4-2024

Expanding analytical capabilities in intrusion detection through ensemble-based multi-label classification

Ehsan Hallaji
University of Windsor

Roozbeh Razavi-Far
University of New Brunswick

Mehrdad Saif
University of Windsor

Computers & Security 139 (2024) 103730

Contents lists available at ScienceDirect

Computers & Security

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

Expanding analytical capabilities in intrusion detection through ensemble-based multi-label classification

Ehsan Hallaji [a,*], Roozbeh Razavi-Far [a,b], Mehrdad Saif [a]

[a] Department of Electrical and Computer Engineering, University of Windsor, 401 Sunset Ave, Windsor, ON N9B 3P4, Canada
[b] Faculty of Computer Science, University of New Brunswick, 550 Windsor St, Fredericton, NB E3B 5A3, Canada

ARTICLE INFO

Keywords:
Multi-label learning
Deep learning
Ensemble learning
Intrusion detection

ABSTRACT

Intrusion detection systems are primarily designed to flag security breaches upon their occurrence. These systems operate under the assumption of single-label data, where each instance is assigned to a single category. However, when dealing with complex data, such as malware triage, the information provided by the IDS is limited. Consequently, additional analysis becomes necessary, leading to delays and incurring additional computational costs. Existing solutions to this problem typically merge these steps by considering a unified, but large, label set encompassing both intrusion and analytical labels, which adversely affects efficiency and performance. To address these challenges, this paper presents a novel framework for multi-label classification by employing an ensemble of sequential models that preserve the original label sets during training. Each model focuses on learning the distribution specifically related to its assigned set of labels, independent of the other label sets. To capture the relationship between different sets of labels, the parameters of each trained model initialize the subsequent model, ensuring that information from unrelated label sets does not interfere with the learning objective. Consequently, the proposed method enhances prediction performance without increasing computational complexity. To evaluate the effectiveness of our approach, we conduct experiments on a real-world dataset related to intrusion detection. The results clearly demonstrate the effectiveness of our proposed method in handling multi-label classification tasks.

1. Introduction

Most recent advancements in intelligent intrusion detection using machine learning tackle single-label classification problems, where a dataset has only a single set of labels consisting of multiple classes (Ferrag et al., 2020; Yang et al., 2022). Nonetheless, in most cyber environments, a single record of data may indicate multiple states or categories (Liu et al., 2022). As an example, deep learning is used in Wang et al. (2020) to detect both false data injection attacks and the location of the injected attack in smart grids simultaneously. As a result, a domain of machine learning, Multi-Label Classification (MLC) is dedicated to facilitating this task (Riera et al., 2022; Liu et al., 2022; Xie et al., 2022).

Research endeavors in MLC generally follow two approaches. The first group is in fact a framework that aims at translating the multi-label problem for single-label classifiers by reformulating the labeling scheme into that of the single-label classifiers. Label Powerset (LP) (Boutell et al., 2004) and Binary Relevance (BR) (Tsoumakas et al., 2010) are the most fundamental algorithms in this category that inspired numerous works advancing and improving their original ideas (Tsoumakas et al., 2011; Read et al., 2011). In short, the idea in LP is to create new classes based on available combinations of classes across different label sets so that they can be integrated into a single label set. BR, on the other hand, forms a set of binary problems that are independent in order to create a single-label multi-class problem.

The second category in the MLC domain includes algorithms specifically designed or adapted for multi-label problems. These designs are mostly inspired or adapted from existing single-label classifiers. For instance, Multi-label k Nearest Neighbor (MLkNN) (Zhang and Zhou, 2005) and Multi-class Multi-label Perceptron (MMP) (Loza Mencia and Furnkranz, 2008) are MLC algorithms that are adapted from k Nearest Neighbors (kNN) and Multi-Layer Perceptron (MLP) algorithms, respectively.

Despite the advantages of the approaches above, these methods are often followed by additional computational burden and sub-optimal prediction performance in Intrusion Detection Systems (IDS). To be-

* Corresponding author.
E-mail addresses: hallaji@uwindsor.ca (E. Hallaji), roozbeh.razavi-far@unb.ca (R. Razavi-Far), msaif@uwindsor.ca (M. Saif).

https://doi.org/10.1016/j.cose.2024.103730
Received 5 July 2023; Received in revised form 4 January 2024; Accepted 15 January 2024
Available online 19 January 2024

gin with, breaking the original data distribution into several binary combinations or creating a single label set from all combinations of classes in all label sets, exponentially increases the computational cost by requiring an excessive number of models and training scenarios. In addition, breaking the original distribution into several smaller subsets usually causes a class imbalance within data, or worsens it if already imbalanced. As a result, the former affects the efficiency, and the latter deteriorates the prediction accuracy. Furthermore, altering the multi-label nature of the data may result in a distribution that does not exactly match that of the problem at hand. This can result in biased data which in turn affects the performance of intrusion identification (Arp et al., 2022; Apruzzese et al., 2023).

To address these issues, this paper proposes a network-based IDS that primarily runs on the server side of the network. The designed IDS utilizes a novel approach for tackling MLC problems, without compromising accuracy or efficiency. The proposed method uses an ensemble of deep learning models, where each model inside the ensemble is trained on only a single label set (i.e., each label set contains several classes). This provides three main advantages for the proposed MLC-based IDS:

- The complexity of the ensemble IDS model is linear and relational to the number of label sets. Compared to the exponential complexity caused by decomposition into all class combinations across different label sets, this is a huge improvement in terms of computational efficiency. In other words, the complexity of the proposed IDS approach is comparable to those that use single-label classifiers.
- Each model in the IDS ensemble can perform a single-label classification on the original label distribution. As a result, class populations are intact and the focus of the neural network is merely on learning the targeted label set, not all label distributions. Our experiments verify that this structure enhances the prediction quality to a great extent.
- In order to take label sets correlation into account, we train these models sequentially, where each model is initialized by the parameters of the previously trained model. This way, information from each label set is conveyed to the next model without interfering with the learning objective.

The rest of the paper is organized as follows. Section 2 includes the preliminaries for this work. Section 3 presents the design of multi-label classifiers. Section 4 illustrates the experimental results and analysis. Finally, Section 5 concludes the paper.

2. Background

The problem this paper aims to address lies at the intersection of intrusion detection and multi-label classification. The connection between these domains may not be clear at first in the realm of security, as they are often studied independently. To clarify this relationship in this study, the problem of MLC and the targeted application are first explained in this section. The required background on important algorithms of MLC is demonstrated afterward.

2.1. Problem statement

Despite the recent advancements in machine learning and the advent of sophisticated deep learning structures, the majority of these methods are designed to work with a single set of labels, where data samples can only fall under a certain category. In the application domain, however, data samples may belong to more than one class (Jethanandani et al., 2020). For instance, in cyber-security, an anomaly in the data stream may indicate an intrusion or a network-related issue in the system (Fernandes et al., 2019). In cyber-physical systems, a fault and a cyber intrusion can take place simultaneously (Hallaji et al., 2022). Moreover, most intrusion detection systems only detect if the traffic is benign or malicious (da Costa et al., 2019; Catillo et al., 2023). Thus, the decision maker in such systems needs to provide more insight into the given input rather than strictly assign it to a certain class. MLC algorithms enable analyzing data across a wider spectrum, which leads to the extraction of more knowledge from data. Despite the aforementioned points, multi-label learning is rarely employed for attack identification and intrusion detection in the literature.

