IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 May 2023, accepted 31 May 2023, date of publication 7 June 2023, date of current version 14 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3283567

RESEARCH ARTICLE

Standard Latent Space Dimension for Network Intrusion Detection Systems Datasets

RICARDO FLORES MOYANO[1], (Member, IEEE), ALEJANDRO DUQUE[11], (Member, IEEE), DANIEL RIOFRÍO[11], (Member, IEEE), NOEL PÉREZ[1], (Senior Member, IEEE), DIEGO BENÍTEZ[1], (Senior Member, IEEE), MARIA BALDEON-CALISTO[11], (Member, IEEE), AND DAVID FERNÁNDEZ[2]

[1]Colegio de Ciencias e Ingenierías "El Politécnico," Universidad San Francisco de Quito (USFQ), Quito 170901, Ecuador
[2]Departamento de Ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid, 28040 Madrid, Spain

Corresponding author: Ricardo Flores Moyano (rflores@usfq.edu.ec)

This work was supported by the Universidad San Francisco de Quito (USFQ) in the Context of Poli-Grants (2021–2022) Program under Grant 17469.

ABSTRACT Machine learning is a branch of artificial intelligence that provides computers the ability to create or improve algorithms without being explicitly programmed by directly learning from data. It is widely used in automation or decision-making tasks in fields such as image or speech recognition, sentiment analysis, or self-driving cars. However, its application in the field of communication networks is limited by the lack of appropriate research resources, such as rich datasets for training or the definition of a standard set of features. In this context, a standard latent space dimension is proposed by performing an autoencoder-based dimensionality reduction process. Different network security datasets are projected onto a lower-dimensional space to determine a standard or convergent dimension. The convergent dimension is determined by identifying the threshold above which diminishing returns begin to occur in the autoencoder loss as the latent space dimension increases. The experimental validation showed that four machine learning classification models, trained with a standard latent space of ten dimensions, performed as well as the models that used the non-reduced versions of the datasets in terms of F1-score and accuracy. Furthermore, a Wilcoxon statistical test showed that the mean accuracy of all classification models trained with the standard latent space dimension had a difference of less than 0.0235 in comparison to the models trained with the original inputs. A negligible difference in accuracy is a significant outcome because researchers can use only the latent space to perform experiments with certainty that the performance of ML models will not be constrained.

INDEX TERMS Standardization, machine learning, autoencoder, latent space, network security.

I. INTRODUCTION

The massive adoption of the Internet and its convenient use as a connectivity medium has driven a significant evolution of communication networks. Consequently, networks have turned into heterogeneous, dynamic, and systematically complex systems. Thus, designing, deploying, managing, and maintaining networks based on traditional techniques is notoriously difficult to perform. As mentioned by Wang et al. [1], one of the most important advantages of machine learning (ML) is the capability to address complex problems that require classification, regression, and decision-making with results close to or even better than those obtained by human beings.

According to Mahadevkar et al. [2] and Samant et al. [3], ML techniques have gained a greater maturity in specific domains such as computer vision or natural language understanding. In others, such as communication networks, the application of ML approaches is still at an early stage. Revising the state of the art reveals the lack of publicly available and rich network traffic datasets for the network research community. In this regard, Barut et al. [4] mention that the lack of comprehensive open datasets not only restricts the evaluation of ML-based proposals to perform network

The associate editor coordinating the review of this manuscript and approving it for publication was Filbert Juwono.

traffic analytics by researchers but also the community is affected due to the inability to reproduce state-of-the-art results. Besides, it is worth mentioning that the available open datasets are mainly focused on security topics. Thus, for the application of ML to improve, for example, the provision of quality of service, it will require the creation of an ad-hoc network dataset.

On the other hand, the state of the art also reveals a lack of consensus regarding the set of features to be considered when creating the dataset. Regarding Network Intrusion Detection Systems (NIDSs), Sarhan et al. [5] mentions that network features are selected based on the author's domain knowledge and the available data collection tools. As a result, the available datasets are quite different in terms of feature sets; therefore, each dataset includes a part of the security events that could be identified. According to Holland et al. [6], the selection and adequate representation of network traffic features, which is related to feature exploration and engineering, determine the effectiveness of ML to a large extent. Thus, considering the most relevant features for the inference process will guarantee a proper model performance.

In this context, determining a standard set of features for network datasets represents a valuable contribution. On one hand, ML-based proposals could be tested in different network setups. On the other hand, testing an ML model in a different network setup would not require repeating the entire feature engineering process, model selection, and parameter tuning. Nonetheless, as mentioned earlier, the dataset's number of features or dimensions is defined based on domain knowledge. Since researchers and practitioners have different points of view, the resulting datasets are heterogeneous. Thus, reaching an agreement about the number and kind of features to be considered is not an easy task. While analyzing network datasets, it is important to note that some features are irrelevant and redundant for the inference process. Besides, datasets have some features in common. Hence, projecting the data to a lower-dimensional subspace that captures the relevant part of the data is an interesting alternative to explore towards providing a standard set of features. In this light, this work proposes identifying a standard low-dimensional or latent space to allow the network research community to perform ML-based analytics with the available open datasets.

In order to identify a standard latent space dimension, the dimensionality reduction technique is applied to five Intrusion Detection Systems (IDS) datasets. According to Li et al. [7], noisy and redundant information could be removed by obtaining a low-dimensional intrinsic space from an original high-dimensional space. In general, dimension reduction methods try to minimize information loss by preserving the original structure of the dataset. Different techniques have been proposed considering the linear or non-linear relationships present in the data. Maaten et al. [8] argue that traditional linear dimensionality reduction techniques such as principal component analysis (PCA) struggle to handle complex non-linear data. Since real-world data is non-linear in nature, non-linear techniques such as autoencoders might offer an advantage. In the presented work, autoencoders are used for the dimensionality reduction process. The common latent space is identified by analyzing the autoencoder loss as a function of the number of dimensions. The experiment consists of measuring the loss as the dataset dimension increases from 1 to the original size of the feature set. The objective is then to determine a threshold where diminishing returns start to occur as the dimension increases. Experiments carried out on five datasets reveal that dimensions greater than 10 do not provide significant improvements in the autoencoder loss. To validate this finding, four classification models corresponding to two neural networks and two extra trees classifiers were trained with the reduced versions of the datasets, using the standard latent space of 10 dimensions, and non-reduced datasets. For the analysis, the F1-score and accuracy evaluation metrics were used. The experiments were validated with an equivalence test. The results show that the reduced versions of the datasets produce results as good as the non-reduced versions with a statistical accuracy difference of less than 0.0235.

The remainder of the article is organized as follows. In section II, a background regarding autoencoders is provided, and an analysis of the related works is carried out. In section III, the proposal of a standard latent space dimension based on dimensionality reduction is described. To better understand the approach and validate the proposal, training and testing processes with four machine learning models are presented in section IV. Finally, the conclusions of the article and future works are reported.

