



Article

Combining Unsupervised Approaches for Near Real-Time Network Traffic Anomaly Detection

Francesco Carrera ¹, Vincenzo Dentamaro ² , Stefano Galantucci ^{2,*} , Andrea Iannacone ¹ ,
Donato Impedovo ²  and Giuseppe Pirlo ² ¹ BVTech SpA, 20123 Milano, Italy; f.carrera@bv-tech.it (F.C.); a.iannacone@bv-tech.it (A.I.)² Department of Computer Science, University of Bari "Aldo Moro", 70125 Bari, Italy; vincenzo.dentamaro@uniba.it (V.D.); donato.impedovo@uniba.it (D.I.); giuseppe.pirlo@uniba.it (G.P.)

* Correspondence: stefano.galantucci@uniba.it

Abstract: The 0-day attack is a cyber-attack based on vulnerabilities that have not yet been published. The detection of anomalous traffic generated by such attacks is vital, as it can represent a critical problem, both in a technical and economic sense, for a smart enterprise as for any system largely dependent on technology. To predict this kind of attack, one solution can be to use unsupervised machine learning approaches, as they guarantee the detection of anomalies regardless of their prior knowledge. It is also essential to identify the anomalous and unknown behaviors that occur within a network in near real-time. Three different approaches have been proposed and benchmarked in exactly the same condition: Deep Autoencoding with GMM and Isolation Forest, Deep Autoencoder with Isolation Forest, and Memory Augmented Deep Autoencoder with Isolation Forest. These approaches are thus the result of combining different unsupervised algorithms. The results show that the addition of the Isolation Forest improves the accuracy values and increases the inference time, although this increase does not represent a relevant problematic factor. This paper also explains the features that the various models consider most important for classifying an event as an attack using the explainable artificial intelligence methodology called Shapley Additive Explanations (SHAP). Experiments were conducted on KDD99, NSL-KDD, and CIC-IDS2017 datasets.

Keywords: unsupervised machine learning; anomaly detection; near real-time; network traffic; explainable artificial intelligence; SHAP



Citation: Carrera, F.; Dentamaro, V.; Galantucci, S.; Iannacone, A.; Impedovo, D.; Pirlo, G. Combining Unsupervised Approaches for Near Real-Time Network Traffic Anomaly Detection. *Appl. Sci.* **2021**, *12*, 1759. <https://doi.org/10.3390/app12031759>

Academic Editor: Luis Javier Garcia Villalba

Received: 30 December 2021

Accepted: 6 February 2022

Published: 8 February 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Cyber attacks can impact the performance of networks (corporate or otherwise), allow access to and the modification of confidential data, and compromise the security of virtually any infrastructure belonging to the network itself. This issue can have a significant economic impact on a smart enterprise. Zero-day attacks are unknown, never-before-seen attacks based on undisclosed and unpublished flaws and the widespread knowledge of “you cannot protect against what is unknown” [1]. Therefore, it is of fundamental importance to create algorithms that are able to predict 0-day attacks in an automatic and fast way (near real-time). Time is a crucial parameter concerning such attacks: in a system with a large volume of traffic, if it cannot be analyzed in a reasonable time to avoid traffic accumulation, and thus a bottleneck is generated. In a near real-time anomaly detection system, it is necessary to consider the processing speed of the algorithm and the pattern variations typical of attacks caused by the different network traffic that can be monitored in a real context. In particular, network traffic is characterized by behaviors that vary based on holidays, business hours, or times of the year characterized by higher or lower network activity [2]. Therefore, the system must be able to respond to changes in the monitored network traffic dynamically.

This work aims to study and design an anomaly detection system through the combination of machine learning algorithms capable of analyzing near real-time network traffic.

The research focuses on classical Machine Learning algorithms and an innovative combination of them and how they can be used in a near real-time context, where it is necessary to identify anomalous network traffic promptly.

The construction of a complete network traffic dataset is challenging as it is difficult to enumerate all possible cases of “normality”, and the boundary between normality and anomaly is often not well defined. This happens because creating a labeled dataset that best reflects a real application context is very expensive. Moreover, in particular situations, due to the frequent change of the characteristics of a process or the continuous emergence of different behaviors, it is impossible to determine if the information is still valid.

Focusing also on unknown attacks, unsupervised approaches must be considered— anomaly detection represents the primary strategy available.

Applying a supervised algorithm can limit the system’s effectiveness as it must consider the variability of the networks, their different types, and the number of hosts that assume unknown behavior. For this reason, in this work, it was chosen to use an unsupervised algorithm, since this allows us to isolate anomalous behavior and also identify unknown attacks [3]. These models implicitly assume that normal data are present in a greater quantity than abnormal samples, so the algorithm can classify normal behavior by clustering similar instances together and highlighting as abnormal a sample that deviates from the cluster representing normality.

The state of the art of unsupervised Anomaly Detection algorithms shows that they achieve lower performance than algorithms that use a supervised approach. Some algorithms, such as the Extended Isolation Forest, reach performances comparable with supervised algorithms, but the time spent in the prediction phase is not suitable for near real-time applications. In order to improve the performance of unsupervised Anomaly Detection algorithms, an approach combining Deep Learning and Shallow Learning algorithms is proposed in this paper. The idea behind the proposed work is to maximize the prediction speed using Deep Learning techniques and at the same time maximize the accuracy index using robust Shallow Learning algorithms. In a near real-time Anomaly Detection system, it is necessary to minimize the prediction time of the anomaly score in order to allow the experts in the field to detect and respond to cyberattacks promptly. Assuming that a large part of the monitored traffic is due to normal network activities, in a first step, a Deep Learning algorithm is used to discriminate normal traffic from suspicious traffic quickly. In the next step, only the suspicious traffic is further analyzed with an Unsupervised Learning algorithm to accurately determine whether it is attributable to abnormal activity or not. In this paper, three approaches are presented, which differ from each other in the Deep Learning algorithm used in the first phase. Three types of algorithms based on the autoencoder architecture have been identified:

- Deep Autoencoder Gaussian Mixture Model;
- Deep Autoencoder;
- Memory-Augmented Deep Autoencoder.

Connections considered normal by the Deep Learning algorithm do not need further analysis; in contrast, for traffic considered suspicious in this approach, the Extended Isolation Forest algorithm is used, which is considered a precise and stable algorithm to be used in an unsupervised Anomaly Detection system. The approach proposed in this paper aims to minimize the prediction time employed by a near real-time Anomaly Detection system adopting an unsupervised approach and, at the same time, maximize the accuracy of an unsupervised Anomaly Detection system.

The main contributions of this work are as follows:

- Proposal of three new algorithms merging unsupervised deep learning with shallow learning using Extended Isolation Forest;
- Analysis of results in terms of precision, recall, F1, AUC ROC, and accuracy;
- Evaluation of training and execution times as expected in the near real-time context;

- Finally, using the explainable artificial intelligence technique called SHAP, it was shown that the features synthesized by the joint use of MemAE and EIF increase the robustness of the results and improve their performance.

In Section 2 of this paper, the state of the art related to Machine Learning and Deep Learning algorithms used to detect anomalous behaviors in a monitored network is described. Subsequently, in Section 3, Deep Autoencoding with GMM and Isolation Forest, Deep Autoencoder with Isolation Forest, and Memory-Augmented Deep Autoencoder with Isolation Forest are proposed. Section 4 describes the setup of the experiments, the results, and reasoning, including also an explainable artificial intelligence in order to understand which are the features that the models identify and discriminate to find a 0-day attack. Section 5 contains conclusions of this work.

2. Related Work

Anomaly Detection systems using a supervised approach achieve high performance, which translates into a very low false alarm rate and a very high detection rate of cyberattacks. Some of the most classical classification algorithms used for this purpose are Random Forest [4], Support Vector Machine [5], and feed-forward neural networks [6]. However, these approaches require a priori knowledge of the attacks to be detected.

In [7], an algorithm called Ensemble Consensus is proposed, which combines three of the main unsupervised algorithms to detect anomalies present in network traffic. The algorithms that make up Ensemble Consensus are Isolation Forest [8], Local Outlier Factor [9], and Elliptic Envelope [10]. The Ensemble Consensus algorithm is tested using KDDCUP99 as the reference dataset. The reported results show that the Ensemble Consensus algorithm achieves comparable performance with supervised algorithms such as Random Forest or Support Vector Machine.