In this paper, eleven attacks categories are considered to simulate an intrusion in computer networks:

1. Reconnaissance: Attack probing and gathering network information to bypass security controls.
2. Backdoor: Creating a hidden entry point in a system or network that allows unauthorized access and control.
3. Denial of Service (DoS): Intrusion disrupting computer resources, overwhelming the system to prevent authorized access.
4. Distributed DoS (DDoS): A distributed DoS attack carried out by several bots in a distributed fashion.
5. Exploit: Utilizing software vulnerabilities to gain unauthorized access, execute harmful code, or compromise a system.
6. Analysis: Intrusion targeting web applications through ports, emails, and web scripts.
7. Fuzzers: Automated tools that input random or invalid data into software to detect vulnerabilities, crashes, or unexpected behavior.
8. Worm: Malware that self-replicates and spreads through networks, exploiting vulnerabilities to infect and compromise multiple systems.
9. Shellcode: Inserting malicious code into a vulnerable program or system to acquire unauthorized control and execute specific commands.
10. Generic: Technique targeting any block cipher with a hash function for collisions, regardless of its configuration.
11. Theft: Adversaries that exploit weak points in a system to penetrate and steal sensitive data.

2.2. Multi-label classification

The majority of MLC approaches are usually in the form of a framework that translates a multi-label scenario into a single-label and multi-class problem. From there, these techniques can be combined with any classifier to make predictions on data (Tsoumakas and Katakis, 2007). While these methods can be combined with any neural network structure, there are also algorithms strictly relying on the neural network structure to classify a multi-label dataset. We select popular algorithms from both groups and explain them in detail in the remainder of this section.

2.2.1. Label powerset

Perhaps, the most common solution for MLC is provided by the Label Powerset (LP) algorithm (Boutell et al., 2004). LP reformulates the multi-label problem into a standard single-label problem by considering distinct combinations of labels among different label sets (i.e., each series of labels in a multi-label scenario) as separate classes. However, as the number of classes and label sets increases, the efficiency of LP deteriorates. Moreover, by regrouping samples into more categories based on every unique combination of labels, samples will be divided into more subsets that decrease the population for some classes and may lead to class imbalance (Tsoumakas et al., 2011).

2.2.2. Random k-label sets

Aiming to eliminate the aforementioned efficiency issues of LP, RAndom k label sets (RAkEL) break the data into smaller subsets (i.e., either joint or disjoint sets) and apply an LP on each of them (Tsoumakas et al., 2011). The final model consists of a set of trained LP models, similar

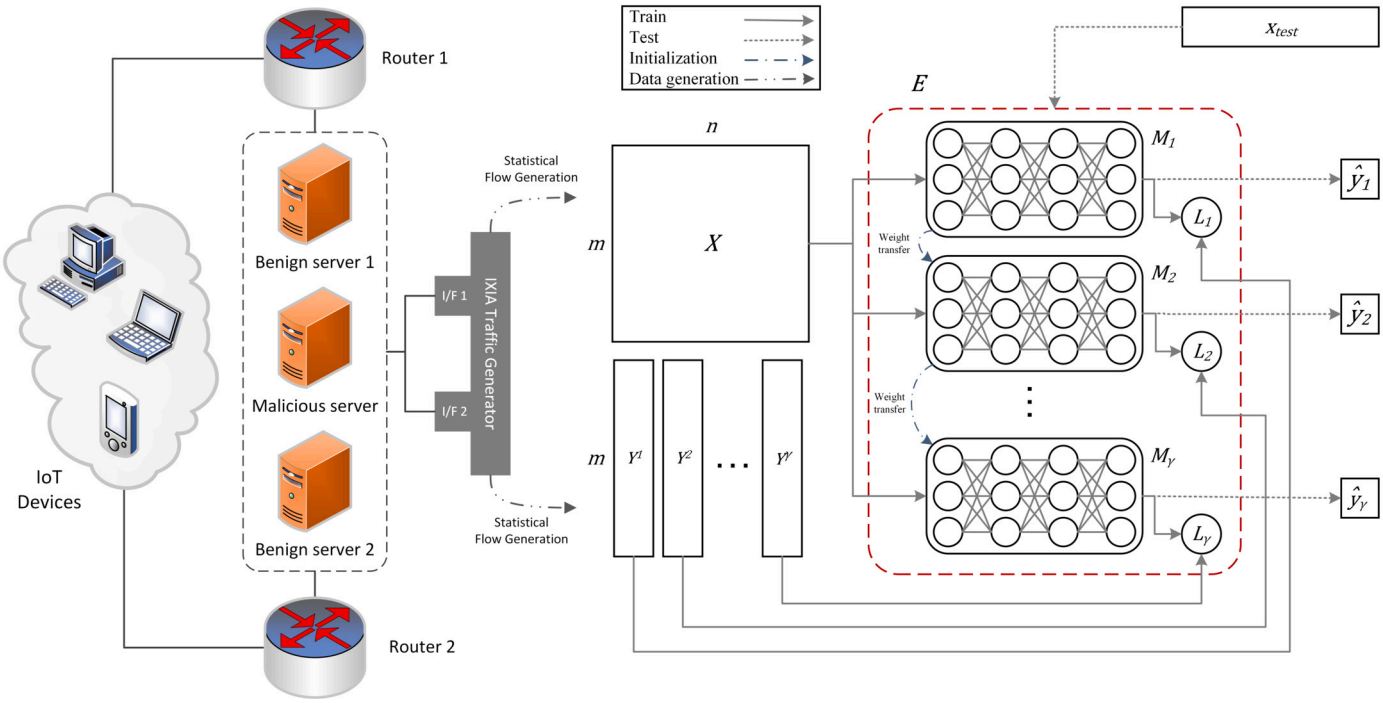

Fig. 1. Illustrative diagram of the proposed IDS method. Steps related to training and test phases are specified in solid and dotted arrows, as specified in the figure. The data generation process of the selected IoT case is also shown on the left.

to the ensemble approach. The results of these LP models will be aggregated on test data to reach the final prediction. Compared to LP, this approach provides enhanced efficiency and classification performance on large-scale training sets with several label sets.

2.2.3. Multi-class multi-label perceptron

Following the idea introduced by BR, MMP trains a perceptron for each pair of labels in the dataset (Loza Mencia and Furnkranz, 2008). These binary models are stored in an ensemble in which the classes are ranked. A voting step is carried out at the end to reach the final prediction.