II. BACKGROUND AND RELATED WORKS

This section introduces the need for mechanisms to reduce the dimensionality of datasets with particular attention to describing autoencoders. Then, the performed efforts regarding the definition of a standard representation of network datasets found in the state of the art are analyzed.

A. AUTOENCODER-BASED DIMENSIONALITY REDUCTION

According to Fournier and Aloise [9], working with a large number of dimensions in the feature space causes the volume of the data space to increase exponentially fast with the dimension and, therefore, the data becomes sparse. This problem, called *curse of dimensionality*, leads to the problem of overfitting as the machine learning model can easily get an accurate solution due to the data sparseness [10]. In order to tackle this problem, high-dimensional data can be projected into a lower-dimensional or latent space using different linear or non-linear techniques. In this light, Wang et al. [11] highlight the ability of a special three-layered neural network, designated as *autoencoder*, to perform dimensionality reduction on non-linear data. Compared with other techniques, autoencoders stand out for the capability of detecting repetitive structures. Besides, when dataset features present complex relationships and using only one autoencoder is not enough

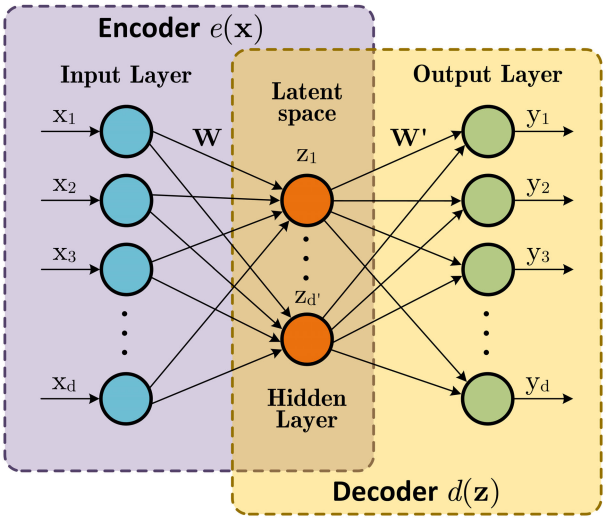

FIGURE 1. Reference architecture for autoencoders. Hidden layers establish a bottleneck. Thus, regardless of the number of input dimensions, the hidden layer provides a fixed dimension for the latent space.

to reduce the dimension, creating stacked autoencoders represent a feasible solution.

Ferreira et al. [12] define autoencoders as unsupervised neural networks that encode input data from $\mathbb{R}^d$ into $\mathbb{R}^{d'}$ and decode the resulting latent space of dimension d' to obtain the original input ($d' < d$). Fig. 1 shows the reference architecture of autoencoders. The encoder function e and the decoder function d of an autoencoder with one hidden layer are as follows:

$$e(\mathbf{x}) = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

$$d(\mathbf{z}) = \sigma'(\mathbf{W}'\mathbf{z} + \mathbf{b}') \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{d \times d'}$, $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{b} \in \mathbb{R}^{d'}$, $\mathbf{z} \in \mathbb{R}^{d'}$, σ and σ' are non-linear activation functions and $\mathbf{W}'$ as well as $\mathbf{b}'$ with adequate size. Parameters $\mathbf{W}$, $\mathbf{b}$, $\mathbf{W}'$ and $\mathbf{b}'$ are calculated by using *backpropagation* in such a way that the *reconstruction loss* is minimized as follows:

$$\min_{\mathbf{W},\mathbf{b},\mathbf{W}',\mathbf{b}'} \sum_i D(\mathbf{x_i}, d(e(\mathbf{x_i}))) \tag{3}$$

where D represents a dissimilarity measure such as mean squared error or cross-entropy. The data points in $\mathbb{R}^d$ are represented by $\mathbf{x}$.

The hidden layer performs the dimensionality reduction as the number of neurons is lower. The neurons of the input layer are fully connected to the neurons of the hidden layer by applying the learned weights (encoder function). Thus, the relevant information of the original dataset is preserved by projecting it into a lower-dimensional subspace. As mentioned by Motoda and Liu [13], the encoder (equation 1) performs a feature extraction process since a set of new features is extracted from the original features through a functional mapping (a non-linear combination of the original attributes).

A different approach for dimensionality reduction consist in applying a feature selection technique. A subset of M features is chosen from the original set of N features ($M < N$) based on a certain criterion. Thus, the feature space is optimally reduced. According to Zebari et al. [14], as the feature extraction technique transforms a considerable number of attributes into a set of reduced features, a lesser information loss and a higher discriminating power can be guaranteed in comparison with the feature selection technique.

Although dimensionality reduction is leveraged in this work towards the definition of a standard way of applying different machine learning models on different network datasets, it is worth mentioning that a reduced version of a dataset provides additional benefits. For instance, fewer computational resources are consumed as data processing complexity is reduced. Besides, the concentration of relevant information enhances not only the classification, clustering, regression, and visualization of data but also the stability and interpretability of the learning model.

B. TOWARDS A STANDARD REPRESENTATION OF NETWORK's DATA

Standardization of datasets is an important step to motivate academia and industry to massively adopt machine learning techniques in communication networks. Thus, different proposals have been presented in alignment with this standardization goal.

Sarhan et al. [5] propose considering the network traffic in terms of flows. A standard version of flows is proposed as the basic unit to create datasets. In order to build network flows, NetFlow is the preferred option as it is the de-facto industry standard. Thus, a NetFlow-based standard set of features for NIDS datasets is conceived. Although a standard representation of network flows is helpful, the main drawback is the manufacturer lock-in effect as Cisco develops NetFlow. Nonetheless, the authors in a different work [15] compared the NetFlow-based set of features and the set created with the CICFlowMeter tool across three network datasets. The best results in terms of accuracy were obtained with NetFlow. On the other hand, licensed tools such as nProbe are required to format the raw network traffic captures (pcap files) into NetFlow format, which might represent a constraint.

In contrast to considering network flows, Holland et al. [6] have developed a standard packet representation designated as nPrint. Raw network packets are transformed into an inherently normalized, binary representation while the underlying semantics of each packet is preserved. Authors demonstrate that a standard format of packets simplifies the integration of network traffic analysis tasks with state-of-the-art automated ML pipelines (AutoML). Although live packet capture makes it possible to identify the root cause of network problems, the main drawback of this approach is the large data storage capacity required. On the other hand, packet capture makes it difficult to identify long-term historical patterns and network trends, which is more feasible with flow-based approaches.