Tien et al. [11] compare stacked-autoencoders with various unsupervised learning algorithms such as one-class SVM, Isolation forest, and others for dimensionality reduction and classification by achieving high F1-score values compared to supervised methods.

A comparison of various deep learning algorithms such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNNs) such as Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) network is instead proposed by Ahmad et al. [12] and used to find zero-day anomalies within an IoT network with a False Acceptance Rate (FAR) ranging from 0.23% to 7.98%.

Jamil et al. [13] use various unsupervised learning algorithms and compare them with a model built through the Kalman filter. Such a model achieved unexpected results with values above 97.02% accuracy.

An updated and detailed review on anomaly detection techniques to detect intrusions and attacks is contained in [14]. Another review of no less importance and clarifying the state of the art, the domain of interest, and the types of attacks is offered in [15].

Hariri et al. [16] propose the Extended Isolation Forest algorithm, which aims to solve some issues identified in the Isolation Forest algorithm. The Isolation Forest algorithm is one of the most robust and widely used Anomaly Detection algorithms and achieves good performance when used to detect attacks contained in the KDDCUP99 dataset. Isolation Forest relies on randomness in the selection of features and values. Because outliers are “few and far between”, they stand out quickly from these random selections. Branch cuts are always horizontal or vertical, introducing distortion and artifacts into the anomaly score map. There is no fundamental aspect of the algorithm that requires this restriction to select a branch cut with a random “slope” at each branch point. Like the standard Isolation Forest algorithm, anomalies can be isolated quickly, whereas regular points require the isolation of many branch cuts.

More recent Anomaly Detection systems use Deep Learning techniques to detect anomalies. Some of the most promising work uses autoencoder-type neural networks to learn the normal behavior of network traffic. Some of the algorithms researched for this purpose are given below.

The idea behind the work presented in [17] concerns the use of a deep autoencoder-type neural network to learn the data describing the regular network traffic to be monitored. Through the constructed model, such Anomaly Detection systems can analyze network traffic and report traffic that deviates from the data learned in the training phase as anomalous.

Zong et al. [18] propose an unsupervised algorithm called Deep Autoencoding Gaussian Mixture Model, which uses in combination a Deep Autoencoder neural network with Gaussian Mixture Model (GMM). In [18], it is used on the KDDCUP99 and NSL-KDD datasets, and for both datasets used, the DAGMM algorithm performed numerically very well.

Another unsupervised approach is the Deep Structured Energy Based Models (DSEBM) algorithm [19], presented by Zhai et al., in whose proposed work Energy Based Model EBMs are extended with deep structured EBMs called DSEBMs, in which an energy function is encoded through a deep neural network. The DSEBM algorithm generalizes classical EBMs by building a new architecture applicable to a broader spectrum of applications, such as anomaly detection.

Gong et al. created the Memory-Augmented Deep Autoencoder [20] algorithm, which combines a deep autoencoder neural network with a memory module to mitigate a problem encountered using a simple deep autoencoder network. The authors state that due to the high generalization ability of the deep autoencoder network, sometimes even outliers are reconstructed correctly in the decoding phase, causing a low reconstruction error failing to detect outliers. In [20], the Memory-Augmented Deep Autoencoder algorithm is used to detect cyber attacks present in the KDDCUP99 dataset. This paper mainly focuses on an unsupervised approach; however, some interesting techniques use semi-supervised approaches. Abdel-Basset et al. [21] propose a semi-supervised architecture oriented to intrusion detection in the IoT context, named SS-Deep-ID. The experiments carried out show that SS-Deep-ID obtains better precision, recall, F1-score, and accuracy results than the other supervised and semi-supervised algorithms analyzed. In [22], Cheng et al. propose, always within the IoT and Intrusion Detection domain, the first semi-supervised algorithm; i.e., semisupervised hierarchical stacking temporal convolutional network (HS-TCN). The primary focus of this work is to perform the labeling of a large amount of data with a small amount of labeled data. Other interesting approaches are those provided by transfer learning. Still in the area of Intrusion Detection aimed at IoT, Vu et al. [23] propose the MMD-AE technique, which uses two autoencoder-type networks and the maximum mean discrepancy metric, aiming at minimizing the distance between multiple hidden layers between the two networks. Gao et al. [24], on the other hand, propose an approach to the problem based on transforming network data into grayscale images, to which transfer learning is added to increase its adaptivity and overall efficiency. The results show that the approach performs better than the most common algorithms, also showing higher robustness and generalization capability.

3. Proposed Approaches

In the application context under consideration, in which the proposed system must detect the presence of abnormal traffic, it is necessary to maximize the accuracy index for the Anomaly Detection algorithm. Low precision involves many false positives, which represent the number of false alarms that a human operator must analyze. A high number of false positives can result in longer latencies for analyzing and responding to real cyberattacks in the monitored network. Therefore, it is necessary to use an algorithm that is oriented to maximize accuracy even if the recall index is penalized in some cases. To achieve these goals, three new algorithms are proposed in this section, which combines a basic algorithm (Deep Learning) with the Extended Isolation Forest. The basic algorithms identified are related to the class of autoencoders and are Deep Autoencoding Gaussian Mixture Model, Deep Autoencoder, and Memory-Augmented Deep Autoencoder, thus giving rise to the following algorithms:

- Deep Autoencoding Gaussian Mixture Model with Extended Isolation Forest (DAGMM-EIF);
- Deep Autoencoder with Extended Isolation Forest (DA-EIF);
- Memory-augmented Deep Autoencoder with Extended Isolation Forest (MemAE-EIF).

Stacked-autoencoders or deep autoencoders are multilevel neural networks built to create a subspace, called embedding, capable of capturing the significant pattern common to various instances. It can intuitively be compared to the vector subspace constructed by low-rank factorization by the singular value decomposition of a matrix. The main difference is that stacked-autoencoders are, due to the activation functions, nonlinear. Specifically, denoising stacked-autoencoders [25] introduce noise into the encoding, resulting in an encoding that is a corrupted version of the original input data. It is precisely the “corrupted” version of the data that is used to train the multilevel neural network, and thanks to the use of appropriate loss functions, the gradient descent optimization algorithm allows us to create a subspace (embedding) of lower dimensionality than the input that, however, captures the most important information to reconstruct the clean pre-image with the lowest possible error. The objective is to reproduce the original and undamaged version of the original information. By comparing the corrupted data to the original data, the network learns which data features are most important and which are unimportant/corrupted. In other words, for a model to denoise the damaged data, it must have extracted the important features of the data.

In addition to the Deep Autoencoder algorithm, two evolutions of this algorithm were therefore considered. The DAGMM algorithm uses two Deep Autoencoder neural networks to learn the network traffic referable to network activities considered normal. The intuition behind the DAGMM algorithm concerns the training phase, in which the algorithm jointly optimizes the parameters of the autoencoder network and the GMM model in an end-to-end manner, exploiting a separate estimation network to facilitate the learning of the mixture model parameters [18]. On the other hand, the MemAE algorithm introduces a memory module into the autoencoder architecture to solve a simple autoencoder-type neural network problem. In [20], the author states that in some cases, the model built through an autoencoder can correctly reconstruct even samples that deviate from the training set records, which could cause the Anomaly Detection system to fail. In the algorithm’s training phase, it is not necessary to have all the classes related to the detected attacks, but it is necessary to build a dataset containing only the network traffic considered normal. The training set is initially divided into two portions:

- The first part comprises 60% of the training set and is used exclusively to train the basic algorithm;
- The second part consists of 40% of the training set and is used to calculate the internal anomaly threshold and train the EIF algorithm.

For each sample analyzed, the basic algorithm produces three new features that represent the following:

- z_c : the compressed sample calculated in the first step of the compression network;
- z_{ed} : the Euclidean distance between the sample under examination X and the sample X' reconstructed by the compression network;
- z_{cs} : the cosine similarity metric calculated between the sample under examination X and the sample X' reconstructed by the compression network.