2.2.4. Attentive interpretable tabular learning

The introduction of Google's Attentive Interpretable Tabular Learning (TabNET) (Arik and Pfister, 2021) model was a step forward towards neural network-based tabular data classification since it outperformed dominant models such as XGBoost (Chen and Guestrin, 2016) on multiple datasets. TabNET was mainly designed for Natural Language Processing (NLP) applications; however, the proposed architecture can be used to complete MLC tasks regardless of the application. The proposed model employs an attention mechanism to prioritize certain features for each decision step (i.e., or in our case each label set).

The building block of the model is based on the same idea as the famous NLP's Transformer (Vaswani et al., 2017) architecture that resulted in models like BERT (Devlin et al., 2018). The main difference here is that the attention mechanism eliminates irrelevant features (instead of picking the most probable ones) at each decision step. Furthermore, the sparse masking approach lets the model mimic a "Decision Tree"-like approach by removing unrelated features step by step. The Sparsemax (Martins and Astudillo, 2016) function results in a soft feature selection process and improvements and makes the model more explainable.

2.3. Intrusion detection in the context of multi-label learning

Currently, the number of research works on MLC-based IDS is very limited. In fact, the majority of titles including the multi-label classification term refer to multi-class classification rather than multi-label learning. Nevertheless, there are a few papers that explore IDS in the context of MLC in the literature. To begin with, a multi-label version of the k Nearest Neighbor (kNN) classifier is used in combination with maximum a posterior principle to detect intrusions in KDD CUP99 dataset under a semi-supervised setting (Qian and Li, 2014). In addition, supervised multi-label learning is used in Roopa and Raja (2018) to design a behavioral-based IDS to secure mobile adhoc networks. Their IDS combines a rule-based approach with a multi-layer neural network that uses sigmoid activation.

3. Ensemble-based multi-label neural network

Fig. 1 presents the illustrative diagram of the proposed IDS method. Given a set of data matrix $X \subseteq \mathbb{R}^{m \times n}$, we assume each data vector $x_i \in X$ corresponds to a vector of labels defined as $y_i = \langle y_i^1, y_i^2, \ldots, y_i^\gamma \rangle$, where $1 \leq i \leq m$ and γ is the number of labels in each label vector. Similar to sample vectors, we consider a set of labels $Y \subseteq \mathbb{N}^{m \times \gamma}$, where $Y = \{y_1, y_2, \ldots, y_m\}$.

3.1. Design of the proposed algorithm

Using ensemble models, we propose modeling each label column in Y under a separate model. We call this structure an Ensemble-based Multi-Label Neural Network (EMLNN). EMLNN enables separate estimation of labels in each column, which facilitates capturing the label distribution due to the primary focus on a certain column of labels in each model. The results of these models will be aggregated once each model has made its prediction to reach the final prediction.

Algorithm 1 contains the pseudo-code of the EMLNN technique. The set of ensemble models is defined as $E = \{M_1, M_2, \ldots, M_\gamma\}$, in which each model M_j is trained w.r.t. y_i^j for all $y_i \in Y$, where $1 \leq j \leq \gamma$. Each M_j is a neural network model that predicts the class label corresponding to column j of Y. The utilized neural network is formulated as follows:

$$M_j(x_i) = (x_i, z_1, z_2, ..., z_L, \hat{y}_i^j), \tag{1}$$

Algorithm 1: EMLNN.

Input: Set of samples X and set of labels Y.
Output: Predicted set of labels $\hat{Y}$.
Definitions:
Card$(\cdot)$ returns the cardinality.
Unique$(\cdot)$ returns unique values.
Y^j denotes j-th column of Y.
Initialization:
Create an empty ensemble $E = \{\varnothing\}$
Initialize M_1 model.
Training:
1 **for** $\forall Y^j \in Y,\ 1 \le j \le \gamma$ **do**
2 c = Card$\left(\text{Unique}(Y^j)\right)$
3 **for** $\forall x_i \in X$ **do**
4 **for** $\forall z_l,\ 1 \le l \le L$ **do**
5 $z_l = \max(0, W_l z_{l-1}) \mid z_0 = x_i$.
6 **end for**
7 $h = W_{L+1} z_L$.
8 **if** $c > 2$ **then**
9 Compute z_{L+1} using Equation (4).
10 **else**
11 Calculate z_{L+1} using Equation (8).
12 **end if**
13 Obtain $\hat{y}_i$ through Equation (5).
14 Estimate L_j using Equation (6).
15 Update $\{W_l\}_{i=1}^{L}$ through gradient decent.
16 **end for**
17 $E \leftarrow E \cup M_j$.
18 Initialize the next model: $M_{j+1} \leftarrow M_j$.
19 **end for**

where z_l is the latent variable formed at hidden layer l, L is the number of hidden layers in M_j, and $\hat{y}_i^j$ is the prediction made by the network. To form each latent variable z_l, first, the output of the previous layer, or the input vector for z_1, is fed to a Relu activation (line 5 in Algorithm 1) as follows:

$$z_l = \max(0, W_l z_{l-1}), \tag{2}$$

where W_l is the weight matrix between layer l and its previous layer, and $z_0 = x_i$. After activation, the obtained representation will undergo a dropout step. For the final layer, a Softmax activation is used to obtain $\hat{y}^j$:

$$h = W_{L+1} z_L = \langle h_1, h_2, \dots, h_c \rangle, \tag{3}$$

$$z_{L+1} = \frac{e^{h_j}}{\sum_{j=1}^{c} e^{h_j}}, \tag{4}$$

where c is the number of classes (line 2 in Algorithm 1), and h is the vector containing the transformed values after applying the weight matrix at the output layer before activation (line 7 in Algorithm 1). Given that z_{L+1} is a vector in the form of $\langle p_1, p_2, \dots p_c \rangle$, $\hat{y}_i^j$ will be the label corresponding to the largest activation value, where c is the number of unique classes in j-th column of Y:

$$\hat{y}_i^j = \arg \max_{p_i \in z_{L+1}} \{p_i \in z_{L+1} \mid 1 \le i \le c\} \tag{5}$$

The network is trained by means of a cross-entropy loss function (line 14 in Algorithm 1) defined as in the following:

$$L_j = -\sum_{i=1}^{m} y_i^j \log \hat{y}_i^j \tag{6}$$

At the end of this iteration, weights of the network are updated based on the obtained L_j through gradient descent (line 15 in Algorithm 1). When M_j is trained, we initialize the parameters of M_{j+1} with those of M_j. Once each model M_j is trained on $\{X \cup Y^j\}$, the predictions are aggregated (line 17 in Algorithm 1) as shown in the following:

$$E(x_i) = \bigcup_{j=1}^{c} M_j(x_i) = \{\hat{y}_i^1, \hat{y}_i^2, \dots, \hat{y}_i^\gamma\} \tag{7}$$

Note that the designed model is formulated for multi-label and multi-class scenarios. In case of binary classes, we change Softmax activation into Sigmoid for the corresponding M (line 11 in Algorithm 1). By doing so, Equation (4) changes into:

$$z_{L+1} = \frac{1}{1 + e^{-W_{L+1} z_L}} \tag{8}$$

Once the training is over, E can be used as the multi-label mode, which takes an unlabeled sample vector, and each $M_j \in E$ predicts a different label resulting in a set of γ predictions.