Regarding using packets or flows to produce feature representations of network traffic for machine learning functioning, Bronzino et al. [16] highlight the need to explore different representations to find the appropriate operating point of a given ML model. In this light, the authors propose a traffic refinery that performs passive traffic monitoring and in-network feature transformations to provide different network traffic feature representations. The tool allows defining a service class and the corresponding features to be collected. The system processes the flows that belong to a given service class and performs transformations such as aggregation or sampling to create the data representation of the class. According to the authors, the traffic refinery system includes default service classes and features, but user-defined traffic representations are also supported. Nonetheless, defining additional service classes may be challenging since functions in charge of updating the flow state metrics and aggregating the collected features must be coded. On the other hand, although a framework to define the set of features of interest contributes to encouraging the use of ML in the networking community, the standardization of network datasets is constrained as the researcher or practitioner criterion drives the dataset creation process.

NetML [4] represents an interesting initiative towards the provision of standard network datasets. To this end, the authors have compiled datasets for malware detection and application categorization. These datasets have been released as an open challenge that serves as common ground for experimenting and benchmarking ML-based approaches. NetML provides two curated datasets for malware detection. The first one is created using the raw traffic data from Stratosphere IPS. The second one is based on the raw traffic capture files of the CICIDS2017 dataset. The raw traffic capture files of the ISCXVPN2016 dataset are used to create a curated version for traffic classification. In order to extract the features from the raw captures, the Intel accelerated feature extraction library is used. The tool provides features regarding metadata, TLS, DNS, and HTTP. Curated datasets are intended for flow classification tasks with different granularity. For instance, the top-level classifies traffic flows as benign or malware. The mid-level identifies applications such as Facebook or skype. Finally, the fine-grained level identifies specific types of malware such as portScan and application traffic such as skype audio. Although NetML aims to provide a common dataset for the research community, the code required to curate different datasets is not included in the GitHub repository. Thus, extending the provided baseline might be cumbersome.

Sharafaldin et al. [17] have developed an IDS dataset, designated as CICIDS2017, in accordance with the criteria for a comprehensive and valid IDS dataset proposed by Gharib et al. [18]. Besides, the dataset includes common updated attacks such as DoS, DDoS, Brute Force, XSS, SQL Injection, Infiltration, Port Scan, and Botnet. In order to obtain the dataset, the network traffic features are extracted from several pcap files using the flow-based feature extractor CICFlowMeter. Considering that 80 features can be extracted, the best feature set for detecting each attack is determined through the RandomForestRegressor algorithm. The main drawback of proposals based on feature extraction tools such as NetFlow or CICFlowMeter is the lack of flexibility in determining the set of features of interest. Although the extraction tools attempt to provide a common set of features, certain ML-based solutions might require features not provided by the tool.

In general, the revision of state of the art reveals that most proposals are focused on proposing a standard feature set or providing a framework to automate the feature extraction process. These efforts significantly contribute toward standard datasets that boost the adoption of machine learning techniques in networks. However, at this point, it is necessary to perform an in-depth analysis of the current open datasets to determine features in common. In addition, exploring the different latent spaces that could be obtained from an original dimension results interesting to determine the latent space where datasets converge. This standard latent space could help to further understand the impact of different features in the performance of the ML models, as well as provide a complementary interpretation of what a standard features set might be.

III. STANDARD LATENT SPACE DIMENSION

Considering that the success of applying machine learning techniques in networks depends largely on the available datasets, analyzing public datasets in pursuing common patterns or similarities represents a step towards standardization. This section presents the proposal of a standard dimension for the latent space. To this end, a dimensionality reduction process is performed with different network datasets to identify similar behavioral patterns in the resulting latent spaces.

A. PROBLEM DESCRIPTION

The starting point assumes that network datasets have certain similarities and, therefore, a projection to a lower subspace where all datasets converge can be determined. It is essential to mention that different latent spaces for each dataset are obtained. A common latent space for all datasets is not determined. Exploring different latent spaces aims to determine a numerical value related to the number of dimensions or features. Fig. 2 shows the workflow defined for determining the point where the latent spaces of the different datasets converge. The following sections describe the steps of the workflow in detail.

B. DATASETS SELECTION

The revision of related works allowed identifying the widely used datasets in networking. It is worth mentioning that most of the datasets are focused on security topics, and the Canadian Institute for Cybersecurity[1] provides most of them.

[1] https://www.unb.ca/cic/datasets/index.html

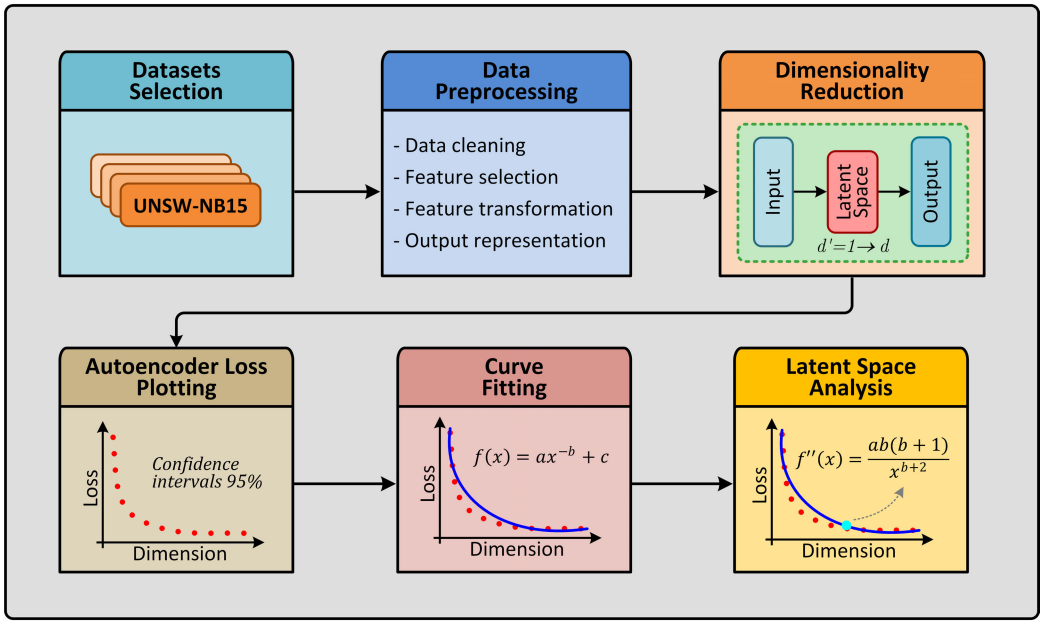

FIGURE 2. Workflow proposal to determine the standard latent space dimension. The objective is to observe the behavior of the autoencoder loss as a function of the latent space dimension in different datasets to identify similar patterns.

With the aim of performing a comprehensive analysis, five datasets for network intrusion detection systems have been considered which are designated as: CIC-IDS2017,[2] UNSW-NB15,[3] NF-UNSW-NB15-v2,[4] CSE-CIC-IDS2018,[5] and NSL-KDD.[6] Regarding the diversity of attacks, Hnamte and Hussain [19] performed an extensive survey that listed 37 datasets that could be used for shallow or deep learning-based intrusion detection systems. This survey aids in making choices based on the characteristics and limitations of a given dataset.