In this way, each sample X to be analyzed is described by the number of features contained in X , plus three features of the vector called Z , where $Z = [z_c, z_{ed}, z_{cs}]$. In the second phase of the training process, it is necessary to construct the forest of trees of the EIF algorithm, in which each sample of the training dataset is composed of the features of X plus those of Z . Figure 1 shows the architecture of the Machine Learning algorithm describing the training and recognition phases.

The proposed algorithm aims to increase the accuracy of the anomaly detection system by exploiting the speed of the basic algorithm and the accuracy of the Extended Isolation

Forest algorithm. Specifically, as can be seen from Figure 1, in the first step, the basic algorithm is used to predict the anomaly score on sample X. If the score evaluated for sample X is greater than an anomaly threshold defined in the training phase, the sample is considered suspicious and must be analyzed with the EIF algorithm. Otherwise, the prediction process ends, and the analyzed sample is considered normal. The samples considered suspicious are analyzed by the EIF algorithm, which calculates the final anomaly score. In particular, the EIF algorithm requires as input not only the features describing the sample X but also the features of Z extracted from the sample reconstructed by the DA algorithm.

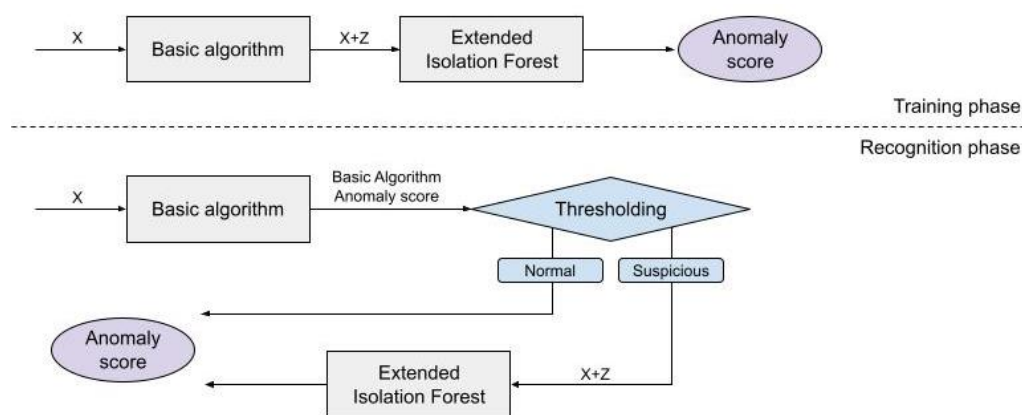


Figure 1. Description of the training phase and recognition phase of the proposed approaches.

In the proposed algorithm, only the samples considered suspicious are also analyzed by the EIF algorithm, so the prediction speed remains the same as in the basic algorithm in the presence of normal samples. The samples considered suspicious are subsequently analyzed by EIF, which can identify a part of the false positives and consequently increase the system’s accuracy for anomaly detection.

As explained in the article presenting the Memory Augmented Deep Autoencoder technique [20], the model uses the embedding autoencoder to create a query vector that can retrieve the most similar elements contained within the memory and perform the reconstruction. During the training phase of the MemAE-EIF algorithm, the normal data model is used to update the memory continuously; in the last phase, the memory is fixed, and the query and thus reconstruction is performed only on a few similar elements contained in the fixed memory.

4. Experiment

All experiments were performed on a machine with an Intel core i7-8665U processor, 16 GB of RAM, and the Windows 10 Pro operating system. Section 4.1 describes the datasets used to evaluate the algorithms examined in this paper. Section 4.2 contains all the information related to the configuration of the algorithms used for the experiment. Section 4.3 reports the results obtained from the experiments performed for the proposed work. Section 4.4 analyzes the importance of the features involved using the SHAP library. Finally, the results are discussed overall in Section 4.5.

4.1. Datasets

One of the main challenges in anomaly-based intrusion detection is the lack of appropriate datasets for evaluating and comparing intrusion detection systems. This gap is caused by the nature of the data, as the inspection of network traffic may expose sensitive information and confidential or personal communications. In addition, publicly sharing raw internet traffic may violate privacy laws. To solve this problem, network traffic is captured in a simulated context. The advantage of using simulation as a network traffic

generator is that it is free of privacy or sensitivity issues, but since real traffic is very complex, simulating real traffic is a challenge.

Evaluating the effectiveness of the network traffic anomaly detection system requires datasets that contain data on network traffic monitored over a given time interval in a context in which activities related to regular network use and activities related to cyberattacks that generate anomalous traffic follow one another.

The Canadian Institute of Cybersecurity (CIC) provides several network traffic datasets used by researchers and companies to evaluate their security systems. The primary datasets identified to evaluate the effectiveness of the proposed Anomaly Detection algorithms are as follows:

- **KDDCUP99:** The KDD Cup 1999 [26] dataset is one of the most widely used datasets in system evaluation for network security. Since the proportion of data belonging to the class identifying attacks is much larger than the proportion of data belonging to the class defining normal traffic, the class relating to attacks is subsampled to a proportion of 20%. A detailed description of the KDDCUP99 dataset has been published in [27]. The KDDCUP99 dataset consists of approximately 4,900,000 single connection vectors, each of which contains 41 features. Each sample in the dataset is labeled with either the class related to normal traffic or the classes identifying the simulated attack;
- **NSL-KDD:** The NSL-KDD dataset [28,29] is an improvement of its predecessor KDDCUP99, in which the issues defined in [27] are resolved. It is distributed as a set of attribute relations (ARFF) and has 41 features per record. The features of the NSL-KDD dataset are the same as the features of the KDDCUP99 dataset.
The NSL-KDD dataset has the following advantages over the original KDDCUP99 dataset:
 - It does not include redundant records in the training dataset, so the classifiers will not be skewed towards more frequent records;
 - No duplicate records are present in the test datasets;
 - The number of records selected from each set of difficulty levels is inversely proportional to the percentage of records in the original KDDCUP99 dataset. As a result, the classification rates of different machine learning methods vary over a broader range, which makes it more efficient to have an accurate evaluation of different learning techniques;
 - The number of records in the training and testing dataset is reasonable, making running the experiments on the entire set convenient without randomly selecting a small portion. As a result, the evaluation results of the different research works will be consistent and comparable.
- **CIC-IDS2017:** The CIC-IDS2017 [30,31] dataset contains network traffic recorded in a controlled network environment for five days and includes traffic related to normal activities performed by network users and traffic generated by cyberattacks such as BruteForce, DoS, and Botnet. The dataset is provided as a set of PCAP files, which can be played on a computer network interface. The dataset also includes a schedule for attacks and a spreadsheet that contains more than 80 features extracted from network traffic.

4.2. Experimental Setup

The unsupervised algorithms examined in this work are as follows:

- Extended Isolation Forest (EIF)
- Ensemble Consensus (EC)
- Deep Autoencoding Gaussian Mixture Model (DAGMM)
- Deep Autoencoding Gaussian - Extended Isolation Forest (DAGMM-EIF)
- Deep Autoencoder (DA)
- Deep Autoencoder - Extended Isolation Forest (DA-EIF)
- Deep Structured Energy Based Models (DSEBM)
- Memory-Augmented Deep Autoencoder (MemAE)

- Memory-Augmented Deep Autoencoder—Extended Isolation Forest (MemAE-EIF)
The metrics examined to assess them are as follows:

- Precision
- Recall
- F1-Score
- Area Under the Receiver Operator Characteristic Curve (AUROC)
- Accuracy

The Recall, Precision, F1-Score, and Accuracy indices reported in the following tables refer to macro measures; i.e., the weighted average over the number of the instances for each class. In the present experiment, the test dataset samples can be labeled with anomalous and normal classes. In unsupervised algorithm testing, the Anomaly Detection system can distinguish the analyzed traffic into two classes, namely the class related to normal traffic, represented by the label “0”, and the class indicating any attack, represented by the label “1”. The reported results were computed through the K-Fold cross-validation [32] technique, in which the value of K was set equal to 10. For the algorithms EIF, DAGMM, DA, DSEBM, MemAE, DAGMM-EIF, MemAE-EIF, and DA-EIF, since they return an anomaly score for each sample analyzed, it is necessary to define a threshold that allows the discrimination of samples considered normal from samples considered anomalous. For the experiments reported in this section, since the percentage of anomalies in the test dataset is known, the anomaly threshold is configured with the value that allows isolating the percentage of higher scores.

Only the numerical features were used to analyze the datasets.

The following are the configurations used to use each algorithm as an anomaly detector:

- **Ensemble Consensus:** The configuration reported in [7] is used, with the “*soft consensus*” mode and a threshold of -0.2 .
- **Deep Autoencoding Gaussian Mixture Model:** The neural network of the DAGMM algorithm used the hyperbolic tangent activation function and was trained through the Adam optimization algorithm [33] with a learning rate of 0.0001. The number of epochs used for the training process is 180. The neural network used by the DAGMM algorithm is described below:
 - Compression network: The compression network has an input layer that contains a dataset record’s features. The input layer thus consists of 38 nodes for the KDDCUP99 and NSL-KDD datasets and 78 nodes for the CIC-IDS2017 dataset. There are then 7 hidden layers, consisting of 60, 30, 10, 1 (which represents the compression carried out on the record given as input), 10, 30, and 60 nodes. The output layer contains the reconstructed features of the input record, so it is composed of 38 nodes for the KDDCUP99 and NSL-KDD datasets; it is instead composed of 78 nodes for the CIC-IDS2017 dataset.
 - Estimation network: The output layer consists of a number of nodes equal to the number of Gaussians used by the GMM algorithm. For the NSL-KDD dataset, the output layer consists of five nodes; instead, for the KDDCUP99 and CIC-IDS2017 datasets, the number of nodes is four.
- **Extended Isolation Forest:** For the KDDCUP99 and NSL-KDD datasets, a forest of 200 trees with parameter `ndim` equal to 38 is used. Parameter `ndim` represents the number of columns to be combined to produce a subdivision within the trees. In the case of the CIC-IDS2017 dataset, a forest of 200 trees with parameter `ndim` equal to 10 is used. In addition, for the CIC-IDS2017 dataset, parameters `prob_pick_pooled_gain` and `ntry` were configured, respectively, equal to 1 and 10. Parameter `prob_pick_pooled_gain` denotes the probability of choosing the threshold on which to split a linear combination of variables as the threshold that maximizes an aggregate standard deviation gain criterion on the linear combination [34]. Parameter `ntry` indicates how many linear combinations to try to determine the best threshold.

- **Deep Autoencoding Gaussian Mixture Model with Extended Isolation Forest:** The same configurations were used to configure the DAGMM-EIF algorithm as were used to evaluate the algorithms individually.
- **Deep Autoencoder:** The neural network used by the Deep Autoencoder algorithm was trained through the Adam [33] optimization algorithm with a learning rate of 0.0001. The number of epochs used in the training phase was 30 for the KDDCUP99 and NSL-KDD datasets and 20 for the CIC-IDS2017 dataset. The neural network uses the Rectified Linear Unit activation function. For the KDDCUP99 and NSL-KDD datasets, the network consists of the following layers: an input layer made up of 38 nodes, which contain the features of a record of the dataset; 3 hidden layers made up of 26, 12 (which represent the compression carried out on the input record), and 26 nodes; and an output layer made up of 38 nodes, which contain the reconstructed features of the input record. For the CIC-IDS2017 dataset, the network has an input layer and an output layer of 78 nodes (corresponding to the dataset features and reconstructed features) and 5 hidden layers made up of 50, 25, 10 (which represent the compression carried out on the input record), 25, and 50 nodes.
- **Deep Autoencoder with Extended Isolation Forest:** The same configurations were used to configure the DA-EIF algorithm as were used to evaluate the algorithms individually.
- **Deep Structured Energy Based Models for Anomaly Detection:** The configuration reported in [19] for the KDDCUP99 dataset was used to configure the DSEBM algorithm. The neural network of the DSEBM algorithm was trained through the Adam optimization algorithm [33] with a learning rate of 0.0001. The number of epochs used for the training process was 20.
- **Memory-Augmented Deep Autoencoder:** The configuration reported in [20] for the KDDCUP99 dataset was used to configure the MemAE algorithm. For the memory module, the parameter N was configured equal to 50. The neural network of the MemAE algorithm was trained through the Adam optimization algorithm [33] using a learning rate of 0.00001 for the KDDCUP99 and NSL-KDD datasets and of 0.0001 for the CIC-IDS2017 dataset. The epochs used in training phase for KDDCUP99 and CIC-IDS2017 numbered 30, while 20 were used for NSL-KDD. The neural network uses the function of hyperbolic tangent activation, and the following layers constitute it: an input layer and output layer with number of nodes equal to the number of features in a single dataset record (38 for KDDCUP99 and NSL-KDD, 78 for CIC-IDS2017), in which the input layer receives the sample and the output layer represents the reconstructed sample; and 9 hidden layers constituted respectively by 120, 60, 30, 10, 3 (which represent the compression made on the record given in input), 10, 30, 60, and 120 nodes.
- **Memory-Augmented Deep Autoencoder with Extended Isolation Forest:** The same configurations were used to configure the MemAE-EIF algorithm as were used to evaluate the algorithms individually.

4.3. Results

Below are tables containing the results obtained from the analyzed algorithms for each dataset used for testing (Table 1).

Table 1. Results obtained from testing unsupervised algorithms on the KDDCUP99 dataset.

Algorithm	Precision	Recall	F1-Score	σ (F1-Score)	AUROC	σ (AUROC)	Accuracy
EC	0.8221	0.8102	0.8162	0.0026	0.8111	0.0026	0.8413
EIF	0.9177	0.9177	0.9177	0.0064	0.9177	0.0064	0.9478
DAGMM	0.9043	0.9056	0.9050	0.0051	0.9057	0.0051	0.9749
DAGMM-EIF	0.9157	0.9158	0.9157	0.0079	0.9158	0.0079	0.9410
DA	0.8657	0.8657	0.8657	0.0025	0.8657	0.0025	0.9150
DA-EIF	0.8914	0.8666	0.8770	0.0020	0.8666	0.0020	0.9234
DSEBM	0.8617	0.8617	0.8617	0.0021	0.8617	0.0021	0.9125
MemAE	0.9084	0.9084	0.9084	0.0021	0.9084	0.0021	0.9420
MemAE-EIF	0.9234	0.9219	0.9226	0.0015	0.9219	0.0015	0.9510

Table 2 shows the prediction times measured in milliseconds for each algorithm used to analyze the KDDCUP99 dataset.

Table 2. Prediction times for analyzing 1000 records of the KDDCUP99 dataset. Times shown in the table are measured in milliseconds.

Algorithm	Prediction Time
EC	1187.610
EIF	274.780
DAGMM	2.810
DAGMM-EIF	211.670
DA	10.145
DA-EIF	451.557
DSEBM	13.620
MemAE	12.481
MemAE-EIF	527.001

Table 3 shows the results obtained from the tests performed on the NSL-KDD dataset.

Table 3. Results obtained from testing unsupervised algorithms on the NSL-KDD dataset

Algorithm	Precision	Recall	F1-Score	σ (F1-Score)	AUROC	σ (AUROC)	Accuracy
EC	0.8635	0.8588	0.8602	0.1326	0.8588	0.1326	0.8617
EIF	0.9120	0.9120	0.9120	0.1530	0.9120	0.1530	0.9124
DAGMM	0.8845	0.8845	0.8845	0.0520	0.8845	0.0511	0.8851
DAGMM-EIF	0.9045	0.9045	0.9045	0.1307	0.9045	0.1307	0.9050
DA	0.8905	0.8905	0.8905	0.1770	0.8905	0.1770	0.8910
DA-EIF	0.9113	0.9113	0.9113	0.1423	0.9114	0.1392	0.9118
DSEBM	0.8698	0.8698	0.8698	0.1781	0.8698	0.1781	0.8704
MemAE	0.9118	0.9118	0.9118	0.1684	0.9118	0.1684	0.9122
MemAE-EIF	0.9172	0.9172	0.9172	0.1275	0.9172	0.1271	0.9175

Table 4 shows the prediction times measured in milliseconds for each algorithm used to analyze the NSL-KDD dataset.

Table 4. Prediction times for analyzing 1000 records of the NSL-KDD dataset. The times shown in the table are measured in milliseconds.

Algorithm	Prediction Time
EC	1198.630
EIF	246.960
DAGMM	1.830
DAGMM-EIF	141.140
DA	11.013
DA-EIF	401.161
DSEBM	14.010
MemAE	12.710
MemAE-EIF	476.712

Table 5 shows the results obtained from the tests performed on the CIC-IDS2017 dataset.

Table 5. Results obtained from testing unsupervised algorithms on the CIC-IDS2017 dataset.

Algorithm	Precision	Recall	F1-Score	σ (F1-Score)	AUROC	σ (AUROC)	Accuracy
EC	0.6264	0.5961	0.6053	0.0381	0.5961	0.0318	0.7804
EIF	0.7063	0.7063	0.7063	0.0019	0.7063	0.0019	0.8142
DAGMM	0.6934	0.6934	0.6934	0.0078	0.6934	0.0078	0.8060
DAGMM-EIF	0.7083	0.7040	0.7056	0.1198	0.7040	0.1200	0.8182
DA	0.6774	0.6774	0.6774	0.0020	0.6774	0.0020	0.7990
DA-EIF	0.7051	0.6912	0.6975	0.0046	0.6912	0.0022	0.8148
DSEBM	0.6800	0.6800	0.6800	0.1879	0.6800	0.1879	0.7975
MemAE	0.7146	0.7146	0.7146	0.0015	0.7146	0.0015	0.8194
MemAE-EIF	0.7431	0.6972	0.7146	0.0001	0.6972	0.0056	0.8350

Table 6 shows the prediction times measured in milliseconds for each algorithm used to analyze the CIC-IDS2017 dataset.

Table 6. Prediction times to analyze 1000 records from the CIC-IDS2017 dataset. Times shown in the table are measured in milliseconds.

Algorithm	Prediction Time
EC	12630.730
EIF	967.554
DAGMM	2.260
DAGMM-EIF	454.896
DA	12.309
DA-EIF	509.840
DSEBM	12.090
MemAE	13.980
MemAE-EIF	603.199

The results obtained show that the MemAE-EIF algorithm achieves the best performance in terms of accuracy and F1-score for all the datasets examined. A high precision rate is equivalent to a low number of false positives, which are false alarms that experts in the field must handle. The new algorithms proposed in this work combine Deep Learning and Unsupervised Learning algorithms to estimate the anomaly score associated with each monitored connection in a network. In detail, Deep Learning algorithms are used to quickly identify suspicious samples, which are subsequently analyzed with the Extended Isolation

Forest algorithm. In addition to identifying connections that are likely to be anomalous, the algorithms of Deep Learning generate new features which, together with the original characteristics of the input, constitute the sample to provide an input to the algorithm Extended Isolation Forest. It is interesting to evaluate the contribution in the new features generated by the Deep Learning algorithms and how useful these features are to the Extended Isolation Forest algorithm in generating the anomaly score for the suspect samples. Considering the datasets used, it is now desired to highlight which features most influence the prediction process of the EIF model used by the MemAE-EIF algorithm. Through the following experiment, we aim to verify if the new features introduced by the MemAE-EIF algorithm affect the prediction process of the EIF algorithm and thus contribute to the increase in metrics observed in the experiments conducted.

4.4. Explainable Artificial Intelligence with SHAP

This section aims to analyze the importance of each feature in the anomaly score prediction process. To do this, the SHAP library [35–37] allows the estimation of the impact of each feature in the prediction process. SHAP values (Shapley Additive Explanations) represent an index that shows the contribution of a feature in driving the model output from the baseline value to the computed value.

The features that push the prediction towards values greater than the base value will have a SHAP value that is proportionally positive and vice versa in the opposite case.

Figure 2 shows a graph describing the SHAP values calculated for features in the KDDCUP99 dataset using the EIF model of the MemAE-EIF algorithm. Specifically, the y-axis presents, in descending order, the 15 features that most influence the predictive model, while on the x-axis are the SHAP values calculated for each feature of each sample analyzed. The dataset used to test the algorithm contains 20% anomalous samples, and the remaining 80% are samples describing regular traffic.

The MemAE-EIF algorithm, in addition to the KDDCUP99 dataset features, generates three new features, namely:

- Z_c : Features of the compressed representation of the input sample. The autoencoder network of the MemAE algorithm compresses the input into a representation consisting of three characteristics indicated by [Zc1, Zc2, Zc3];
- euclidean distance: Euclidean distance between the sample given as input and the sample reconstructed by the autoencoder network;
- cosine similarity: Cosine similarity between the sample given as input and the sample reconstructed by the autoencoder network.

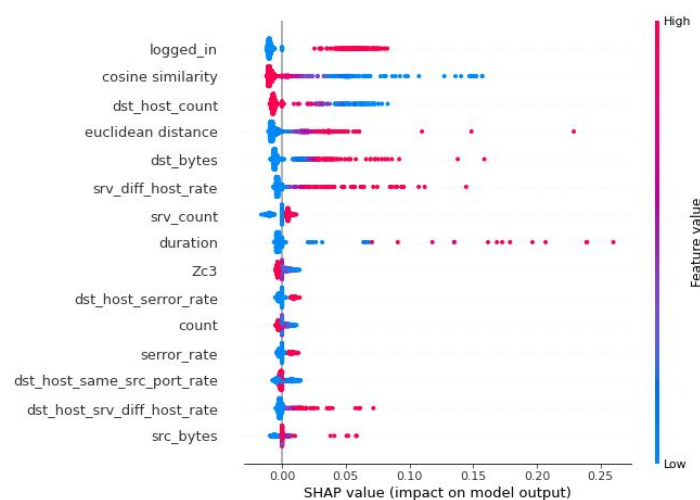


Figure 2. Summary plot of SHAP values calculated for the KDDCUP99 dataset using the EIF model of the MemAE-EIF algorithm.

From the graph in Figure 2, it can be observed that the features that most influence the predictive process of the EIF algorithm are the features produced in the first stage of the MemAE algorithm. In particular, the second most influential feature is the cosine similarity measure calculated between the given input sample and the sample reconstructed by the autoencoder network. The fourth most influential feature is the Euclidean distance between the input sample and the sample reconstructed by the autoencoder network. Moreover, from the same figure, it is possible to see that some of the features related to the compressed representation Zc also rank among the 15 most influential features in the anomaly score prediction process. Notably, the third feature of the compressed sample ranks ninth. The graph provides information on how the values of a feature affect the analyzed Machine Learning model. In particular, it can be seen that low values for the feature cosine similarity (indicated with the color blue) imply a higher SHAP value for the analyzed feature. High values of the feature cosine similarity imply a lower influence of the analyzed feature using the EIF model. The inferred information is consistent with the definition of cosine similarity, which states that if the cosine similarity metric tends to zero, it implies that the compared samples are very different, while a metric with a value tending to 1 implies that the given input sample is very similar to the sample reconstructed by the autoencoder network. Thus, in the presence of anomalies, the value of the cosine similarity metric is very low, implying a high SHAP value. Similarly, it can be observed that high values of the characteristic euclidean distance (indicated with the color magenta) imply a high SHAP value. In contrast, low values of the feature euclidean distance imply a lower influence of the analyzed feature for the EIF model. In this case, the deduced information is also consistent with the definition of Euclidean distance; if the calculated Euclidean distance between a sample given as input and the reconstructed sample of the autoencoder network has a high value, it implies that the autoencoder network was not able to reconstruct the sample given as input. If the autoencoder network cannot reconstruct a sample, the analyzed data are probably very different from the data in the training dataset, so it can be considered an anomalous sample. A bar graph is shown in Figure 3 that clearly shows the influence that each feature has on the prediction model. In particular, on the abscissa axis are reported, in ascending order, the 15 most influential features, while on the ordinate axis is reported the average value of SHAP for each feature of the analyzed dataset.