3.2. Complexity analysis

For the sake of simplicity, we assume that all hidden layers have n neurons. The complexity of the initialization part of the algorithm involves creating an empty ensemble with $O(1)$ and initializing a neural network with L layers, each with n neurons, which yields a complexity of $O(Ln^2)$.

The complexity of the training process of the neural network is mainly affected by the matrix operations between each layer, activation, and gradient descent. Matrix multiplication adds a $O(Ln^3)$ complexity whereas activation functions cause linear complexity at each layer, $O(Ln)$. The complexity of gradient descent, however, is different for hidden layers and the final layer. It is known that gradient estimation for n neurons has a $O(n^2)$ complexity at the final layer (Hallaji et al., 2023). This is while this complexity increases to $O(n^3)$ for the rest of the hidden layers. Given that there are $L-1$ layers before the final layer, the complexity order of this step would be $O(n^2) + O\left((L-1)n^3\right)$.

The process of training each neural network model M_j is repeated for γ times, which equals the number of label sets. Moreover, the training process should be repeated for m samples (i.e., we omit the batching process for the sake of simplicity). This will multiply the complexity of training one model by $m\gamma$. Adding the complexities of these steps together results in:

$$O(1) + m\gamma\left(O(Ln^2) + O(Ln^3) + O(Ln) + O(n^2) + O\left((L-1)n^3\right)\right)$$

Finally, taking the maximum complexity of each variable into account the above equation leads to $O(m\gamma Ln^3)$. Given that L and γ are very small compared to data size, the complexity further simplifies into $O(mn^3)$, which is equivalent to the complexity of a single-label neural network. Hence, the proposed EMLNN only increases the complexity by $O(\gamma)$, which can be disregarded due to its insignificance.

4. Experimental results

In this section, we initially explain the utilized setting for each of the compared methods. Then, the experimental results obtained from MLC classification are presented and analyzed in terms of Exact Match Ratio (EMR) and Hamming Loss (HL), as formulated in the following:

$$EMR = \frac{1}{m} \sum_{i=1}^{m} I(y_i = \hat{y}_i), \tag{9}$$

$$HL = \frac{1}{m\gamma} \sum_{i=1}^{m} \sum_{j=1}^{\gamma} I(y_i^j \neq \hat{y}_i^j), \tag{10}$$

where $I(\cdot)$ is an indicator function that returns one if the input condition is satisfied, and zero otherwise. Moreover, we use the area under the Receiver Operating Characteristic (ROC) curve to evaluate the overall intrusion detection performance in addition to class-wise accuracy. Standard deviation is also analyzed to assess the stability of the selected algorithms.

4.1. Simulation data

The first experiment is performed on UNSW-NB15 dataset (Moustafa and Slay, 2015), which contains several types of data files such as pcap,

Table 1
Label sets considered in UNSW-NB15 and Bot-IoT datasets.

Data	Label set 1		Label set 2	
	Label	Attack	Label	Service
UNSW-NB15	1	Normal	1	None
	2	Reconnaissance	2	FTP
	3	Backdoor	3	SMTP
	4	DoS	4	SNMP
	5	Exploits	5	HTTP
	6	Analysis	6	FTP-Data
	7	Fuzzers	7	DNS
	8	Worms	8	SSH
	9	Shellcode	9	Radius
	10	Generic	10	POP3
	-	-	11	DHCP
	-	-	12	SSL
	-	-	13	IRC
Bot-IoT	1	DDoS	1	HTTP
	2	DoS	2	Key logging
	3	Normal	3	Normal
	4	Reconnaissance	4	OS Fingerprint
	5	Theft	5	Service Scan
	-	-	6	TCP
	-	-	7	UDP

Argus, Bro, and CSV files for evaluating network intrusion detection systems. The data is generated using three servers, two of which always generate benign data whereas the third one acts maliciously. Both benign and malicious traffic are generated using IXIA PerfectStorm tool. Details of feature extraction from these files to make a dataset processible by machine learning algorithms can be found in Moustafa and Slay (2015). The collection data in UNSW-NB15 includes authentic modern normal and abnormal network traffic. The data has 2,540,044 samples and 49 features. Train and test subsets are sampled from the original dataset with 175,341 and 82,332 samples, respectively. We consider two sets of labels for attacks and services, as listed in Table 1.

The second experimental data is a combination of normal and botnet traffic collected into the Bot-IoT dataset (Koroniotis et al., 2019). Bot-IoT data comes in several formats such as PCAP, Argus, and CSV; however, only CSV files are considered in these experiments. The data contains two sets of labels, namely category and subcategory. The former specifies the type of attack whereas the latter indicates the utilized protocol. Specific details about the feature extraction process that led to the prepared CSV file can be found in Koroniotis et al. (2019). Detailed labeling of this dataset used in this work is shown in Table 1.

While the utilized datasets may not comprehensively represent the current threat landscape, the goal is to showcase the ability of the proposed model in capturing attack patterns from the provided training dataset and identifying those patterns in each label set. Upon industrial use, it is recommended to fine-tune the model using a more comprehensive dataset that encompasses specific characteristics of the selected network or system.

4.2. Experimental setting

Table 2 lists the parameter setting of each MLC method used in our experiments. LP and RAkEL are combined with MLP to focus our study on neural networks. All methods are optimized using the Adam optimizer (Kingma and Ba, 2015) with a batch size of 100 and through 1000 epochs.

4.3. Results analysis

In order to ensure the statistical reliability of the results, all experiments are repeated ten times. Fig. 2 shows the overall performance of MLC methods through all experiments. In this figure, solid circles show the performance distribution for each algorithm over several experiments. Solid squares and plus signs are used to indicate the average performance and outliers of the performance distribution. In addition, the height of each box implies the variance of the results.

Table 2
Parameter setting of MLC techniques.

Algorithms	Parameter Setting
LP	Classifier: MLP
MMP	N/A
RAkEL	$k = 2$
TabNET	Feature re-usage in mask = 1.3 Mask type: entmax #. independent phases = 2 #. share phases = 2 Attention hidden size = 8 Shared layer hidden size = 32 Hidden layer size = 16
EMLNN	Learning rate = 0.001 Layer size = [256, 256, 128, 64] Dropout ratio = {0.2, 0.4}
MLP	#. hidden layers = 4 Learning rate = 0.001 Layer size = [512, 256, 128, 64]

Results are divided into four groups based on the recorded performance measure and the selected dataset. Fig. 2(a, c) shows the experimental results in terms of HL. HL indicates the relative correctness of results, that is it also takes partial correctness in label prediction into account. It can be seen in Fig. 2(a) that EMLNN outperforms all other methods in terms of HL (i.e., lower HL means better performance) in experiments with UNSW-NB15 dataset. This is while LP, the simplest of the selected algorithms, seems to yield a higher HL compared to the rest of the methods. TabNET, RAkEL, and MMP are ranked second to fourth. In terms of stability, EMLNN, MMP, and TabNET seem to exhibit a similar amount of variance in the results, placing above RAkEL and LP in our comparison. These conclusions are also confirmed in Fig. 3, which illustrates the difference between the averaged performance of the selected methods. Fig. 3(a) indicates that the lowest standard deviation is resulted by TabNET, with a slight difference from that of EMLNN. It can be also seen from this figure that LP and MMP perform similarly in terms of average HL. The same behavior is observed between TabNET and RAkEL.