Datasets are structured as network flows. The traffic flows are labeled to differentiate attacks from benign traffic and recognize a subtype of attack. Table 1 summarizes the characteristics of the datasets under analysis in terms of the number of instances (labeled flows), number of features, number of attack categories, and the proportion of attacks with respect to the total number of instances. Datasets are provided in different formats, such as raw pcap files or more readable CSV files. The CSV files are used because the present work focuses on exploring the different projections into a lower-dimensional subspace instead of creating a new set of features. For a further revision of the structure of each dataset, the reader could visit the URLs listed above as footnotes.

C. DATA PREPROCESSING

According to Kotsiantis et al. [20], ML algorithms may produce less accurate and less understandable results if the quality of data is not as required. In this regard, the data preprocessing step helps solve several of problems such as noisy data, redundancy data, and missing values, among others. The authors mention that data cleaning, normalization, transformation, feature extraction, and selection represent some techniques considered in the preprocessing step. Nonetheless, there is not a single pipeline that meets all the requirements. The techniques used in this work are detailed next to obtain the final training set used within the dimensionality reduction step and the subsequent validation.

- *Dataset cleaning:* This step consists of removing flow instances that contain features with a NaN or Inf values.
- *Feature selection:* Features related to ports, IP addresses, and timestamp information are removed. As stated by Sarhan et al. [5], this selection is required to avoid introducing circumstantial flow bias into the model.
- *Feature Transformation:* This step involves feature encoding and normalization. Categorical features are mapped to numerical values using label encoding. Then, all features are normalized using the min-max method to have the same scale within the autoencoder.
- *Output representation:* Regarding the output labels, 0 and 1 are used to differentiate the benign traffic from attack categories when implementing binary classifiers. On the other hand, as stated by Bhattacharya et al. [21], categorical data, such as types of attacks, must be converted into a suitable format for ML algorithms. Thus, when implementing multiclass classification models, the attack category labels are mapped to numerical values using one-hot encoding.

After data preprocessing, the resulting datasets consist exclusively of numerical input features normalized between 0 and 1, and the corresponding label encoded outputs.

[2]https://www.unb.ca/cic/datasets/ids-2017.html
[3]https://research.unsw.edu.au/projects/unsw-nb15-dataset
[4]https://staff.itee.uq.edu.au/marius/NIDS_datasets/#RA6
[5]https://www.unb.ca/cic/datasets/ids-2018.html
[6]https://www.unb.ca/cic/datasets/nsl.html

TABLE 1. Datasets characteristics before data preprocessing.

Dataset	Instances	Features	Attack Categories	Proportion of Attacks
CIC-IDS-2017	3056488	78	15	0.224
UNSW-NB15	3240048	47	15	0.106
NF-UNSW-NB15-v2	2390275	43	10	0.040
CSE-CIC-IDS2018	1048575	79	15	0.273
NSL-KDD	148517	41	40	0.481

The proportion of attacks makes it possible to observe how instances of network traffic are distributed between benign or malicious

D. DIMENSIONALITY REDUCTION

After providing the datasets with a suitable format, the next step consists of obtaining a set of reduced versions from the original dimensions. This set will be used in a subsequent step for an in-depth analysis. The dimensionality reduction is performed using the autoencoder architecture depicted in Fig. 1. The architecture consists of an input feature layer, a dimensionality reduction layer, and a feature reconstruction layer.

In order to implement the neural networks that support the autoencoder, the open-source library Keras[7] is used. The input layer is in charge of receiving the features vector. Both the reduction and reconstruction layers are fully connected dense layers. The reduction layer applies a ReLU activation function to introduce the desired non-linearity into the dimensionality reduction technique. The reconstruction layer applies a sigmoid activation function to map the outputs (predicted inputs) between 0 and 1, which is needed to compare those outputs to the original normalized inputs. Adam is selected as the optimizer using the default Keras hyperparameter configuration with an initial learning rate of 1×10^{-3}. Regarding the loss function, binary cross-entropy is used as follows:

$$-\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} x_{ij}\log\left(\hat{x_{ij}}\right) \tag{4}$$

where N is the batch size, M is the original size of the dataset, x_{ij} is the value of the min-max normalized feature, and $\hat{x_{ij}}$ is the prediction of the feature reconstructed by the decoder.

The number of neurons allocated to the reduction layer of an autoencoder corresponds to the latent space dimension in which the original features are projected. Since the dimensionality reduction process is intended to provide a wide set of latent spaces for subsequent analysis, it is required to use as few dimensions as possible while keeping most of the original dataset information. Finding the optimal number of neurons is not as simple as choosing the best performing model after conducting a grid search in the latent space dimension. The reasoning behind this statement is that increasing the number of neurons up to the total number of original features will constantly improve the performance of the model due to the nature of an autoencoder.

In this light, the dimensionality reduction process must focus on achieving a balance between information loss and generalization power while exploring the latent spaces that can be obtained from the original size. A grid search in the latent space is mandatory, preserving the bottleneck principle of autoencoders. This approach obtains a family of autoencoders, all trained with the original feature set. However, each is configured with a specific number of neurons in the middle layer to obtain a specific latent space. The training process runs for 3 epochs using a batch size of 64. A relatively small batch size is used, considering the small proportion of attack flows available in the datasets. On the other hand, considering the large number of instances on the datasets and the small batch size used, 3 epochs are sufficient to allow the autoencoder loss function to converge.

E. AUTOENCODER LOSS PLOTTING

This step shows the behavior of the autoencoder loss as a function of the latent space dimension. To this end, the number of neurons of the hidden layer is modified from 1 to the original size of the dataset being analyzed. Then, for each resulting latent space dimension, the autoencoder loss is measured. In order to provide a reliable measure of the autoencoder loss in the presence of computational random noise, confidence intervals of 95% are calculated by repeating the measuring process 10 times. This confidence interval represents the certainty about the difference between the obtained measure at a fixed confidence level [22].

The red dots of Fig. 3 represent the sample mean of the autoencoder loss for different dimensions of the latent space. The behavioral pattern in the figure is present in all datasets and represents an expected result since the reconstruction loss tends to be minimal as the dimension increases to the original size. Moreover, it is important to note that minimal increments in the dimension size result in significant improvements in the loss below a certain threshold.

