Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Web traffic anomaly detection using C-LSTM neural networks

Tae-Young Kim, Sung-Bae Cho*

Department of Computer Science, Yonsei University, Seoul, Republic of Korea

A R T I C L E   I N F O

Article history:
Received 16 January 2018
Revised 3 April 2018
Accepted 4 April 2018
Available online 5 April 2018

Keywords:
Web traffic
Anomaly detection
Deep learning
C-LSTM

A B S T R A C T

Web traffic refers to the amount of data that is sent and received by people visiting online websites. Web traffic anomalies represent abnormal changes in time series traffic, and it is important to perform detection quickly and accurately for the efficient operation of complex computer networks systems. In this paper, we propose a C-LSTM neural network for effectively modeling the spatial and temporal information contained in traffic data, which is a one-dimensional time series signal. We also provide a method for automatically extracting robust features of spatial-temporal information from raw data. Experiments demonstrate that our C-LSTM method can extract more complex features by combining a convolutional neural network (CNN), long short-term memory (LSTM), and deep neural network (DNN). The CNN layer is used to reduce the frequency variation in spatial information; the LSTM layer is suitable for modeling time information; and the DNN layer is used to map data into a more separable space. Our C-LSTM method also achieves nearly perfect anomaly detection performance for web traffic data, even for very similar signals that were previously considered to be very difficult to classify. Finally, the C-LSTM method outperforms other state-of-the-art machine learning techniques on Yahoo's well-known Webscope S5 dataset, achieving an overall accuracy of 98.6% and recall of 89.7% on the test dataset.

1. Introduction

As Internet technology develops and computers become more popular, the importance of computer networks is increasing. Massive infrastructure based on complex networks and internet of things (IoT) technology has a significant impact on society and the economy (Kim & Cho, 2017; Ronao & Cho, 2017). A large amount of information is exchanged through web servers and a variety of services are provided (Huang & Huang, 2013). However, with the increase in internet services, malicious attacks through networks are gradually becoming more advanced and diversified. Various network attacks can cause serious damage to web service operation, leading to social and economic losses (Ahmed, Mahmood, & Hu, 2016; Ronao & Cho, 2016a). Table 1 lists several types of common network attacks. Proactive management and prevention of various attacks that threaten network infrastructure are essential (Jiang, Xu, Zhang, & Zhu, 2014).

Detection of traffic anomalies in web servers is a univariate time-series classification problem. Using a specific window of the primary sensor signal, one can extract differentiating features to recognize activity by using classifiers (Zheng, Liu, Chen, Ge, & Zhao, 2014). However, it is very difficult to detect abnormal patterns using statistical approaches because web traffic has different characteristics depending on the type of service provided and user connection patterns, and the distribution of patterns is very irregular (Yu, Liu, Zhou, & Yu, 2014). Anomalies in these irregular patterns can be categorized in three ways. Fig. 1 presents anomaly forms from the three different categories (Ahmed & Mahmood, 2014).

Fig. 1(a) contains a point anomaly and refers to an instance in which an individual data sample is considered abnormal in relation to the surrounding data. Fig. 1(b) contains a contextual anomaly and refers to cases where a data instance is abnormal in a specific context, but not otherwise. Fig. 1(c) contains a collective anomaly, where the collection of related data instances is exceptional as a whole, even though individual values may be normal. Fig. 2 presents a diagram of attack-type and anomaly-type mapping.

In addition, these abnormal patterns may contain both local and global abnormalities. Although global anomalies can be easily identified by visual inspection, local anomalies have patterns similar to that of a normal signal, meaning detection based on outliers is difficult (Goldstein & Uchida, 2016). Fig. 3 presents two plots of web traffic anomaly patterns. Fig. 3(a) shows a local anomaly where the anomaly exists inside the traffic and Fig. 3(b) shows a global anomaly in which the anomaly exists outside the traffic.

Yahoo in the United States provides a dataset consisting of 367 time series, each of which consists of 1500 data points, for a total of 5,050,000 data points (https://research.yahoo.com/). The dataset contains four classes: A1, A2, A3, and A4. The classes contain 67,

* Corresponding author.
E-mail addresses: taeyoungkim@yonsei.ac.kr (T.-Y. Kim), sbcho@yonsei.ac.kr (S.-B. Cho).

https://doi.org/10.1016/j.eswa.2018.04.004
0957-4174/© 2018 Elsevier Ltd. All rights reserved.

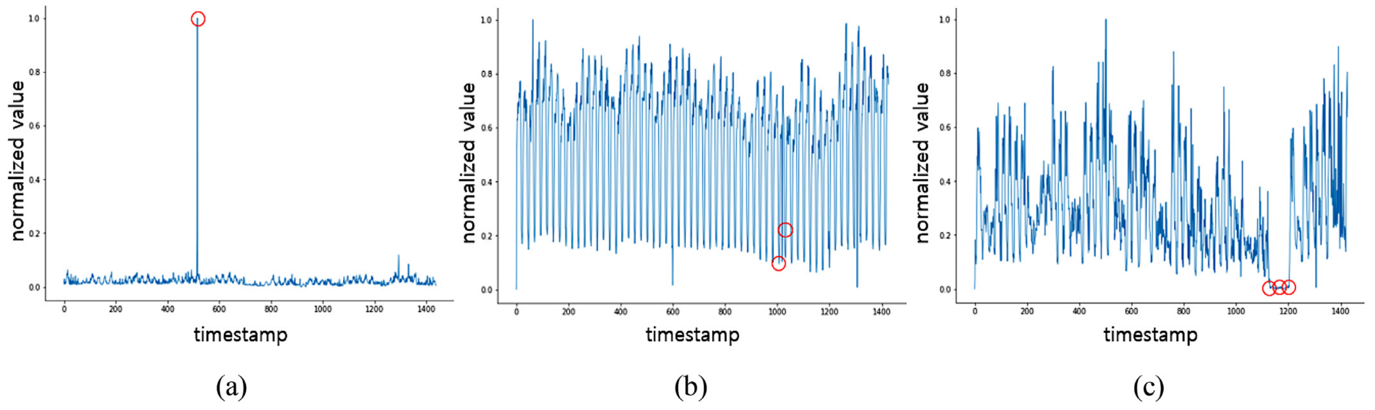

Fig. 1. Three types of anomaly categorization.

Table 1
Types of network attacks.

Network attack method	Description
Denial of Service (DoS)	A malicious attack on a system that causes the system to consume extra resources, which negatively impacts its intended use.
Probe	Collects information about a target network or host and checks which devices are connected to the network.
User to Root (U2R)	An attempt to illegally access a managed account to modify, manipulate, or exploit a client's critical resources.
Remote to User (R2U)	An attempt to obtain local user access on a target computer and gain permission to send packets over the network.

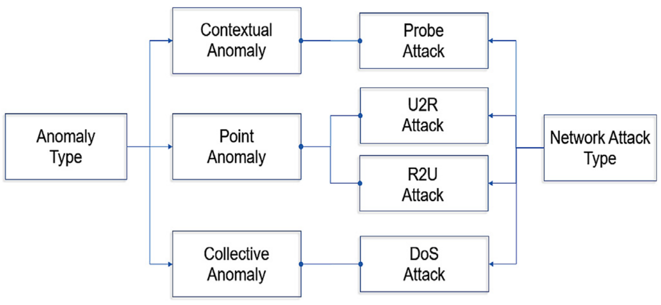

Fig. 2. Anomaly and attack type mapping.

Table 2
Characteristics of the Yahoo Webscope S5 data.

Class	Real traffic	Synthetic traffic	Total length	Total anomalies
A1	O	X	94,866	1669
A2	X	O	142,100	466
A3	X	O	168,000	943
A4	X	O	168,000	837

100, 100, and 100 files, respectively (Laptev & Amizadeh, 2015). Class A1 contains traffic data from actual web services, but classes A2, A3 and A4 contain synthetic anomaly data with increasing levels of complexity (Thill, Konen, & Bäck, 2017). Table 2 contains the characteristics of the Yahoo Webscope S5 data.