Another indicator in evaluating the performance of MLC methods is EMR, which estimates the ratio of samples whose predicted labels are correct across all label sets. Fig. 2(b) illustrates the overall performance of all methods in terms of EMR. Similar to the previous analysis, EMLNN again outperforms all techniques. Moreover, MMP and LP are ranked fourth and fifth. However, in contrast to HL analysis, here RAkEL surpasses TabNET in terms of EMR. In other words, RAkEL results in more fully correct predictions whereas TabNET makes more partially correct predictions. Nevertheless, comparing the performance of both in Fig. 3(b), one can conclude that the difference is negligible and they are both on par. Fig. 3(b) also shows that EMLNN is followed by the lowest standard deviation. TabNET is almost as stable as EMLNN when looking at EMR. RAkEL and MMP, on the other hand, exhibit relatively higher standard deviations compared to their HL. LP, RAkEL, and MMP are ranked from third to fifth based on the standard deviation of recorded EMR.

Experiments with the Bot-IoT dataset result in similar conclusions, albeit with a slight difference. In terms of the overall HL and EMR, Fig. 2(c, d) shows that EMLNN outperforms all competitors in terms of both HL and EMR. In contrast to the previous experiment, Fig. 3(b, d) indicates that EMLNN also surpasses TabNET in terms of standard deviation for both performance measures. When the standard deviation is averaged for both datasets, they are on par with each other. Fig. 2 and Fig. 3 also show that all the selected algorithms generally perform better on Bot-IoT compared to the UNSW-NB15 dataset. RAkEL, TabNET,

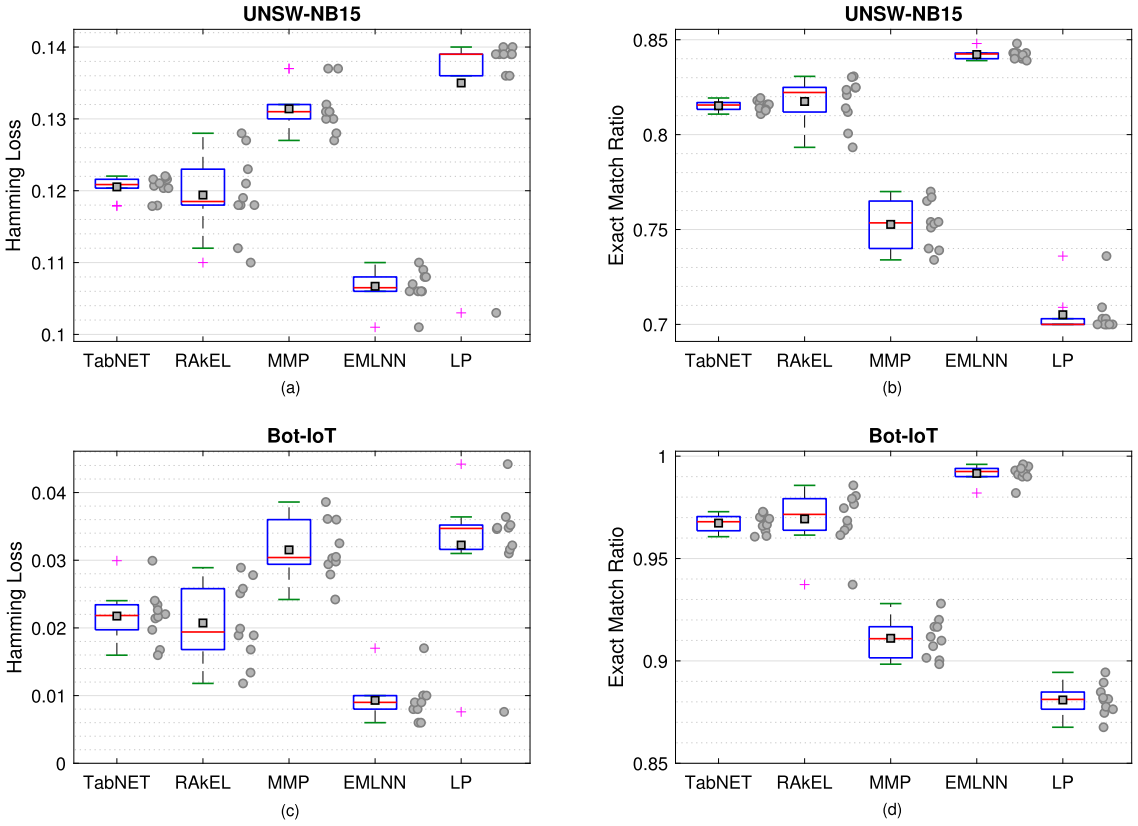

Fig. 2. Performance of MLC algorithms over 10 runs in terms of HL and EMR. Solid circles, squares, and plus signs denote recorded performance in each experiment, their average, and outliers, respectively.