F. CURVE FITTING

The plot of the autoencoder loss allows understanding the behavior concerning different dimensions of the latent space. As mentioned before, there is a threshold where improvements in the loss start to be negligible. In order to determine this threshold, a mathematical expression for the loss that allows applying calculus principles is required. In this light, a curve fitting process is applied by using the Curve Fitting toolbox of Matlab. The best results are obtained with the power function in the form of $f(x) = ax^{-b} + c$. The resulting fitted curves for all datasets are depicted in Fig. 3 in orange

[7]https://keras.io/

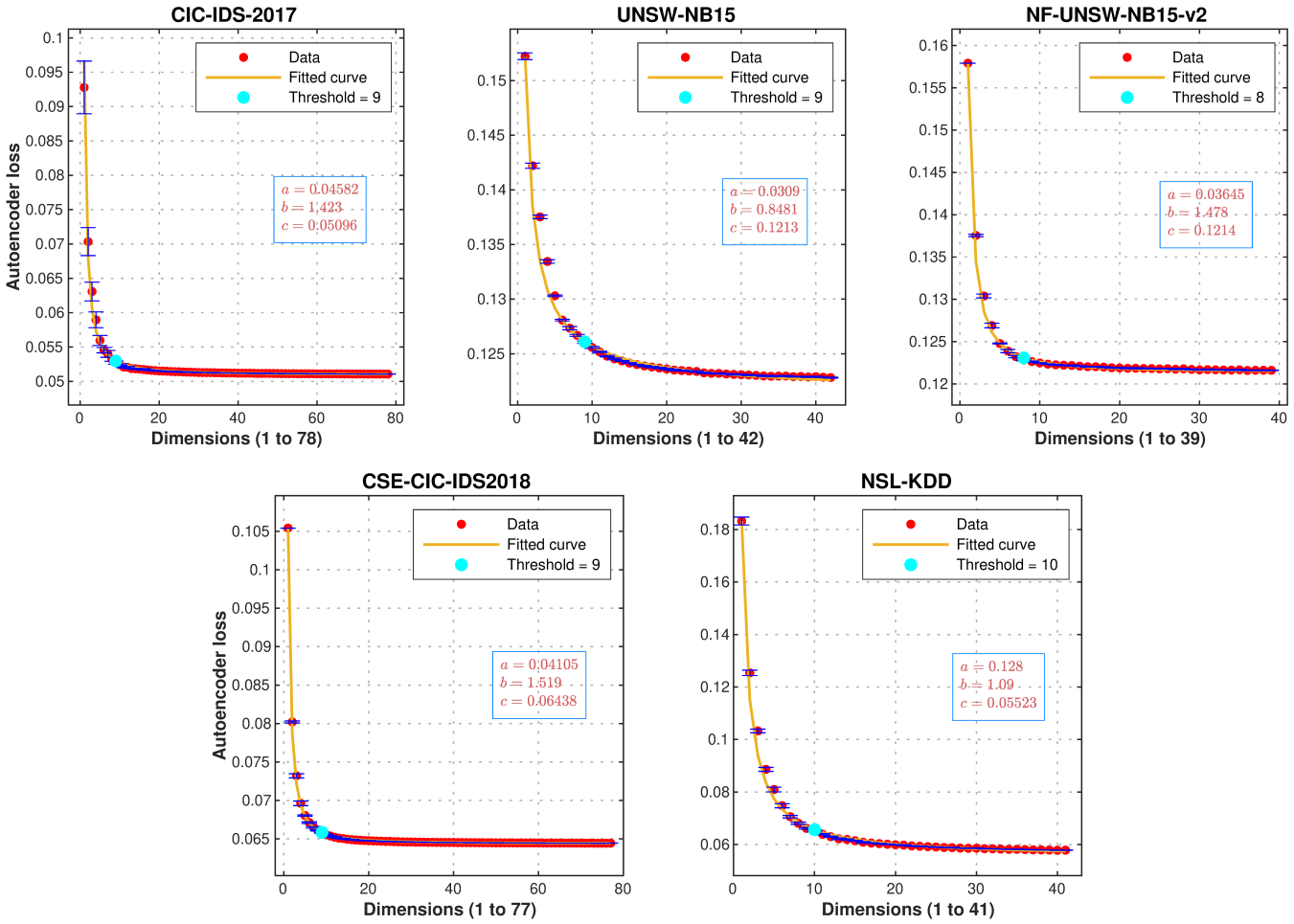

FIGURE 3. Behavior of the autoencoder loss as a function of the latent space dimension for five network intrusion detection systems datasets. The resulting plots show how diminishing returns start to occur in the same interval (8^{th} - 10^{th} dimension) for all datasets.

color. The values for the coefficients a, b and c are also included.

G. LATENT SPACE ANALYSIS

This process aims to observe the behavior of the resulting latent spaces to determine whether a point of convergence exists. This point of convergence is represented by the threshold on the latent space dimension where diminishing returns in the autoencoder loss start to occur as the dimension increases. Above this threshold, increasing the latent space dimension is pointless due to the lack of significant improvements in the loss. As observed in Fig. 3, this threshold is approximately at the ninth dimension for the CIC-IDS-2017 dataset. From the plots of the remaining datasets, it is possible to notice that the threshold is also around the ninth and tenth dimensions. In order to analytically determine the threshold for each dataset, the second derivative criterion is used.

The first derivative represents the instantaneous rate of change of the dependent variable with respect to the independent variable [23]. Thus, the growth, constancy, or decrease of the dependent variable might be analyzed in detail. The second derivative is not intended to measure the growth or decrease of the original function but rather the growth or decrease rate. In this light, the second derivative is used to observe how fast or slow the autoencoder loss decreases as the latent space dimension grows. Equation 5 represents the second derivative of the fitted curves of Fig. 3.

$$f''(x) = \frac{ab(b+1)}{x^{b+2}} \tag{5}$$

where x represent the dimension of the latent space; a and b are the coefficients of the fitted curves.

The second derivative values are calculated by replacing the variable x in equation 5 with the corresponding latent space dimensions (from 1 to the original size of the dataset). For instance, 77 values are calculated for the second derivative of the CSE-CIC-IDS2018 dataset fitted curve. These results show that at the beginning of the interval (dimension = 1 and 2), a fast or significant decrease rate of the loss with respect to the dimension occurs. However, from the tenth dimension, the decrease rate of the loss experiences a significant slowdown and, therefore, can be considered negligible. Thus, the ninth dimension is determined as

