
hiroshi@wide.ad.jp

Hideya Ochiai
Graduate School of Information Science and Technology
The University of Tokyo
Tokyo, Japan
jo2lxq@hongo.wide.ad.jp

***Abstract*— Information systems accelerate the advancement of society. However, malicious manipulation of information would bring great harm. Recently, criminal groups are increasingly involved in cybercrime, especially in the Local Area Network (LAN). Several methods are being used to analyze network traffic in LAN such as extracting the transition patterns in traffic flows, however, research on visualization of network traffic, thus detecting and classifying various abnormal events, is insufficient. In this research, we propose visual analytics for generating feature maps of network events based on protocol information. We extract protocol information of ARP, TCP, and UDP from network traffic and generate each type of feature maps. Then for each event, we merge these three types into one image by putting them into different channels, to represent features. We simulate and visualize eight types of network events in LAN which are the normal, arp scan, tcp scan, scan of tcp port 23, scan of tcp port 80, udp scan, scan of udp port 137 and scan of udp port 1900. Then for this multiclass classification problem, we adopt a deep convolutional neural network (CNN) to differentiate between these network events, with these eight types as labels and generated feature maps as inputs. We evaluated the scheme using precision, recall, and F-measure in two LANs, at last, achieving an average F-measure of 0.76.**

Keywords—visual analytics, LAN, cybersecurity, machine learning, convolutional neural network

I. INTRODUCTION

The implementation of resilient and robust networks is continually imperative. However, an effective and explainable solution to network anomaly detection has not been made clear. Especially, attacks such as phishing, delivering and spreading malware in networks by spammers have been monitored frequently in recent years. At the same time, network systems are becoming so complicated that even the manager of networks has a problem with manipulating them.

General approaches being used for detection usually have a limit to extremely changeable malware and are short of explainability. Therefore, it is considerable and meaningful to represent hidden features of network traffic using visual analytics, thus we can observe and detect anomaly from visualization results. Moreover, through a combination with machine learning, automatic classification between various abnormal events can be considered.

In this research, we adopt channel-based visual analytics based on the Hilbert curve to represent protocol information of network events. Each feature map of network events consists of three channels, which are used to save information about a specific protocol during a predetermined recording period (Fig. 1). The protocols used in this research are ARP, TCP, and UDP. Consequently, we use these feature maps to represent network traffic features of different network events.

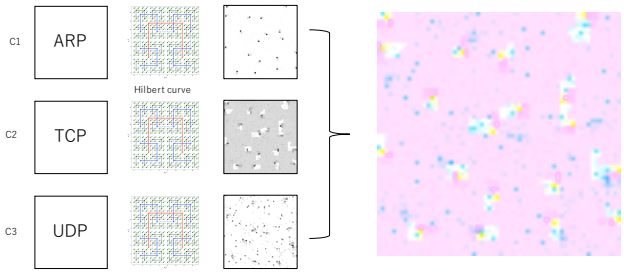

Fig. 1. The scheme of generating a feature map from network traffic using the channel-based visual analytics (under a case of arp scan).

We simulate eight types of known network events in LAN, using a terminal connected to this LAN as the attacker who manipulates various commands, and another terminal to collect all network traffic flowing through the LAN. Then we generate feature maps of these network events using the approach discussed above.

After that, we adopt a deep CNN model based on the VGG-16 which consists of thirteen convolutional layers accompanied by five maxpooling layers and three fully connected layers. We use several optimizing approaches to machine learning to stabilize the progress of training as well as improving performance of this model. We train this model from scratch using generated feature maps. It is supposed that this CNN model could give out confidence of each network event cluster, thus identifying the type of a network event based on confidence.

This paper is organized as follows. Section 2 discusses related work about visualization of network traffic and anomaly detection in networks based on machine learning. Section 3 provides an overview of the scheme, consisting of generating feature maps and implementing the CNN model. Section 4 presents performance evaluation using precision, recall, and F-measure in two networks. Section 5 discusses the advantages and disadvantages of this scheme. Section 6 concludes the paper.