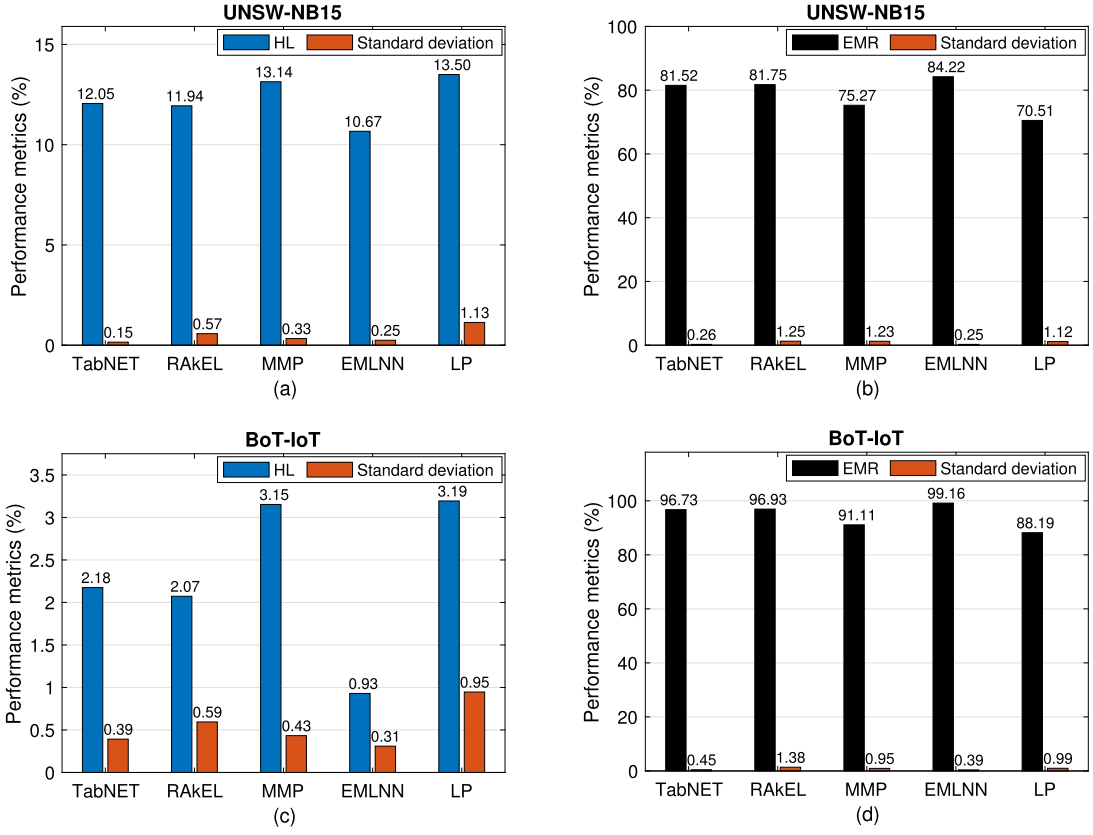

Fig. 3. Averaged performance and standard deviation of MLC methods in terms of HL and EMR. Results are devised based on experiments and performance measures.

Table 3
Ranking of MLC algorithms in terms of HL, EMR, and standard deviation. Lower numbers indicate better ranks.

Algorithms	HL	EMR	HL Std.	EMR Std.	Overall
LP	5	5	5	3	5
RAkEL	2	3	4	5	3
MMP	4	4	3	4	4
TabNET	2	3	1	2	2
EMLNN	1	1	2	1	1

Table 4
AUROC values associated with ROC curve in Fig. 4.

Dataset	EMLNN	MMP	TabNET	RAkEL	LP
UNSW-NB15	0.9161	0.8738	0.8976	0.9118	0.8483
Bot-IoT	0.9992	0.9473	0.9760	0.9862	0.9416
Rank	1	4	3	2	5

MMP, and LP can be ranked from second to fifth in terms of both HL and EMR for experiments with the Bot-IoT dataset. This is while TabNET and RAkEL are very close in terms of EMR. Furthermore, MMP and LP result in a roughly similar HL.

A review of the performed analysis is included in Table 3. Considering both HL and EMR, and the stability of algorithms, we conclude that EMLNN, TabNET, RAkEL, MMP, and LP are ranked from first to last. This indicates the effectiveness of the proposed EMLNN algorithm.

4.4. Security analysis

Fig. 4 illustrates the ROC curve resulting from intrusion detection based on the attack label sets. The positive class indicates the occurrence of an intrusion (i.e., the union of all attack classes). The Area Under the ROC curve (AUROC) is an indicator of the intrusion detection success rate. It can be seen in Fig. 4(a) that EMLNN and RAkEL result in the largest area under the ROC curve in experiments with UNSW-NB15. Table 4, which lists the precise estimated values of AUROC, confirms this statement. Moreover, comparing the AUROC values, one can conclude EMLNN, RAkEL, TabNET, MMP, and LP are ranked first to last in terms of the success rate of intrusion detection on UNSW-NB15. In contrast to the previous analysis of the classification results, RAkEL outperforms TabNET for intrusion detection. This is probably due to the higher dependency of the intrusion detection process on the distribution of all attacks rather than separate distributions associated with each attack subcategory. Fig. 4 also illustrates the results of intrusion detection for Bot-IoT dataset. Comparing the curves in this figure, it can be concluded that EMLNN, RAkEL, TabNET, MMP, and LP can be ranked from first to last for this experiment as well. Moreover, it seems that the IDS can handle cyber threats more efficiently on this dataset. This could be due to the smaller number of attack types included in this data.

Fig. 5 shows the class-wise detection accuracy estimated for each group of cyber-attacks separately. The results are divided into two heatmaps for UNSW-NB15 and Bot-IoT datasets. In addition, a color index is shown beside each heatmap that defines a color spectrum across the range of recorded measurements. The warmer colors indicate higher performance whereas the cooler colors are associated with lower performance. The color spectrum helps in observing the overall performance in identifying each attack and using each algorithm more conveniently.

As shown in Fig. 5(a), EMLNN has the highest ratio of correct predictions for each attack type for both datasets. For all methods, it seems Analysis attacks are easily distinguished from other attack types. In contrast, Reconnaissance and Backdoor attacks are more challenging to deal with. The normal class which is associated to benign traffic is also not detected desirably by TabNET and RAkEL, which means they result in a higher false alarm rate. In addition, Exploit attacks are not recognized with a high success rate for most algorithms. Regardless, EMLNN exhibits the best performance in identifying Backdoor and Reconnaissance attacks. DoS attacks are robustly identified by EMLNN, TabNET, and RAkEL. However, LP and MMP do not show optimal performance on DoS attacks. Moreover, Worms are identified roughly similarly by all algorithms. Fuzzers are also best distinguished using EMLNN, TabNET, and RAkEL. Although the performance of MMP and LP is somewhat acceptable on Fuzzers, the aforementioned methods lead to a better success rate in Fuzzer identification. Shellcode attacks challenge all MLC algorithms except for EMLNN and TabNET. Finally, the only technique that can robustly identify generic attacks in these experiments is EMLNN. Fig. 5(b) lists class-wise performance for attacks considered in Bot-IoT dataset. While the majority of attacks are detected with satisfying performance for most algorithms, it appears that detecting data theft attacks is more challenging for all detectors. In contrast to the results of Fig. 5(a), the normal class is identified with the highest accuracy by all algorithms. EMLNN, RAkEL, and TabNET identify reconnaissance, DoS, and DDoS attacks with satisfying performance; however, LP and MMP fall behind others in detecting these classes.

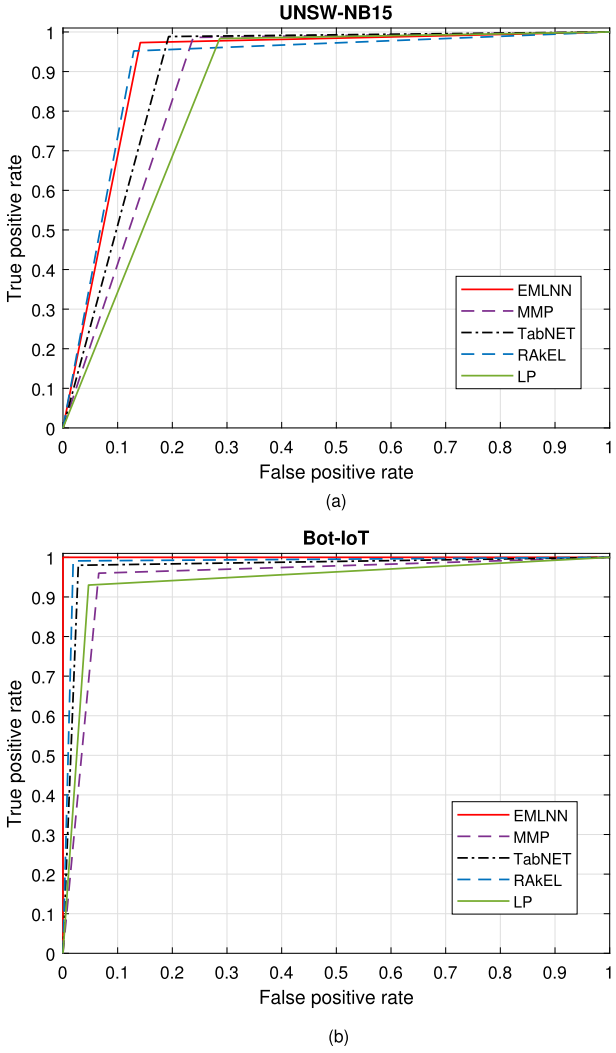

Fig. 4. ROC curve obtained based on the intrusion detection results (i.e., attack versus benign) using the attack label sets.

