future internet

MDPI

Article

A Transferable Deep Learning Framework for Improving the Accuracy of Internet of Things Intrusion Detection

Haedam Kim, Suhyun Park, Hyemin Hong, Jieun Park and Seongmin Kim *

Department of Convergence Security Engineering, Sungshin Women's University, Seoul 02844, Republic of Korea; haedam615@naver.com (H.K.); suhyunpark0410@naver.com (S.P.); haemin1107@gmail.com (H.H.); pjieun73@gmail.com (J.P.)
* Correspondence: sm.kim@sungshin.ac.kr

Abstract: As the size of the IoT solutions and services market proliferates, industrial fields utilizing IoT devices are also diversifying. However, the proliferation of IoT devices, often intertwined with users' personal information and privacy, has led to a continuous surge in attacks targeting these devices. However, conventional network-level intrusion detection systems with pre-defined rulesets are gradually losing their efficacy due to the heterogeneous environments of IoT ecosystems. To address such security concerns, researchers have utilized ML-based network-level intrusion detection techniques. Specifically, transfer learning has been dedicated to identifying unforeseen malicious traffic in IoT environments based on knowledge distillation from the rich source domain data sets. Nevertheless, since most IoT devices operate in heterogeneous but small-scale environments, such as home networks, selecting adequate source domains for learning proves challenging. This paper introduces a framework designed to tackle this issue. In instances where assessing an adequate data set through pre-learning using transfer learning is non-trivial, our proposed framework advocates the selection of a data set as the source domain for transfer learning. This selection process aims to determine the appropriateness of implementing transfer learning, offering the best practice in such scenarios. Our evaluation demonstrates that the proposed framework successfully chooses a fitting source domain data set, delivering the highest accuracy.

Keywords: IoT; intrusion detection; transfer learning

Citation: Kim, H.; Park, S.; Hong, H.; Park, J.; Kim, S. A Transferable Deep Learning Framework for Improving the Accuracy of Internet of Things Intrusion Detection. *Future Internet* **2024**, *16*, 80. https://doi.org/10.3390/fi16030080

Academic Editor: Massimo Cafaro

Received: 8 January 2024
Revised: 22 February 2024
Accepted: 25 February 2024
Published: 28 February 2024

1. Introduction

The rapid innovation in Internet of Things (IoT) technology has accelerated data-driven strategies, harnessing vast volumes of data originating from a tremendous number of IoT devices in various domains, including digital healthcare [1], smart factories [2], industrial control systems [3], and transportation systems [4]. The IoT service market is poised to reach $575 billion in 2027 with an impressive annual growth rate of 18.8% [5]. Nevertheless, the ubiquitous deployment of IoT devices poses numerous threats to IoT networks, such as Distributed Denial-of-Service (DDoS), Botnets, and Backdoors, echoing the security concerns commonly encountered in traditional networked systems [6]. Unfortunately, the conventional Network Intrusion Detection Systems (NIDSs) based on Deep Packet Inspection (DPI) with pre-defined rulesets to address vulnerabilities within IoT networks are gradually losing their efficacy. Such ineffectiveness is attributable to the growing prevalence of zero-day attacks and the widespread adoption of encryption protocols [7].

To address this problem, researchers have leveraged machine learning algorithms on intrusion detection systems to analyze incoming network traffic to identify suspicious or anomalous activities in IoT environments [8–12]. Based on the labeled data categorizing packets as either benign or malicious, these models acquire the ability to discern patterns associated with different types of attacks. Throughout the training phase, relevant features representing various aspects of network behavior (e.g., inter-arrival time and flow

rates) are used [13–15]. One of the remarkable advantages of ML-based intrusion detection mechanisms is their capacity to detect unforeseen attacks and adapt to varying IoT network environments.

Nonetheless, there are several obstacles to building an effective intrusion detection model. Unlike rule-based Deep Packet Inspection (DPI), the ML-based approach is susceptible to generating false alarms, potentially misidentifying legitimate activities as malicious. The quality and completeness of the training data profoundly impact the model's performance, particularly its ability to detect unseen attacks. In practice, obtaining sufficient labeled data for various attack types is challenging. Furthermore, the class imbalance (e.g., benign data substantially outweighing malicious ones) within the training data can introduce bias into the model, and, consequently, overfitting becomes a risk in such scenarios. These limitations become even more pronounced in the diverse and dynamic landscapes of IoT networks and environments, accentuating the challenges faced in developing robust intrusion detection systems.

Notably, the emergence of Transfer Learning [16] helps IDSs improve the detection performance in the context of IoT environments. Transfer learning entails the process of addressing a new problem, known as the target domain, by leveraging knowledge gleaned from a previously well-understood domain, termed the source domain. This approach capitalizes on the wealth of knowledge from the source domain to bolster performance and expedite the learning process in the target domain. Researchers have elaborated on enhancing the detection accuracy of IDSs in IoT environments by utilizing various machine learning and deep learning models with knowledge transfer [17–19]. These endeavors have predominantly focused on enhancing the models themselves by exploring knowledge transfer strategies. However, there remains a gap when it comes to determining which data sets are suitable as source domains. From the perspective of the target domain, it remains unclear which data set is appropriate as a source domain to optimize the margin of improvement and contribute most effectively to the transfer learning process.

In this paper, we propose a novel framework to identify the most analogous source domain to a given target domain, leveraging deep learning as our foundational approach. Through a meticulous comparative analysis of the outcomes of similarity assessments and intrusion detection based on deep transfer learning, we significantly enhance the overall rigor of our experimental methodology. To explain in detail, similarity assessment consists of a framework that uses deep learning to determine which source domains are most similar to the target domain when there are multiple source domains. Furthermore, in intrusion detection based on deep transfer learning, we conduct an accuracy experiment, which exists to make sure that the results obtained in a similarity assessment are appropriate. Then, by comparing the results of these two steps, the source domain that is most similar to the target domain is finally selected. This study can provide a guide for identifying which data sets are best suited to the source domain in the transfer learning process and, in turn, contribute to improving the detection accuracy of IDSs in IoT environments. By utilizing four public intrusion detection data sets, we demonstrate the effectiveness of our framework in successfully choosing a fitting source domain data set, ultimately achieving the highest performance in transfer learning.

The organizational structure of this paper is outlined as follows: Section 2 is dedicated to substantiating the research's necessity through an exhaustive examination of the relevant literature. Section 3 systematically organizes crucial contextual information. Subsequently, in Section 4, we delve into an intricate exposition of the framework conceived by our research team, accompanied by a detailed presentation of the experimental methodology executed within this framework. Section 5 adeptly illustrates the results obtained through the proposed experimental approach. Lastly, Section 6 summarizes our research findings and delineating prospective avenues for future research endeavors.

2. Related Work

2.1. ML-Based Intrusion Detection

For several decades, there has been a surge in research pertaining to Intrusion Detection Systems (IDSs), aiming to enhance intrusion detection performance across diverse security frameworks. Rule-based Deep Packet Inspection (DPI) has persisted as the dominant approach in network-based intrusion detection, particularly for proactively alerting and the identifying anomalies. Nonetheless, the situation exacerbates with the emergence of attack variants that exploit zero-day vulnerabilities, rendering the existing rule-sets for pattern matching increasingly ineffective and obsolete [7]. Moreover, the pervasive adoption of cryptographic protocols (such as HTTPS, SSH, and SSL) to establish end-to-end encryption has gained substantial traction, posing a formidable challenge to the comprehensive scrutiny of packet payloads [20,21].

Within this realm, there is a notable emphasis on leveraging machine learning techniques to improve the efficacy of intrusion detection processes. Numerous methodologies and models have emerged with the intent of bolstering the efficiency of IDSs, catering to the detection of an array of attack types. Notably, a significant focus lies on scrutinizing intrusion detection approaches that harness deep learning strategies. M. El-Shrkaway et al. [8] proposed the Multi-layer Feature Selection and Reduction IDS (MFSR-IDS) algorithm to provide a high level of protection against DoS and Probe attacks using an anomaly-based approach. Liloja et al. [11] developed an effective intrusion detector using both a deep learning model and a machine learning model. To obtain high-accuracy scores, a deep learning model was used in the process of extracting features, and a machine learning model was used to measure the intrusion detection performance of intrusion detection. Andrea Ranieri et al. [12] proposed a 'deep adversarial learning response' to machine learning attacks targeting traffic from IoT devices using AdaBoost, DT, KNN, RF, and neural networks metrics. At this time, a new test bed was developed and tested, which collected the network traffic data set of IoT devices in general use situations such as media playback and then changed the statistics of outbound traffic using deep adversarial learning technology.

In a study by Nana Kwame Gyampf et al. [9], a binary data set was converted to a binary code format, which was further converted into a file. Based on the hex code generated from the file, the acquired binary image was fed into a Convolutional Neural Network (CNN). The authors argued that while conventional models generally demonstrated improved performance, the CNN-based models exhibited even higher performance levels in terms of accuracy. Riccardo Recori et al. [10] constructed a large integrated data set by merging multiple data sets. They then proceeded with experiments using multi-classification based on types of attacks.

We note that previous studies focused on improving intrusion detection performance through ML-based and DNN-based feature analysis and selection, whereas our research differentiates itself by focusing on evaluating domain similarity to leverage deep transfer learning.

2.2. Transfer Learning

Transfer learning [16] is a machine learning paradigm encompassing the utilization of acquired knowledge from a source model to enhance the performance on another distinct yet interconnected target model. It empowers models to extract broader insights from previously assimilated features to address restricted data availability and computational constraints. The utility of transfer learning spans diverse domains, including but not limited to computer vision [22,23], natural language processing [24,25], and anomaly detection [26,27], where pre-trained models can be fine-tuned for specific tasks. Transfer learning has demonstrated significant success in cases where labeled data for a target task are limited [28], making it an essential technique in modern machine learning. Figure 1 illustrates the comparison between transfer learning and traditional machine learning.

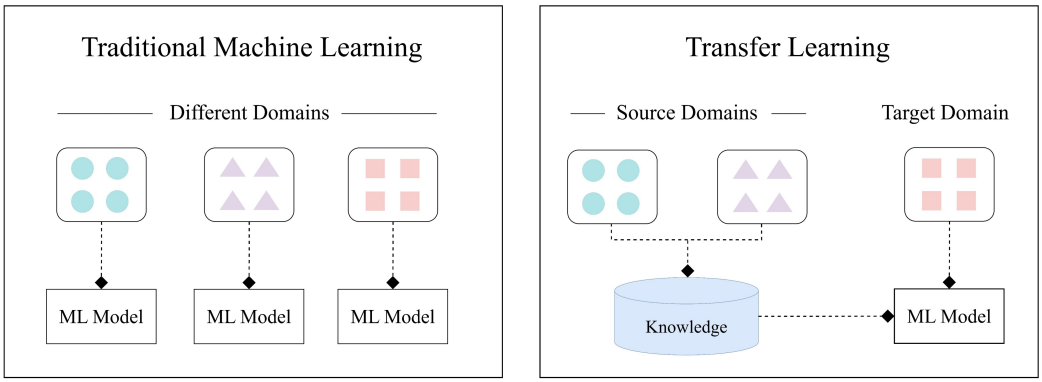

Figure 1. Traditional machine learning vs. transfer learning.

M. El-Shrkaway et al. [17] proposed a novel Deep Transition Learning (DTL) method to learn from data collected from multiple unlabeled IoT devices. Labeled and unlabeled samples were trained on a data set containing both source and target domains, respectively, and after training, MMD-AE was used to predict IoT attacks in incoming traffic in the target domain. MMD-AE is divided into AE1 and AE2, and it uses the Maximum Mean Mismatch (MMD) metric to minimize the distance between AE1's multiple hidden layers and AE2's multiple hidden layers. MMD has helped improve the effectiveness of knowledge transmitted from source to target domains in IoT attack detection systems.

To simulate a zero-day attack scenario, a study by Islam Debicha et al. [18] evaluated the performance of well-detecting attacks that were not present in the training phase when tested. Based on the concept of transfer learning, a new neural network was designed through the fine-tuning of parameters in the original deep neural network. The aim was to enhance performance by leveraging transfer learning to create a new deep neural network through parameter fine-tuning in existing networks.

A study by Ehsan Tanghatari et al. [19] proposed an approach to distribute DNN training across IoT edge devices based on transfer learning. Delegating DNN training onto edge devices from the central cloud reduces communication costs and preserves data privacy on edge devices. The authors demonstrated that the proposed scheme only incurs a 3.5% loss of accuracy compared to the conventional DNN training conducted on the cloud.

The primary objective of our framework is to identify a suitable source domain data set from among potential candidates, which is important in the context of IoT ecosystems due to its diverse and heterogeneous nature. For a modeler aiming to construct an intrusion detection classifier, computational resources may be limited, making it essential to prioritize the selection of an appropriate source domain for implementing transfer learning. It is noteworthy that existing approaches fall short in providing a definitive answer to this challenge, whereas our framework excels in this regard. Our framework embodies a complete process, starting from a proper source data set selection appropriate for the target domain to conventional instance-based transfer learning. We note that the state-of-the-art transfer learning techniques [29] and improved models [6,30] are applicable in the transfer learning phase of our proposed framework (see Section 4.3). In summary, our ultimate goal is to surpass the mere utilization of transfer learning and strive for efficient adaptation to new IoT environments.

3. Intrusion Detection Data Sets

In this study, we chose four representative intrusion detection data sets that are publicly available, as summarized in Table 1. These data sets not only provide flow characteristics extracted from unprocessed TCP/UDP packet capture traces but also encompass labels denoting whether a given flow corresponds to a specific attack or benign activity. They commonly feature a substantial number of instances classified into both benign and diverse attack categories, rendering them appropriate for the implementation of machine

learning-based intrusion detection methodologies. Note that two of these data sets (Bot-IoT and IoT Intrusion data sets) were curated specifically for IoT environments, while the remaining data sets (CIC-IDS2017 and UNSW-NB15 data set) cater to generic intrusion detection scenarios. Table 1 summarizes the specifications of data sets.

Table 1. Data set description.

		Bot-IoT	IoT Intrusion	CIC-IDS2017	UNSW-NB15
Type of Attack	Benign	o	o	o	o
	DDoS	o	-	-	-
	DoS	o	o	o	o
	Mirai	-	o	-	-
	Botnet	-	-	o	-
	Worms	-	-	-	o
	Reconnaissance	o	-	-	o
	PortScan	-	o	o	-
	MITM	-	o	-	-
	Web Attack	-	-	o	-
	Exploits	-	-	-	o
	Fuzzers	-	-	-	o
	SSH-Patator	-	-	o	-
	FTP-Patator	-	-	o	-
	Analysis	-	-	-	o
	Backdoor	-	-	-	o
	Shellcode	-	-	-	o
	Generic	-	-	-	o
	Theft	o	-	-	-
	Infiltration	-	-	o	-
	Heartbleed	-	-	o	-
Number of Instances	Attack	3,668,045	5,805,710	846,248	321,283
	Benign	477	40,073	2,273,097	2,218,760
Number of Attack Types		4	4	9	9
Number of Features		46	83	84	49

3.1. Bot-IoT Data Set

The Bot-IoT data set [14] is a data set created by designing a real-world network environment in UNSW Canberra's Cyber Range Lab and was collected in an environment through a combination of general and botnet traffic. In the positive scenario, a typical smart home is designed and configured in a testbed environment, and the normal traffic of a realistic smart home network is generated using a total of five IoT devices: a smart refrigerator, smart garage door, weather monitoring, smart lighting, and smart thermostat. In malicious scenarios, malicious traffic is generated through probing attacks that collect information through remote system scanning such as fingerprint collection, DoS, and DDoS using the Hping3 tool, and keylogging attacks for data theft, DDoS, DoS, OS, and service scanning, including keylogging and data exfiltration attacks.

3.2. IoT Intrusion Data Set

This open-source data set [31] was initially created by IEEE DataPort and is a new data set created using new technologies and detection algorithms requiring well-designed data sets for IoT networks. The entire IoT testbed architecture is a typical smart home environment. It includes two smart home devices that generate data, SKT NGU and EZVIZ Wi-Fi cameras, and smartphones and computers are connected via wireless networks to attack other IoT devices. The SKT NGU (NU100) and EZVIZ Wi-Fi cameras (C2C Mini O Plus 1080P) are IoT victim devices and include four types of attacks: Mirai, DoS, MITM, and Scanning. Then, we use the CICflowmeter log to extract features from the pcap file and generate a CSV, where the features consist of 80 network features and three labels.

3.3. CIC-IDS2017 Data Set

The CIC-IDS2017 data set [13] comprises unprocessed packet capture traces of network transactions encompassing both legitimate and malicious activities, which is well-suited for the assessment of network intrusion detection methodologies [32,33]. The data set comprises PCAP and CSV formats, facilitating the analysis of network flows, notably featuring 80 attributes extracted through CICFlowmeter [34], a comprehensive toolchain for processing raw packet capture traces. By profiling 25 distinct categories of user behaviors generated in a virtual environment, it encompasses network traffic derived from HTTP, HTTPS, FTP, SSH, and email protocols, amassed over a span of five days. The data set contains eight distinct attack classifications, namely, Brute Force FTP, Brute Force SSH, Denial of Service (DoS), Heartbleed, Web Attack, Infiltration, Botnet, and Distributed Denial of Service (DDoS).

3.4. UNSW-NB15 Data Set

The UNSW-NB15 data set [15] is a hybrid data set developed by the CyberScope Laboratory at UNSW Canberra in 2015, collecting genuine contemporary normal activities alongside synthetically generated modern attack behaviors. The attack behaviors were created using the IXIA PerfectStorm (https://www.ixiacom.com/products/perfectstorm, accessed on 1 January 2024), and the feature extraction process involves employing the Bro IDS [35]. This data set comprises nearly 2 million packet flows with 49 flow features, and the malicious traffic corresponds to nine distinct attack categories, such as DoS Exploits, Backdoors, Shellcode, and Worms.

4. Framework Design

This section outlines the proposed framework for knowledge transfer, aimed at identifying an appropriate source domain for intrusion detection in the IoT environment. The ultimate goal is to build a binary classifier for determining whether the given flow from the target domain is malicious or benign based on the knowledge transfer. The framework is structured around two key phases: (1) similarity assessment with the target domain (Phase 1) and (2) the application of deep transfer learning for intrusion detection (Phase 2). An overview of the entire process is depicted in Figure 2.

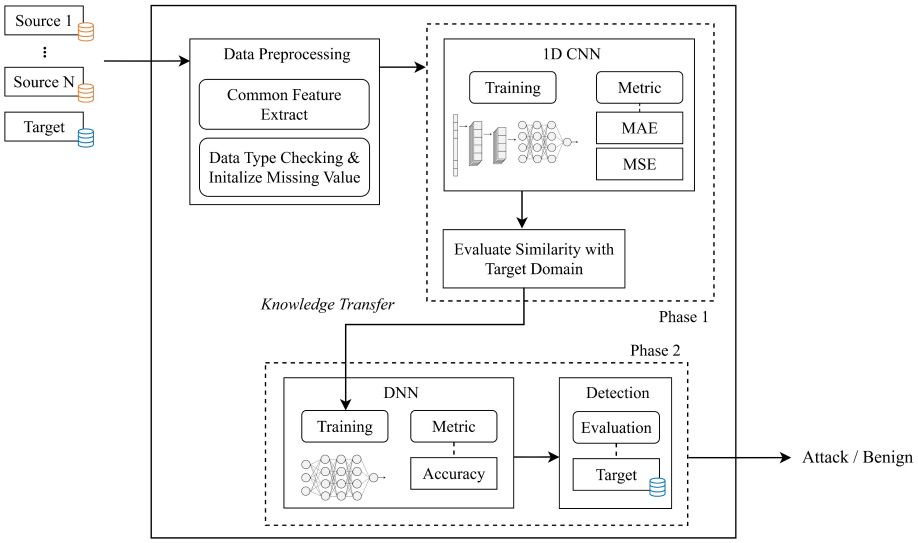

Figure 2. A system overview of the proposed framework.

First, candidate source data sets, along with the target domain data set, undergo a preprocessing phase. Then, a metric is employed to assess the similarity between these data-sets. This evaluation serves to determine which source domain exhibits the highest

degree of similarity with the target domain. The insights achieved from Phase 1 are then leveraged in Phase 2, where a model is constructed for handling inference data originating from the target domain.

4.1. Preprocessing

Given the diverse deployment scenarios and specialized devices inherent in the IoT ecosystem, it is essential to acknowledge that the data collection environments for each candidate source data set can significantly vary. These differences manifest in various ways, including variations in the number of features, data set composition, and the quantities of attack and benign instances. Consequently, preprocessing is necessarily required to unify the feature configurations across multiple data sets and address uninitialized values before embarking on the knowledge transfer process. This ensures that the source and target domains are effectively aligned, evaluating the similarity between data sets and facilitating a seamless transfer of knowledge between them.

Based on a feature analysis of the four data sets that we utilized, we first identified a common set of 12 features shared among all the source domain candidates and the target domain data set. Table 2 summarizes the result.

Table 2. Common feature description.

No.	Feature Names	Feature Description
1	Source IP	Source IP address
2	Source Port	Source port number
3	Destination IP	Destination IP address
4	Destination Port	Destination port number
5	Protocol	Internet Protocol used
6	Flow Duration	Duration of the flow in microsecond
7	Fwd Packet Length Mean	Mean size of packet in forward direction
8	Bwd Packet Length Mean	Mean size of packet in backward direction
9	Total Fwd Packets	Total packets in the forward direction
10	Total Bwd Packets	Total packets in the backward direction
11	Timestamp	Timestamp of the packet
12	Label	Category of attack or normal

To acquire common features across the four data sets, we focused on unifying features whose names were different but virtually identical in representing flow aspects (e.g., sbytes in Bot-IoT and Fwd_Pkt_Len_Mean in IoT Intrusion data sets). We note that the four data sets used in our study are representative and widely used data sets with extensive flow characteristics, which means that our approach can be extended to apply to other IoT intrusion detection data sets. The remaining features include 'Source IP', 'Source Port', 'Destination IP', 'Destination Port', 'Protocol', 'Flow Duration', 'Fwd Packet Length Mean', 'Bwd Packet Length Average', 'Total Delivered Packets', 'Total Bandwidth Packets', 'Timestamp', and 'Label', which signify the maliciousness of the data. We also note that the previous studies showed that anomaly detection performance is affected by feature importance (e.g., the weight of the information gain) rather than the sheer number of features [36]. In a comparative analysis, previous studies grouped features into sets of 4, 15, and 22, demonstrating a consistently high accuracy ranging from 96% to 99%. We confirmed that our evaluation result aligns with such a result to distinguish benign and malicious traffic, as our proposed framework delivers a 97–99% accuracy (see Section 5.3). To ensure a balanced representation among the source data sets, we employed a sampling strategy for the number of attack instances in each attack type. Specifically, we adjusted the total count of benign and attack instances in each data set to align with 625,783, which is the smallest instance number among the data sets (IoT Intrusion data set).

4.2. Phase 1: Similarity Assessment

In our study, we based our approach on *similarity* to select the optimal source domain to enhance the attack detection performance in the target domain. We designed the similarity assessment based on the following intuition: the closer the data are to the target domain, the better the detection performance will be during transfer learning. In phase 1, we evaluate the similarity between each source domain and the target domain. The deep learning model used for this has layers as illustrated in Figure 3, consisting of three 1D CNN layers and five Dense layers.

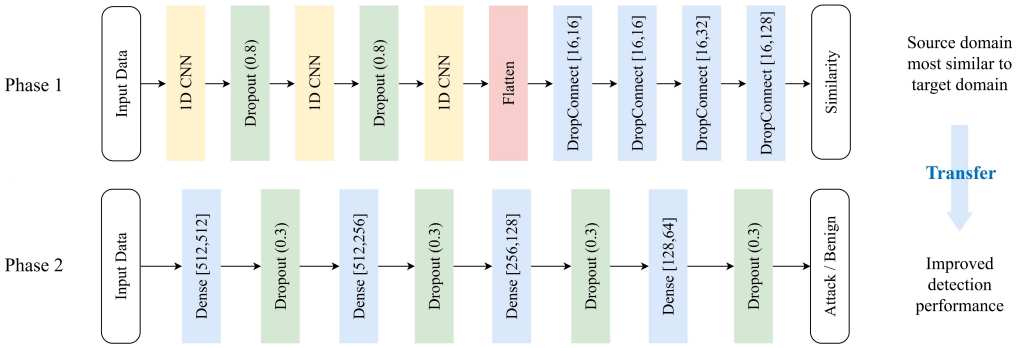

Figure 3. Deep learning layer configuration.

All 1D CNN layers and Dense layers use the LeakyReLU activation function, and a Flatten layer is added between the 1D CNN and Dense to match dimensions. The dropout layer is one of the regularization techniques used to prevent overfitting in deep learning models. The reason the dropout layer parameter was set to 0.8 was to deactivate neurons with a 20% probability during training, preventing the network from being overly dependent on specific neurons. Thus, 0.8 was the optimal parameter value obtained through experimentation. The rationale for layer selection and arrangement lies in the hierarchical nature of deep neural networks in transfer learning. In the context of transfer learning, lower layers are known to capture more fundamental knowledge from the input data of network traffic, while upper layers specialize in learning higher-level abstractions by combining or synthesizing these basic features, respectively. In our approach, we maintain the lower layer (1DCNN) in a fixed state and manipulate the parameters of the upper layer (Dense and Dropconnect). Additionally, we employ batch normalization to standardize the data distribution. Batch normalization adjusts data distribution within each batch, enhancing the stability of the learning process. Finally, to prevent overfitting, early stopping is implemented based on the point where the validation loss starts increasing. This layered configuration aims to leverage the specialized capabilities of each layer, facilitating the effective transfer of knowledge across varying domains.

In phase 1, we use the Mean Squared Error (MSE) loss function and the Mean Absolute Error (MAE) loss sum to conduct similarity assessments between the source and target domains. MSE and MAE are representative metrics for regression problems, and both are based on the difference between the actual and predicted values, which has the advantage of being intuitive and involving less computational complexity. In addition, the MSE loss function is typically used in deep learning models that train the closest value to the actual label to be predicted. Because it averages the square of the errors, large errors are more emphasized, which is useful for evaluating the overall performance of the model. Mean Squared Error (MSE) is defined as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (1)$$

Through the MSE loss function, we calculate the mean squared error, where y_i represents the predicted values, $\hat{y}_i$ represents the actual values, and N denotes the number of training samples.

To evaluate the performance of the regression model and determine how well it has learned, we utilize the MAE evaluation metric. MAE measures the absolute differences between the actual ground truth values and the predicted values, averaging these absolute differences. A lower MAE indicates a higher similarity of the model's predictions to the actual values. MAE is defined as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2}$$

In this context, y_i represents the actual values, while $\hat{y}_i$ denotes the predicted values. The error is calculated by subtracting the predicted value from the actual value. This difference is then taken in absolute terms. The resulting value is divided by the number of training samples, N, to compute the average.

Algorithm 1 describes the detailed workflow of phase 1.

Algorithm 1: Similarity evaluation between source and target domains

Input: Source Domain S_n for $n = 1, 2, 3$, Target Domain T
Output: Trained Model M_n for $n = 1, 2, 3$
Initialization:
Uniformize the size of the data sets.
Sample attack types and standardize the number of labels to 625,783.
Transfer learning:
Evaluate the similarity between S_n data sets before proceeding with transfer learning.
Phase 1 process:
while $n \leq 3$ **do**
    Input Train data S_n 100%, Test data T 70%.
    Train data S_n learning through DNN layer with LeakyReLU.
    Compile the model with an MSE loss function for similarity evaluation and Adam optimizer.
    while *epoch* ≤ 100 **do**
        Check that overfitting does not occur at epochs 20/40/60/80/100.
        if *validation loss* < 1 **then**
            Early stopping for overfitting prevention.
    Evaluate the performance of the model using MAE function on Test T data set.
    Save the trained model.
return Trained Model M_n for $n = 1, 2, 3$.

The goal is to evaluate the similarity between source domains (S_n) and a target domain (T) before conducting knowledge transfer. After the preprocessing and sampling procedure, our framework conducts deep learning with the aforementioned network layers for similarity assessment. Note that we use LeakyReLU as an activation function to evaluate which of the source domain is most similar to the target. The MSE and MAE loss functions are used as performance evaluation indicators of the regression model to compare the performance of the trained model (M_n). The training is repeated until the epoch reaches 100, and the appearance of each epoch is observed at 20, 40, 60, 80, 100. If there is overfitting at this time, the training undergoes early stopping to prevent overfitting.

4.3. Phase 2: Transfer Learning

In phase 2, our framework acquires a training model for intrusion detection against the target domain by utilizing the knowledge transferred from phase 1. For evaluation, the Binary Cross Entropy (BCE) loss function is employed to assess the accuracy of malicious traffic detection for source domains with a high similarity. We selected BCE because it measures the difference between the predictive probability of the model and the actual class and is a suitable loss function for binary classification. During the experiment, the binary classifier is trained by setting malicious traffic to 1 and positive traffic to 0. The BCE loss function is defined as follows:

$$BCE = -\frac{1}{N}\sum_{i=1}^{N}[t_i \log(y_i) + (1 - t_i)\log(1 - y_i)] \tag{3}$$

Here, t_i represents the actual label (either 0 or 1), and y_i is the predicted binary probability ($0 < y_i < 1$) for the data. As the prediction becomes more accurate, the loss decreases. Our aim is to minimize the total loss through training.

To prevent overfitting, we introduce five Dense layers and four Dropout layers (see Figure 3). The Dropout layer deactivates random neurons during training to enhance generalization capabilities. To reduce the model's complexity and enhance generalization, a Dropout rate of 0.3 is applied. In the phase 2 experiments, a ratio of 0.3 showed the most optimal results. Additionally, to simplify the model, we reduce the number of units in the deep learning model and utilize DropConnect in the Dense layers. For binary classification, the Leaky ReLU activation function is used in the four Dense layers excluding the last Dense layer, which requires a Sigmoid activation function. This choice effectively controls the model complexity and mitigates overfitting.

As shown in Algorithm 2, the goal of phase 2 is to acquire a binary classifier (K) to judge whether the given target domain instance is benign or malicious. Like the typical architecture composition in a binary classification problem, we use LeakyReLU as an activation function for four dense layers and Sigmoid for the last one. Comparing the Accuracy value, the result obtained from this experiment allows us to determine the highest accuracy when using any of the three S_ns as the source domain. As a result, phase 2 demonstrates the correlation of similarity and performance between data sets based on the accuracy of the target-domain-like source domain identified in phase 1.

Algorithm 2: Transfer learning with source and target domains

Input: Source Domain S_n for $n = 1, 2, 3$, Target Domain T
Output: Trained Model K

Initialization: Load the source domain evaluated for similarity in Experiment 1.
Transfer learning:
Train using similar source domain data, evaluate performance on target domain.
Phase 2 process:
while $n < 3$ **do**
    Input Train data ($S'_n = S_n + T$ 70%), Test data T 30%.
    Train data S_n through DNN layer with LeakyReLU and Sigmoid.
    Compile the model with a BCE loss function for binary classification and Adam optimizer.
    while *epoch* ≤ 100 **do**
        Check that overfitting does not occur at epochs 20/40/60/80/100.
        if *validation loss* < 1 **then**
            Early stopping for overfitting prevention.
    Evaluate the performance of K by Accuracy on Test data set.
    Update K if the model corresponding to S_n has the best performance.
return Trained Model K.

5. Evaluation

5.1. Experimental Environment

To evaluate the performance of our proposed framework, we used four public data sets mentioned above. The Bot-IoT data set and IoT Intrusion data set were selected as target domains, which were collected from real IoT environments. The experiment included the number of both cases in which each of these two data sets (Bot-IoT and IoT Intrusion) was set as the target domain, and, in each case, the source domain was the other three except that data set. These source domains contain various types of network attacks and provide appropriate data to evaluate their similarities with the target domain. Note that each source domain consists of 70% of the three data sets and 30% of the data set selected as the target domain. As an experimental environment, we used a Windows 11 Pro 64-bit operating system, equipped with an Intel(R) Core(TM) i7-1065G7 CPU @ 1.30 GHz 1.50 GHz processor, 16 GB of RAM, and an Intel Iris Plus Graphics graphics card. We utilized Python 3.11.4 64-bit and leveraged the Pandas, Scikit-learn, and Numpy libraries. Table 3 summarizes our evaluation environment.

Table 3. Experimental environment.

Platforms	Windows 11 pro 64 bit
Processor	Intel(R) Core(TM) i7-1065G7 CPU @ 1.30 GHz 1.50 GHz
RAM	16 GB
Graphics Card	Intel Iris Plus Graphics
Library	Pandas, Scikit-learn, Numpy
Programming Language	Python 3.11.4 64 bit

5.2. Similarity Assessment

First, we measured the MAE and MSE of each source domain candidate for knowledge transfer to figure out the source domain most adventurous to the target domain. The result acquired from phase 1 was further used in the subsequent phase, phase 2, to conduct a more accurate and effective transfer learning. The experimental results are shown in Tables 4 and 5. We confirmed that the Bot-IoT data set was the IoT Intrusion data set with the lowest MAE and MSE metric values, which means that it was the most similar to the target domain. This suggests that the IoT Intrusion data set has similar features and patterns to the Bot-IoT data set. Likewise, even when the IoT Intrusion data set was a target, it was confirmed that the MAE and MSE of the Bot-IoT data set were the lowest. As a next step, we verified that intrusion detection based on deep transfer learning in each case utilizing IoT Intrusion data set and Bot-IoT data set delivered the best inference performance for the source data set conducted in phase 2.

Table 4. The results of the similarity evaluation experiment in the case where the target domain was Bot-IoT.

	MAE	**MSE**
CIC-IDS2017	0.9933	0.9873
IoT Intrusion	0.0781	0.0066
UNSW-NB15	1.0015	1.0036

Table 5. The results of the similarity evaluation experiment in the case where the target domain was IoT Intrusion.

	MAE	**MSE**
CIC-IDS2017	0.8938	0.8859
Bot-IoT	0.0641	0.0638
UNSW-NB15	0.8874	0.8660

5.3. Intrusion Detection Performance

To verify the assessment result, we constructed multiple models for each source domain combined with the target domain data set, a typical way of instance-based transfer learning. For this, three training data sets were constructed by combining 100% of the source data set and 70% of the target data set for each of the three source data sets. Note that the remaining 30% of the target data set was used as a test data set. The accuracy evaluation index employed in the performance assessment was derived from a confusion matrix and adhered to the following standard measurements:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Here, TP (True Positive) represents the count of inference results in which the model correctly predicted a malicious flow as an attack class. In essence, the accuracy metric serves as an indicator of how precise the model's predictions are, with a higher value approaching 1 signifying superior performance.

Tables 6 and 7 show the experimental results of phase 2.

Table 6. The results of the performance evaluation experiment in the case where the target domain was Bot-IoT.

	Accuracy	**Loss**	**F1-Score**	**Recall**	**Precision**
CIC-IDS2017	0.9928	0.0278	0.9792	0.9605	0.9994
IoT Intrusion	**0.9994**	**0.0049**	**0.9997**	**1.0000**	**0.9994**
UNSW-NB15	0.9993	0.0083	0.9997	1.0000	0.9993

Table 7. The results of the performance evaluation experiment in the case where the target domain was IoT Intrusion.

	Accuracy	**Loss**	**F1-Score**	**Recall**	**Precision**
CIC-IDS2017	0.9753	0.0452	0.9865	0.9776	0.9960
Bot-IoT	**0.9994**	**0.0053**	**0.9997**	**1.0000**	**0.9994**
UNSW-NB15	0.9765	0.0600	0.9937	0.9995	0.9882

As expected, the highest Accuracy value and the lowest loss value were identified in the case of using the IoT Intrusion data set as the source data set. To cross-check whether the selected source domain based on the assessment and the target domain were indeed similar, we performed a qualitative analysis by scrutinizing the data sets, for example, by examining the number of attack types and the instance ratio of common attack types.

First, when the number of common attack types between the target and source domains was examined, UNSW-NB15 overlapped the most with two types: DoS and Reconnaissance. Meanwhile, IoT Intrusion and CIC-IDS2017 had only DoS in common. Considering that all three source domains shared the DoS attack type with the target domain, similarity was assessed based on the number of DoS instances. The counts for each source domain were as follows: IoT Intrusion with 59,391 instances, UNSW-NB15 with 16,353 instances, and CIC-IDS2017 with 252,661 instances, out of a total of 625,783 instances across the domains. The percentage of DoS instances from the total for each source domain was calculated as follows: IoT Intrusion at 9.48%, UNSW-NB15 at 2.61%, and CIC-IDS2017 at

40.35%. Interestingly, this reveals that CIC-IDS2017 significantly had the highest number of instances for the common attack type.

Then, we reviewed the data collection environment for each public data set. The data sets used as source domains were collected from diverse environments ranging from actual IoT setups to virtual networks. The IoT Intrusion data set originated from a smart home environment, specifically from SKT NUGU (NU 100) and the EZVIZ Wi-Fi Camera (C2C Mini O Plus 1080P). The UNSW-NB15 data set is a hybrid blend of genuine modern network traffic and synthesized modern attack behaviors. The CIC-IDS2017 data set was collected in a virtual environment designed to emulate real PCAP data. Lastly, the Bot-IoT data set comprises data from five IoT devices, including smart fridges, garage doors, weather monitors, lights, and thermostats.

While UNSW-NB15 dominated in terms of the number of attack types and CIC-IDS2017 led in the instance ratio of shared attack types with the target domain, ultimately, the IoT Intrusion data set, generated from the same smart home environment, demonstrated the highest performance. This underscores the significant influence of the data collection environment on the target domain's performance. In situations requiring the selection of a source domain for transfer learning, it is recommended, as evidenced by our paper, to prioritize the collection environment over the shared attack types for better effectiveness.

5.4. Overfitting Mitigation

To address overfitting in the model training, we employed various techniques. First, we utilized the dropout layer and empirically determined the dropout rate to mitigate overfitting. For mild overfitting, a dropout rate between 0.2 and 0.5 is recommended, and for severe cases, a rate of 0.8 is beneficial. Therefore, we applied a rate of 0.8 to ensure a robust prevention of overfitting. Additionally, as the complexity of the model increases, it becomes more prone to overfitting. Hence, we adjusted the number of layers and parameters in the model. Finally, we incorporated the BatchNormalization() layer into the deep learning model to further mitigate overfitting. The BatchNormalization() Layer normalizes the output value of the activation function to distribute it appropriately. If the weights are not appropriate, the model may become sensitive to even small changes in the input data, which may lead to overfitting. In this case, BatchNormalization() improves the learning speed and suppresses overfitting by maintaining good weight values and appropriately distributing activation values without depending on the initial weight value. Finally, it is worth noting that existing overfitting mitigation methods, including the aforementioned mitigation strategies, can be also leveraged to our proposed framework.

To validate the overfitting mitigation strategies applied in our framework, we measured the trend of training and validation accuracies in phase 2 for the IoT Intrusion data set while varying the epochs until 100. Figure 4 shows the result after adopting the aforementioned mitigation methods.

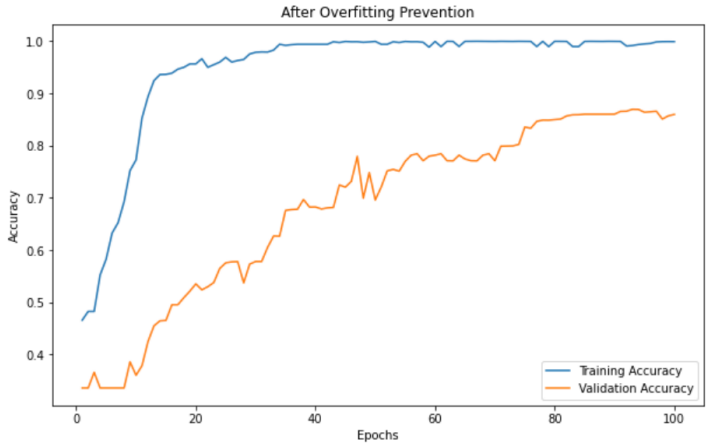

Figure 4. Training and validation accuracies after adopting overfitting mitigation.

Typically, a model is likely to be overfitted when the slope of training and validation accuracy gradually decrease, reaching a floor. In our observation, we noted a smooth increase in both training and validation accuracy, with a consistent plateau observed from epoch 20 onwards, maintaining a stable shape. This is interpreted as the effective contribution of the overfitting technique mentioned above.

5.5. Discussion and Limitations

Our research emphasizes the importance of choosing the right source domain for transfer learning, as this choice impacts learning performance. This perspective, which highlights the significance of selecting an appropriate source domain, has yet to be explored in existing research. The effectiveness of our framework is demonstrated through our evaluation results on the IoT Intrusion and Bot-IoT data set case studies. Remarkably, these data sets, sharing similar collection environments and attack labels compared to other data sets, exhibited the best accuracy as a source domain to each other when processed using our framework. This result highlights the framework's efficacy when collection environments and attack labels align closely.

Compared to previous studies, our approach sets itself apart by not only focusing on the types and frequencies of attacks but also taking into account the environmental context of data collection based on the inherent similarity with the target domain. While existing approaches are primarily focused on the technical enhancement of trained intrusion detection models by utilizing diverse machine learning techniques, we recognize the importance of providing practical implications and insights for knowledge distillation. These insights are particularly valuable for practitioners and researchers in IoT security, who face challenges posed by diverse and heterogeneous IoT network environments. We believe the importance of our research lies not only in developing an effective intrusion detection model but also in emphasizing the crucial role of selecting an appropriate source domain based on its similarity to the target domain.

However, our knowledge distillation strategy based on data set similarity would be inefficient in scenarios where the intersection across the source domain candidates and the target domain are limited and scarce. As our framework relies on a subset of shared features from varied candidate data set pools in the preprocessing phase, it might fail to accurately identify all the attacks specified within the data sets. We note that information loss due to the common feature extraction can be ameliorated if the raw packet data (e.g., PCAP files) are accessible. By utilizing the same flow analyzer (e.g., CICFlowMeter [34]) against candidate data set pools to acquire the flow information from the raw packet data, it is possible to fairly utilize all the flow features for transfer learning.

If raw packet data are not provided, the number of common features could be drastically reduced when using datasets with vastly different characteristics, which could lead to performance degradation. However, these concerns can be alleviated primarily through screening by security operators before performing intrusion detection. For example, they can conduct qualitative analysis to ensure the collection environment is similar or the considered attack label is being targeted. Subsequently, they might encounter a situation in which it is hard to intuitively determine the most suitable source dataset among the filtered candidate data set pools. In such cases, we believe leveraging the proposed framework is considered best practice for comparing datasets and resolving uncertainties.

In addition, the aforementioned limitation aligns with the inherent constraints of knowledge distillation based on data set similarity, particularly in scenarios involving the detection of unforeseen attack patterns (e.g., zero-day attacks). We believe such limitation can be addressed by leveraging the state-of-the-art ML techniques. For example, recent approaches that enhances the calibration of neural network confidence by leveraging outlier exposure [37] can be applicable to phase 2 in our framework to detect false alarms raised by zero-day attacks.

6. Conclusions

In this paper, we propose a comprehensive end-to-end deep transfer learning framework for IoT intrusion detection. Our approach starts with thoroughly selecting suitable data sets from uncertain candidate pools tailored to the source domain and extends to leveraging knowledge transfer into the inference phase for target domain data. We conducted experiments using four of the most prevalent and widely used data sets of network traffic collected in IoT environments. We confirmed that the proposed deep transfer learning framework appropriately selects suitable source domain data sets from the candidates and delivers the best accuracy for the test data. Our framework not only enhances the efficiency of the transfer learning process but also serves as a guiding mechanism for selecting the source domain that best enhances the performance of the target domain across the entire transfer learning process, making it universally applicable when applying different data sets in the future. We believe it relieves the burden of security managers having to assess each source domain suitable for intrusion detection tailored to their custom IoT environment. The proposed framework can be further extended to support multi-class classification, enabling the identification of specific attack types by utilizing deep transfer learning. Furthermore, future efforts could focus on optimizing the time efficiency, such as Time-to-Accuracy, in phase 2. Exploring variations in neural network architectures is another avenue for investigation to understand their impact on accuracy. We leave these as future work.

Author Contributions: Conceptualization, H.K. and S.P.; Methodology, H.H. and J.P.; Supervision, S.K.; Writing-review & editing, H.K., S.P., H.H. and J.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Sungshin Women's University Research Grant of H20220028.

Data Availability Statement: The data presented in this study are publicly available. Bot-IoT Dataset: https://research.unsw.edu.au/projects/bot-iot-dataset, accessed on 1 January 2024. IoT Intrusion Dataset: https://ocslab.hksecurity.net/Datasets/iot-network-intrusion-dataset, accessed on 1 January 2024. CIC-IDS2017 Dataset: https://www.unb.ca/cic/datasets/ids-2017.html, accessed on 1 January 2024. UNSW-NB15 Dataset: https://research.unsw.edu.au/projects/unsw-nb15-dataset, accessed on 1 January 2024.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
DoS	Denial-of-Service
DDoS	Distributed Denial of Service
NIDS	Network Intrusion Detection System
DPI	Deep Packet Inspection
ML	Machine Learning
IDS	Intrusion Detection System
SSL	Secure Sockets Layer
HTTPS	HyperText Transfer Protocol over Secure Socket Layer
SSH	Secure Shell
MFSR-IDS	Multi-layer Feature Selection and Reduction IDS
DT	Decision Tree
KNN	K-Nearest Neighbor
RF	Random Forest
CNN	Convolutional Neural Network
DNN	Deep Neural Network
MMD	Maximum Mean Discrepancy
MDM-AE	Mobile Device Management-AE

MITM	Man-in-the-Middle
MSE	Mean Squared Error
MAE	Mean Absolute Error
BCE	Binary Cross Entropy
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative

References

1. Zhao, Z.Z.; Guo, F.; Wu, G.; Susilo, W.; Wang, B. Secure infectious diseases detection system with iot-based e-health platforms. *IEEE Internet Things J.* **2022**, *9*, 22595–22607. [CrossRef]
2. Shariatzadeh, N.; Lundholm, T.; Lindberg, L.; Sivard, G. Integration of digital factory with smart factory based on Internet of Things. *Procedia Cirp* **2016**, *50*, 512–517. [CrossRef]
3. Alladi, T.; Chamola, V.; Zeadally, S. Industrial control systems: Cyberattack trends and countermeasures. *Comput. Commun.* **2020**, *155*, 1–8. [CrossRef]
4. Muthuramalingam, S.; Bharathi, A.; Rakesh Kumar, S.; Gayathri, N.; Sathiyaraj, R.; Balamurugan, B. IoT based intelligent transportation system (IoT-ITS) for global perspective: A case study. In *Internet of Things and Big Data Analytics for Smart Generation*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 279–300.
5. MarketsandMarkets. *IoT Solutions and Services Market by Component (Solutions and Services), Organization Size, Deployment Mode, Focus Area (Smart Manufacturing, Smart Energy and Utilities, and Smart Retail) and Region-Global Forecast to 2027*; Technical Report; MarketsandMarkets: Pune, India, 2022.
6. Rodríguez, E.; Valls, P.; Otero, B.; Costa, J.J.; Verdú, J.; Pajuelo, M.A.; Canal, R. Transfer-learning-based intrusion detection framework in IoT networks. *Sensors* **2022**, *22*, 5621. [CrossRef]
7. Kruegel, C.; Toth, T. Using decision trees to improve signature-based intrusion detection. In *Proceedings of the International Workshop on Recent Advances in Intrusion Detection*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 173–191.
8. Elshrkawey, M.; Alalfi, M.; Al-Mahdi, H. An enhanced intrusion detection system based on multi-layer feature reduction for probe and dos attacks. *J. Internet Serv. Inf. Secur. (JISIS)* **2021**, *11*, 40–57.
9. Gyamfi, N.K.; Goranin, N.; Čeponis, D.; Čenys, A. Malware detection using convolutional neural network, a deep learning framework: Comparative analysis. *J. Internet Serv. Inf. Secur.* **2022**, *12*, 102–115. [CrossRef]
10. Pecori, R.; Tayebi, A.; Vannucci, A.; Veltri, L. IoT Attack detection with deep learning analysis. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8.
11. Liloja; Ranjana, P. An Intrusion Detection System Using a Machine Learning Approach in IOT-based Smart Cities. *J. Internet Serv. Inf. Secur. (JISIS)* **2023**, *13*, 11–21.
12. Ranieri, A.; Caputo, D.; Verderame, L.; Merlo, A.; Caviglione, L. Deep adversarial learning on google home devices. *arXiv* **2021**, arXiv:2102.13023.
13. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In Proceedings of the International Conference on Information Systems Security and Privacy, Funchal, Portugal, 22–24 January 2018.
14. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [CrossRef]
15. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015; pp. 1–6.
16. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
17. Vu, L.; Nguyen, Q.U.; Nguyen, D.N.; Hoang, D.T.; Dutkiewicz, E. Deep transfer learning for IoT attack detection. *IEEE Access* **2020**, *8*, 107335–107344. [CrossRef]
18. Debicha, I.; Bauwens, R.; Debatty, T.; Dricot, J.M.; Kenaza, T.; Mees, W. TAD: Transfer learning-based multi-adversarial detection of evasion attacks against network intrusion detection systems. *Future Gener. Comput. Syst.* **2023**, *138*, 185–197. [CrossRef]
19. Tanghatari, E.; Kamal, M.; Afzali-Kusha, A.; Pedram, M. Distributing DNN training over IoT edge devices based on transfer learning. *Neurocomputing* **2022**, *467*, 56–65. [CrossRef]
20. Sherry, J.; Lan, C.; Popa, R.A.; Ratnasamy, S. Blindbox: Deep packet inspection over encrypted traffic. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, London, UK, 17–21 August 2015; pp. 213–226.
21. Han, J.; Kim, S.; Ha, J.; Han, D. Sgx-box: Enabling visibility on encrypted traffic using a secure middlebox module. In Proceedings of the First Asia-Pacific Workshop on Networking, Hong Kong, China, 3–4 August 2017; pp. 99–105.
22. Guo, Y.; Shi, H.; Kumar, A.; Grauman, K.; Rosing, T.; Feris, R. Spottune: Transfer learning through adaptive fine-tuning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4805–4814.

23. Gopalakrishnan, K.; Khaitan, S.K.; Choudhary, A.; Agrawal, A. Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. *Constr. Build. Mater.* **2017**, *157*, 322–330. [CrossRef]
24. Ruder, S.; Peters, M.E.; Swayamdipta, S.; Wolf, T. Transfer learning in natural language processing. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials, Minneapolis, Minnesota, 3–5 June 2019; pp. 15–18.
25. Peng, Y.; Yan, S.; Lu, Z. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. *arXiv* **2019**, arXiv:1906.05474.
26. Baireddy, S.; Desai, S.R.; Mathieson, J.L.; Foster, R.H.; Chan, M.W.; Comer, M.L.; Delp, E.J. Spacecraft time-series anomaly detection using transfer learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Online, 19–25 June 2021; pp. 1951–1960.
27. Wen, T.; Keyes, R. Time series anomaly detection using convolutional neural networks and transfer learning. *arXiv* **2019**, arXiv:1905.13628.
28. Weiss, K.; Khoshgoftaar, T.M.; Wang, D. A survey of transfer learning. *J. Big Data* **2016**, *3*, 1–40. [CrossRef]
29. Huang, S.; Guo, Y.; Liu, D.; Zha, S.; Fang, W. A two-stage transfer learning-based deep learning approach for production progress prediction in IoT-enabled manufacturing. *IEEE Internet Things J.* **2019**, *6*, 10627–10638. [CrossRef]
30. Tien, C.W.; Huang, T.Y.; Chen, P.C.; Wang, J.H. Using autoencoders for anomaly detection and transfer learning in iot. *Computers* **2021**, *10*, 88. [CrossRef]
31. Kang, H.; Ahn, D.H.; Lee, G.M.; Yoo, J.D.; Park, K.H.; Kim, H.K. IoT Network Intrusion Dataset. Available online: http://ocslab.hksecurity.net/Datasets/iot-network-intrusion-dataset (accessed on 5 December 2019).
32. Stiawan, D.; Idris, M.Y.B.; Bamhdi, A.M.; Budiarto, R. CICIDS-2017 dataset feature analysis with information gain for anomaly detection. *IEEE Access* **2020**, *8*, 132911–132921.
33. Yulianto, A.; Sukarno, P.; Suwastika, N.A. Improving adaboost-based intrusion detection system (IDS) performance on CIC IDS 2017 dataset. *J. Phys. Conf. Ser.* **2019**, *1192*, 012018. [CrossRef]
34. Lashkari, A.H.; Zang, Y.; Owhuo, G.; Mamun, M.; Gil, G. CICFlowMeter. GitHub. Available online: https://github.com/ahlashkari/CICFlowMeter/blob/master/ReadMe.txt (accessed on 10 August 2021).
35. Paxson, V. Bro: A system for detecting network intruders in real-time. *Comput. Netw.* **1999**, *31*, 2435–2463. [CrossRef]
36. Kurniabudi.; Stiawan, D.; Darmawijoyo.; Bin Idris, M.Y.; Bamhdi, A.M.; Budiarto, R. CICIDS-2017 Dataset Feature Analysis With Information Gain for Anomaly Detection. *IEEE Access* **2020**, *8*, 132911–132921. [CrossRef]
37. Hendrycks, D.; Mazeika, M.; Dietterich, T. Deep anomaly detection with outlier exposure. *arXiv* **2018**, arXiv:1812.04606.