II. RELATED WORK

The research of anomaly detection in networks with machine learning has a long history. In previous research, a non-linear technique called support vector machine (SVM) is used to explore hidden relations between time-series traffic data. For instance, Palmieri et al. [1] have proposed a method to extract the transition patterns based on the analysis of non-stationary properties and hidden recurrence patterns occurring in the aggregated IP traffic flows. Moreover, Hsu and Lin [2], they demonstrated an approach of using SVM models to detect DDoS attacks in an efficient and accurate way.

978-1-7281-9331-1/20/$31.00 ©2020 IEEE

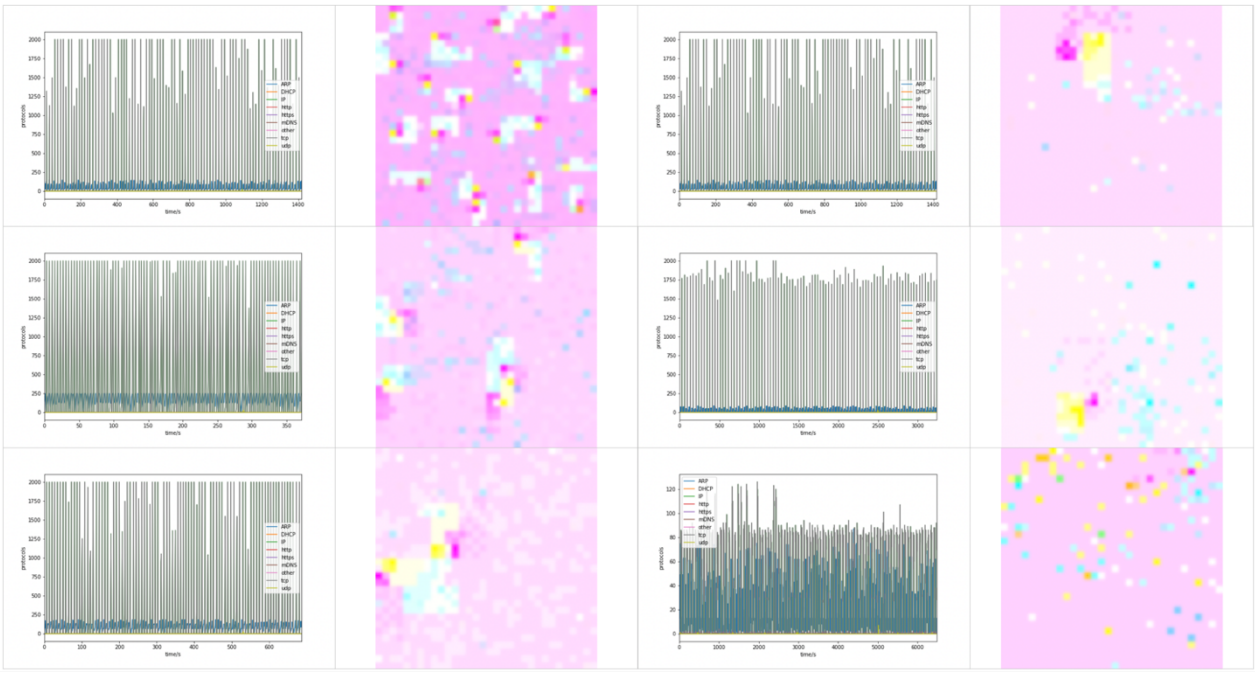

Fig. 3 The record of traffic data under arp scans and the corresponding feature maps when adopting different fineness. Col 1: from top to bottom, when the fineness is 5, 1, and 0.5; col 2: from top to bottom, when the fineness is 0.1, 0.05, and 0.01.

Besides SVM, there are also other machine learning methods used to solve network anomaly detection. For instance, in the research of Mehdi et al. [3], they used Deep Learning (DL) to extract and analyze the features of IoT domains. In addition, Krokos et al. [4] adopted an autoencoder to process raw pcap files to Query-Space visualization, revealing potential DDoS attacks and unforeseen changes in the distribution of queries received.

The research discussed above is aimed to detect anomaly in networks from the perspective of machine learning. However, most of them lack enough explainability regarding feature representation of network traffic. In this research, we are aimed to visualize features of network traffic to achieve more intuition of anomaly in LAN, by exploring a hidden relationship among various protocols of network traffic. We adopt a supervised machine learning called the convolutional neural network for further classifying different abnormal events.

III. Visual Analytics for Feature Maps Generation

A. Visualization of network traffic in LAN with Hilbert curve

We adopt two active network environments here and implement simulations of various network events in each network for collecting traffic data. In detail, one terminal is used as a simulator for generating various network events in LAN and another one is used to collect traffic data when simulating a specific event (Fig. 2).

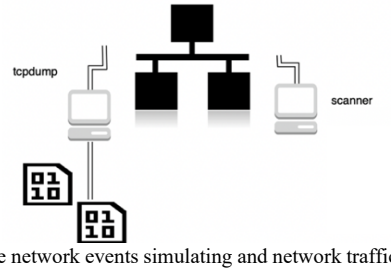

Fig. 2. The network events simulating and network traffic collecting system.

In this research, we use a tool called tcpdump for collecting the traffic data in LAN. And for simulating these network events, we adopt a tool called nmap with several types of manipulating commands. Moreover, simulations of each type of network event is conducted continually for two whole days. From this collected traffic data, we extract all protocol information and time stamp information to further visualize features of network traffic.

We compute how many communications of each protocol have been monitored in LAN during a fixed recording time unit, which is influenced by the value of fineness, as discriminators for representing features of network traffic. Considering the network events we experiment on in this research, from all protocol information, communication information about ARP, TCP, and UDP is extracted and computed to represent features. Then, we convert frequency information of these protocols into pixel values using (1), for visualization with feature maps.

$$p_i = \frac{c_i}{Max(c)} \times 255 \quad (1)$$

Where c_i is the frequency of a specific type of protocols' communication, and the denominator is the maximum from all frequency values within the duration for generating a feature map, and p_i is the pixel value of a pixel point in an image.

Here, the parameter of fineness mentioned above is used to show how precisely we compute the frequency of protocol communication. For a discussion about the influence of fineness on the representation result of network traffic's features, we verify results of visualization when fineness has a value of 5.0, 1.0, 0.5, 0.2, 0.1, 0.05, 0.02, and 0.001(Fig. 3). In this research, we found that when adopting fineness with a value of 0.5, in which case the fixed recording time unit is 0.5 seconds, the representation of network traffic's features shows the best result.

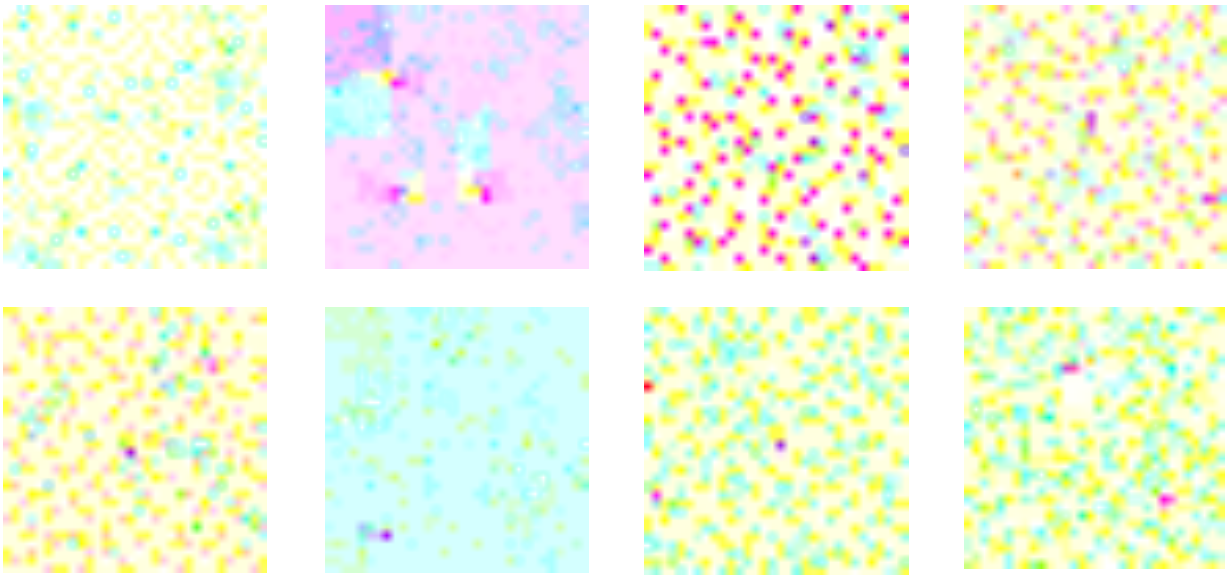

Fig. 5 Feature maps of network events generated with a fineness value of 0.5. Line 1, from left to right: feature maps of normal, arp scan, tcp scan and scan of tcp port 23; Line 2, from left to right: feature maps of tcp port 80, udp scan, scan of udp port 137 and scan of udp port 1900. (in LAN A)

Then, we adopt a geometric structure called Hilbert curve to project these pixel values converted from communication frequency information into different pixel points in a feature map (Fig. 4). Here, the merit of Hilbert curve is that we can suppress time-related data into a 2-Dimension image, amplifying the hidden relationship between data. As discussed above, a fineness value of 0.5 is adopted, thus each feature map consists of 1024 records, each of which shows features of a specific protocol's communication within 0.5 seconds.

After projecting every pixel value to the corresponding positions in a feature map image, we combine these three feature maps of ARP, TCP and UDP into one feature map through putting each of them into different channels of an image, merging all three types of protocol information and representing the features of a network event's traffic data with this combined feature map (Fig. 1).

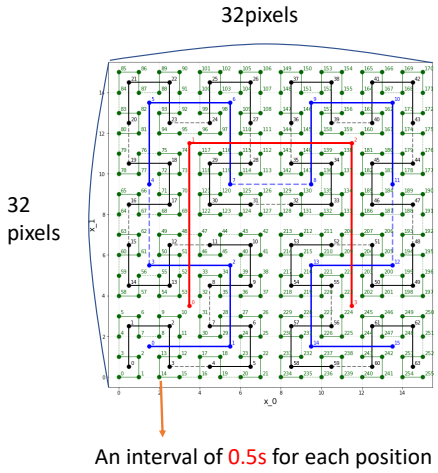

Fig. 4. The geometric structure of Hilbert curve, where frequency information is projected into each position of it through pixel values. Moreover, every four neighboring pixel points are represented with the central point. For example, all black points can be considered as a compressed representation of all information. Blue points can be considered as a compressed representation of all information in black points. As such, the four red points in this structure can be used to represent all information saved in this image.

B. *Dataset*

Network traffic from two network environments, LAN A and LAN B, was collected and used for generating feature maps. Here, LAN A is a network with a variable-length subnet mask with a length of 25 digits. And it is a network of the institute's critical infrastructures. On the other hand, LAN B is a network serving for general purposes in several labs, such as research and other daily operations.

In this research, we simulated eight types of network events in LAN A and LAN B separately, including normal, arp scan, tcp scan, scan of tcp port 23, scan of tcp port 80, udp scan, scan of udp port 137, and scan of udp port 1900. Then we collected network traffic from these two networks, generating feature maps of network events based on the approach discussed above (Fig. 5). At last, we achieved 2000 feature maps in LAN A and 2800 feature maps in LAN B in total.

IV. Network Events Classification using Deep Convolutional Neural Network

A. *Adaptivity of convolution operation to feature maps*

The convolution operation corresponds to the filter operation in image processing. It computes the convolution of the input data and the filter as the output. As a result, it has great adaptivity to the geometric structure of Hilbert curve. Furthermore, results from multiple channels are added up to obtain the output, adaptable to this channel-based feature representation of network events (Fig. 6).

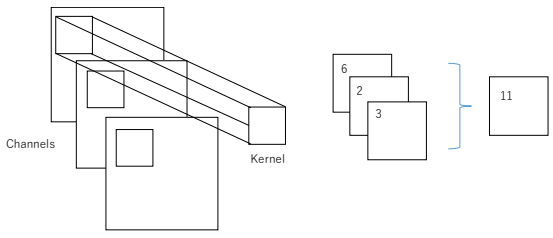

Fig. 6. The function of convolution operation, where a kernel is used to compute the convolution of each area of the input data and the filter.

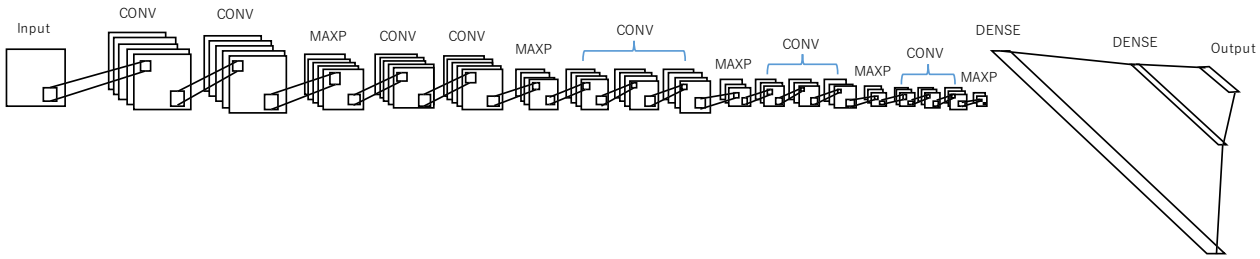

Fig. 7. The structure of the deep CNN model we adopt in this research, consisting of thirteen convolutional layers accompanied by five maxpooling layers and three fully connected layers.

B. The structure of the deep CNN model

Considering the task of multiclass classification between eight types of network events using generated feature maps, we adopt a deep CNN model in order to identify subtle differences hidden in the visualized network traffic data. At the same time, we utilize several optimization methods of deep learning to stabilize the progress of model training, as well as improving the performance of this model.

Moreover, CNN is one kind of supervised learning. Usually, a CNN model consists of several convolution layers, sometimes accompanied by pooling layers, and several fully connected layers at the end of the model. Moreover, the pooling operation in the pooling layer is used for aggregating features and reducing the size of the input in the vertical and horizontal directions. Furthermore, fully connected layers are used to narrow the output to a specific range, which consists of numerical labels of the input data. In this research, we transfer the eight types into numerical vectors as the labels to train the model.

We build a deep CNN model based on the VGG-16 model [5], consisting of thirteen convolution layers accompanied by five maxpooling layers and three fully connected layers, with an output of eight values (Fig. 7). By adopting this deep CNN model, the hidden features of network events inside a feature map can be extracted. Then three fully connected layers are used to flatten the output matrix of the convolution layers and maxpooling layers to a one-dimension array and compress information to a matrix, which gives out the confidence percentage of this model's prediction regarding each type of network events.

Moreover, the activation function used in this research is ReLU which is defined as (2). After the computation of each layer, we use ReLU function to convert the output to a non-linear distribution. And at the last layer, for this multiclass problem, we use an activation function called softmax, which is defined as (3). Through using softmax instead, each component including negative, greater than one, or might not sum to 1, will be in the interval (0, 1), with a sum of 1.

$$f(x) = max(0, x) \tag{2}$$

Where x is input, f(x) is an output of the node, and "max" is a function used to get a maximum between 0 and x.

$$y_i = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}} \tag{3}$$

Where x_i represents each element of the input vector x. Softmax is used to normalize these values through dividing by the sum of all the exponentials of these elements.

In order to speed up and stabilize the training progress, the learning function called RMSprop is used, where the emphasis is placed on the latest gradient information more than the past gradient information and gradually the past gradient information is forgotten, instead, the new gradient information is greatly reflected. This learning function is defined as (4) (5).

$$h_t = \rho * h_{t-1} + (1-\rho) * \frac{\partial L}{\partial W_t} \odot \frac{\partial L}{\partial W_t} \tag{4}$$

$$W_{t+1} = W_t - \eta \frac{1}{\sqrt{h_t + \epsilon}} \odot \frac{\partial L}{\partial W_t} \tag{5}$$

Where L is the loss, W is the weight of the node, and ρ is the decay rate with a value of 0.9 here.

This deep CNN model was trained from scratch based on the datasets from LAN A and LAN B separately, with a batch size of 40 and a learning rate of 0.00002. The early stopping was adopted while training to prevent the overfitting, by monitoring the validation loss for every five epochs. After training, the model is supposed to identify eight types of network events in LAN based on confidence scores of the model prediction. The corresponding training graphs in the two network environments are shown below (Fig. 8) (Fig. 9).

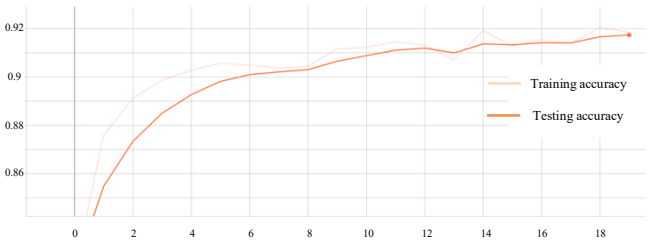

Fig. 8. The training accuracy and the testing accuracy in LAN A.

TABLE I. EVALUATION OF THE SCHEME IN TWO DIFFERENT NETWORK ENVIRONMENTS

Events	LAN A			LAN B		
	Precision	***Recall***	***F-measure***	***Precision***	***Recall***	***F-measure***
normal	0.8384	0.7829	0.7938	0.8563	0.8108	0.8273
scan (arp)	0.9935	0.9922	0.9921	0.8276	0.8084	0.8161
scan (tcp)	0.9123	0.7122	0.7822	0.9945	0.9964	0.9952
scan (tcp port 23)	0.5790	0.6184	0.5686	0.7578	0.5239	0.6059
scan (tcp port 80)	0.6519	0.6307	0.6122	0.3613	0.3324	0.3103
scan (udp)	0.6340	0.8442	0.7058	0.9900	1.0000	0.9945
scan (udp port 137)	0.9097	0.4766	0.6023	0.4294	0.8875	0.5600
scan (udp port 1900)	0.6145	0.9370	0.7236	0.8669	0.7308	0.7664
Average	0.7667	0.7493	0.7576	0.7605	0.7613	0.7597

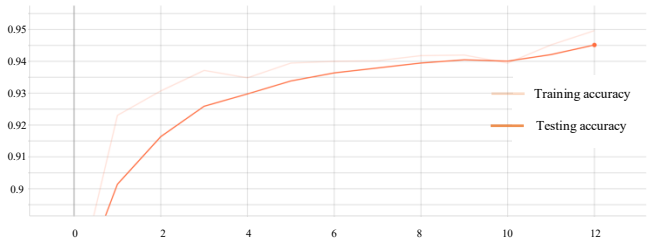

Fig. 9. The training accuracy and the testing accuracy in LAN B.

V. PERFORMANCE EVALUATION

We evaluate the scheme in two active networks, LAN A and LAN B, each of which includes the system structure we mentioned before, with a terminal for simulation of network events and another one for collecting network traffic in LAN. We evaluate the performance of it using precision, recall, and F-measure.

The precision is a parameter used to show how many events are successfully classified in all test data; the recall is a parameter that is used to show how many times a specific event is successfully classified in all test data of that event. They are defined as (6) and (7).

$$Precision = \frac{TP}{TP+FP} \tag{6}$$

$$Recall = \frac{TP}{TP+FN} \tag{7}$$

Where TP (True Positives) indicates the number of events successfully detected by the scheme, and FN (False Negative) represents the number of events unsuccessfully classified.

On the other hand, the F-measure is a parameter used to show the comprehensive performance of a model. And it is defined as (8).

$$F-measure = \frac{2\times Precision\times Recall}{Precision+Recall} \tag{8}$$

While training, we adopted the evaluation methods above for each epoch. We compute the precision, recall, and F-measure scores of each event as the measure of the classification ability of the scheme. The corresponding result of the evaluation is shown above (TABLE I).

From the result above, we can see that the classification between normal, arp scan, tcp scan, and udp scan shows relatively great performance. Whereas, the classification between scans of some specific ports such as tcp port 80 and udp port 137 comes out relatively low F-measure scores, which shows that it is more difficult to classify between scans of specific ports than to classify between the normal and the abnormal in LAN. At last, an average F-measure score of 0.76 is achieved for this anomaly classification in LAN.