4.5. Computational resources

Experiments were simulated in Python using TensorFlow in a Conda virtual environment created on Windows Subsystem for Linux (WSL) on Ubuntu kernel. Experiments were executed on a computer equipped

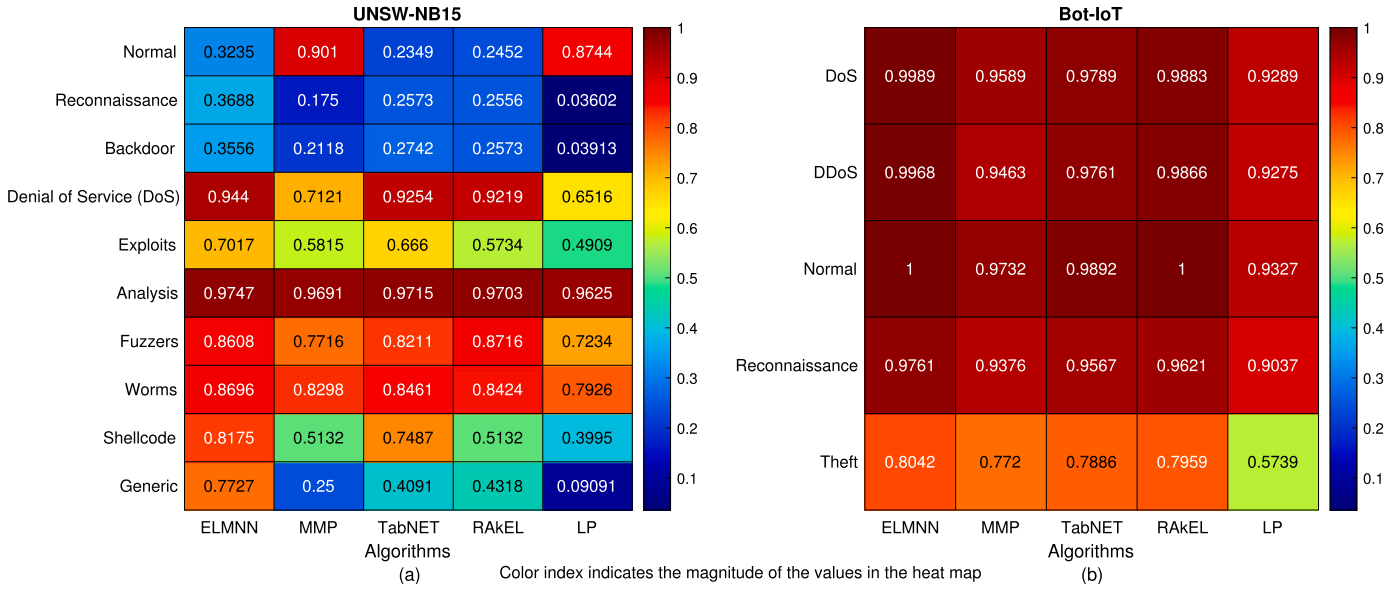

Fig. 5. Class-wise detection accuracy for each of the cyber-attacks using MLC algorithms. Results for UNSW-NB15 and Bot-IoT datasets are shown in separate heat maps. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Table 5
Averaged resource usage of EMLNN algorithm in all experiments.

Metrics	UNSW-NB15	Bot-IoT
Memory usage	4.9 GB	4.8 GB
GPU usage	9 GB	9 GB
CPU usage	13%	13%
Train run time	11,080 s	66,395 s
Test run time	6.8 s	57 s
Train speed	15.82 sample/s	15.79 sample/s
Test speed	12,108 sample/s	12,872 sample/s

with RTX 3080 GPU with 12Gb of memory, Intel Core i7-12700 processor, and 32 Gb of RAM.

Table 5 shows the averaged resource usage of EMLNN through the aforementioned experiments. Using the hardware mentioned above, we achieved an inference rate above 12,000 samples per second. The aforementioned setting sufficed for our research purpose and highlighted the effectiveness of EMLNN in performing multi-label analysis while identifying intrusions. In an industrial setting, equipping the server with high-end GPUs, running the algorithm natively on a Linux kernel, and using faster languages such as C++ will significantly boost the processing speed.

5. Conclusion

A novel MLC algorithm, EMLNN, was proposed to tackle efficiency and prediction performance issues existing in MLC problems. The proposed method works based on ensemble models and deep learning. This approach trains a set of sequential neural network models separately, where each model only targets a specific label set. Moreover, once each model is trained, the network parameters are used to initialize the next model. This ensures transferring knowledge of each label set to the next model. This training scheme allows each model to concentrate on learning the targeted label set without compromising the prediction performance by involving other label sets in the learning objective. Another advantage of this structure is that, in contrast to the majority of MLC solutions in the literature, EMLNN does not tamper with the label and data distribution (e.g., binarizing the problem or merging label sets into one). This prevents further complications such as causing class imbalance or insufficient labeled data for certain combinations. The proposed method is also evaluated in a real-world case of intrusion detection. Experimental results and analysis indicate the superiority of the proposed algorithm over comparable methods in terms of HL, EMR, and standard deviation.

CRediT authorship contribution statement

Ehsan Hallaji: Data curation, Formal analysis, Investigation, Methodology, Validation, Visualization, Writing – original draft. **Roozbeh Razavi-Far:** Conceptualization, Project administration, Supervision, Writing – review & editing. **Mehrdad Saif:** Funding acquisition, Project administration, Resources, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The code and the utilized dataset are publicly available on: https://github.com/h-ehsan/EMLNN.

Acknowledgement

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under funding reference numbers CGSD3-569341-2022 and RGPIN-2021-02968.

References

Apruzzese, G., Laskov, P., Schneider, J., 2023. SoK: pragmatic assessment of machine learning for network intrusion detection. In: 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P), pp. 592–614.

Arik, S.O., Pfister, T., 2021. Tabnet: attentive interpretable tabular learning. Proc. AAAI Conf. Artif. Intell. 35 (8), 6679–6687.

Arp, D., Quiring, E., Pendlebury, F., Warnecke, A., Pierazzi, F., Wressnegger, C., Cavallaro, L., Rieck, K., 2022. Dos and don'ts of machine learning in computer security. In: Proc. of USENIX Security Symposium.