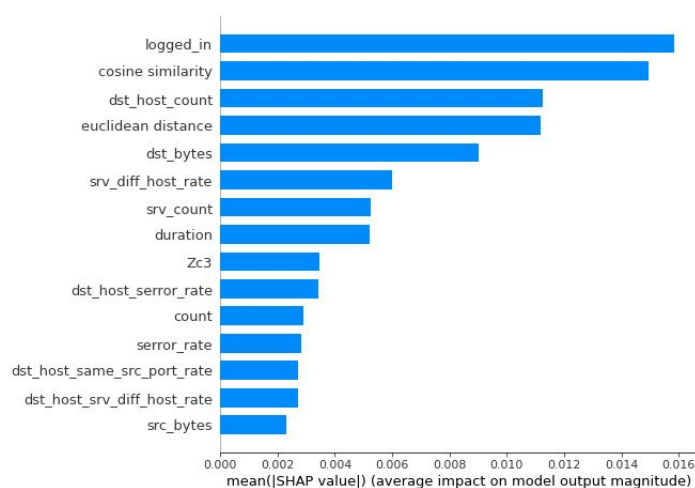


Figure 3. Bar graph of SHAP values calculated for the KDDCUP99 dataset using the EIF model of the MemAE-EIF algorithm.

From Figure 3, it is possible to observe that among the first 15 features with a higher average SHAP value are the features euclidean distance, cosine similarity and some features related to the compressed sample Zc. Therefore, it is possible to state that the new

features calculated by the MemAE-EIF model affect the prediction process of the anomaly score.

As seen for the KDDCUP99 dataset, it is possible to analyze the SHAP values calculated for the NSL-KDD dataset features using the MemAE-EIF algorithm. The dataset used to test the algorithm contains 46% anomalous samples, and the remaining 54% are samples describing normal traffic. For the NSL-KDD dataset, the records used for the training and prediction phases of the EIF model related to the MemAE-EIF algorithm consist of 43 features. The MemAE-EIF algorithm and the features in the NSL-KDD dataset generate new features for each sample. Figure 4 shows a graph describing the SHAP values computed for the features of the NSL-KDD dataset using the EIF model of the MemAE-EIF algorithm.

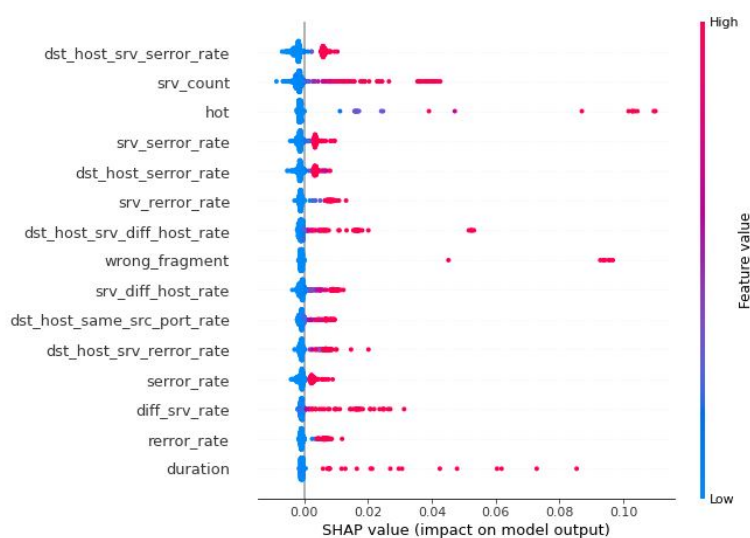


Figure 4. Summary plot of SHAP values calculated for the NSL-KDD dataset using the EIF model of the MemAE-EIF algorithm

From the graph in Figure 4, it can be observed that the features introduced in the MemAE-EIF algorithm do not rank among the top 15 most influential features. However, 3 of the 6 features introduced by the proposed algorithm rank among the top 25 most influential features: in detail, the features of the Zc1 Zc2 and Zc3 tablet representation rank at position 25, 30, and 19, respectively, whereas the cosine similarity measure and the Euclidean distance rank at position 31 and 24, respectively. From the SHAP analysis of the 43 features used to train the EIF model of the MemAE-EIF algorithm, it is observed that the introduced features affect the prediction process of the anomaly score. Figure 5 shows a bar graph displaying the influence of each feature in an analogous way to what was conducted for the KDDCUP99 dataset.

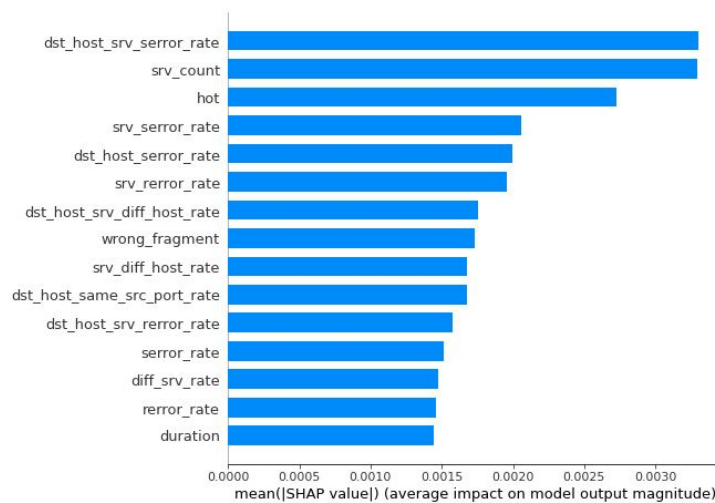


Figure 5. Bar graph of SHAP values calculated for the NSL-KDD dataset using the EIF model of the MemAE-EIF algorithm.

As performed for the other two datasets, it is possible to analyze the SHAP values calculated for the features in the CIC-IDS2017 dataset using the MemAE-EIF algorithm. The dataset used to test the algorithm contains 20% anomalous samples, and the remaining 80% are samples describing normal traffic. The records used for both the training and prediction phase of the EIF model related to the MemAE-EIF algorithm consist of 83 features. Figure 6 shows the graph describing the SHAP values calculated for the features in the CIC-IDS2017 dataset using the EIF algorithm of the MemAE-EIF model.

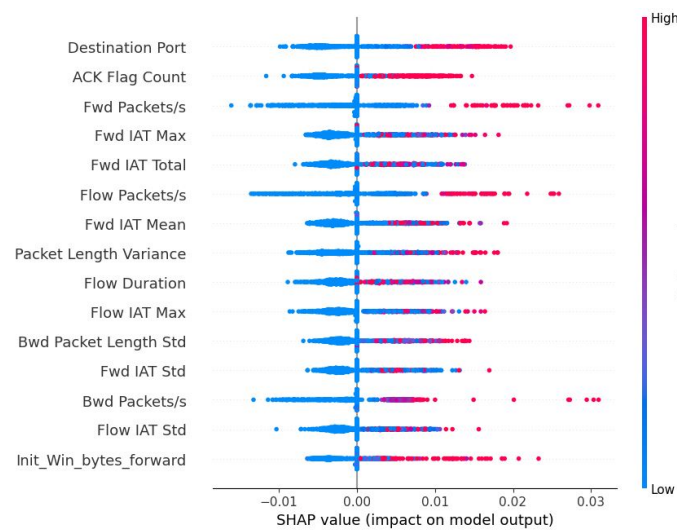


Figure 6. Summary plot of SHAP values calculated for the CIC-IDS2017 dataset using the EIF model of the MemAE-EIF algorithm.

Looking at Figure 6, it can be seen that the features introduced in the MemAE-EIF algorithm do not rank among the top 15 most influential features. However, four of the five features introduced by the proposed algorithm rank among the top half of the most influential features. Specifically, the compressed representation features Zc1 Zc2, and Zc3 rank at positions 32, 61, and 28, respectively, while the cosine similarity measure and Euclidean distance rank at positions 41 and 30, respectively. From the SHAP analysis of the 83 features used to train the EIF model of the MemAE-EIF algorithm, it was observed that the introduced features affect the prediction process of the anomaly score.

Figure 7 shows the bar graph for the CIC-IDS2017 dataset, already produced for the other two datasets.

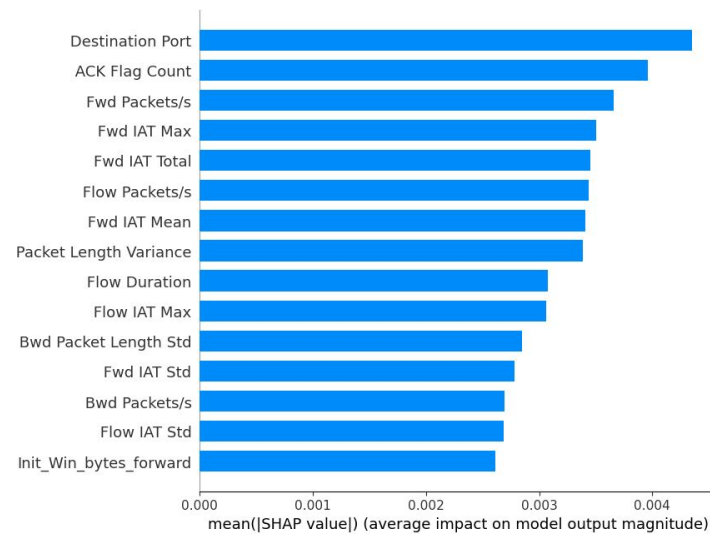


Figure 7. Bar graph of SHAP values calculated for the CIC-IDS2017 dataset using the EIF model of the MemAE-EIF algorithm.

From the analyses performed in the present section, it has been verified that the features of the sample compressed by the autoencoder network and the features related to Euclidean distance and cosine similarity represent influential features in the prediction process the MemAE-EIF algorithm. Therefore, it is possible to argue that the introduced features improve the prediction process of the EIF model of the MemAE-EIF algorithm. In conclusion, it is possible to state that, in the present section, it has been verified that for all the datasets examined, the features introduced by the MemAE-EIF algorithm are useful to the prediction process of the proposed Anomaly Detection algorithm, which achieves results that exceed the performance obtained using the EIF model without the features introduced in the same.

4.5. Discussion

From the results obtained in the experiments carried out in the present work, it is possible to observe that the proposed algorithms have reached performances comparable with the state-of-the-art Anomaly Detection algorithms. Particularly, for the dataset KDDCUP99, the best performances are obtained using the MemAE-EIF algorithm, which reaches an F1-score index equal to 0.9234 and an AUROC index equal to 0.9219. Immediately after the MemAE-EIF algorithm, the EIF and DAGMM-EIF algorithms are ranked with an F1-score value of 0.9177 and 0.9157 and an AUROC value of 0.9177 and 0.9158, respectively.

For the NSL-KDD dataset, the best performance is obtained using the MemAE-EIF algorithm, which achieves an F1-score index of 0.9172 and an AUROC index of 0.9172. It can be observed how the proposed algorithms achieve performances that improve the Deep Learning algorithm used as a base. In particular, the performance obtained using the DAGMM algorithm is lower than the performance obtained using the DAGMM-EIF algorithm for all the datasets examined. Similarly, the performance obtained using the DA algorithm is lower than the performance obtained using the algorithm in combination with the Extended Isolation Forest in each of the three datasets evaluated.

For the CIC-IDS2017 dataset, the performance obtained using the DAGMM-EIF and DA-EIF algorithm is higher than the performance obtained using the DAGMM and DA algorithms, respectively. However, the performance obtained using the MemAE-EIF algorithm is superior to that obtained using the MemAE algorithm only in some of the computed metrics. In particular, both algorithms achieve the same F1-score value, but the MemAE-EIF algorithm's accuracy index is higher than the accuracy index calculated for the basic MemAE algorithm.

For the CIC-IDS2017 dataset, the best performance is obtained using the MemAE-EIF and MemAE algorithms. In particular, both algorithms achieve the same result for the F-score index, but the MemAE-EIF algorithm allows for a sharp increase in precision at the expense of recall. In particular, for the dataset CIC-IDS2017, the MemAE and MemAE-EIF algorithms reach an F1-score index of 0.7146 and an AUROC index of 0.7146 and 0.6972, respectively. It is interesting to highlight how the precision index calculated for the MemAE-EIF algorithm reaches a score of 0.7431, while the basic MemAE algorithm obtains a precision score of 0.7146. From the obtained results, it is possible to prove that all the proposed algorithms improve the accuracy of the Deep Learning algorithms used as a base. Moreover, the MemAE-EIF algorithm achieves the best precision and F1-score for all the datasets examined. From the analysis carried out in Section 4.4, it is possible to observe that the internal characteristics introduced by the MemAE-EIF algorithm influence the prediction process of the EIF model, which allows a high precision to be reached in reporting anomalous events and therefore reduces the number of false alarms. This result is relevant in a real-world context in which false alarms are handled by experts in the field, taking their time away from analysis and response related to real cyber attacks.

5. Conclusions

The present work analyzes the state of the art of Anomaly Detection systems and how innovative Machine Learning algorithms can be used for this purpose. In particular, the study of Anomaly Detection systems has been guided by the application focus examined; i.e., identifying near real-time anomalous behaviors within a network. Therefore, such analysis has been focused on unsupervised Machine Learning algorithms, which allow the detection of anomalous network traffic, knowing only the traffic of normal network activities. In fact, unlike supervised approaches, it is unnecessary to have, and often not possible to create, a training dataset that contains all the cyber attacks one wants to detect for network anomaly detection systems.

Referring to the state of the art, three new Anomaly Detection algorithms have been developed in the present work, which combines Unsupervised Learning and Deep Learning techniques to detect in near real-time anomalous behaviors in a network. The excellent results obtained in the evaluation phase of these proposals affirm that the proposed Anomaly Detection algorithms can detect cyberattacks by analyzing the network traffic, improving the performance of their original components. From the results reported in Section 4.3, it can be seen that the proposed algorithms, when used to analyze the KDDCUP99, NSL-KDD and CIC-IDS2017 datasets, improve the performance of their respective base algorithms.

Moreover, the MemAE-EIF algorithm obtained the highest values for both the precision index and the F1-score ratio for all the examined datasets.

The Deep Learning algorithms used have a higher prediction speed than tree-based algorithms such as Extended Isolation Forest. On the other hand, algorithms such as Extended Isolation Forest achieve outstanding performance in anomaly detection.

In the proposed algorithms, the prediction time is equivalent to the time taken by the Deep Learning algorithm in the presence of normal samples, while when suspicious samples are detected, it is necessary to add the time taken to extract the new features and the time taken by the EIF algorithm to compute the anomaly score.

The algorithms employ an analysis time that allows the use of the Anomaly Detection models built to analyze in near real-time the network traffic, maximizing the performance in Anomaly Detection and thus obtaining an optimal compromise between temporal performance and the performance of the classification itself. The prediction time of the anomaly score used by each algorithm presented makes it possible to use the proposed Anomaly Detection models to analyze in near real-time the network traffic. The increase obtained in the classification scores permits, in a real-world context, the elimination of many alerts that would have occurred on large volumes of network traffic. Furthermore, assuming that in a real context, the traffic is mainly composed of connections that can be traced back to network activities considered normal, the algorithms proposed in the present

work can achieve a prediction time that tends towards the time taken by the Deep Learning algorithm to identify suspicious samples. However, this approach has some limitations: it is necessary to have GPU hardware to make the inference, which makes it difficult to apply directly on devices such as routers, but requiring analysis servers or dedicated hardware; it is also necessary to take into account strict memory constraints and high computation times for the Extended Isolation Forest algorithm, as it is not parallelizable. Future research directions are oriented to testing the system on other datasets, in particular in a cross-dataset way (i.e., train, for example, the thresholds with a dataset and test with a second dataset, obviously taking into account the constraints due to keeping the same features or a subset of these) in order to evaluate the generalization capability of the system.

Author Contributions: Methodology, software, writing—original draft preparation: F.C., A.I.; Conceptualization, validation: V.D., S.G., D.I., G.P.; Supervision, Project Administration: D.I.; Project Administration, Funding Acquisition: G.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Fondo Europeo di Sviluppo Regionale Puglia Programma Operativo Regionale (POR) Puglia 2014-2020-Axis I-Specific Objective 1a-Action 1.1 (Research and Development)-Project Title: CyberSecurity and Security Operation Center (SOC) Product Suite by BV TECH S.p.A., under Grant CUP/CIG B93G18000040007.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Abbreviations

AUROC	Area Under the Receiver Operation Characteristic Curve
CNN	Convolutional Neural Network
DA	Deep Autoencoder
DA-EIF	Deep Autoencoder with Extended Isolation Forest
DAGMM	Deep Autoencoding Gaussian Mixture Model
DAGMM-EIF	Deep Autoencoding Gaussian Mixture Model with Extended Isolation Forest
DSEBM	Deep Structured Energy Based Models
EBM	Energy Based Models
EC	Ensemble Consensus
EIF	Extended Isolation Forest
FAR	False Acceptance Rate
GMM	Gaussian Mixture Model
GRU	Gated Recurrent Unit
LSTM	Long Short Term Memory
MemAE	Memory Augmented Deep Autoencoder
MemAE-EIF	Memory Augmented Deep Autoencoder with Extended Isolation Forest
RNN	Recurrent Neural Network
SHAP	Shapley Additive Explanations
SVM	Support Vector Machine

References

1. Sun, X.; Dai, J.; Liu, P.; Singhal, A.; Yen, J. Using Bayesian networks for probabilistic identification of zero-day attack paths. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2506–2521. [\[CrossRef\]](#)
2. Zhao, S.; Ramos, J.; Tao, J.; Jiang, Z.; Li, S.; Wu, Z.; Pan, G.; Dey, A.K. Discovering different kinds of smartphone users through their application usage behaviors. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, Heidelberg, Germany, 12–16 September 2016; pp. 498–509.
3. Oliveira, N.; Praça, I.; Maia, E.; Sousa, O. Intelligent cyber attack detection and classification for network-based intrusion detection systems. *Appl. Sci.* **2021**, *11*, 1674. [\[CrossRef\]](#)
4. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [\[CrossRef\]](#)

5. Hu, W.; Liao, Y.; Vemuri, V.R. Robust anomaly detection using support vector machines. In Proceedings of the International Conference on Machine Learning, Washington, DC, USA, 21–24 August 2003; Citeseer: University Park, PA, USA, 2003; pp. 282–289.
6. Zhang, G.P. Neural networks for classification: A survey. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2000**, *30*, 451–462. [[CrossRef](#)]
7. Dentamaro, V.; Convertini, N.; Galantucci, S.; Giglio, P.; Impedovo, D.; Pirlo, G. Ensemble Consensus: An Unsupervised Algorithm for Anomaly Detection in Network Security Data. In Proceedings of the Itasec21, Virtual Event, 7–9 April 2021; pp. 309–318.
8. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422.
9. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 16–18 May 2000; pp. 93–104.
10. Rousseeuw, P.J.; Driessen, K.V. A fast algorithm for the minimum covariance determinant estimator. *Technometrics* **1999**, *41*, 212–223. [[CrossRef](#)]
11. 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](#)]
12. Ahmad, Z.; Shahid Khan, A.; Nisar, K.; Haider, I.; Hassan, R.; Haque, M.R.; Tarmizi, S.; Rodrigues, J.J. Anomaly Detection Using Deep Neural Network for IoT Architecture. *Appl. Sci.* **2021**, *11*, 7050. [[CrossRef](#)]
13. Jamil, F.; Kim, D. An Ensemble of a Prediction and Learning Mechanism for Improving Accuracy of Anomaly Detection in Network Intrusion Environments. *Sustainability* **2021**, *13*, 10057.
14. Alsoufi, M.A.; Razak, S.; Siraj, M.M.; Nafea, I.; Ghaleb, F.A.; Saeed, F.; Nasser, M. Anomaly-Based Intrusion Detection Systems in IoT Using Deep Learning: A Systematic Literature Review. *Appl. Sci.* **2021**, *11*, 8383. [[CrossRef](#)]
15. Riera, T.S.; Higuera, J.R.B.; Higuera, J.B.; Herraiz, J.J.M.; Montalvo, J.A.S. Prevention and Fighting against Web Attacks through Anomaly Detection Technology. A Systematic Review. *Sustainability* **2020**, *12*, 4945. [[CrossRef](#)]
16. Hariri, S.; Kind, M.C.; Brunner, R.J. Extended isolation forest. *IEEE Trans. Knowl. Data Eng.* **2019**, *33*, 1479–1489. [[CrossRef](#)]
17. Mirsky, Y.; Doitshman, T.; Elovici, Y.; Shabtai, A. Kitsune: An ensemble of autoencoders for online network intrusion detection. *arXiv* **2018**, arXiv:1802.09089.
18. Zong, B.; Song, Q.; Min, M.R.; Cheng, W.; Lumezanu, C.; Cho, D.; Chen, H. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
19. Zhai, S.; Cheng, Y.; Lu, W.; Zhang, Z. Deep structured energy based models for anomaly detection. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 19–24 June 2016; pp. 1100–1109.
20. Gong, D.; Liu, L.; Le, V.; Saha, B.; Mansour, M.R.; Venkatesh, S.; Hengel, A.v.d. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 1705–1714.
21. Abdel-Basset, M.; Hawash, H.; Chakraborty, R.K.; Ryan, M.J. Semi-supervised Spatio-Temporal Deep Learning for Intrusions Detection in IoT Networks. *IEEE Internet Things J.* **2021**, *8*, 12251–12265. [[CrossRef](#)]
22. Cheng, Y.; Xu, Y.; Zhong, H.; Liu, Y. Leveraging Semisupervised Hierarchical Stacking Temporal Convolutional Network for Anomaly Detection in IoT Communication. *IEEE Internet Things J.* **2020**, *8*, 144–155. [[CrossRef](#)]
23. 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](#)]
24. Gao, M.; Song, Y.; Xin, Y. Intrusion detection based on fusing deep neural networks and transfer learning. In *Digital TV and Wireless Multimedia Communication, Proceedings of the 16th International Forum, IFTC 2019, Shanghai, China, 19–20 September 2019*; Revised Selected Papers; Springer Nature: Berlin/Heidelberg, Germany, 2020; Volume 1181, p. 212.
25. Vincent Pascalvincent, P.; Larocheh, L.; Autoencoders, H.S.D. Learning Useful Representations in a Deep Network with a Local Denoising Criterion Pierre-Antoine Manzagol. *J. Mach. Learn. Res.* **2010**, *11*, 3371–3408.
26. KDDCUP99 Dataset. Available online: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html> (accessed on 13 December 2021).
27. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
28. Canadian Institute for Cybersecurity. NSL-KDD Dataset. Available online: <https://www.unb.ca/cic/datasets/nsl.html> (accessed on 13 December 2021).
29. Dhanabal, L.; Shantharajah, S. A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *Int. J. Adv. Res. Comput. Commun. Eng.* **2015**, *4*, 446–452.
30. Canadian Institute for Cybersecurity. CIC-IDS2017 Dataset. Available online: <https://www.unb.ca/cic/datasets/ids-2017.html> (accessed on 13 December 2021).
31. Panigrahi, R.; Borah, S. A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. *Int. J. Eng. Technol.* **2018**, *7*, 479–482.

32. Rodriguez, J.D.; Perez, A.; Lozano, J.A. Sensitivity analysis of k-fold cross validation in prediction error estimation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *32*, 569–575. [CrossRef]
33. Kingma, D.; Ba, J. A method for stochastic optimization. In Proceedings of the International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015.
34. Cortes, D. Revisiting randomized choices in isolation forests. *arXiv* **2021**, arXiv:2110.13402.
35. Štrumbelj, E.; Kononenko, I. Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst.* **2014**, *41*, 647–665. [CrossRef]
36. Lundberg, S.M.; Lee, S.I. A unified approach to interpreting model predictions. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 4768–4777.
37. Lundberg, S. SHAP library Documentation. Available online: <https://shap-lrjball.readthedocs.io/en/latest/index.html> (accessed on 13 December 2021).