VI. DISCUSSION

The visualization of network traffic allows some explainability to anomaly detection in LAN. Furthermore, a deep CNN model is adopted to classify these reoccurring patterns in feature maps of various network events. And simulation experiments under two different networks are conducted to evaluate the scheme.

On the other hand, it is still possible that an adversary could forge these features inside a feature map by adjusting the communication frequency. A more delicate and complete experiment in a real-world setting should be verified. Furthermore, besides the proposed eight types of network events in this research, how additional, non-explicit network events would influence the classification result should be considered.

VII. CONCLUSION

In this research, we propose channel-based visual analytics, which extracts protocol information of ARP, TCP, and UDP from network traffic, generating feature maps of eight types of network events in LAN using this information. We are aimed to visualize features of network traffic by representing a relationship between communications of protocols based on Hilbert curve. We adopt a deep CNN model to approach this multiclass classification, conducting performance evaluation using the precision, recall, and F–measure in two different network environments. As a result,

the scheme achieves an average F-measure score of 0.76 for the anomaly classification in LAN.

REFERENCES

[1] F. Palmieri, U. Fiore, Network anomaly detection through nonlinear analysis, Comput. Secur. 29 (7), 737–755, 2010.

[2] Hsu Chih-Wei, Lin Chih-Jen, A comparison of methods for multiclass support vector machines, IEEE Trans. Neural Netw. 13 (2), 415–425, 2002.

[3] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, Mohsen Guizani, Deep Learning for IoT Big Data and Streaming Analytics: A Survey, IEEE COMMUNICATIONS SURVEYS & TUTORIALS, 2018.

[4] Eric Krokos, Alexander Rowden, Kirsten Whitley, and Amitabh Warshney, "Visual Analytics for Root DNS Data", IEEE, 2018.

[5] Simonyan, K., Zisserman, A., Very deep convolutional networks for large-scale image recognition, ICLR 2015.

[6] Yann LECun, Patrick Haffner, Leon Bottou, and Yoshua Bengio, Object Recognition with Gradient-Based Learning, Shape, Contour and Grouping in Computer Vision, p.319, 1999.

[7] Kazumasa Y amauchi, Junpei Kawamoto, Y oshiaki Hori, Kouichi Sakurai, Evaluation of Machine Learning Techniques for C&C Traffic Classification, Information Processing Society of Japan Vol.56 No.9 1745–1753, 2015.

[8] F. Palmieri, U. Fiore, A nonlinear, recurrence-based approach to traffic classification, Comput. Networks 53 (6), 761–773, 2009.

[9] S. Lau, The spinning cube of potential doom, Communications of the ACM, 47(6), 25–26, 2004.

[10] J. Erman, M. Arlitt, and A. Mahanti, Traffic classification using clustering algorithms, MineNet'06, pp. 281–286, 2006.

[11] H. B. McMahan, E. Moore, D. Ramage, S. Hampson et al., "Communication-efficient learning of deep networks from decentralized data", arXiv:1602.05629, 2016.

[12] Samaneh Mahdavifar, Ali A. Ghorbani, Application of deep learning to cybersecurity: A survey, Neurocomputing 347, 149–176, 2019.

[13] Eric Ke Wang,Yunming Ye, Xiaofei Xu, S.M.Yiu, L.C.K.Hui, K.P.Chow, Security Issues and Challenges for Cyber Physical System, IEEE/ACM International Conference on Green Computing and Communications & IEEE/ACM International Conference on Cyber, Physical and Social Computing, 2010.

[14] Sommer, Robin, Paxson, Vern, Outside the Closed World: On Using Machine Learning for Network Intrusion Detection, IEEE Symposium on Security and Privacy, 305-316. 10.1109/SP.2010.25., 2010.

[15] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, Asaf Shabtai, Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection, Network and Distributed Systems Security Symposium (NDSS), 2018.

[16] Yuwei Sun, Hideya Ochiai, Hiroshi Esaki. Detection and Classification of Network Events in LAN Using CNN. IEEE International Conference on Information Technology, 2019.