Fig. 4 presents a statistical analysis of the A1 class of the Yahoo Webscope S5 dataset. The A1 class consists of 67 real web traffic files and each file has a different distribution of traffic. Fig. 4(a) is a standard deviation graph based on the average of the normalized amount of data in each file. From this graph, it can be seen that traffic for various services has different distributions. Fig. 4(b) shows the number of outliers and anomalies calculated in each file. From this figure, we can see that it is hard to predict actual anomalies by simply detecting statistical outliers. Therefore, it is very difficult to perform anomaly detection using statistical analysis techniques on files with different distributions.

Recently, deep learning has been actively studied for image and signal processing applications. It is a technique that finds key func-

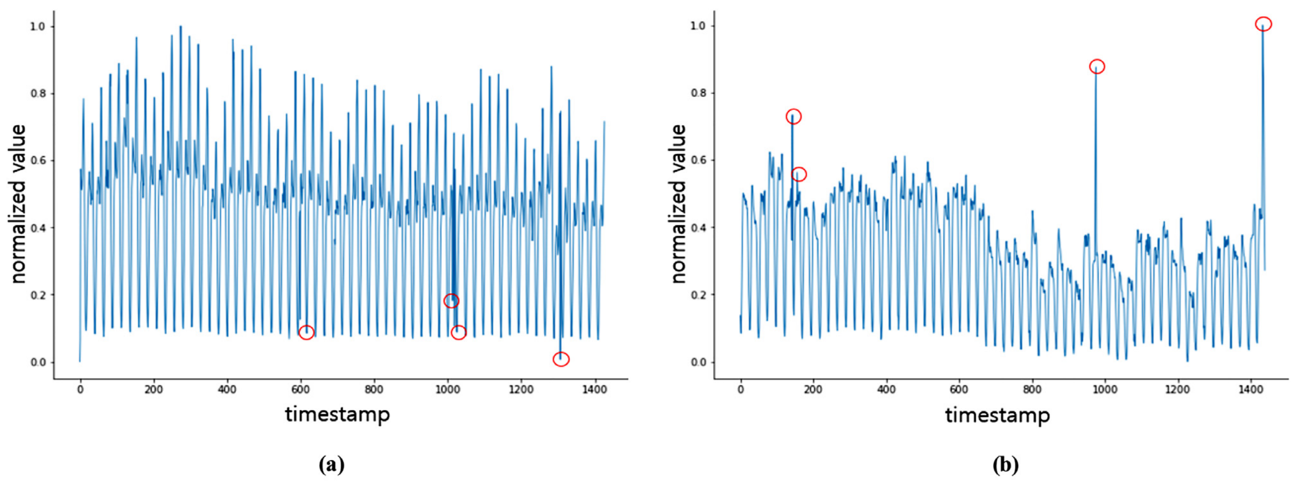

Fig. 3. Example plots of two patterns of web traffic anomalies.

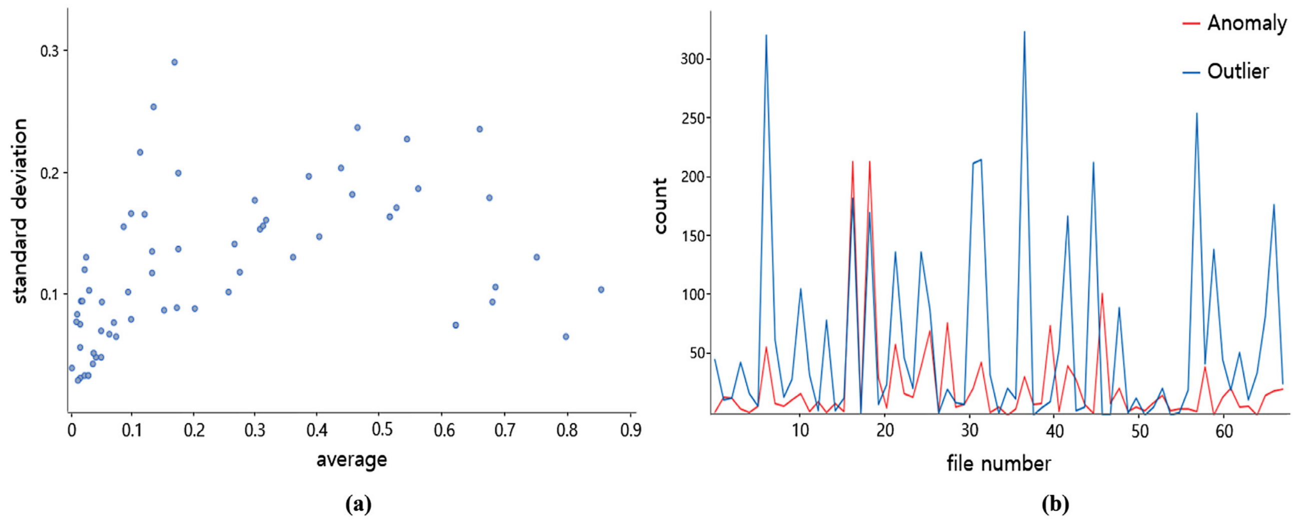

Fig. 4. Statistical graphs for Yahoo Webscope S5 dataset A1 class.

tions in a large amount of data or very complex data through a combination of several nonlinear transformation methods (Chen & Lin, 2014). Deep learning is best represented by two algorithms: convolutional neural networks (CNNs) for image recognition and recurrent neural networks (RNNs), which are mainly used for natural language processing and speech recognition (Donahue et al., 2015). CNNs have a local receptive field and shared weight kernel, which can reflect spatial characteristics by extracting basic visual features, such as oriented edges, end-points, and corners (LeCun, Bottou, Bengio, & Haffner, 1998). An RNN has a very deep structure that connects basic neural units in chronological order and is typically effective for modeling sequential data by learning using gate units such as long short-term memory (LSTM) units (Sak, Senior, & Beaufays, 2014).

Also, the combination of CNN and LSTM layers is being studied to extract temporal and spatial features (Zhou, Hu, Chen, & Wang, 2018). Since speech recognition and natural language processing have temporal and spatial information, the combination of CNN and LSTM can effectively extract features (Xu, Kong, Huang, Wang, & Plumbley, 2017). Currently, these advantages are being applied to classifying and predicting sensor data occurring in the industrial domain (Oehmcke, Zielinski, & Kramer, 2018).

In this paper, we propose a C-LSTM neural network that combines a CNN and RNN for automatic feature extraction and detection from web traffic signals. Web traffic data is recorded over time and contains specific patterns of spatial and temporal information. By recognizing such spatial-temporal information patterns, an administrator can classify normal and abnormal patterns occurring in traffic (Paschalidis & Smaragdakis, 2009).

Our proposed C-LSTM reduces the data spectrum by transforming the temporal context using a relatively simple CNN layer. The output of this CNN layer is used as the input for several LSTM layers to reduce temporal variations. The output of the final LSTM layer is then fed into several fully connected DNN layers, making it easier to classify the output by adding functionality to the data space (Greff, Srivastava, Koutník, Steunebrink, & Schmidhuber, 2017). Finally, we combine multiple scales of information to explore whether or not further improvements can be made (Sermanet & LeCun, 2011). In particular, we explore how to extract spatial features from time-series data by using a CNN. Passing these features through the LSTM helps us to identify how temporal modeling of spatial characteristics in data affects performance. We also explore the complementarity among the CNN, LSTM, and DNN layers.

The proposed method is evaluated by using a 1D sequence traffic input signal, as shown in Fig. 5, and the spatial and temporal features in the sequence are extracted to achieve high performance. This is the first time a C-LSTM has been designed and trained for anomaly detection in web traffic. We also provide a basis for the empirical rules and principles of C-LSTM architecture design for detecting anomalies in web traffic.

The remainder of this paper is organized as follows. In Section 2, we discuss the related work on web traffic anomaly detection. Section 3 details the proposed C-LSTM neural network architecture. Section 4 presents experimental results and Section 5 concludes the paper.

2. Related works

As presented in Table 3, many researchers have studied the classification of normal and abnormal patterns by extracting data characteristics in the field of abnormal sequence detection, such as web traffic. Sequence anomaly detection approaches can be divided into three categories: statistical modeling, temporal feature modeling, and spatial feature modeling.

Münz et al. used a K-means clustering algorithm to perform anomaly detection in network traffic data (Münz, Li, & Carle, 2007). They calculated the centroid of a cluster by analyzing the statistical characteristics of real data and performed anomaly detection by calculating the distance between a centroid and traffic value. As a result, they obtained high classification performance for outliers with large Euclidean distances between normal and abnormal sequences. Zhang and Zulkernine used a random forest method to extract outlier patterns based on unsupervised learning (Zhang & Zulkernine, 2006). They proposed a mathematical criterion to distinguish between normal data and outliers using statistical algorithms. Through this process, outliers were discovered using decision trees and anomaly detection was performed through an ensemble technique. These methods achieve good performance at classifying statistical anomalies in web traffic data. However, there is a disadvantage in that they cannot properly classify abnormal traffic data with the same distribution as normal traffic data.

Cheng et al. used preprocessed network traffic data with a sliding window algorithm. They extracted temporal information by using an LSTM model on the preprocessed data (Cheng et al., 2016).

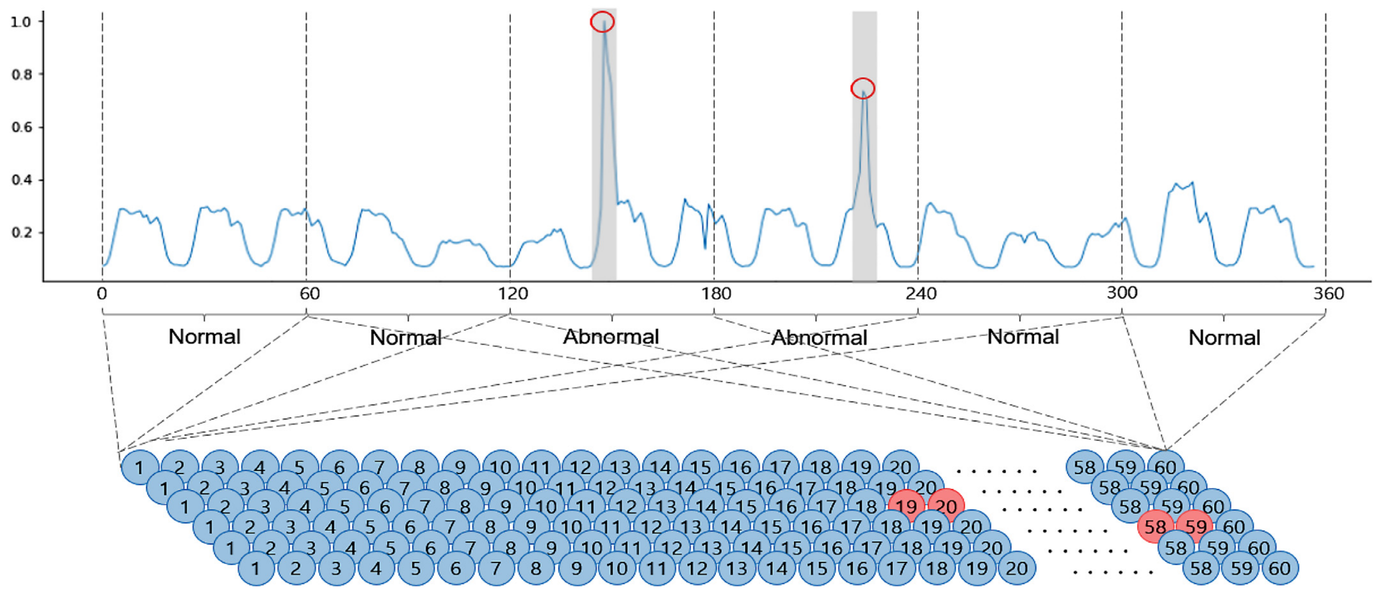

Fig. 5. 1D sequence traffic signal input.

Table 3
Related works on sequence anomaly detection.

Category	Author	Year	Data	Method	Description
Statistical modeling	Alizadeh, Khoshrou, and Zuquete (2015)	2015	Network traffic	Gaussian mixture model	Probabilistic model
	Münz, Li, and Carle (2007)	2007	Network traffic	K-means clustering	Classification using cluster centroids
	Zhang and Zulkernine (2006)	2006	Network traffic	Random forest	Unsupervised outlier pattern extraction
	Moore and Zuev (2005)	2005	Network traffic	Naïve Bayes	Bayes theorem assuming independence
Temporal feature modeling	Cheng et al. (2016)	2016	Network traffic	Multi-scale LSTM	Sliding window-based sequence classification
	Ding, Li, Batta, and Trajković (2016)	2016	Network traffic	SVM, LSTM	Performance comparison using SVM and LSTM
	Malhotra et al. (2015)	2015	Multi-sensor data	Stacked LSTM	Using prediction error distribution
	Chauhan and Vig (2015)	2015	ECG signal	LSTM	Normal signal learning and prediction
Spatial feature modeling	Wang et al. (2017)	2017	Network traffic	1D CNN	Image mapping and classification
	Zeng et al. (2014)	2014	Multi-sensor data	1D CNN	CNN-based feature extraction
	Zheng et al. (2014)	2014	Medical sensor data	Multi-channel DCNN	Signal concatenation using multiple channels
	Ren and Wu (2014)	2014	EEG signal	Convolutional DBN	Scaling of high-dimensional data

Malhotra et al. studied the LSTM model using sensor data with a normal signal (Malhotra, Vig, Shroff, & Agarwal, 2015). They used an LSTM model to predict future signals and then calculated error distributions using real signals to perform anomaly detection. These methods model the temporal features of sequence data and classify them using prediction-based algorithms. After learning only normal sequences using an RNN, the predicted traffic data and actual traffic data were compared and anomaly detection was performed based on a threshold value. This method has the advantages of predicting web traffic data with periodicity and achieving high classification performance. However, if the pattern of the web traffic data does not have a certain period, the predicted traffic data does not properly classify the actual traffic data.

Finally, Wang et al. created a specific pattern by mapping network traffic into a two-dimensional image (Wang, Zhu, Wang, Zeng, & Yang, 2017). A convolution operation was then applied and the features of the image were extracted. Ren and Wu extracted features using a convolutional algorithm on an EEG signal, which is high-dimensional data, and classified ideal values using a generative graphical model called a deep brief network (DBN) (Ren & Wu, 2014). These methods model spatial features by mapping complex sequences into images. CNN algorithms were mainly used to extract spatial features. These methods efficiently reflect the spatial information in sequence data with complex patterns and have obtained the highest classification performance compared to previous studies. However, when extracting features from time series data, time information loss occurs in the convolution and pooling operations.

As described above, there have been many attempts to perform anomaly detection in sequences using various methods, such as statistical, temporal, and spatial modeling. However, few attempts have been made to classify such sequences using spatial-temporal information. Most studies have proposed anomaly detection models that model only one important feature in the data. Therefore, a

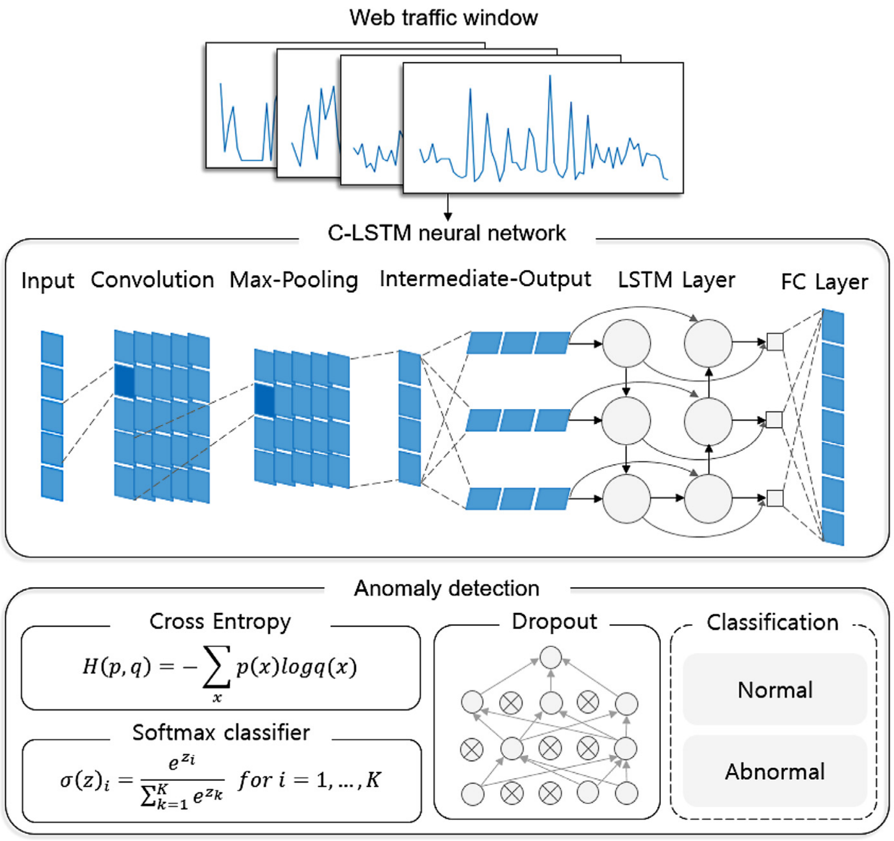

Fig. 6. The proposed detection structure.

proper learning method is required to perform anomaly detection using both time information and spatial information from complex web traffic sequences.

3. The proposed method

3.1. C-LSTM neural network

The proposed C-LSTM consists of CNN and LSTM layers, and is connected in a linear structure (Zhou, Sun, Liu, & Lau, 2015). Fig. 6 presents the structure for anomaly detection of web traffic using the proposed C-LSTM. The C-LSTM uses preprocessed data as inputs. The spatial features in the traffic window are extracted by the convolution and pooling layers. The temporal features are then extracted by the LSTM layers. The trained model then performs anomaly detection on the test data using a softmax classifier.

First, the CNN consists of several convolution and pooling layers, which are used to automatically extract higher-level sequences of web traffic spatial features (Masci, Meier, Cireşan, & Schmidhuber, 2011). These two-dimensional convolution operations utilize several filter vectors that slide over the sequence and detect features in order (Ronao & Cho, 2016b). A convolution layer is followed by an activation function. This allows the CNN to capture complex features in the input signal.

Assume that $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ is a web traffic data input vector, n is the number of values per window, and x_i is normalized traffic values. i is the index of the feature value, and j is the index of feature map for each traffic window. Eq. (1) derives the output value y_{ij}^1 from the first convolution layer. y_{ij}^1 is calculated using the value x_{ij}^0 from input data. b_j^0 represents the bias for the jth feature map, W is the weight of the kernel, M represents the size of the filter, and σ is an activation function, such as tanh or ReLU. Eq. (2) derives the output value y_{ij}^l from the lth convolution layer.

$$y_{ij}^1 = \sigma\left(b_j^0 + \sum_{m=1}^{M} W_{m,j}^0 x_{i+m-1,\ j}^0\right) \tag{1}$$

$$y_{ij}^l = \sigma\left(b_j^{l-1} + \sum_{m=1}^{M} W_{m,\ j}^{l-1} x_{i+m-1,\ j}^{l-1}\right) \tag{2}$$

The pooling layers decrease the spatial size of the representation to reduce the number of parameters and computational complexity of the network. This also has the effect of preventing overfitting. These layers effectively reduce spatial size by applying a max operation independently for each depth slice. Eq. (3) shows the operation of a pooling layer. R is a pooling size of less than the size of the input y and T is the stride that determines how far to move the pooled area. Max pooling is a type of pooling that selects the largest number in the subarea. It is effective to select the largest value in the subarea, since it indicates a large activation when the feature exists. This pooling also achieves better performance than average pooling or L2-norm pooling.

$$p_{ij}^l = \max_{r\ \in R} y_{i\times T+r,\ j}^{l-1} \tag{3}$$

In the LSTM layers, we use memory cells rather than simple recurrent units to store and output temporal features of web traffic data. This capability makes it simpler to understand temporal relationships on large time scale. The output value of the previous pooling layer is used as the input and the concept of gating is applied (Hochreiter & Schmidhuber, 1997). Gating is a mechanism based on multiplication, where a component of the input defines the behavior of each individual memory cell. Each LSTM unit updates its cell state according to the activation of each gate and is controlled by continuous values between 0 and 1. The inputs provided to the LSTM are fed into operations that control the input, output, and forgetting gates, which are managed in cell memory. $\boldsymbol{h}_t$, which is the hidden value of the LSTM cell, is updated at every time step t.

$$\boldsymbol{i}_t = \sigma\left(W_{pi}\boldsymbol{p}_t + W_{hi}\boldsymbol{h}_{t-1} + W_{ci} \circ \boldsymbol{c}_{t-1} + \boldsymbol{b}_i\right) \tag{4}$$

$$\boldsymbol{f}_t = \sigma\left(W_{pf}\boldsymbol{p}_t + W_{hf}\boldsymbol{h}_{t-1} + W_{cf} \circ \boldsymbol{c}_{t-1} + \boldsymbol{b}_f\right) \tag{5}$$

$$\boldsymbol{o}_t = \sigma\left(W_{po}\boldsymbol{p}_t + W_{ho}\boldsymbol{h}_{t-1} + W_{co} \circ \boldsymbol{c}_t + \boldsymbol{b}_o\right) \tag{6}$$

Eqs. (4)–(6) use the notations $\boldsymbol{i}$, $\boldsymbol{f}$ and $\boldsymbol{o}$, which are the input, forget, and output gates, respectively. Eqs. (7) and (8) use the notations $\boldsymbol{c}$ and $\boldsymbol{h}$, which are the cell state and hidden value, respectively. These two values are determined by the outputs of the three types of gates. σ is an activation function such as tanh. The term $\boldsymbol{p}_t$ is used as the input of a memory cell layer and is the output of a pooling layer at time t. W is a weight matrix, $\boldsymbol{b}$ is a bias vector, and $\circ$ denotes the Hadamard product. Networks using LSTM cells provide superior performance through time information modeling of signals, which has provided state-of-the-art results in anomaly detection (Ordóñez & Roggen, 2016).

$$\boldsymbol{c}_t = \boldsymbol{f}_t \circ \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \circ \sigma\left(W_{pc}\boldsymbol{p}_t + W_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c\right) \tag{7}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \circ \sigma\left(\boldsymbol{c}_t\right) \tag{8}$$

The combination of a fully connected layer and softmax classifier can be utilized to detect anomalies in traffic. These are the top-most layers of the C-LSTM system. The output of the LSTM unit is flattened into a feature vector $\boldsymbol{h}^l = (h_1, h_2, \ldots h_n)$, where n is the number of units in the last layer of LSTM. This vector is used as the input to a fully connected layer. Eq. (9) is used in this layer. σ is the activation function, W is the weight of the ith node in layer $l-1$ and the jth node in layer l, and b_i^{l-1} is a bias. d represents the output of a fully connected layer.

$$d_i^l = \sum_j \sigma\left(W_{ji}^{l-1}\left(h_i^{l-1}\right) + b_i^{l-1}\right) \tag{9}$$

The output of the fully connected layer is classified as either 0 or 1 by softmax. Eq. (10) calculates the classification probability used by the softmax layer. C is the activity class, L is the last layer index, and N_c is the total number of activity classes. The softmax layer classifies traffic test data into two classes: normal and abnormal.

$$P(c|d) = argmax_{c\in C} \frac{\exp\left(d^{L-1}w^L\right)}{\sum_{k=1}^{N_c} \exp\left(d^{L-1}w_k\right)} \tag{10}$$

3.2. Architecture

The design of the C-LSTM can include various structures depending on various parameters, such as the number of CNN layers, the number of kernels, the number of layers of LSTM, and the number of units of LSTM. These parameters can affect learning performance by extracting more characteristics of the training data and affect learning speed by increasing or decreasing the number of parameters (He & Sun, 2015). To determine the optimal architecture for the C-LSTM, including parameters, one must understand the characteristics of the input data. In our case, we wish to perform anomaly detection on web traffic data. Because this data is input into the C-LSTM in the form of sequences of length 60 by using a sliding window algorithm with spatial-temporal information, a kernel of size 5 should be used to minimize the loss of information. Zero padding was used for the convolution operation and not used for the pooling operation. We designed the parameters of the C-LSTM as listed in Table 4. The input of the C-LSTM is a data vector of length 60 that passes through the LSTM after passing through the convolution and pooling layers. We used tanh as an activation function of C-LSTM. Tanh is a function that rescales and shifts the sigmoid function, so that tanh is faster in learning than sigmoid when it is used as an activation function. There is another activation function like ReLU other than tanh. ReLU computes the function $f(x) = max(0,\ x)$. In other words, it thresholds the activation at zero. Its advantage is that it greatly accelerates the convergence of the stochastic gradient descent compared to the tanh function, which is claimed to be due to its linear form. Also, when compared to a tanh neuron containing expensive operations, ReLU can be computed with a simple operation. Although the proposed architecture uses tanh, there could be improvements with ReLU or its relatives.

Table 4
The proposed C-LSTM architecture.

Type	#Filter	Kernel size	Stride	# Param
Convolution	64	5	1	384
Activation (tanh)	–	–	–	0
Pooling	–	2	2	0
Convolution	64	5	1	20,544
Activation (tanh)	–	–	–	0
Pooling	–	2	2	0
LSTM (64)	–	–	–	262,400
Dense (32)	–	–	–	2080
Activation (tanh)	–	–	–	0
Dense (2)	–	–	–	66
Softmax	–	–	–	0
Total number of parameters				285,474

3.3. Hardware and software setup

We used the following hardware and software to detect web traffic anomaly using C-LSTM. We used four GeForce GTX1080 GPU and one Intel Xeon E5 CPU for C-LSTM learning. High performance hardware is required because the maximum size of the network that can be trained depends on the performance of the GPU (Krizhevsky, Sutskever, & Hinton, 2012). Four GeForce GTX1080 GPUs have 8 gigabytes of RAM, 2560 CUDA cores, and a maximum bandwidth of 320 GB/sec. The Intel Xeon E5-2680V4 CPU has 14 cores and 28 threads and is powerful enough to run GeForce GTX1080 GPUs. We also used 64GB of DDR4 RAM and used the Samsung 960 PRO M.2 2280 2 TB SSD as a storage device. The software used to implement C-LSTM was installed with the Ubuntu Desktop 16.04.2 LTS operating system. We installed the NVIDIA 384.59 graphic driver to run the GPU. NVIDIA cuda 8.0.44 GPL License 3.0 was installed as the GPU platform. The programming language is Python 3.6.1, which is widely applied to artificial intelligence for machine learning and deep learning. We also installed Anaconda 4.4.0 for large-scale data processing, predictive analytics, and scientific computing. We used numpy 1.13.1 and pandas 0.20.1 libraries to simplify matrix operations. The tensorflow-gpu 1.2.1 and keras 2.0.6 libraries were used to run deep learning using the GPU. In particular, using the keras library, the researcher can easily configure the network model because it includes functions such as data preprocessing, and parameter tuning as well as deep learning layers in block form. We have trained C-LSTM with 512 batch sizes for 500 epochs using the above hardware and software, and it took 769 seconds.

Table 5
Performance comparison of different model combinations.

Method	Accuracy	Precision	Recall	F1 score
LSTM	92.9	82.4	39.8	53.7
CNN	96.5	94.5	86.2	90.1
LSTM + DNN	93.1	88.2	34.5	49.6
CNN + DNN	96.7	95.1	86.5	90.6
CNN + LSTM	97.1	95.4	87.6	91.3
CNN + LSTM + DNN	**98.6**	96.2	**89.7**	92.3

4. Experiments

4.1. Yahoo S5 Webscope dataset

In this paper, we used data from Yahoo, which is a company that operates a US portal site. We utilized the A1 class of the Yahoo Webscope S5 anomaly benchmark dataset, which consists of 67 files, to validate the proposed anomaly detection architecture. The collected data is represented by time series of traffic measurement values from actual web services in one hour units. Abnormal values were labeled manually and the data has a relatively large variation in traffic compared to other available datasets. There are a total of 94,866 traffic values in 67 different files, but only 1669 of these values are abnormal. Therefore, traffic anomaly detection is a problem with data imbalance. The data used has an unusually small ratio of 0.02% abnormal values, meaning the data imbalance is severe (Sainath, Vinyals, Senior, & Sak, 2015). Therefore, we applied a sliding window algorithm to solve the data imbalance problem (Catania, Bromberg, & Garino, 2012). To test the proposed method, 90,910 windows were created by applying the sliding window algorithm and 8470 abnormal windows were labeled. The inputs for the C-LSTM are values between 0 and 1, so we had to preprocess the traffic values for anomaly detection. The values were normalized using Eq. (11). x represents the value of the actual traffic data and x' represents the normalized value.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{11}$$

4.2. Results and analysis

4.2.1. Comparison to other machine learning algorithms

In order to verify the usefulness of the proposed method, 10-fold cross validation was performed to compare its performance to those of other machine learning algorithms. We adjusted the parameters to compare the highest performance of the machine learning method with those of the proposed C-LSTM. Support vector machine (SVM) kernel was set to linear and L2-norm was used as penalty. Loss was squared hinge and tolerance for stopping criteria was set to 1e−4. The maximum number of iterations to be run is set to 1000. K-nearest neighbor (KNN) sets the number of neighbors to 7. All points in each neighborhood are weighted equally. Random forest set the number of trees in the forest to 20 and we used gini to measure the quality of the split. Gini is used as entropy for information gain. The proposed C-LSTM has the highest performance compared to other machine learning methods, followed by the random forest, multilayer perceptron (MLP), and KNN algorithms. Fig. 7 shows the accuracy from 10-fold cross validation.

4.2.2. Performance comparison of model combinations

Table 5 contains a performance comparison of different deep learning model combinations. CNN, LSTM, and DNN were used as baselines and experiments were conducted in all combinations of the three models. Because the proposed C-LSTM model has a structure combining them, it achieved better performance than other combinations. We also confirmed that modeling spatial and temporal features using the CNN, LSTM, and DNN models yields better performance than other models (Bell, Upchurch, Snavely, & Bala, 2015).

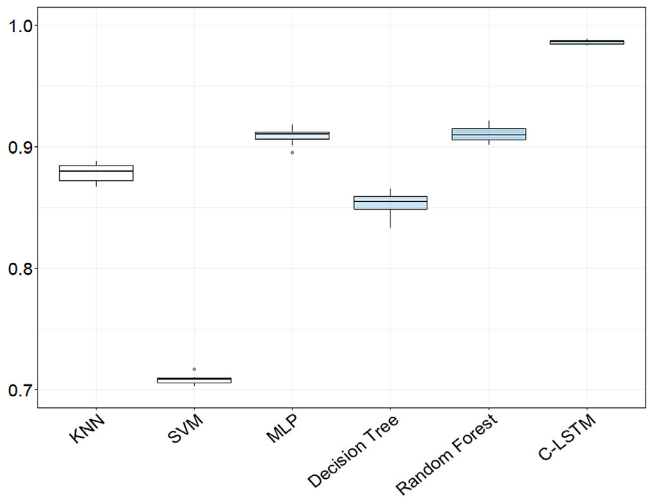

Fig. 7. Comparison of accuracy from 10-fold cross validation.

Table 6
Confusion matrix.

Predict true	Normal	Abnormal
Normal	24,663	69
Abnormal	264 (a)	2277 (b)

An error occurs if an abnormal instance is marked as a normal instance, or vice versa. The former type of error is a false negative (FN) and the latter is a false positive (FP). Similarly, a true positive (TP) occurs if a normal instance is correctly identified and a true negative (TN) occurs if an abnormal instance is correctly classified. In our experiments, the evaluation of individual algorithms was performed using TP, TN, FP and FN rates.

$$precision = \frac{TP}{TP + FP} \tag{12}$$

$$recall = \frac{TP}{TP + FN} \tag{13}$$

The precision in Eq. (12) is the percentage of relevant instances among all retrieved instances and the recall in Eq. (13) is the percentage of relevant instances retrieved among all relevant instances. Recall is a particularly important metric for anomaly detection because it represents the ratio of the number of abnormalities found over the total number of abnormal instances.

$$F1\ Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{14}$$

Using precision and recall, we can derive an F1 score, as shown in Eq. (14). We quantitatively compared the test results using these evaluation criteria. Table 5 contains the results of comparative experiments using the above metrics. Fig. 8 presents the cross entropy and accuracy per experimental epoch of the C-LSTM, CNN, and LSTM models. C-LSTM has the highest accuracy and lowest cross entropy.

4.2.3. Misclassification data analysis

Table 6 represents the confusion matrix for the C-LSTM model. Because data imbalance in the anomalous traffic detection problem is significant, it is important to classify a small number of abnormal data instances very accurately. Table 6 contains the results of classifying 30% of the data as test data after training the C-LSTM using 70% of the data as training data.

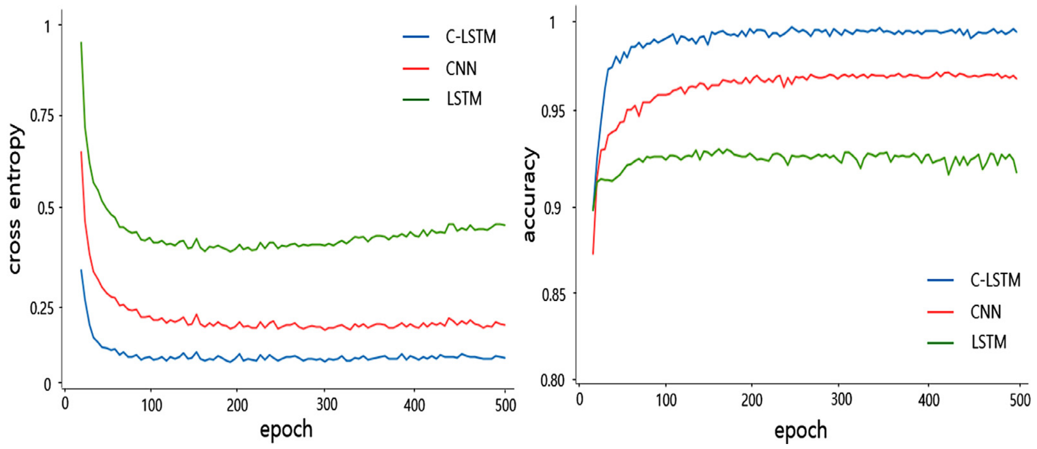

Fig. 8. Cross entropy and accuracy per epoch.

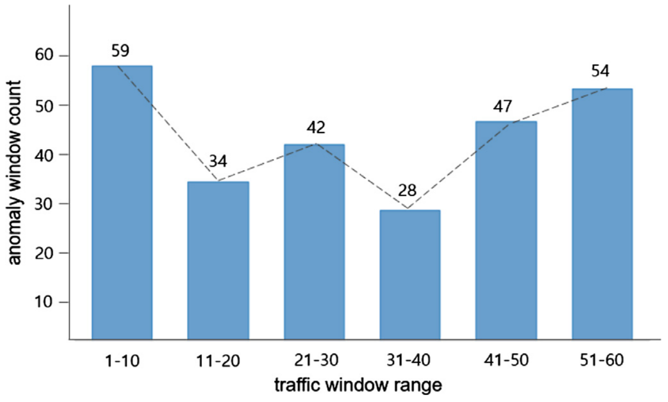

Fig. 9. Number of misclassifications for each anomaly value position.

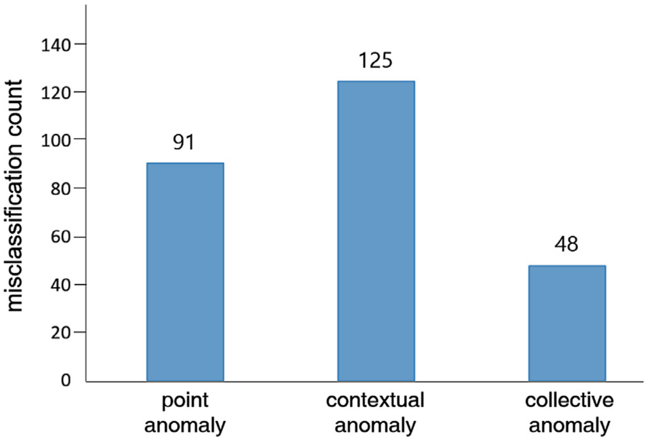

Fig. 10. Number of misclassifications of each kinds of anomalies.

(a) is the number of classification failures of actual anomalies, and (b) is the number of traffic windows containing successful anomaly detection. It is important to analyze (a) in order to understand the characteristics of misclassification. For this analysis, traffic windows of length 60 were divided into 6 segments and categorized based on the anomaly locations within each window. Fig. 9 illustrates the number of misclassifications for each abnormal position with respect to (a). When the abnormal position is located at either end of the traffic window, there is a tendency to misclassify. In contrast, when the abnormal value was located at the center, we confirmed that the number of misclassifications decreased. We also analyzed the types of anomalies in (a). Fig. 10 shows the number of point anomaly, contextual anomaly, and collective anomaly that were not detected by C-LSTM. Analysis showed that it was the most difficult to detect the context anomalies, followed by point anomalies and collective anomalies.

4.2.4. Internal analysis of C-LSTM model

Deep learning has the disadvantage that we cannot interpret the learning process. We cannot fully understand the network, but we used a visualization approach to see how some of the behaviors of the architecture work. Through the visualization, we can see how the input data changes in each layer. Fig. 11 shows the intermediate outputs of the CNN in the C-LSTM system. Each CNN layer consists of 64 kernels, each of which has a different weight. Each kernel is a weighting matrix for filtering the spatial characteristics of web traffic by applying convolution operations to the web traffic signals. An input window produces a different output for each kernel. A window of length 60 is reduced in size to a length of 15 by the convolution and pooling operations. However, we confirmed that the spatial-temporal characteristics of the input window were maintained. C-LSTM has a structure in which CNN and LSTM are linearly connected. Therefore, it is important to minimize the loss of the temporal and spatial information of the web traffic signal input to the CNN layer and transmit it to the LSTM layer. We show in Fig. 11 that each kernel filter on the CNN layer reduces the noise of the web traffic data. The intermediate output filtered by kernel A and B preserves both local and global features, even though the noise is reduced. Web traffic anomalies occur locally and globally, it is important to preserve features by the kernel. We assumed that the features extracted through convolution operations can be passed to the LSTM layer to help classify various anomalies. The visualization is useful for analyzing the intermediate output of the C-LSTM.

4.2.5. Cluster analysis through t-SNE

Fig. 12 is a visualization of the last fully connected layer based on t-distributed stochastic neighbor embedding (t-SNE). t-SNE is a technique for visualizing high-dimensional data through dimension reduction (Maaten & Hinton, 2008). We confirmed that anomaly detection was successfully accomplished based on the cluster as shown in Fig. 12. Green indicates normal window and red indicates abnormal window. Through t-SNE visualization, we can see that data with similar characteristics form clusters. Since normal windows are composed of various characteristics, they form several clusters. We can see that abnormal windows form a cluster as shown in the left of Fig. 12. Analysis of the cluster characteristics

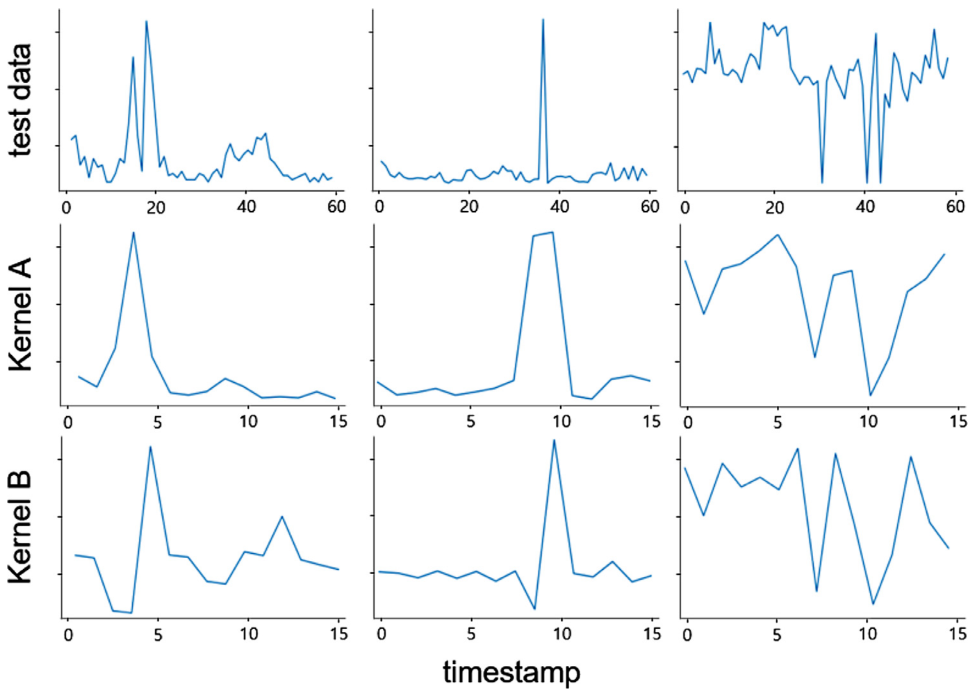

Fig. 11. C-LSTM intermediate outputs for various kernels.

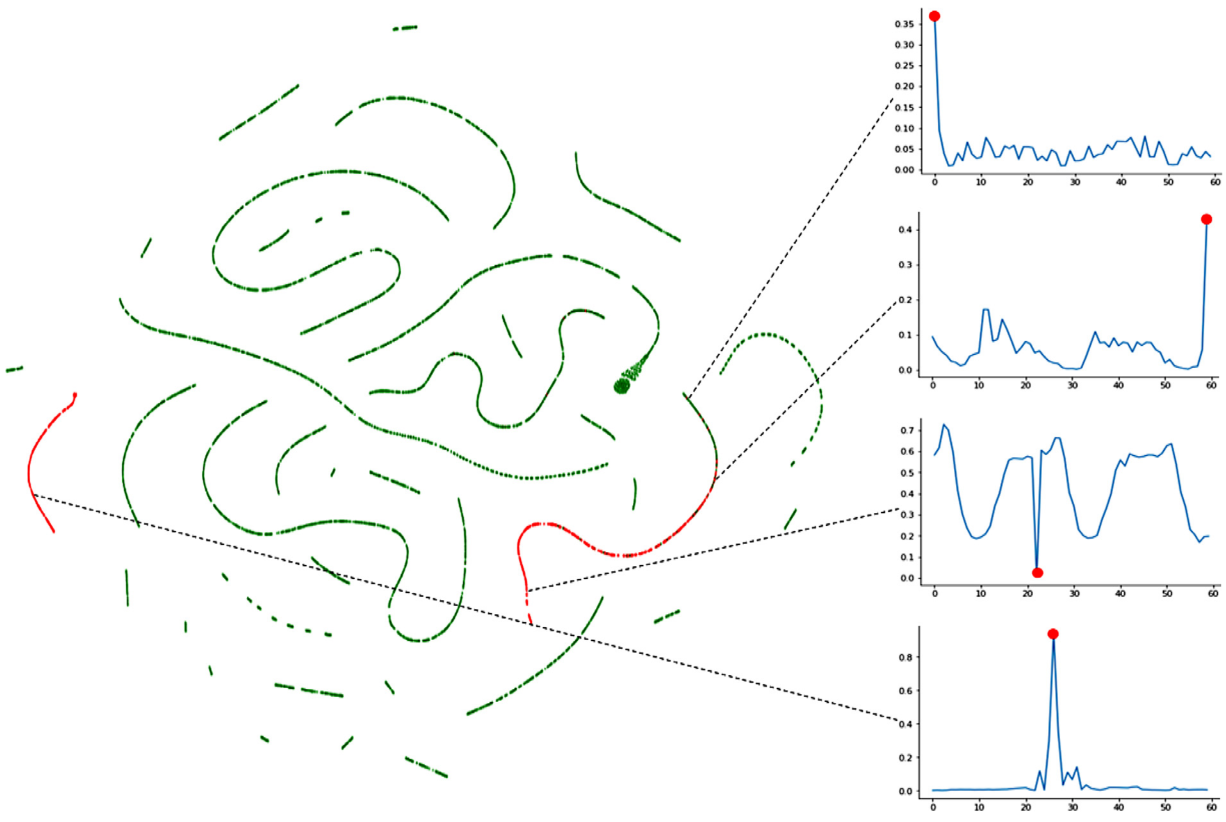

Fig. 12. Visualization of the final fully connected layer based on t-SNE.

revealed that the cluster with anomalies on the left is mostly located at the center of a window or is clearly distinguished from normal values. In contrast, in the case of the cluster on the right, we can see that it consists of normal and abnormal windows. In order to detect anomalies in web traffic, a sliding window is applied. Even if there is an abnormal value at the end of the window, the abnormal label is included. If an abnormal value exists at the end of a window, it is labeled as an abnormal window. In the case of the abnormal window with a similar distribution to normal values, the anomaly frequently occurred at either end of the window. It is difficult to distinguish point, contextual, and collective anomaly when abnormal value is located at the end of window.

Table 7
Additional experiments: compare with other deep learning methods.

Method	Accuracy	Precision	Recall	F1 score
LSTM	95..9	63.8	14.1	23.1
GRU	95.9	63.1	11.6	19.6
CNN	97.2	98.5	96.9	97.7
C-LSTM	**98.7**	95.9	**98.5**	97.2

4.2.6. Additional experiment

We have experimented with the twitter data provided by Numenta (Lavin & Ahmad, 2015). Twitter data is web traffic that represents a collection of Twitter mentions of large publicly-traded companies like Google and IBM. Anomaly in Twitter data is manually labeled. Anomaly detection is a data imbalance problem. Therefore, to solve the data imbalance problem, the sliding window algorithm was applied to label the abnormal window. Twitter data has a total of 158,036 traffic windows, of which 2100 are anomaly windows. The experiment classifies normal and abnormal windows. Table 7 shows that C-LSTM performance achieves higher accuracy and recall compared to other competing approaches such as CNN and LSTM.

Even though the precision and F1 scores are lower compared to CNN, the recall performance, which is a metric of observed anomaly, represents the possibility for detecting web traffic anomalies. Although C-LSTM is not always the best for all web traffic data, there is the possibility of improvement for anomaly detection.

5. Conclusion

We proposed a C-LSTM architecture for anomaly detection in web traffic. We found an optimal model through parametric experiments, model comparison experiments, and data analysis. We demonstrated the usefulness and superiority of the proposed model by comparing it to other machine learning methods. It can detect various anomalies in complex network streams. We used the C-LSTM to automatically extract patterns in web traffic data containing spatial-temporal information. A confusion matrix and t-SNE analysis revealed the characteristics of normal and abnormal data classified by the C-LSTM. The C-LSTM model proposed in this paper classifies and extracts features that could not be extracted in previous anomaly detection studies using the conventional machine learning methods. However, its accuracy tends to decrease when detecting abnormal values generated at the boundaries of preprocessed data windows. Additionally, because the proposed method preprocesses data using a sliding window, there is a delay for detecting anomalies in real data. This issue remains a challenge to be addressed in the future. Further research is also required to automatically find the optimal parameters for the C-LSTM system.

Acknowledgments

This work was supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract (UD160066BD).

References

Ahmed, M., & Mahmood, A. N. (2014). Network traffic pattern analysis using improved information theoretic co-clustering based collective anomaly detection. In *International conference on security and privacy in communication systems* (pp. 204–219).
Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications, 60*, 19–31.
Alizadeh, H., Khoshrou, A., & Zuquete, A. (2015). Traffic classification and verification using unsupervised learning of Gaussian Mixture Models. In *2015 IEEE international workshop on measurements & networking (M&N)* (pp. 1–6).
Bell, S., Upchurch, P., Snavely, N., & Bala, K. (2015). Material recognition in the wild with the materials in context database. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 3479–3487).
Catania, C. A., Bromberg, F., & Garino, C. G. (2012). An autonomous labeling approach to support vector machines algorithms for network traffic anomaly detection. *Expert Systems with Applications, 39*(2), 1822–1829.
Chauhan, S., & Vig, L. (2015). Anomaly detection in ECG time signals via deep long short-term memory networks. In *IEEE international conference on data science and advanced analytics (DSAA)* (pp. 1–7).
Chen, X. W., & Lin, X. (2014). Big data deep learning: Challenges and perspectives. *IEEE Access, 2*, 514–525.
Cheng, M., Xu, Q., Lv, J., Liu, W., Li, Q., & Wang, J. (2016). MS-LSTM: A multi-scale LSTM model for BGP anomaly detection. In *IEEE international conference on network protocols (ICNP)* (pp. 1–6).
Ding, Q., Li, Z., Batta, P., & Trajković, L. (2016). Detecting BGP anomalies using machine learning techniques. In *IEEE international conference on systems, man, and cybernetics (SMC)* (pp. 3352–3355).
Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., et al. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2625–2634).
Goldstein, M., & Uchida, S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLoS ONE, 11*(4), 1–31.
Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems, 28*(10), 2222–2232.
He, K., & Sun, J. (2015). Convolutional neural networks at constrained time cost. In *IEEE conference on computer vision and pattern recognition* (pp. 5353–5360).
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.
Huang, S. Y., & Huang, Y. N. (2013). Network traffic anomaly detection based on growing hierarchical SOM. In *Annual IEEE/IFIP international conference on dependable systems and networks (DSN)* (pp. 1–2).
Jiang, D., Xu, Z., Zhang, P., & Zhu, T. (2014). A transform domain-based anomaly detection approach to network-wide traffic. *Journal of Network and Computer Applications, 40*(1), 292–306.
Kim, K.-H., & Cho, S.-B. (2017). Modular Bayesian networks with low-power wearable sensors for recognizing eating activities. *Sensors, 17*(12), 1–16.
Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).
Laptev, N., & Amizadeh, S.. *A labeled anomaly detection dataset S5* Yahoo Research, v1. https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70.
Lavin, A., & Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark. In *IEEE 14th international conference on machine learning and applications (ICMLA)* (pp. 38–44).
LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324.
Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research, 9*, 2579–2605.
Malhotra, P., Vig, L., Shroff, G., & Agarwal, P. (2015). Long short term memory networks for anomaly detection in time series. In *European symposium on artificial neural networks, computational intelligence* (p. 89).
Masci, J., Meier, U., Cireşan, D., & Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. In *International conference on artificial neural networks and machine learning (ICANN)* (pp. 52–59).
Moore, A. W., & Zuev, D. (2005). Internet traffic classification using Bayesian analysis techniques. In *ACM SIGMETRICS international conference on measurement and modeling of computer systems: 33* (pp. 50–60).
Münz, G., Li, S., & Carle, G. (2007). Traffic anomaly detection using k-means clustering. *GI/ITG Workshop MMBnet*.
Oehmcke, S., Zielinski, O., & Kramer, O. (2018). Input quality aware convolutional LSTM networks for virtual marine sensors. *Neurocomputing, 275*, 2603–2615.
Ordóñez, F. J., & Roggen, D. (2016). Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors, 16*(1), 115.
Paschalidis, I. C., & Smaragdakis, G. (2009). Spatio-temporal network anomaly detection by assessing deviations of empirical measures. *IEEE/ACM Transactions on Networking, 17*(3), 685–697.
Ren, Y., & Wu, Y. (2014). Convolutional deep belief networks for feature extraction of EEG signal. In *International joint conference on neural networks (IJCNN)* (pp. 2850–2853).
Ronao, C. A., & Cho, S.-B. (2016a). Anomalous query access detection in RBAC-administered databases with random forest and PCA. *Information Sciences, 369*, 238–250.
Ronao, C. A., & Cho, S.-B. (2016b). Human activity recognition with smartphone sensors using deep learning neural networks. *Expert Systems with Applications, 59*, 235–244.
Ronao, C. A., & Cho, S.-B. (2017). Recognizing human activities from smartphone sensors using hierarchical continuous hidden Markov models. *International Journal of Distributed Sensor Networks, 13*(1).
Sainath, T. N., Vinyals, O., Senior, A., & Sak, H. (2015). Convolutional, long short-term memory, fully connected deep neural networks. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 4580–4584).
Sak, H., Senior, A., & Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *Annual conference of international speech communication association*.

Sermanet, P., & LeCun, Y. (2011). Traffic sign recognition with multi-scale convolutional networks. In *International joint conference on neural networks (IJCNN)* (pp. 2809–2813).

Thill, M., Konen, W., & Bäck, T. (2017). Online anomaly detection on the webscope S5 dataset: A comparative study. In *Evolving and adaptive intelligent systems (EAIS)* (pp. 1–8).

Wang, W., Zhu, M., Wang, J., Zeng, X., & Yang, Z. (2017). End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *IEEE international conference on intelligence and security informatics (ISI)* (pp. 43–48).

Xu, Y., Kong, Q., Huang, Q., Wang, W., & Plumbley, M. D. (2017). Convolutional gated recurrent neural network incorporating spatial features for audio tagging. In *International joint conference on neural networks (IJCNN)* (pp. 3461–3466).

Yu, J., Liu, F., Zhou, W., & Yu, H. (2014). Hadoop-based network traffic anomaly detection in backbone. In *IEEE international conference on cloud computing and intelligence systems (CCIS)* (pp. 140–145).

Zeng, M., Nguyen, L. T., Yu, B., Mengshoel, O. J., Zhu, J., Wu, P., et al. (2014). Convolutional neural networks for human activity recognition using mobile sensors. In *International conference on mobile computing, applications and services (MobiCASE)* (pp. 197–205).

Zhang, J., & Zulkernine, M. (2006). Anomaly based network intrusion detection with unsupervised outlier detection. In *IEEE international conference on communications: 5* (pp. 2388–2393).

Zheng, Y., Liu, Q., Chen, E., Ge, Y., & Zhao, J. L. (2014). Time series classification using multi-channels deep convolutional neural networks. In *International conference on web-age information management* (pp. 298–310).

Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015). A C-LSTM neural network for text classification. arXiv preprint arXiv:1511.08630.

Zhou, X., Hu, B., Chen, Q., & Wang, X. (2018). Recurrent convolutional neural network for answer selection in community question answering. *Neurocomputing, 274*, 8–18.