Boutell, M.R., Luo, J., Shen, X., Brown, C.M., 2004. Learning multi-label scene classification. Pattern Recognit. 37 (9), 1757–1771.

Catillo, M., Pecchia, A., Villano CPS-GUARD, U., 2023. Intrusion detection for cyber-physical systems and IoT devices using outlier-aware deep autoencoders. Comput. Secur. 129, 103210.
Chen, T., Guestrin, C., 2016. XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16. ACM, pp. 785–794.
da Costa, K.A., Papa, J.P., Lisboa, C.O., Munoz, R., de Albuquerque, V.H.C., 2019. Internet of things: a survey on machine learning-based intrusion detection approaches. Comput. Netw. 151, 147–157.
Devlin, J., Chang, M., Lee, K., Toutanova, K., 2018. BERT: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805.
Fernandes, G., Rodrigues, J.J.P.C., Carvalho, L.F., Al-Muhtadi, J.F., Proença, M.L., 2019. A comprehensive survey on network anomaly detection. Telecommun. Syst. 70 (3), 447–489.
Ferrag, M.A., Maglaras, L., Moschoyiannis, S., Janicke, H., 2020. Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. J. Inf. Secur. Appl. 50, 102419.
Hallaji, E., Razavi-Far, R., Wang, M., Saif, M., Fardanesh, B., 2022. A stream learning approach for real-time identification of false data injection attacks in cyber-physical power systems. IEEE Trans. Inf. Forensics Secur. 17, 3934–3945.
Hallaji, E., Farajzadeh-Zanjani, M., Razavi-Far, R., Palade, V., Saif, M., 2023. Constrained generative adversarial learning for dimensionality reduction. IEEE Trans. Knowl. Data Eng. 35 (3), 2394–2405.
Jethanandani, M., Sharma, A., Perumal, T., Chang, J.-R., 2020. Multi-label classification based ensemble learning for human activity recognition in smart home. Int. Things 12, 100324.
Kingma, D.P., Ba, J., 2015. Adam: a method for stochastic optimization. arXiv:1412.6980.
Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B., 2019. Towards the development of realistic botnet dataset in the Internet of things for network forensic analytics: bot-IoT dataset. Future Gener. Comput. Syst. 100, 779–796.
Liu, W., Wang, H., Shen, X., Tsang, I.W., 2022. The emerging trends of multi-label learning. IEEE Trans. Pattern Anal. Mach. Intell. 44 (11), 7955–7974.
Loza Mencia, E., Furnkranz, J., 2008. Pairwise learning of multilabel classifications with perceptrons. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pp. 2899–2906.
Martins, A.F.T., Astudillo, R.F., 2016. From softmax to sparsemax: a sparse model of attention and multi-label classification. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning. In: ICML'16, vol. 48, pp. 1614–1623.
Moustafa, N., Slay, J., 2015. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: Military Communications and Information Systems Conference (MilCIS), pp. 1–6.
Qian, Y., Li, Y., 2014. An intrusion detection algorithm based on multi-label learning. In: IEEE Workshop on Electronics, Computer and Applications, pp. 602–605.
Read, J., Pfahringer, B., Holmes, G., Frank, E., 2011. Classifier chains for multi-label classification. Mach. Learn. 85 (3), 333–359.
Riera, T.S., Higuera, J.-R.B., Higuera, J.B., Herraiz, J.-J.M., Montalvo, J.-A.S., 2022. A new multi-label dataset for web attacks capec classification using machine learning techniques. Comput. Secur. 120, 102788.
Roopa, M., Raja, S.S., 2018. Intelligent intrusion detection and prevention system using smart multiinstance multi-label learning protocol for tactical mobile adhoc networks. KSII Trans. Int. Inf. Syst. 12 (6), 2895–2921.
Tsoumakas, G., Katakis, I., 2007. Multi-label classification: an overview. Int. J. Data Warehous. Min. 3 (3), 1–13.
Tsoumakas, G., Katakis, I., Vlahavas, I., 2010. Mining Multi-Label Data. Springer US, pp. 667–685.
Tsoumakas, G., Katakis, I., Vlahavas, I., 2011. Random k-labelsets for multilabel classification. IEEE Trans. Knowl. Data Eng. 23 (7), 1079–1089.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc.
Wang, S., Bi, S., Zhang, Y.-J.A., 2020. Locational detection of the false data injection attack in a smart grid: a multilabel classification approach. IEEE Int. Things J. 7 (9), 8218–8227.
Xie, J., Li, S., Zhang, Y., Sun, P., Xu, H., 2022. Analysis and detection against network attacks in the overlapping phenomenon of behavior attribute. Comput. Secur. 121, 102867.
Yang, Z., Liu, X., Li, T., Wu, D., Wang, J., Zhao, Y., Han, H., 2022. A systematic literature review of methods and datasets for anomaly-based network intrusion detection. Comput. Secur. 116, 102675.
Zhang, M.-L., Zhou, Z.-H., 2005. A k-nearest neighbor based algorithm for multi-label classification. In: 2005 IEEE International Conference on Granular Computing, vol. 2, pp. 718–721.

Ehsan Hallaji holds a B.Sc. degree in software engineering from Shahid Rajaee University, Iran (2015) and an M.A.Sc. degree in electrical engineering from the University of Windsor (2018), where he is currently pursuing a Ph.D. degree in electrical engineering. He has been the recipient of prestigious scholarships from organizations such as NSERC and the Ontario Ministry of Education. His research interests include machine learning, federated learning, and cybersecurity. He serves as a reviewer for various journals and conferences in his field. He served as the Vice-Chair of the IEEE Systems, Man, and Cybernetics Society, Windsor Section from 2019 to 2022.

Roozbeh Razavi-Far is an Assistant Professor at the Faculty of Computer Science and Canadian Institute for Cybersecurity, at the University of New Brunswick. His research focuses on machine learning, big data analytics, and cybersecurity. Stanford lists his name among the top two percent most cited researchers for 2022. He is the recipient of several awards and grants including NSERC-DG, NSERC-ECR, NBIF, USRG and NSERC-PDF. He is an Associate Editor at several journals, including *Neurocomputing*, *Machine Learning with Applications*, *Discover Artificial Intelligence*, and IEEE TRANSACTIONS ON INDUSTRIAL CYBER-PHYSICAL SYSTEMS. He was a Guest Editor and Chair for several journals and conferences.

Mehrdad Saif is a Professor with the Department of Electrical and Computer Engineering, University of Windsor, where he was the Dean of the Faculty of Engineering between 2011 and 2022. From 2002 to 2011, he was the Director of the School of Engineering Science, Simon Fraser University. He has been a consultant to several organizations, including GM, NASA, B.C. Hydro, and Ontario Council of Graduate Studies. His research interests include control systems and cyber-physical security. He is a member of the editorial board of IEEE ACCESS, IEEE SYSTEMS, MAN AND CYBERNETICS MAGAZINE, IEEE TRANSACTIONS ON INDUSTRIAL CYBER-PHYSICAL SYSTEMS, and IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS.