electronics

MDPI

Article

PF-TL: Payload Feature-Based Transfer Learning for Dealing with the Lack of Training Data

Ilok Jung, Jongin Lim and Huy Kang Kim *

School of Cybersecurity, Korea University, Seoul 02841, Korea; okkida@korea.ac.kr (I.J.); jilim@korea.ac.kr (J.L.)
* Correspondence: cenda@korea.ac.kr

Abstract: The number of studies on applying machine learning to cyber security has increased over the past few years. These studies, however, are facing difficulties with making themselves usable in the real world, mainly due to the lack of training data and reusability of a created model. While transfer learning seems like a solution to these problems, the number of studies in the field of intrusion detection is still insufficient. Therefore, this study proposes payload feature-based transfer learning as a solution to the lack of training data when applying machine learning to intrusion detection by using the knowledge from an already known domain. Firstly, it expands the extracting range of information from header to payload to accurately deliver the information by using an effective hybrid feature extraction method. Secondly, this study provides an improved optimization method for the extracted features to create a labeled dataset for a target domain. This proposal was validated on publicly available datasets, using three distinctive scenarios, and the results confirmed its usability in practice by increasing the accuracy of the training data created from the transfer learning by 30%, compared to that of the non-transfer learning method. In addition, we showed that this approach can help in identifying previously unknown attacks and reusing models from different domains.

Keywords: knowledge transfer; intrusion detection; machine learning; payloads; transfer learning

Citation: Jung, I.; Lim, J.; Kim, H.K. PF-TL: Payload Feature-Based Transfer Learning for Dealing with the Lack of Training Data. *Electronics* **2021**, *10*, 1148. https://doi.org/10.3390/electronics10101148

Academic Editor: Krzysztof Szczypiorski

Received: 3 April 2021
Accepted: 8 May 2021
Published: 12 May 2021

1. Introduction

With the advance of information technology, cyberattacks are becoming more intelligent and mass-produced, overwhelming the detection, analysis, and response abilities of traditional security approaches [1,2]. Thus, the number of studies applying artificial intelligence technology to the cybersecurity field is increasing [3–5]. Among these studies, intrusion detection is one of the particular fields where machine learning is showing higher detection rates and fewer false positive cases than the conventional signature-based detection methods [5–8].

In order to understand how machine learning works better than signature-based methods, one must first thoroughly understand the structure of the data of a common intrusion detection event. These data consist of two main fields (header and payload), shown in Figure 1. The header contains network information and the flow of the source IP and destination IP, while the payload contains the server and user information as well as data on requests and responses [9]. As cyberattacks continue to evolve, the number of cases where potential threats are hidden within payloads is increasing.

As demonstrated above, the traditional signature-based detection system uses an attack pattern, such as "admin or 1=1", to detect SQL injection, despite the user information in the user-agent section implying that it is, indeed, an attack via an automated attack tool (sqlmap) [10]. This is the epitome of a situation where the decision regarding intrusion should be made based on the attack information, such as attack pattern and related characters as well as domain knowledge, such as URL information, body, user-agent, referrer and so on.

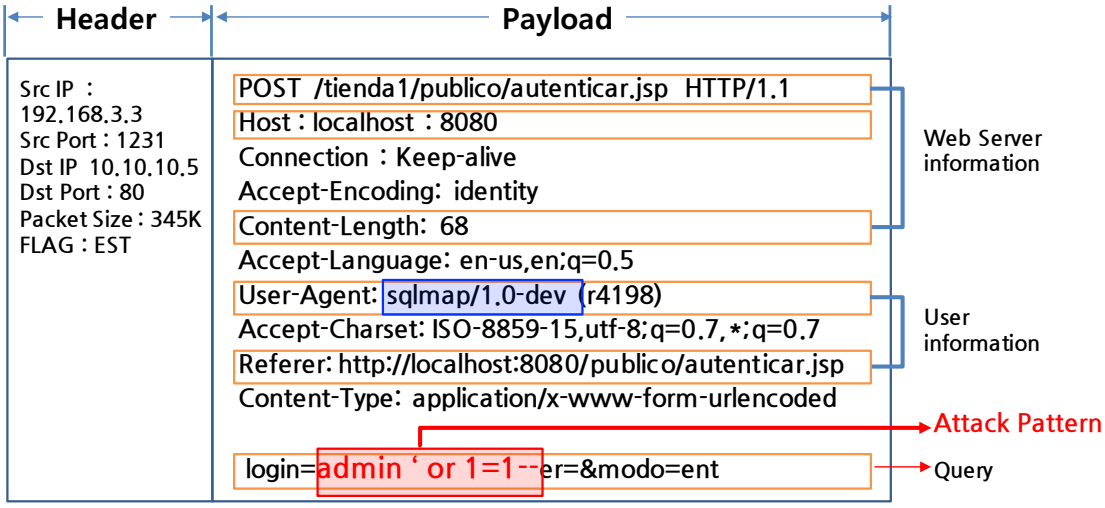

Figure 1. Example of intrusion detection event.

Recent studies have expanded the detection area from the header to payload to maximize accuracy and minimize false positives [8,11,12].

Torrano-Gimenez et al. used n-gram to extract features from the payload [13], and Betarte et al. extracted text-based features from the payload based on professional opinions [14], while Min et al. extracted statistical, vectorized features from the payload. These studies demonstrate the expanding the detection area from the header with limited information to the payload with rich information can prove to be successful not only in the conventional signature-based intrusion detection system, but also in the field of machine learning for the same field. However, the application of such supervised machine learning requires sufficient training data before being deployed into production [15]. Obtaining these training data requires a lot of resources and professional manpower, especially when one tries to collect data that contain the payload. Even after collecting enough labeled data and, thus, building a model with good performance, it is still difficult to reuse the model in other domains [3,6,16].

One of the machine-learning technologies to solve this problem is transfer learning, which allows information learned from an existing domain to be transferred to another domain [17–20]. Transfer learning is widely applied to natural language processing and visual recognition, but studies using transfer learning in intrusion detection are lacking. Recently, studies have attempted to use transfer learning in malware classification [21,22] and network intrusion detection [23–25]. Wu et al. used deep learning for feature extraction-based transfer learning, while Zhao et al. proposed the HeTL approach to detect unknown attacks, using already-known detection techniques. However, their applicability in the real environment is insufficient due to the use of structured datasets [5], such as KDD99 [26] and NSL-KDD [22], or the limited experiments through hypotheses within the same dataset [27].

Therefore, we propose payload feature-based transfer learning (PF-TL) to solve the shortage of training data in the field of intrusion detection. For our proposal, the following points were considered.

Firstly, by limiting the domain subject of this study to an intrusion detection event for webservices [14,28–33], transfer learning could be applied since intrusion detection events share similar characteristics with a variation of already-known web attacks. Secondly, the method to select and extract the features of information were carefully chosen. Well-known signatures were used to extract the characteristics of the attacks, while text vectorization was used to extract unstructured domain knowledge, such as user and application information [34,35]. We call this the hybrid feature extraction method. With this new method, we aim to transfer the detection methodology of the source domain to the target domain [36–38].

In addition, the transfer learning algorithm, based on heterogeneous features seeking optimized space between the two domains, is applied to enhance accuracy [18–25].

This study makes the following contributions.

First, the proposed transfer learning approach makes it easier to obtain optimized label data for the target domain, which saves the time and manpower needed in creating training data (Section 5).

Second, an effective feature extraction method for transfer learning between intrusion detection domains is presented. Moreover, feature extraction was extended to the payload, including headers, the signature-based feature in the payload, and text vectorization-based feature methods (Section 3).

Third, an improvement over the optimization method and the manual configuration of HeTL suggested in another study [25] is presented. This leads to optimized transfer learning for the web application intrusion detection field (Section 4).

Fourth, a benchmark against the HeTL using the training datasets from the same domain, which in turn proved the necessity of PF-TL, is presented and validated using two publicly available datasets and one real-world dataset (Section 5).

Fifth, we showed that it is possible to use the training data generated by the proposed approach to identify the new types of attack detection that do not exist in the training data, and therefore demonstrated that the model could be reused in other domains (Section 5).

2. Preliminaries

In this section, we will first describe transfer learning and then explain the intrusion detection system.

2.1. Transfer Learning

Traditional machine learning assumes that training and test data are imported from the same domain so that the feature spaces and the distributions of two datasets are similar [17–20]. Training data are often difficult to obtain and time-consuming and expensive to collect [39]. Therefore, when applying machine learning to other domains, training with data that can be obtained more easily from a particular domain and using the knowledge gained from it is desirable. This methodology is called transfer learning [18–20,40].

When implementing transfer learning, the following points should be considered. First, what source domain information is useful? Second, what is the best way to send this information to the target domain? Third, what can be done to avoid sending unnecessary information to the desired results?

2.1.1. Categorization According to Similarity between Domains and Feature Space

Transfer learning can be categorized into homogenous and heterogeneous transfer learning, depending on the similarities between source and target domains and the feature space in between [20].

Homogenous transfer learning [20] is used in the same kind of transfer learning for which the feature space of the data in the source and target domain is represented by the same feature ($X_s = X_t$) and label ($Y_s = Y_t$), and the space itself comprises the same dimension (ds = dt). Therefore, this method focuses on closing the gap in the distribution of data between domains, which has already been experienced in the cross-domain transition.

As for heterogeneous transfer learning [20], the source and target domain may not share features ($X_s \neq X_t$) and labels ($Y_s \neq Y_t$), and the size of the feature space may also vary. Therefore, this method requires the transformation of feature and label spaces to bridge the gaps between knowledge areas and to deal with differences in data distribution between domains. This method is even more difficult because little expressive common points are present between domains. That is, knowledge in the source domain is available, but it can be displayed in a different way than the target domain, and the most important thing is its extraction method.

2.1.2. Categorization According to Transition Content

Based on the transition content, transfer learning can be divided into four categories (instance-, feature-, parameter-, and relational-based) [20].

Instance-based transfer learning assumes that a particular part of the data in the source domain can be re-weighted and re-used in the target domain for learning. In this situation, instance weight and importance can be sampled.

Feature-based transfer learning can be applied to both homogenous and heterogenetic transfer learning. It aims to reduce the feature distribution gap between the feature spaces of the source and target domain through a "good" feature.

Parameter-based transfer learning is applied by extracting each shared parameter and priority between the source and target domain models.

Relational-based transfer learning forms transfer learning for the relational area. The underlying assumption for this learning is that some relations between the source and target domain data are similar. Thus, the knowledge to be transmitted is the relation between the data. The recent statistical relation learning technique is similar to this one [20].

2.2. Intrusion Detection

Intrusion detection aims to identify various types of malicious activities on a network and computer. How the system responds to the identified malicious activities differs in IDS and IPS; IDS responds by passively monitoring the activities, while IPS actively restricts the traffic of the activities. These intrusion detection techniques are continuously developed and can be largely divided into the signature-based intrusion detection system (SIDS) [5] and anomaly-based intrusion detection system (AIDS) [5].

2.2.1. Signature-Based Intrusion Detection

SIDS uses pattern matching technology to find known attacks. This is also known as knowledge-based detection or misuse detection [5,41]. The main concept is to build an intrusion signature database, compare it with the data in the packets, and raise an alert if a match is found. SIDS generally shows excellent accuracy in detecting previously known intrusions, but it shows a poor detection rate when a new attacks occurs until the signature database is updated [42].

2.2.2. Anomaly-Based Intrusion Detection

AIDS uses machine learning as well as statistics- and knowledge-based methods to overcome the limitations of SIDS. The main concept is that malicious and benign behaviors are different. Therefore, this detection comprises two phases that define normal behavior: a learning phase and a test phase. The advantage of AIDS is that it can identify zero-day attacks because the recognition of abnormal user activities does not rely on the signature database [42].

3. PF-TL: Payload Feature-Based Transfer Learning

3.1. Overall Process of Proposed Approach

In this section, we propose payload feature-based transfer learning (PF-TL) as described in Figure 2 in which the intrusion detection knowledge is transferred from a source domain to a target domain in order to train an unlabeled dataset.

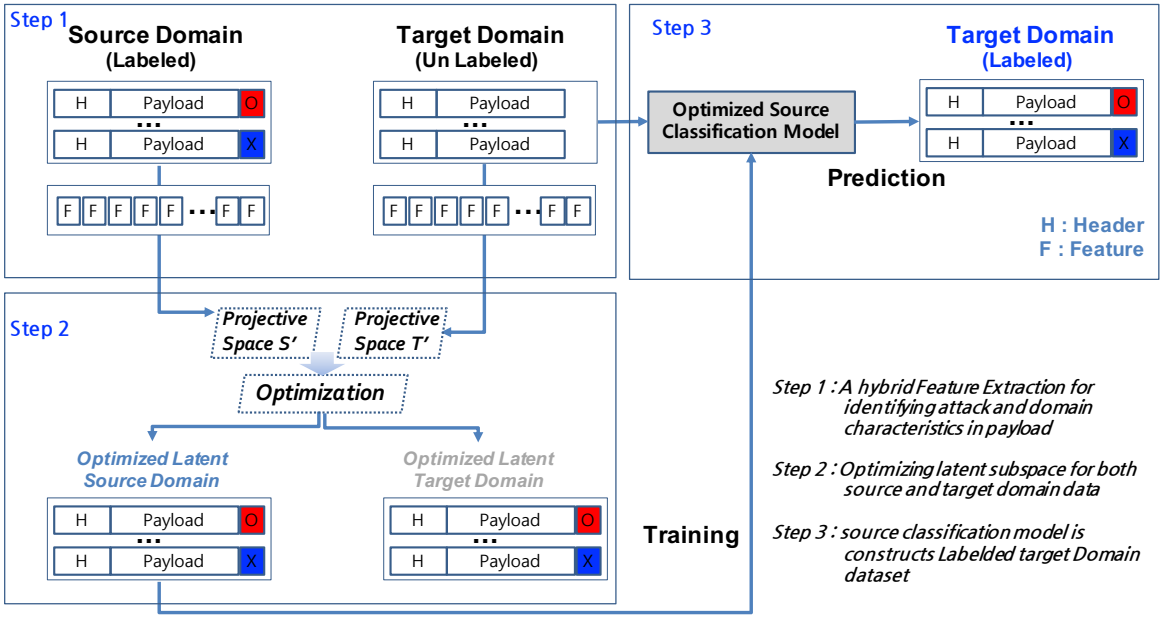

Figure 2. The proposed steps for PF-TL (payload feature-based transfer learning).

The source domain contains labeled intrusion detection data that can be used as training data. The target domain contains unlabeled intrusion detection data that can be categorized as test data. The source and target domain data comprise attack and normal data and we assume that the source domain is labeled by an attack type and the target domain is unlabeled.

This proposal is comprised of the following steps.

Step 1: Features are extracted from the header and payload data of the source domain and target domain, using both attack feature extraction and text vectorization, in order to effectively transfer the attack knowledge.

Step 2: An optimized latent subspace between the source domain and target domain's feature space is created, using PF-TL on the features from the extracted payload. These optimized latent subspaces are then used to create an optimized source domain and optimized target domain datasets.

Step 3: The optimized source classification model is created through the optimized source domain dataset created in Step 2. It uses the Random Forest, SVM, MLP, and KNN algorithms [2] as the model classification and selects a titration algorithm.

The subsequently generated model (optimized source classification model) can then be used to obtain the target domain (labeled) that is formed by adding a label to the target domain (unlabeled). This allows labeled data to be generated, even in the target domain where training data are limited, and machine learning can be applied through the subsequently created model.

3.2. Hybrid Feature Extraction for Identifying Attack and Domain Characteristics

In this section, we propose a payload-based feature extraction method, which enhances the knowledge transfer between two domains for transfer learning.

Our proposal includes not only the statistical feature extraction of header—the common approach used in many research—but also a hybridized feature extraction method that extracts both attack features and domain characteristics from the payload as shown in the Figure 4. The attack features are extracted from the payload based on the keyword or special characters commonly used in the attacks [14,37,43]. The domain characteristics are extracted using latent semantic analysis (LSA) on the vectorized payloads.

In order to do so, the collected source data must include a payload of the intrusion detection event, which contains a variety of information. As shown in the Figure 3, the

header of the source data includes the IP, port, packet size, etc., of a specific network connection. The payload is comprised of text data that contain the signature used in a detection system as well as related domain information, such as application, server and users [9].

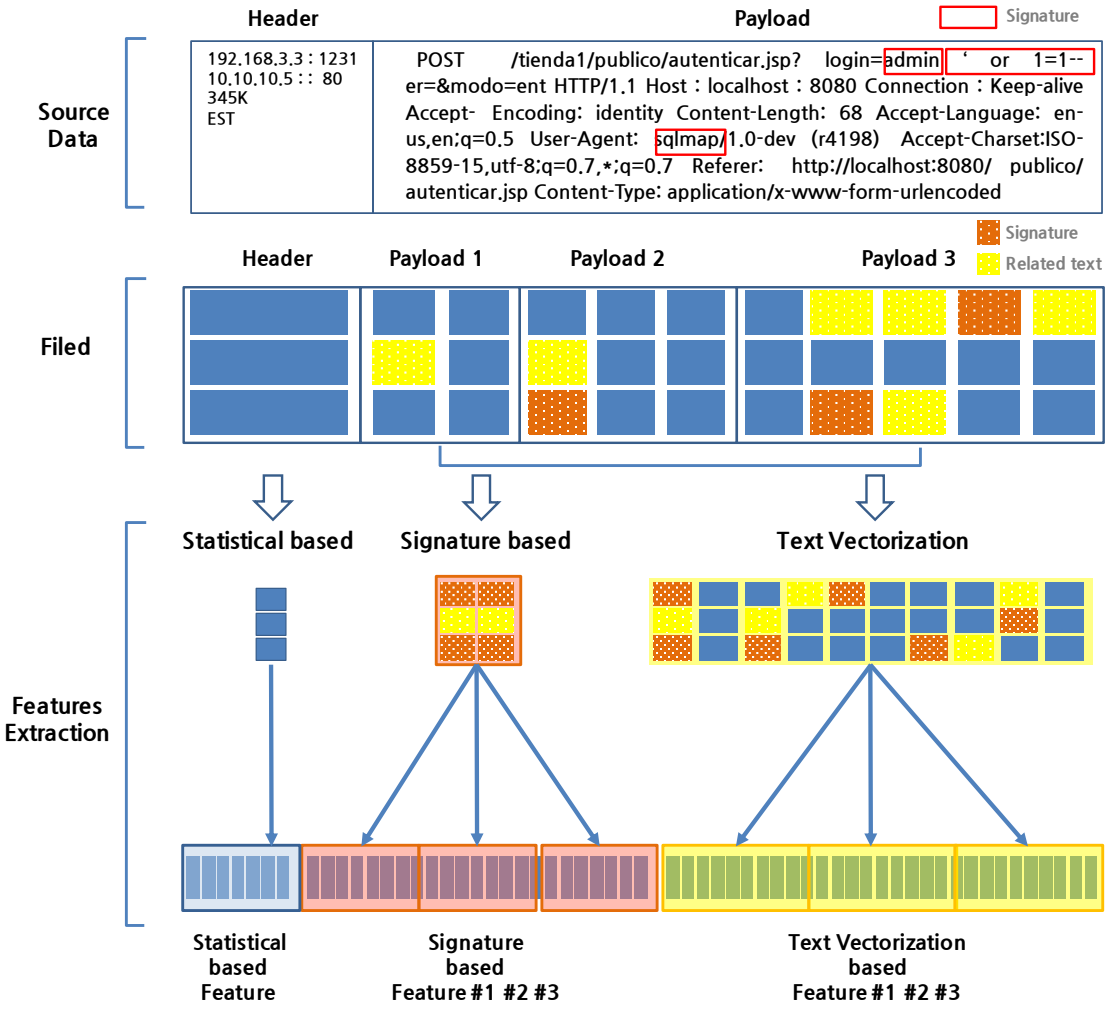

Figure 3. Payload based Feature Extraction.

First, the payload can be distinguished according to the characteristics of the service. In particular, in the case of web service, it can be divided into the URI, query, body, etc., within the payload. URI refers to the area up to the previous part of the string "?" that contains a specific location for resources on a particular server, including domains. Query refers to the area after "?" that contains the remaining area from the URI area. Body means the area containing head information, excluding the URI and Query areas. Payloads divided, such as those in Figure 3 (URI, Query, and Body), are more representative of the characteristics of each domain (Step 2).

Therefore, for each payload area, the feature required for transfer learning can be extracted through the following tasks (Step 3).

First, as proposed by Latha et al. [44], to extract the characteristics of each type of attack from the payload, the keywords and characters used for each type of attack, identified by security experts, were extracted to create the feature, shown in Table 1. Table 1 is similar to the character extraction method used by experts to determine attacks when using a signature. It differs from signature-based detection, as it does not simply consider one pattern but considers various signature combinations for related types, such as that in Table 2, extracting them to values such as frequency and length.

Table 1. Example of features by the security specialist [44].

Attack Class	Feature List
XSS	'&', '%', '/', '\\', '+', '"', '?', '!', ';', '#', '=', '[', ']', '$', '(', ')', '^', '*', ',', , '<', '>', '@', ' ', ':', '{', '}', ' ', '.', ' ', '|', '"', '<>', '"', '<>', '[]', "==", 'createelement', 'cookie', 'document', 'div', 'eval()', 'href', 'http', 'iframe', 'img', 'location', 'onerror', 'onload', 'string.fromcharcode', 'src', 'search','this', 'var', 'window', '.js', '<script'
SQL Injection	char', ',', '', '<', '>', ' ', '.', '|', '"', '(', ')', '<>', '<=', '>=', '&&', '||', ':', '!=', '+', ';', ' ', 'count', 'into', 'or', 'and', 'not','null', 'select', 'union', '#', 'insert', 'update', 'delete', 'drop', 'replace', 'all', 'any', 'from', 'count', 'user', 'where', 'sp', 'xp', 'like', 'exec', 'admin', 'table', 'sleep', 'commit', '()', 'between'
LDAP Injection	'\\', '*', ,(', ')', '/', '+', '<', '>', ';', '"', '&', '|', '(&', '(|', ')(', ',', '!', '=', ')&', ' ', '*)', '))', '&(', '+)', '=)','cn=', 'sn=', '=*', '(|','mail', 'objectclass', 'name'
SSL	'<!−', '−−>', '#', '+', ',', '"', 'access.log', 'bin/', 'cmd', 'dir', 'dategmt', 'etc/', '#exec', 'email', 'fromhost', 'httpd', 'log/' 'message', 'odbc','replyto', 'sender', 'toaddress', 'virtual', 'windows','#echo', '#include', 'var', '+connect', '+statement', '/passwd', '.conf', '+l', '.com', ':\\'

Table 2. Some of the feature frequency (Table 1) by attack type and length content depending on the payload field area.

Feature	Description
url_length	Length of URL area
url_kwd_wget_cnt	Number of frequency of inclusion of 'wget' in URL area
url_kwd_cat_cnt	Number of frequency of inclusion of 'cat' in URL area
url_kwd_passwd_cnt	Number of frequency of inclusion of 'passwd' in URL area
url_query_length	Length of QUERY area
query_kwd_wget_cnt	Number of frequency of inclusion of 'wget' in QUERY area
body_length	Length of BODY area
body_kwd_wget_cnt	Number of frequency of inclusion of 'wget' in BODY area
http_method_HEAD	Whether http method value is 'HEAD'
http_method_PUT	Whether http method value is 'PUT'
digits_freq	Frequency of inclusion of numbers in URL and QUERY areas

Such extracted features show that even within the same dataset, the distribution of features can differ by event, and even in the same type of attack, the distribution of features differ depending on the domain. In addition, even for features that characterize the type of attack, feature distribution varies depending on the characteristics of the classified fields.

The second is the feature extraction method for domain characteristics.

In order to apply LSA on the payload as a feature extractor, the payload data must be vectorized first. In this paper, we used term frequency-inverse document frequency (TF-IDF) [45] to achieve that. Nevertheless, since the payload often consists of seemingly unmeaningful combination of characters instead of general words and sentences, we obtained the vectorized values, using a hash of a limited size of characters instead of words.

Furthermore, it removes noise by applying a truncated SVD for LSA and extracts only the characteristics.

Using this method, 593 and 300 features have been extracted herein based on the characteristics of the domain information. Moreover, some additional common information was used to extract the method, version, and host related information for HTTP as eight binary values, and the information about the character format for URI and Query areas was also extracted as a feature.

The extracted features, using the methods suggested in this study, seemed to characterize the intrusion detection events better than the existing methods.

3.3. Optimizing Latent Subspace for Both Source and Target Domain Data

This chapter proposes feature-based transfer learning [25] to optimize the space and distribution of features between source and target domains based on the payload-based extracted features in the previous chapter. This helps in obtaining optimized latent subspaces S and T, as they are similar to each other. In addition, we can create a model using optimized latent subspace S and make predictions with optimized latent subspace T to obtain labeled data of target domain T [46].

The optimization methods suggested in this chapter are shown in Figure 4. We apply principal component analysis (PCA) to extract characteristics only for given S and T and to obtain S and T with a projection of the same size. Optimization is performed through similarity parameters β and distance measures $D(*, *)$ for the two S' and T' to obtain V_s and V_t, respectively, which are similar to each other.

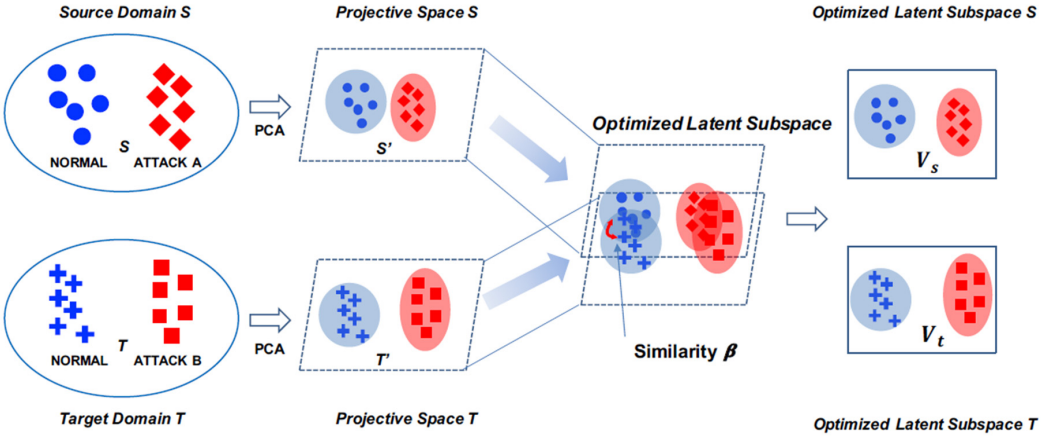

Figure 4. Optimizing latent subspace for both the source and target domain.

To summarize, the issue of optimizing feature-based similarity for the two domains that have been projected by PCA can be considered as the well-known Equation (1). However, here, $l(*, *)$ is the distance measure for the two given matrixes.

$$\underset{V_s V_s^T = I,\ V_t V_t^T = I}{min}\ l(V_s, S') - l(V_t,\ T') + \beta D(V_s,\ V_t) \tag{1}$$

By introducing the distance measures $l(*, *)$ and $D(*, *)$, used in Zhao et al. [41], Equation (1) can be expressed as Equation (2). However, here, P_s is a projection matrix that converts V_s to the S' coordinate system, and P_t is a projection matrix that converts V_s to the T' coordinate system.

$$G(V_s,\ V_t,\ P_{s'},\ P_{t'}) = \underset{V_s V_s^T = I,\ V_t V_t^T = I}{min}\ ||S' - V_s P_s|| - ||T' - V_t P_t|| + \beta ||V_s - V_t|| \tag{2}$$

Therefore, the optimal value is obtained by applying AdaDelta [47], which is quantified for Equation (2) to obtain the optimization goals of similar source domain V_s and target domain V_t.

First, the partial differential coefficient for each application of Equation (2) is as follows:

$$\frac{\partial G}{\partial V_s} = 2\left(V_s P_s P_s^T - S' P_s^T + \beta(V_s - V_t)\right) \tag{3}$$

$$\frac{\partial G}{\partial V_t} = 2\left(V_t P_t P_t^T - T' P_t^T + \beta(V_t - V_s)\right) \tag{4}$$

$$\frac{\partial G}{\partial P_s} = 2\left(V_s S' - V_s^T V_s P_s\right) \tag{5}$$

$$\frac{\partial G}{\partial P_t} = 2\left(V_t T' - V_t^T V_t P_t\right) \tag{6}$$

Here, if the variables V_s and V_t, which we have final interest in, are fixed, the optimal values of Equations (5) and (6) are zero, and accordingly, P_s and P_t are arranged into the following:

$$P_s = \left(V_s^T V_s\right)^{-1}\left(V_s^T S'\right) \tag{7}$$

Therefore, it is possible to obtain V_s and V_t, by repeating optimization through AdaDelta, along with Equation (7). However, since existing AdaDelta performs optimization for each individual element, it does not consider the relation between matrix elements for V_s and V_t. Thus, to consider the relation between matrix elements for V_s and V_t, AdaDelta is quantified and applied in a matrix-approximation-based optimization method by introducing the following.

1. All operations are calculated based on matrix operations.
2. Initialize with random number N (0, 1) for the initial value of V_s and V_t.
3. Estimates of changes (g_s, g_t, z_s, z_t) are estimated using Frobenius Norm.
4. Then, select the relation that is orthogonal to each other through QR decomposition as the result value for V_s and V_t, either at the initial or calculated value.

The finally proposed feature-based transfer learning algorithm can be presented as shown in Algorithm 1 below.

Algorithm 1: Optimization performance algorithm.

input: T, S, β,
feature k, step = 10,000, tol = 1×10^{-4},
$\gamma_s = 0.1$, $\gamma_t = 0.1$, $h_s = 0.1$, $h_t = 0.1$,
$b_s = 1 \times 10^{-7}$, $b_t = 1 \times 10^{-7}$
Output: V_t, V_s
Step 1. Normalize T', S'
Step 2. $S' = \text{PCA(S)}$, $T' = \text{PCA(T)}$, for Features k
Step 3. Initialize: V_s, $V_t \sim N(0, 1)$
$V_s = Q \leftarrow QR(V_s)$ ***QR is QR Decomposition
$V_t = Q \leftarrow QR(V_t)$ ***QR is QR Decomposition
$P_s = {V_s}^T S'$, $P_t = {V_t}^T T'$,
Step 4. While Optimized Step 4-1~4-6 not converge or step < steps
Step 4-1. Calculate Gradient, J_s, J_t
$J_s = \frac{\partial G}{\partial V_s}$, $J_t = \frac{\partial G}{\partial V_t}$
Step 4-2. Calculate Frobenius Norm, g_s, g_t
$g_s = \left|\left|{J_s}^T J_s\right|\right|_F$, $g_t = \left|\left|{J_t}^T J_t\right|\right|_F$
Step 4-3. Update b_s, b_t
$b_s \leftarrow \sqrt{\gamma_s b_s + (1-\gamma_s) g_s}$,
$b_t \leftarrow \sqrt{\gamma_t b_t + (1-\gamma_t) g_t}$
Step 4-4. Update V_s, V_t
$K_s = \frac{h_s}{b_s} J_s$, $K_t = \frac{h_t}{b_t} J_t$,
$V_s \leftarrow V_s - K_s$, $V_t \leftarrow V_t - K_t$
Step 4-5. Update h_s, h_t
$z_s = \left|\left|{K_s}^T K_s\right|\right|_F$, $z_t = \left|\left|{K_t}^T K_t\right|\right|_F$,
$h_s \leftarrow \sqrt{\gamma_s h_s + (1-\gamma_s) z_s}$,
$h_t \leftarrow \sqrt{\gamma_t h_t + (1-\gamma_t) z_t}$
Step 4-6. Calculate P_s, P_t
$P_s = \left({V_s}^T V_s\right)^{-1}\left({V_s}^T S'\right)$, $P_t = \left({V_t}^T V_t\right)^{-1}\left({V_t}^T T'\right)$

End while

Step 5. Calculate orthogonal matrix Q
$V_s = Q \leftarrow QR(V_s)$ ***QR is QR Decomposition
$V_t = Q \leftarrow QR(V_t)$ ***QR is QR Decomposition

4. Experiments and Evaluation

In this chapter, experiments were conducted and evaluated through three scenarios, as shown in Table 3 below, for transfer learning, which is proposed to solve the shortage of training data through a knowledge transfer between the two domains. In addition, in the case when transfer learning is not used, the performance comparison between No-TL and heterogeneous-based transfer learning, proposed by Zhao et al. [24], evaluates the superiority of the transfer learning proposed in this study. Thus, the accuracy and F1-score are compared for the labeled data generated from each experiment.

Table 3. Description and configuration by scenario.

	Scenario 1	Scenario 2	Scenario 3
Objective	Is an accurate detection possible when a new type of attack occurs that does not exist in the training dataset?	Is a well-trained model reusable in other network environments?	Will it perform well in the real environment?
Experiment setting	Transfer learning comparison experiment for different attack types on the same equipment	Transfer learning comparison experiment for attack types on different equipment of the same model	Scenario 1 Scenario 2
Source Domain	Specific attack type dataset with label	Specific target dataset with label	Labeled dataset
Target Domain	Unlabeled specific attack type dataset	Unlabeled specific target dataset	Unlabeled dataset
Inter-domain Dataset	Same (PKDD2007)	Different (PKDD2007 and CSIC2012)	Different (PKCS, WAF2020)
Inter-domain Feature space	Same	Same	Same
Inter-domain Feature distribution	Different	Different	Different

In the first scenario, experiments are performed when labels are produced for different types of attacks, using the labels granted for a particular type of attack, provided that the distributions between the features differ due to the transfer learning according to the type of attack that occurred in the same dataset.

The second scenario is the same model, but it assumes that the distribution between each feature is different for different equipment due to the transfer learning that occurred in the different datasets. The labeled, well-known source domain of the existing equipment is used to apply to the other target domain of the same model.

In the third scenario, experiments are performed on the dataset in the real environment.

In addition, in Section 4.5, a performance comparison and parameter sensitivity analysis of various algorithms are performed.

4.1. Dataset

The selection of the dataset used in this experiment considered the following conditions to implement the three scenarios presented in Table 3: the dataset's intrusion detection area, presence of labels, multi class, payload, etc. Two of the public datasets in various fields of intrusion detection, such as those in Table 4, and the web application firewall dataset in the real environment were selected as datasets to satisfy this requirement. The selected datasets are PKDD2007 (ECML PKDD2007 Discovery Challenge dataset) [48], CSIC2012 (HTTP CSIC Torpeda 2012 dataset) [49], and WAF2020 (web application firewall real dataset) [50].

4.1.1. PKDD2007

The PKDD2007 dataset was created in 2007 through Challenger for web traffic analysis at the 18th European Machinery Conference (ECML) and the 11th European Conference (PKDD) on knowledge discovery principles and practices in databases. As part of Challenger, the dataset was provided with normal traffic and seven types of attack traffic [48,51].

The dataset included 35,006 requests as normal traffic and 15,110 requests as attacks. The PKDD2007 dataset recorded and generated traffic and processed some information, including replacing all URL, parameter, and values with randomly strings. The seven types of attacks were Normal, SQL Injection, Path Traversal, XSS, Command Extraction, XPATH Injection, LDAP Injection, and SSI Attack. The dataset is in the XML file format, comprising reqContext, class, and request; request is divided into Method, Protocol, URI, Query, and Headers.

Table 4. Description and datasets (PKDD, CSIC2012, and WAF2020).

Datasets	Labeled	Class	Payload	Sum	Etc.
PKDD2007	O	7 EA	O	50,116	Normal: 35,006 Attack: 15,110
CSIC2012	O	10 EA	O	65,767	Normal: 8363 Attack: 57,404
WAF2020	O	13 EA	O	67,407	Normal: 10,000 Attack: 57,407

4.1.2. CSIC2012

The CSIC2012 dataset was presented in the TORPEDA Framework in RECSI2012 in 2012. The TORPEDA framework is used for generating labeled web traffic for the evaluation and testing of web application firewalls (WAF) [49–53]. The data comprises 10 classes, including 8363 requests classified as normal, 16,456 requests classified as anomalous, and 40,948 requests classified as attacks. The 10 types of attacks were Normal, SQLi, Format String, Buffer Overflow, XPath, XSS, CRLFi, SSI, LDAP, and Anomalous. The dataset is in the XML file format, comprising label and request. Request is divided into Method, Protocol, Path, Headers, and Body.

4.1.3. WAF2020

The WAF2020 dataset is the data collected over a year from the web application firewall (WAF) established and operated by Company A's SOC [50]. The dataset was detected in the WAF based on users that accessed the portal-based web application environment. The dataset was divided into 13 types of attacks, among which the false-detection event that detected a normal user as an attack was selected as Normal. The data comprised 10,000 requests as Normal and 57,407 requests as attack. The thirteen types of attacks were Normal, Default Page Access, HTTP Method Restrictions, Directory Access, URL Extension Access, Command Injection, XSS, SQL Injection, Header Vulnerability, Application Vulnerability, SSI Injection, LDAP Injection, and Directory Listing. The dataset is in csv file format and is divided into Method, Protocol, URI, URI_Query, Body, etc.

In particular, the attack type of each dataset was defined with a different name and similar attack types were matched and sorted out for experimentation. Selected attack types were Normal, XSS, SQLi (defined as SQL Injection in PKDD2007), LDAPi (defined as LDAP Injection in PKDD2007), SSI (defined as SSI Attack in PKDD2007), OS Commander.

4.2. Data Preprocessing

At this stage, the data are structured and preprocessed so that the collected datasets (PKDD2007, CSIC2012, and WAF2020) can be applied to machine learning.

Pre-processing steps are performed in the following order: Normalization, Field Selection, Feature Extractor and Selection, and Sampling, as shown in Figure 5.

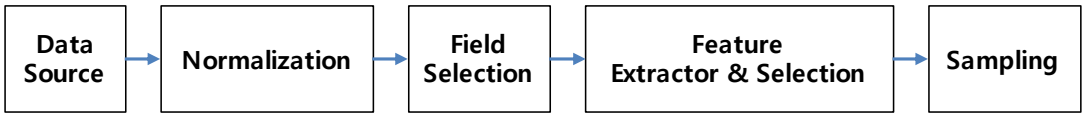

Figure 5. Data Preprocessing.

4.2.1. Normalization

Among the datasets collected, PKDD2007 and CSIC2012 are in the form of unstructured XML, shown in Sample 1 [54] below, and WAF2020 is composed of structured data. To start with, we equally perform normalization with Method, Version, URI, Query, and Body for the applicable dataset. In particular, values that represent user information, etc., are included in the Body field. Then, the letter "\n" will be removed and URI decoding will be applied for URI, Query, and Body.

Sample 1: Contents of xml File [54].

```
<sample id="88888">
    <reqContext>
        <os>WINDOWS</os>
      ... ..
        </reqContext>
    <class>
        <type>xxxxxxx</type>
      <inContext>FALSE</inContext>
        <attackIntervall>xxxxxx</attackIntervall>
    </class>
    <request>
        <method>POST</method> <protocol>HTTP/1.0</protocol>
        <uri></cxxxxcktp/WKKj_l3333/iAdgxxxx6mMT.gif]></uri>
      <query><D1d=%5Bddt%2Fsxl&loh=5nddd5Ni=aL1]></query>
      ... ...
        Accept-Language: ++insert+??+++from+++ith+
      Referer: http://www.xxxxx.biz/icexxx/uoxxx.zip
      User-Agent: Mozilla/7.9 (Machintosh; K; PPD 6.1]></headers>
    </request>
</sample>
```

4.2.2. Field Selection

For each dataset, we select the field that is going to be used for the experiment. In PKDD2007 and CSIC2012, the category type can be divided into Class, Method, and Version, and the text type selects URI, Query, and Body. At this time, missing values occur if a Query does not exist in the generated data of CSIC2012. Therefore, the missing values are batch-processed with "?" for the field.

4.2.3. Feature Extractor and Selection

Feature extraction is performed with signature-based feature extraction and text-vectorization-based feature extraction according to the selected field. Signature-based feature extraction first performs numericalization for Method and Version. Method is expressed as three features with numerical values of "0" and "1" depending on the existence of GET, POST, and PUT, and Version is expressed as two features with numerical values of "0" or "1" depending on the existence of HTTP/1.0 and HTTP/1.1. Second, character-based features by attack type are extracted as numerical features in URI, Query, and Body fields. Hence, eight common-related features, 196 URI-related features, 196 Query-related features, and 196 Body-related features are created. Text-vectorization-based feature extraction performs text vectorization for URI, Query, and Body characters using LSA, which compresses the hashing vector, comprising 10,000 features, into 100 features again. As a result, URI, Query, and Body strings are converted into features with a size of 100 each. This merges the extracted features into an index basis and will eventually produce a dataset with 901 features.

4.2.4. Sampling

After preprocessing, each dataset is randomly sampled so that it contains 1000 samples from the attacker's label and 4000 samples from the normal label for experiments by scenario.

4.3. Evaluation Environment & Metrics

This experimental environment was implemented using Python in Ubuntu 18.04.2 LTS. The classical machine learning library Scikit-learn 0.20.4 was used. Hardware specifications include NVidia GeForce RTX 2060 * 2 for GPU, 128 GB RAM, an 8 TB hard disk, and an AMD Ryzen Threadripper 1900X 8-core processor environment.

Accuracy and F1-score for the predicted target data after transfer learning were selected as the evaluation method for this experiment. The three scenario approaches (No-TL, HeTL, and PF-TL) presented in Table 3 were evaluated using Random Forest, SVM, MLP, and KNN as classification algorithms. To calculate the selected evaluation method, confusion matrix metrics were used, which are commonly used in machine learning. As shown in Table 5, the confusion matrix [55] comprises four information points.

Table 5. Confusion matrix.

		Predicted	
		Positive	**Negative**
Labeled	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Based on these four information points, four scales can be evaluated, as shown in Table 6 [55]. First, accuracy is a measure of evaluating how accurately a model classifies. Second, precision is a measure of evaluating how reliable the results that were predicted through the model are. Third, recall is a measure compared to precision, which can indicate how well the predicted results reflect the actual results. In other words, it is a measure related to the practicality of the model. Fourth, F1-score is a number that divides the product of precision and recall by the sum of the two and then multiplies by two.

Table 6. Performance Measurement.

Rule	**Formula**
Accuracy	(TP + TN)/(TP + TN + FP + FN)
Precision	TP/(TP + FP)
Recall	TN/(TP + FN)
F1-Score	(2 × Recall × Precision)/(Recall + Precision)

Therefore, in this study, we are going to measure performance through the accuracy and F1-score of the model.

4.4. Experimental Result

As presented in Table 3, the experiment was conducted through three scenarios, and results were obtained.

4.4.1. Scenario 1: Transfer Learning Comparison for Different Attack Types on the Same Equipment

The first scenario is based on "Can we accurately detect when a new type of unknown attack occurs?" in a security system. In other words, the dataset being targeted is the same, the feature space between domains is the same, and the feature distribution is different.

To test this, the PKDD2007 dataset was used, the web application intrusion detection dataset was set to be identical, and the source and target domains were set differently

for each type of labeled attack. The feature space used the 901 features presented earlier. Feature distribution is differently distributed for each detection event and for HTTP header and request values.

For the types of attacks used in the experiment, SQLi, LDAPi, XSS, and SSI in the PKDD2007 dataset were selected as four types of attacks that are uniformly distributed, and source and target domains were set up and experimented, as shown in Table 7. In addition, for more accurate results, the experiments were performed by changing the target domain type (LDAPi, XSS, and SSI) to the same type of source domain (SQLi), as shown in Table 8.

Table 7. Performance comparison of transfer learning on Scenario 1-1.

Source	**PKDD2007**	**SQLi**	**LDAPi**	**XSS**
Target	**PKDD2007**	**LDAPi**	**XSS**	**SSI**
Accuracy	No-TL	0.6548	0.8134	0.8004
	HeTL(2017)	0.9696	0.9690	0.9668
	PF-TL	0.9988	0.9990	0.9996
F1-Score	No-TL	0.4418	0.4398	0.0060
	HeTL(2017)	0.9180	0.9109	0.9096
	PF-TL	0.9997	0.9990	0.9990
Source	**CSIC2012**	**SQLi**	**LDAPi**	**XSS**
Target	**CSIC2012**	**LDAPi**	**XSS**	**SSI**
Accuracy	No-TL	0.9730	1	0.8638
	HeTL(2017)	0.9676	0.9686	0.9678
	PF-TL	1	1	1
F1-Score	No-TL	0.9368	1	0.4837
	HeTL(2017)	0.9120	0.9150	0.9126
	PF-TL	1	1	1

Table 8. Performance comparison of transfer learning on Scenario 1-2.

Source	**PKDD2007**	**SQLi**	**LDAPi**	**XSS**
Target	**PKDD2007**	**LDAPi**	**XSS**	**SSI**
Accuracy	No-TL	0.6548	0.7954	0.8598
	HeTL(2017)	0.9696	0.9678	0.9704
	PF-TL	0.9988	0.9994	0.9992
F1-Score	No-TL	0.4418	0.6129	0.5980
	HeTL(2017)	0.9180	0.9127	0.9203
	PF-TL	0.9997	0.9985	0.9980
Source	**CSIC2012**	**SQLi**	**SQLi**	**SQLi**
Target	**CSIC2012**	**LDAPi**	**XSS**	**SSI**
Accuracy	No-TL	0.9730	0.9730	0.8176
	HeTL(2017)	0.9696	0.9678	0.9704
	PF-TL	0.9988	0.9994	0.9992
F1-Score	No-TL	0.9368	0.0.936	0.1648
	HeTL(2017)	0.9180	0.9127	0.9203
	PF-TL	0.9997	0.9985	0.9980

As seen in Tables 7 and 8, the experimental results show that the predictive model performance for different types of attacks in the same dataset is improved, as No-TL < HeTL(2017) < PF-TL in accuracy and F1-score. In particular, the accuracy and F1-score vary widely by type when transfer learning is not used. However, the application of transfer learning shows that the deviation is not significant. This enables transfer learning

to increase the training dataset for the deficient attack types when the variation in the amount of training data by attack type is significant. This can also be used by transfer learning to increase detection accuracy when events that are not included in the training data occur.

4.4.2. Scenario 2: Transfer Learning Comparison for Attack Types from other Equipment

The second scenario is the view on whether a well-trained model can perform equally well in other network environments. In other words, the target dataset is different, the feature space between domains is the same, and the feature distribution is different. To test this, the PKDD2007 dataset, which is a web application intrusion detection dataset, was selected as the source domain, the CSIC2012 dataset was selected as the target domain, and 901 features were used in common, as shown in Scenario 1 for feature space.

The results of Table 9 show that the accuracy and F1-score performed better when transfer learning was applied compared to when it was not applied. The results show that accuracy is high, but the F1-score is low when transfer learning takes place from XSS to SSI. This suggests that the performance was very poor in recall or precision. This also shows that the use of transfer learning reduces bias against data, which increases model performance, such as the F1-score. Moreover, while the previous model does not perform well when applied to other security equipment, it can be used for other security equipment if the training data are optimized for the target security equipment through transfer learning.

Table 9. Performance comparison of transfer learning on Scenario 2.

Source	**PKDD2007**	**XSS**	**XSS**	**XSS**	**XSS**
Target	**CSIC2012**	**XSS**	**SSI**	**SQLi**	**LDAPi**
Accuracy	No-TL	0.1204	0.8086	0.8370	0.2166
	HeTL(2017)	0.9690	0.9694	0.9698	0.9682
	PF-TL	0.9998	1	0.9994	0.9996
F1-Score	No-TL	0.1804	0.0825	0.3369	0.3301
	HeTL(2017)	0.9162	0.9172	0.9187	0.9138
	PF-TL	0.9995	1	0.9985	0.9990

4.4.3. Scenario 3: Application of Scenario 1 and 2 to the Real Environment

The third scenario is the view on whether what was done in Scenarios 1 and 2 can be well applied in the real environment. To this end, we applied the web application firewall dataset WAF2020 collected in the real environment to Scenarios 1 and 2.

First, Scenario 1 was applied to the WAF2020 Dataset. As shown in Table 10, the use of PF-TL in the real environment shows high accuracy and F1-score. In particular, if the domain applies the model without using transfer learning from OS Commander to SQLi, it shows a low accuracy of 0.21; however, when transfer learning is applied, high accuracy of 0.99 or higher is obtained.

Table 10. Performance comparison of transfer learning on Scenario 3-1.

Source	**WAF2020**	**SQLi**	**XSS**	**OS Cmd**
Target	**WAF2020**	**XSS**	**OS Cmd** [1]	**SQLi**
Accuracy	No-TL	0.8755	0.8157	0.2116
	HeTL(2017)	0.9720	0.9726	0.9732
	PF-TL	0.9999	0.9999	0.9999
F1-Score	No-TL	0.5546	0.1456	0.2196
	HeTL(2017)	0.9247	0.9265	0.9282
	PF-TL	0.9999	0.9999	0.9999

[1] OS Cmd = OS Commander.

Second, transfer learning was tested with WAF2020 for models created in PKDD2007+ CSIC2012, as shown in Scenario 2. Table 11 contains a new dataset called PKCS, created by merging existing common datasets (PKDD2007 and CSIC2012) and applying the model obtained from this to the WAF2020 dataset. As a result, an accuracy of lower than 0.53 was obtained when transfer learning was not applied, whereas a 0.99 or higher accuracy was obtained when transfer learning was applied.

Table 11. Performance comparison of transfer learning on Scenario 3-2.

Source	**PKCS**	**SQLi**	**SQLi**	**XSS**	**XSS**
Target	**WAF2020**	**SQLi**	**XSS**	**XSS**	**SQLi**
Accuracy	No-TL	0.1951	0.2555	0.5296	0.5333
	HeTL(2017)	0.9715	0.9727	0.9708	0.9716
	PF-TL	0.9998	0.9998	0.9999	0.9999
F1-Score	No-TL	0.2512	0.3112	0.4596	0.4405
	HeTL(2017)	0.9234	0.9269	0.9215	0.9236
	PF-TL	0.9995	0.9995	0.9997	0.9997

This shows that transfer learning can be applicable to data in the real environment by applying it to the existing public dataset.

4.5. Limitations and Discussion

In this study, three different scenarios for domain knowledge transfer were used to create optimized training data for target domain, using payload feature-based transfer learning. In addition, by comparing with No-TL and HeTL, we showed that the proposed transfer learning could create the dataset that might effectively be used to improve model accuracy. Considering how the machine learning applications in intrusion detection have been dependent on the training data, the research result is very promising. Through the aforementioned three scenarios, our proposal can significantly enhance the detection accuracy of unknown attacks based on the same dataset and guarantee understandable performance on a dataset of different security devices when using the transfer learning.

Nevertheless, in order to achieve the desired outcomes, the following limitations regarding the labels of source and target domains, feature spaces and distributions, were put into the consideration [56].

First, the source and target domains must have the same feature spaces. To solve this, PCA was performed to adjust the number of features.

Second, the source and target domains must have the same number of records. Thus, the number between domains was identically adjusted through sampling.

Third, the similarity between the source and target domain is high and must be entered in pairs when being input. Hence, similarity parameter β was used and matched and was set as $\beta = 1$ for this experiment.

In this experiment, we will look into the proposed algorithm and the similarity parameter β.

4.5.1. Algorithm Selection

To select the best algorithm, the PKDD2007 dataset and the CSIC2012 dataset were selected as the target and source domains, respectively.

The dataset configuration used the entire datasets without distinction of attack types. The data used were applied with Random Forest, Linear SVM, MLP, and KNN algorithms to compare the transfer learning results. As shown in Table 12, the comparison shows that Random Forest has the best performance. In particular, it has the highest accuracy and a high improvement rate of 95.4% when transfer learning is applied. In addition, the performance of the KNN algorithm was excellent when transfer learning was not used, but

its performance improvement was not high when transfer learning was applied. Therefore, in this paper, Random Forest was applied in the experiment.

Table 12. Algorithm comparison of transfer learning on Scenario 2.

Source Target	PKDD2007 CSIC2012	RF [1]	SVM	MLP	KNN
Accuracy	No-TL	0.2010	0.1900	0.2112	0.6946
	HeTL(2017)	0.9624	0.5162	0.7414	0.8132
	PF-TL	0.9638	0.8988	0.8966	0.9146
F1-Score	No-TL	0.3135	0.3193	0.3219	0.1724
	HeTL(2017)	0.9260	0.3091	0.1465	0.2634
	PF-TL	0.9032	0.6645	0.7608	0.7303

RF [1] = Random Forest.

4.5.2. Parameter Sensitivity

For inter-domain optimization, two hyperparameters must be set: similarity parameter β [46] and the size of new feature space k. Therefore, we decided on the sensitivity of parameters through an experiment. First, we analyzed the sensitivity to similarity parameter β.

To analyze the sensitivity to similarity parameter β, the size of the new feature space k was fixed with the same value of 100 as the control variable.

As seen in Figure 6, as the similarity parameter β increases, the similarity between the target and the source domains increases. In particular, the source and target domains are almost identical when β is greater than 0.8. In this experiment, the β value was set to 1 for more obvious results.

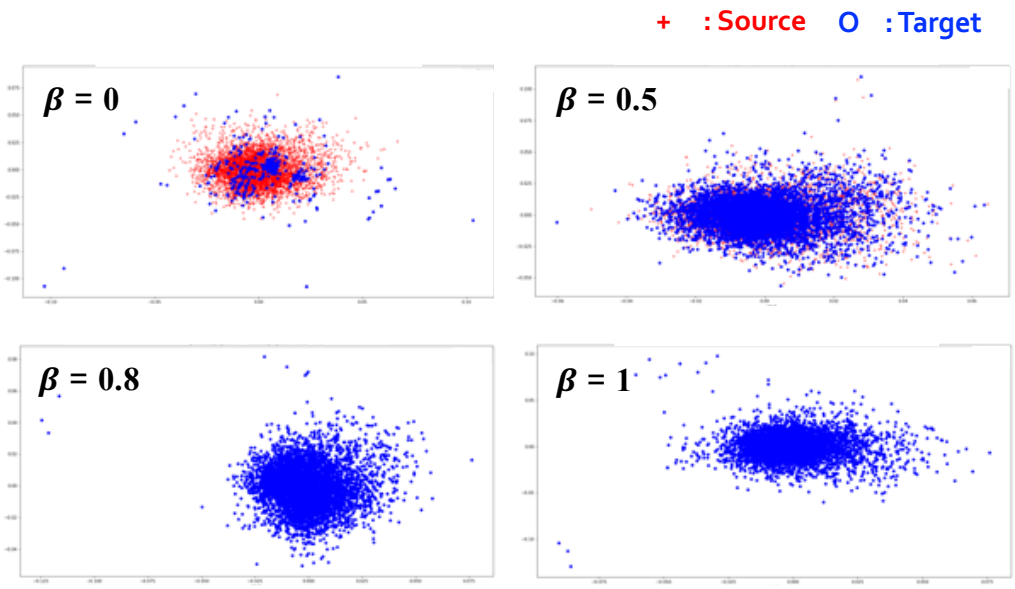

Figure 6. Distribution diagram according to similarity parameter β (class: LDAP injection, source domain: PKDD2007, target domain: CSIC2012).

4.5.3. Comparison of Methods of Creating Training Data

While there are multiple ways to create training data, including Generative Adversarial Net (GAN) and SVM Feature selection, many studies suggest creating training data using the same datasets that are labeled together. These methods focus on creating new training datasets based on the same dataset [57–61]. Nonetheless, our proposal focuses on transferring data from both the same and different datasets using three scenarios.

5. Related Works

5.1. Payload Based Intrusion Detection

Torrano-Gimenez et al. [13] proposed a method for extracting feature from the payload using n-gram to increase the accuracy of web intrusion detection. Generic feature selection (GFS) measurements were applied to decrease the number of features extracted by n-gram, and the accuracy tended to improve through classification algorithms (Random Forest, Random Tree, CART, and C4.5). Betarte et al. [14] extracted character-based features with the help of an expert's analysis in the payload, based on which classification algorithms (RF, KNN, and SVM) suggested better performance than conventional signature-based detection technology. Min et al. [62] proposed statistical methods that are important in network flows for efficient feature extraction in packet headers and payloads, as well as methods using the Random Forest algorithm after word vector followed by feature extraction via Text-CNN.

However, in these studies, a large number of training data is needed to achieve good results and learned models cannot be reused in other domains.

5.2. Transfer Learning for Intrusion Detection

Transfer learning has commonly been applied to natural language processing and visual recognition; however, its application in the cybersecurity field is insufficiently studied. Intel Labs and Microsoft researchers Chen and Parich [21] proposed a malware classification method using deep-learning transfer learning technology that has low false detection rates and high accuracy through a static malware as the image network (STAMINA). Microsoft's REA+L dataset was used as an applicable dataset and showed improvements over previous studies [22]. Wu et al. [23] proposed the TL-ConvNet model in transfer learning for network intrusion detection. Furthermore, the deep learning algorithm ConvNet was used for feature extraction. Source domain (KDDTest+) was learned through the NSD-KDD dataset and transfer learning was implemented through the prediction of the target domain (KDDTest-21). Moreover, Zhao et al. [24,25] proposed a feature-based HeTL approach to detect unknown attacks based on known detection techniques. NSD-KDD was used as a dataset, and the source and target domains were selected and tested for each type of attack within the same dataset. Subsequent studies [25] also suggest transfer learning that uses clustering to improve the optimization of the source and target domains. However, studies of transfer learning in the field of network intrusion detection have so far been conducted using formalized datasets, such as KDD99 and NSL-KDD, and have been experimented with different types and distinctions of attacks within the same dataset. Therefore, a limitation exists in the application of transfer learning in a real network environment [63,64].

6. Conclusions and Future Work

In this paper, we proposed payload feature-based transfer learning as a solution to the shortage of training data in the field of intrusion detection, which extracts features from the payload of the intrusion detection event. Our proposal confirmed that transfer learning can create optimized training data for a domain that has insufficient labeled data or no label data at all. In addition, the proposal demonstrated three distinctive scenarios in which we proved the proposal's capability of detecting new types of attack events and reusing a well-trained model in another network environment by utilizing a newly created training dataset from PF-TL. Furthermore, by validating real-life datasets, we showed that our proposal can indeed be used in practice. In the future, we will cover methods to process imbalanced training data and to create big data specifically optimized for the intrusion detection field to increase the utility of the training data created from transfer learning.

Author Contributions: All authors contributed to this work. Conceptualization, I.J.; methodology, I.J. and J.L.; software, I.J.; validation I.J. and J.L.; writing—original draft preparation, I.J.; writing—review

and editing, I.J.; visualization, I.J. and J.L.; supervision, J.L. and H.K.K.; project administration, J.L. and H.K.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

References

1. Hussain, A.; Mohamed, A.; Razali, S. A Review on Cybersecurity: Challenges & Emerging Threats. In Proceedings of the Proceedings of the 3rd International Conference on Networking, Information Systems & Security, Marrakech, Morocco, 31 March 2020; pp. 1–7.
2. Ibor, A.E.; Oladeji, F.A.; Okunoye, O.B. A Survey of Cyber Security Approaches for Attack Detection, Prediction, and Prevention. *IJSIA* **2018**, *12*, 15–28. [CrossRef]
3. Vinayakumar, R.; Soman, K.P.; Poornachandran, P.; Akarsh, S. Application of Deep Learning Architectures for Cyber Security. In *Cybersecurity and Secure Information Systems*; Hassanien, A.E., Elhoseny, M., Eds.; Advanced Sciences and Technologies for Security Applications; Springer International Publishing: Cham, Switzerland, 2019; pp. 125–160. ISBN 978-3-030-16836-0.
4. Prasad, R.; Rohokale, V. Artificial Intelligence and Machine Learning in Cyber Security. In *Cyber Security: The Lifeline of Information and Communication Technology*; Springer Series in Wireless Technology; Springer International Publishing: Cham, Switzerland, 2020; pp. 231–247. ISBN 978-3-030-31702-7.
5. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
6. Pachghare, V.K.; Khatavkar, V.K.; Kulkarni, P.A. Pattern Based Network Security Using Semi-Supervised Learning. *IJINS* **2012**, *1*, 228–234. [CrossRef]
7. Gou, S.; Wang, Y.; Jiao, L.; Feng, J.; Yao, Y. Distributed Transfer Network Learning Based Intrusion Detection. In Proceedings of the 2009 IEEE International Symposium on Parallel and Distributed Processing with Applications, Chengdu, China, 10–12 August 2009; pp. 511–515.
8. Epp, N.; Funk, R.; Cappo, C. Anomaly-based Web Application Firewall using HTTP-specific features and One-Class SVM. In Proceedings of the 2nd Workshop Regional de Segurança da Informação e de Sistemas Computacionais, UFSM, Sta María, Brazil, 25–29 September 2017.
9. RFC 7540-Hypertext Transfer Protocol Version 2 (HTTP/2). Available online: https://tools.ietf.org/html/rfc7540 (accessed on 18 February 2021).
10. Ojagbule, O.; Wimmer, H.; Haddad, R.J. Vulnerability Analysis of Content Management Systems to SQL Injection Using SQLMAP. In Proceedings of the SoutheastCon 2018, St. Petersburg, FL, USA, 19–22 April 2018; pp. 1–7.
11. Blowers, M.; Williams, J. Machine Learning Applied to Cyber Operations. In *Network Science and Cybersecurity*; Pino, R.E., Ed.; Advances in Information Security; Springer: New York, NY, USA, 2014; Volume 55, pp. 155–175. ISBN 978-1-4614-7596-5.
12. Zhang, M.; Xu, B.; Bai, S.; Lu, S.; Lin, Z. A Deep Learning Method to Detect Web Attacks Using a Specially Designed CNN. In *Neural Information Processing*; Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.-S.M., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2017; Volume 10638, pp. 828–836. ISBN 978-3-319-70138-7.
13. Pastrana, S.; Torrano-Gimenez, C.; Nguyen, H.T.; Orfila, A. Anomalous Web Payload Detection: Evaluating the Resilience of 1-Grams Based Classifiers. In *Intelligent Distributed Computing VIII*; Camacho, D., Braubach, L., Venticinque, S., Badica, C., Eds.; Studies in Computational Intelligence; Springer International Publishing: Cham, Switzerland, 2015; Volume 570, pp. 195–200. ISBN 978-3-319-10421-8.
14. Betarte, G.; Pardo, A.; Martinez, R. Web Application Attacks Detection Using Machine Learning Techniques. In Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 1065–1072.
15. Chen, C.; Gong, Y.; Tian, Y. Semi-Supervised Learning Methods for Network Intrusion Detection. In Proceedings of the 2008 IEEE International Conference on Systems, Man and Cybernetics, Singapore, 12–15 October 2008; pp. 2603–2608.
16. Kim, J.; Jeong, J.; Shin, J. M2m: Imbalanced Classification via Major-to-Minor Translation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–18 June 2020; pp. 13893–13902.
17. Csurka, G. A Comprehensive Survey on Domain Adaptation for Visual Applications. In *Domain Adaptation in Computer Vision Applications*; Csurka, G., Ed.; Advances in Computer Vision and Pattern Recognition; Springer International Publishing: Cham, Switzerland, 2017; pp. 1–35. ISBN 978-3-319-58346-4.
18. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A Comprehensive Survey on Transfer Learning. *Proc. IEEE* **2021**, *109*, 43–76. [CrossRef]
19. Weiss, K.; Khoshgoftaar, T.M.; Wang, D. A Survey of Transfer Learning. *J. Big Data* **2016**, *3*, 9. [CrossRef]
20. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
21. Microsoft Researchers Work with Intel Labs to Explore New Deep. Available online: https://www.microsoft.com/security/blog/2020/05/08/microsoft-researchers-work-with-intel-labs-to-explore-new-deep-learning-approaches-for-malware-classification/ (accessed on 18 February 2021).

22. Use Transfer Learning for Efficient Deep Learning Training on Intel. Available online: https://software.intel.com/content/www/us/en/develop/articles/use-transfer-learning-for-efficient-deep-learning-training-on-intel-xeon-processors.html (accessed on 18 February 2021).
23. Wu, P.; Guo, H.; Buckland, R. A Transfer Learning Approach for Network Intrusion Detection. In Proceedings of the 2019 IEEE 4th International Conference on Big Data Analytics (ICBDA), Suzhou, China, 15–18 March 2019; pp. 281–285.
24. Zhao, J.; Shetty, S.; Pan, J.W.; Kamhoua, C.; Kwiat, K. Transfer Learning for Detecting Unknown Network Attacks. *EURASIP J. Inf. Secur.* **2019**, 1. [CrossRef]
25. Zhao, J.; Shetty, S.; Pan, J.W. Feature-Based Transfer Learning for Network Security. In Proceedings of the MILCOM 2017-2017 IEEE Military Communications Conference (MILCOM), Baltimore, MD, USA, 23–25 October 2017; pp. 17–22.
26. KDD Cup 1999 Dataset. Available online: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed on 18 February 2021).
27. Wang, J.; Chen, Y.; Yu, H.; Huang, M.; Yang, Q. Easy Transfer Learning By Exploiting Intra-Domain Structures. In Proceedings of the 2019 IEEE International Conference on Multimedia and Expo (ICME), Shanghai, China, 8–12 July 2019; pp. 1210–1215.
28. Buczak, A.L.; Guven, E. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1153–1176. [CrossRef]
29. Althubiti, S.; Yuan, X.; Esterline, A. Analyzing HTTP requests for web intrusion detection. In Proceedings of the 2017 KSU Conference on Cybersecurity Education, Research and Practice, Kennesaw, GA, USA, 20–21 October 2017; pp. 1–11.
30. Appelt, D.; Nguyen, C.D.; Panichella, A.; Briand, L.C. A Machine-Learning-Driven Evolutionary Approach for Testing Web Application Firewalls. *IEEE Trans. Rel.* **2018**, *67*, 733–757. [CrossRef]
31. Wang, J.; Zhou, Z.; Chen, J. Evaluating CNN and LSTM for Web Attack Detection. In Proceedings of the 2018 10th International Conference on Machine Learning and Computing, Macau, China, 26 February 2018; pp. 283–287.
32. Betarte, G.; Giménez, E.; Martínez, R.; Pardo, Á. Machine Learning-Assisted Virtual Patching of Web Applications. *arXiv* **2018**, arXiv:1803.05529.
33. Mac, H.; Truong, D.; Nguyen, L.; Nguyen, H.; Tran, H.A.; Tran, D. Detecting Attacks on Web Applications using Autoencoder. In Proceedings of the Ninth International Symposium on Information and Communication Technology (SoICT 2018), Danang, Vietnam, 6–7 December 2018; pp. 1–6.
34. Yin, C.; Zhu, Y.; Fei, J.; He, X. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access* **2017**, *5*, 21954–21961. [CrossRef]
35. Xin, Y.; Kong, L.; Liu, Z.; Chen, Y.; Li, Y.; Zhu, H.; Gao, M.; Hou, H.; Wang, C. Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access* **2018**, *6*, 35365–35381. [CrossRef]
36. Wang, K.; Stolfo, S.J. Anomalous Payload-Based Network Intrusion Detection. In *Recent Advances in Intrusion Detection*; Jonsson, E., Valdes, A., Almgren, M., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3224, pp. 203–222. ISBN 978-3-540-23123-3.
37. Torrano-Gimenez, C.; Nguyen, H.T.; Alvarez, G.; Petrovic, S.; Franke, K. Applying Feature Selection to Payload-Based Web Application Firewalls. In Proceedings of the 2011 Third International Workshop on Security and Communication Networks (IWSCN), Gjovik, Norway, 18–20 May 2011; pp. 75–81.
38. Raissi, C.; Brissaud, J.; Dray, G.; Poncelet, P.; Roche, M.; Teisseire, M. Web Analysis Traffic Challenge: Description and Results. In Proceedings of the ECML/PKDD, Warsaw, Poland, 17–21 September 2007; pp. 1–6.
39. Liu, X.; Liu, Z.; Wang, G.; Cai, Z.; Zhang, H. Ensemble Transfer Learning Algorithm. *IEEE Access* **2018**, *6*, 2389–2396. [CrossRef]
40. Ge, W.; Yu, Y. Borrowing Treasures from the Wealthy: Deep Transfer Learning through Selective Joint Fine-Tuning. *arXiv* **2017**, arXiv:1702.08690.
41. Gao, X.; Shan, C.; Hu, C.; Niu, Z.; Liu, Z. An Adaptive Ensemble Machine Learning Model for Intrusion Detection. *IEEE Access* **2019**, *7*, 82512–82521. [CrossRef]
42. Kreibich, C.; Crowcroft, J. Honeycomb: Creating intrusion detection signatures using honeypots. *ACM SIGCOMM Comput. Commun. Rev.* **2004**, *34*, 51–56. [CrossRef]
43. Nguyen, H.; Franke, K.; Petrovic, S. Improving Effectiveness of Intrusion Detection by Correlation Feature Selection. In Proceedings of the 2010 International Conference on Availability, Reliability and Security, Krakow, Poland, 15–18 February 2010; pp. 17–24.
44. Chanthini, S.; Latha, K. Log based internal intrusion detection for web applications. *Int. J. Adv. Res. Ideas Innov. Technol.* **2019**, *5*, 350–353.
45. Vinayakumar, R.; Alazab, M.; Soman, K.P.; Poornachandran, P.; Al-Nemrat, A.; Venkatraman, S. Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access* **2019**, *7*, 41525–41550. [CrossRef]
46. Zhong, X.; Guo, S.; Shan, H.; Gao, L.; Xue, D.; Zhao, N. Feature-Based Transfer Learning Based on Distribution Similarity. *IEEE Access* **2018**, *6*, 35551–35557. [CrossRef]
47. Zeiler, M.D. ADADELTA: An Adaptive Learning Rate Method. *arXiv* **2012**, arXiv:1212.5701.
48. Gallagher, B.; Eliassi-Rad, T. Classification of HTTP Attacks: A Study on the ECML/PKDD 2007 Discovery Challenge. Available online: https://www.osti.gov/biblio/1113394-classification-http-attacks-study-ecml-pkdd-discovery-challenge (accessed on 18 February 2021).
49. Csic Torpeda 2012, http Data Sets. Available online: http://www.tic.itefi.csic.es/torpeda (accessed on 18 February 2021).

50. IGLOO Security. Available online: http://www.igloosec.com/ (accessed on 18 February 2021).
51. Kok, J.; Koronacki, J.; Mantaras, R.L.; Matwin, S.; Mladeni, D.; Skowron, A. Knowledge Discovery in Databases: PKDD 2007. In Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, 17–21 September 2007.
52. Torrano-Gimenez, C.; Perez-Villegas, A.; Alvarez, G. *TORPEDA: Una Especificacion Abierta de Conjuntos de Datos para la Evaluacion de Cortafuegos de Aplicaciones Web*; TIN2011-29709-C0201; RECSI: San Sebastián, Spain, 2012.
53. Web Attacks Detection Based on CNN-Csic Torpedo 2012 http Data Sets-GitHub. Available online: https://github.com/DuckDuckBug/cnn_waf (accessed on 18 February 2021).
54. ECML/PKDD2007 Datasets. Available online: http://www.lirmm.fr/pkdd2007-challenge/index.html#dataset (accessed on 18 February 2021).
55. Kim, A.; Park, M.; Lee, D.H. AI-IDS: Application of Deep Learning to Real-Time Web Intrusion Detection. *IEEE Access* **2020**, *8*, 70245–70261. [CrossRef]
56. Wu, H.; Yan, Y.; Ye, Y.; Min, H.; Ng, M.K.; Wu, Q. Online Heterogeneous Transfer Learning by Knowledge Transition. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–19. [CrossRef]
57. Yuan, D.; Ota, K.; Dong, M.; Zhu, X.; Wu, T.; Zhang, L.; Ma, J. Intrusion Detection for Smart Home Security Based on Data Augmentation with Edge Computing. In Proceedings of the ICC 2020–2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
58. Wang, H.; Gu, J.; Wang, S. An Effective Intrusion Detection Framework Based on SVM with Feature Augmentation. *Knowl.-Based Syst.* **2017**, *136*, 130–139. [CrossRef]
59. Gu, J.; Wang, L.; Wang, H.; Wang, S. A Novel Approach to Intrusion Detection Using SVM Ensemble with Feature Augmentation. *Comput. Secur.* **2019**, *86*, 53–62. [CrossRef]
60. Zhang, H.; Yu, X.; Ren, P.; Luo, C.; Min, G. Deep Adversarial Learning in Intrusion Detection: A Data Augmentation Enhanced Framework. *arXiv* **2019**, arXiv:1901.07949. [CrossRef]
61. Merino, T.; Stillwell, M.; Steele, M.; Coplan, M.; Patton, J.; Stoyanov, A.; Deng, L. Expansion of Cyber Attack Data from Unbalanced Datasets Using Generative Adversarial Networks. In *Software Engineering Research, Management and Applications*; Lee, R., Ed.; Studies in Computational Intelligence; Springer International Publishing: Cham, Switzerland, 2020; Volume 845, pp. 131–145. ISBN 978-3-030-24343-2.
62. Min, E.; Long, J.; Liu, Q.; Cui, J.; Chen, W. TR-IDS: Anomaly-Based Intrusion Detection through Text-Convolutional Neural Network and Random Forest. *Secur. Commun. Netw.* **2018**, *2018*, 4943509. [CrossRef]
63. The Benefits of Transfer Learning with AI for Cyber Security. Available online: https://www.patternex.com/blog/benefits-transfer-learning-ai-cyber-security (accessed on 18 February 2021).
64. Kneale, C.; Sadeghi, K. Semisupervised Adversarial Neural Networks for Cyber Security Transfer Learning. *arXiv* **2019**, arXiv:1907.11129.