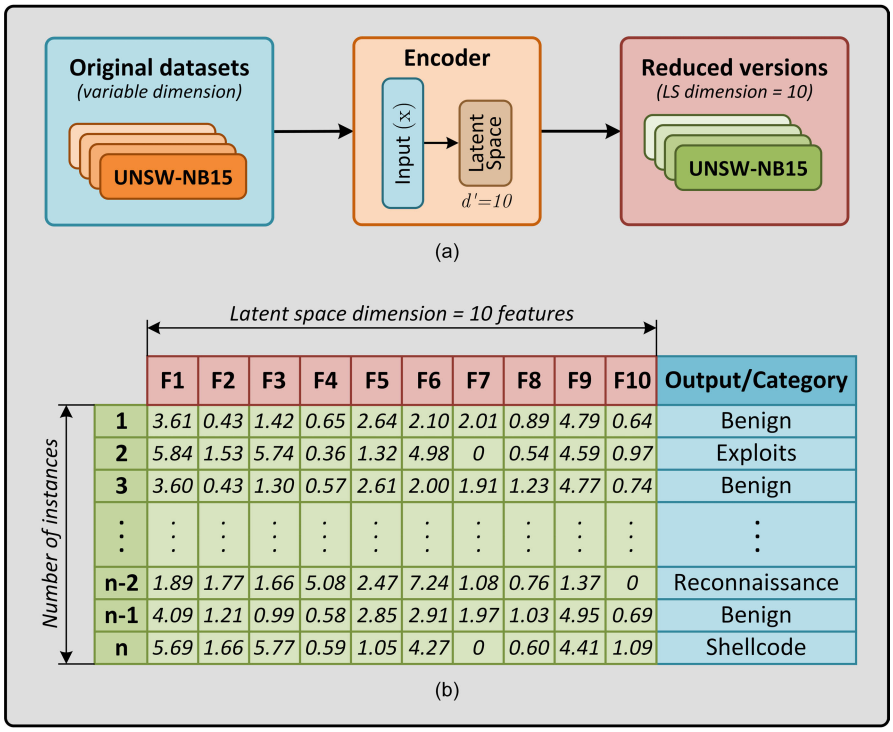

FIGURE 4. Obtaining the reduced versions of the datasets. (a) The encoder provides the reduced versions as the latent space is configured with a fixed dimension of 10. (b) The reduced versions are composed of 10 features that result from a non-linear combination of the variable set of features performed by the encoder.

the threshold. Dimensions that exceed this threshold do not improve the autoencoder loss.

The thresholds determined for all datasets are included in Fig. 3. The minimum threshold (dimension = 8) was obtained for NF-UNSW-NB15-v2 dataset, while the maximum threshold (dimension = 10) was obtained for NSL-KDD dataset. These findings are significant as they allow identifying an interval where the dimension of the latent spaces converges. The existence of a convergence point was the starting hypothesis which has been supported by these experiments. Based on these results, the tenth dimension is selected as the standard value for the latent space.

IV. EXPERIMENTAL VALIDATION

The exploration of different datasets representations to a lower subspace allowed establishing a dimension where all datasets converge and, therefore, it becomes standard. In order to validate this finding, neural networks and extra trees classification models are trained with the reduced and non-reduced versions of the datasets. Extra trees, a more randomized kind of random forest, are used due to the considerable speed-up, and therefore CPU efficiency, compared to random forests which can be beneficial when working with large datasets in terms of both examples and features [24]. On the other hand, neural networks are the preferred choice to obtain a better prediction accuracy as multiple nonlinear measures in estimating the output are introduced. In addition, neural networks have great architectural flexibility to solve problems across multiple domains, leveraging structured and unstructured data [25].

To obtain the reduced versions, the dimensionality reduction process is performed again with all the original datasets considering a standard latent space dimension. As Fig. 4(a) depicts, the encoder receives different datasets with variable set of features and provides a fixed latent space composed of 10 features per dataset ($d' = 10$). As mentioned in section II, the encoder performs a feature extraction and therefore a new set of features or latent space is obtained. Fig. 4 (b) depicts the graphical interpretation of the latent space. Note that although a dimensionality reduction is performed, the resulting dataset preserves the n instances from the original dataset.

After obtaining all the latent spaces, it is required to evaluate the performance of the ML models. Fig. 5 depicts the pipeline defined for this purpose. In order to differentiate the datasets, the reduced version is colored in red and the non-reduced version is colored in blue. In general, binary classifiers differentiate between benign and attack traffic, while multiclass classifiers identify the subtype of attack. The accuracy and F1-score classification metrics are used to evaluate the results of the defined ML pipeline. Accuracy measures the percentage of correctly classified instances, as presented in Eq. 6 where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives. Meanwhile, the F1-score is a harmonic mean between precision and recall

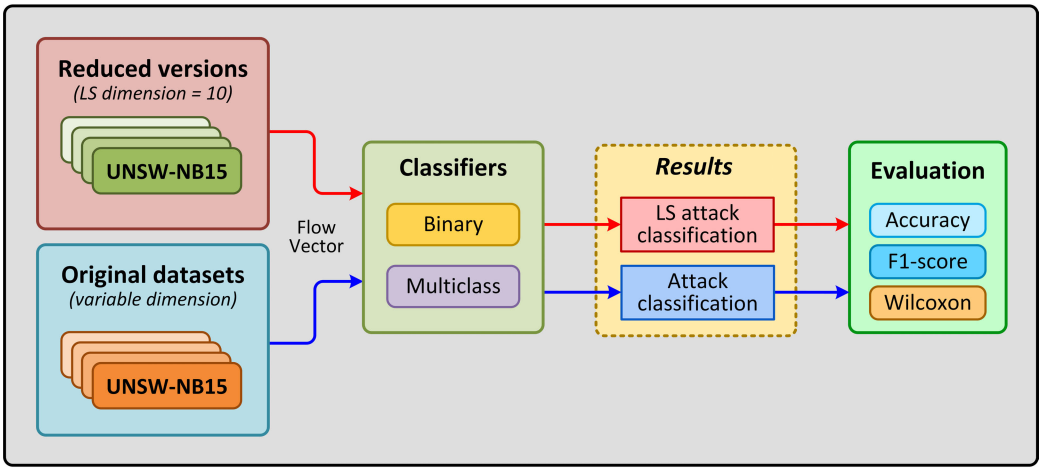

FIGURE 5. Pipeline proposal for the standard latent space dimension validation. LS is used to denote Latent Space. The pipeline is required to compare the performance of ML models when the reduced and non-reduced versions of datasets are used. Besides, it is required to determine if the difference in performance is significant.

(Eq. 9), which provides an adequate assessment of the model when the dataset has an imbalance between classes [24].

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{6}$$

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$F1 - score = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \tag{9}$$

The evaluation process is complemented with the Wilcoxon statistical test on the accuracy metric. It is important to note that the training process with the non-reduced versions of the datasets aims to define a baseline of classification metrics for both binary and multiclass classifiers. Thus, the standard latent space dimension proposal is validated by comparing the classification metrics obtained with the reduced versions of the datasets against the baseline metrics. In this light, the comparison of the baseline metrics presented in this work against the state-of-the-art classification metrics is not required.

Both neural network models are implemented by using Keras. The binary neural network consists of an input layer, an 8-neuron dense hidden layer with ReLU activation, and a single-neuron output layer. The output layer applies sigmoid activation to map the outputs between 0 and 1, a value that can be rounded off and used to predict benign or attack flows. The model is optimized with mean square loss using the Adam optimizer. Regarding the multiclass classifier, the neural network implementation consists of an input layer, a 15-neuron hidden layer with ReLU activation, and an output layer with a number of neurons equal to the number of categories represented on the dataset. The output layer applies Softmax activation to transform the numerical values into estimated probabilities corresponding to each attack category. The model is optimized with categorical cross-entropy loss using the Adam optimizer. Both neural networks are trained for 10 epochs using a batch size of 32.

Extra trees classifiers are implemented by using the scikit-learn[8] open-source library. The binary extra trees model uses 25 estimators, and the multiclass extra trees classifier uses 50 estimators. Both models consider the square root of the number of features when a split is performed. Besides, entropy is used as the optimization metric.

ML models are trained separately with the original feature set and with the latent space that has a standard dimension. After the training process, the classifiers are evaluated with the test set. The comparison between binary classifiers is performed with the accuracy and F1-score classification metrics. On the other hand, for the multiclass models, the accuracy per class is considered. Then, a weighted average is calculated with respect to the proportion of attacks.

A. RESULTS AND DISCUSSION

The training and testing processes of the considered ML models were conducted for 10 iterations using a Monte Carlo cross-validation. For each iteration, a random sample of 75% of the dataset is considered for the training set, while the remaining 25% is considered for the testing set. 10-fold cross-validation was considered; however, due to the small proportion of attacks on the dataset, the assignment of only 10% of the total instances to the testing set would not be considered a representative sample of the dataset. A Monte Carlo cross-validation is selected because it has proved to be asymptotically consistent, especially when using large datasets [26]. The classification metrics results were averaged. Table 2 shows the mean accuracy of classifiers trained with the original and latent space inputs. F1-score is also shown for binary classifiers. As noted from the table,

[8]https://scikit-learn.org/stable/

TABLE 2. Classification metrics obtained with the reduced and non-reduced versions of the datasets under analysis.

Dataset	Classifier	Accuracy	LS Accuracy	F1-score	LS F1-score
CIC-IDS-2017	*Binary ANN*	0.978	0.963	0.952	0.916
	Binary ET	0.999	0.998	0.997	0.995
	Multiclass ANN	0.982	0.969	-	-
	Multiclass ET	0.982	0.968	-	-
UNSW-NB15	*Binary ANN*	0.996	0.996	0.930	0.921
	Binary ET	1.000	1.000	0.996	0.990
	Multiclass ANN	0.990	0.988	-	-
	Multiclass ET	0.990	0.988	-	-
NF-UNSW-NB15-v2	*Binary ANN*	0.996	0.995	0.947	0.941
	Binary ET	0.998	0.997	0.970	0.969
	Multiclass ANN	0.983	0.979	-	-
	Multiclass ET	0.983	0.979	-	-
CSE-CIC-IDS2018	*Binary ANN*	0.997	0.997	0.994	0.994
	Binary ET	0.999	1.000	0.999	0.999
	Multiclass ANN	0.998	0.996	-	-
	Multiclass ET	0.997	0.995	-	-
NSL-KDD	*Binary ANN*	0.974	0.953	0.973	0.951
	Binary ET	0.994	0.992	0.994	0.992
	Multiclass ANN	0.969	0.952	-	-
	Multiclass ET	0.970	0.951	-	-

ANN: Artificial Neural Network; *ET*: Extra Trees; *LS*: Latent Space; *F1-score*: harmonic mean of the precision and recall.

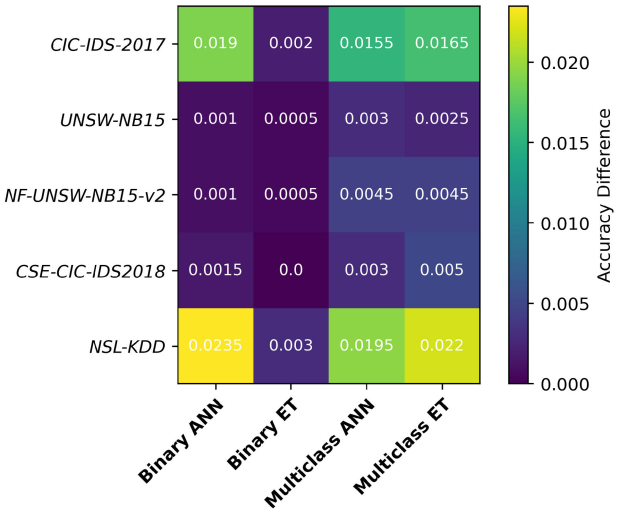

FIGURE 6. Results of the Wilcoxon equivalence test. The NSL-KDD dataset presents the highest difference value (0.0235). However, this value is practically negligible and, therefore, the use of a fixed ten-dimensional latent space provides results as good as the full-dimensional datasets.

multiclass and binary classifiers trained with the reduced versions (LS inputs) of the five network datasets performed as well as those models trained with the non-reduced versions. These remarkable results validate the use of a standard latent space dimension. In summary, feeding an ML model with a fixed ten-dimensional network dataset (a reduced version) allows to successfully accomplish the classification task regardless of the original number of dimensions or features of the dataset.

Considering that a dimensionality reduction technique is employed, losing a certain percentage of information is unavoidable as the original dataset structure is projected into a lower subspace. Consequently, the mean accuracy achieved by a ML model trained with the non-reduced versions of the dataset should be as good or better than the ML model trained with the reduced version. However, reducing the training dataset size significantly decreases the training time, computational costs, and memory requirements. To determine whether the classification results between the reduced and non-reduced versions of the datasets have a significant effect on the classification accuracy, a statistical analysis must be performed. In this work, the effectiveness of using a standard latent space dimension is validated by conducting a Wilcoxon signed-rank equivalence test on the accuracy results of both cases (reduced and non-reduced). The Wilcoxon test is required to verify that the minimal difference in accuracy is not attributed to randomness. According to Demšar, a Wilcoxon test is appropriate for comparing ML models because it is non-parametric, which means that a normal distribution is not assumed [27].

For the equivalence test, the accuracy difference between the results obtained by the implemented ML classifiers is designated as delta (Δ). Thus, the smallest value Δ such that the statistical test would hold up to a confidence level (α) of 0.05 is calculated. The results are summarized in Fig. 6. As can be noted, there is statistical evidence that the accuracy of all classification models is reduced by less than 0.0235 when a standard latent space dimension is used for the training process. Furthermore, in most cases, the accuracy difference does not exceed 0.005. A virtually negligible difference in accuracy is a significant outcome because researchers can use only the latent space to perform experiments with certainty that the performance of ML models will not be constrained.

In comparison with the non-reduced versions, the obtained latent spaces present a significant reduction in size. For instance, the original version of UNSW-NB15 is 775 MB while the corresponding latent space is 186.5 MB. A size reduction of 75.94% is achieved, which decreases the training time without worsening the previously obtained baseline classification metrics. Similarly in the CSE-CIC-IDS2018 dataset, a 70.72% reduction is achieved from 352.4 MB to 103.2 MB. The latent spaces with a standard dimension of 10 of the five datasets used in this work are published in a GitHub repository[9] along with the code of the experiments developed to validate the presented proposal. Published latent spaces are intended to provide the network research community with reduced standard versions of datasets to encourage the use of ML. Therefore, different ML pipelines could be tested with different fixed-dimensional latent spaces.

V. CONCLUSION AND FUTURE WORKS

Compared with other knowledge domains, such as computer vision, the networking domain is experiencing a slow adoption of machine learning techniques to improve processes related to design, management, deployment, and performance tuning, among others. The main reason behind this claim is the lack of appropriate resources for the network research community. Thus, network datasets receive special attention as they represent the fundamental part that enables applying ML techniques. In this context, the proposal of a standard set of features composed of those most relevant to the classification process represents an important contribution to the provision of appropriate resources that guarantee the promotion of ML usage. The classical application of dimensionality reduction involves eliminating redundancies or irrelevant features to improve the learning process. However, a different approach has been proposed in which dimensionality reduction is used to achieve standardization across security datasets. Standardization is achieved by identifying the commonalities between the datasets. In this study, we identified the dimension of the latent space where different datasets converge. Therefore, this convergence dimension can be considered a standard.

As mentioned in the background section, the relevant features might be determined by applying a dimensionality reduction process, either by means of feature extraction or feature selection. This work has presented a workflow to explore the different latent spaces or reduced versions that can be obtained from a full-dimensional dataset. To this end, the encoder part of an autoencoder is configured to perform a feature extraction operation. A comprehensive analysis of five datasets for network intrusion systems is carried out. The selected datasets range from 148,517 to 3,056,488 instances, 41 to 78 features, and 10 to 40 attack categories. Hence, a diversity of datasets in terms of characteristics and sizes is guaranteed. The experimental results have allowed determining a point of convergence of the latent spaces, which can be considered a standard dimension. To validate this finding, Artificial Neural Networks and Extra Trees were trained and tested with the reduced and non-reduced versions of the datasets. The results shown that ML models obtained practically the same accuracy with a minimal difference. In a further step, a Wilcoxon statistical test was performed to quantify such difference in terms of mean accuracy. The test provided a difference of less than 0.0235, which can be considered negligible and, therefore, the use of a fixed ten-dimensional latent space is supported.

Although determining a standard latent space dimension represents an important step towards standardization, the lack of a single dataset with a standard dimension that could be applied to different ML-based solutions still persists. To address this concern, the five obtained latent spaces should be further processed to determine a unified standard latent space that could be distributed to the research community. On the other hand, the presented work might be extended by applying the feature selection as dimensionality reduction technique. In this light, an identification of the most relevant features on the five datasets under analysis should be performed. Then, with the reduced versions of the datasets, a search should be conducted to identify common features or intersect regions. The resulting set of features could be considered standard. This extended experimental scenario would allow determining which dimensionality reduction technique provides the best results in terms of performance metrics.

Regarding feature extraction, advanced techniques such as stacked autoencoders may be explored to further analyze the behavior of latent-space convergence when datasets present complex relationships. A comparison between the results provided by encoders that implement a single hidden layer and multiple hidden layers would enrich the discussion on a standard latent space dimension for network datasets.

In addition, considering the significant advances that artificial intelligence has experienced recently, the research community has made efforts to release new network datasets that must be used to analyze whether this standard latent-space dimension is also identified on these new datasets. Similarly, further analysis will be required with datasets of a single attack class, such as Distributed Denial-of-Service, to observe how the latent-space dimension changes in comparison with datasets composed of several attack classes.

ACKNOWLEDGMENT

The authors would like to thank the Applied Signal Processing and Machine Learning Research Group of USFQ for providing the computing infrastructure (NVidia DGX workstation) to implement and execute the developed source code.

REFERENCES

[1] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: Workflow, advances and opportunities," *IEEE Netw.*, vol. 32, no. 2, pp. 92–99, Mar. 2018.

[9] https://github.com/grimloc-aduque/Standard-Latent-Space-Dimension-For-Network-Intrusion-Detection-Systems-Datasets

[2] S. V. Mahadevkar, B. Khemani, S. Patil, K. Kotecha, D. R. Vora, A. Abraham, and L. A. Gabralla, "A review on machine learning styles in computer vision—Techniques and future directions," *IEEE Access*, vol. 10, pp. 107293–107329, 2022.

[3] R. M. Samant, M. R. Bachute, S. Gite, and K. Kotecha, "Framework for deep learning-based language models using multi-task learning in natural language understanding: A systematic literature review and future directions," *IEEE Access*, vol. 10, pp. 17078–17097, 2022.

[4] O. Barut, Y. Luo, T. Zhang, W. Li, and P. Li, "NetML: A challenge for network traffic analytics," 2020, *arXiv:2004.13006*.

[5] M. Sarhan, S. Layeghy, and M. Portmann, "Towards a standard feature set for network intrusion detection system datasets," *Mobile Netw. Appl.*, vol. 27, pp. 357–370, Nov. 2021.

[6] J. Holland, P. Schmitt, N. Feamster, and P. Mittal, "New directions in automated traffic analysis," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2021, pp. 3366–3383.

[7] M. Li, H. Wang, L. Yang, Y. Liang, Z. Shang, and H. Wan, "Fast hybrid dimensionality reduction method for classification based on feature selection and grouped feature extraction," *Exp. Syst. Appl.*, vol. 150, Jul. 2020, Art. no. 113277.

[8] L. Van Der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: A comparative," *J. Mach. Learn. Res.*, vol. 10, nos. 66–71, p. 13, 2009.

[9] Q. Fournier and D. Aloise, "Empirical comparison between autoencoders and traditional dimensionality reduction methods," in *Proc. IEEE 2nd Int. Conf. Artif. Intell. Knowl. Eng. (AIKE)*, Jun. 2019, pp. 211–214.

[10] B. Janakiramaiah, G. Kalyani, S. Narayana, and T. B. M. Krishna, "Reducing dimensionality of data using autoencoders," in *Smart Intelligent Computing and Applications*. Berlin, Germany: Springer, 2020, pp. 51–58.

[11] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, pp. 232–242, Apr. 2016.

[12] D. C. Ferreira, F. I. Vázquez, and T. Zseby, "Extreme dimensionality reduction for network attack visualization with autoencoders," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–10.

[13] H. Motoda and H. Liu, "Feature selection, extraction and construction," *Commun. IICM*, vol. 5, p. 2, May 2002.

[14] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *J. Appl. Sci. Technol. Trends*, vol. 1, no. 2, pp. 56–70, May 2020.

[15] M. Sarhan, S. Layeghy, and M. Portmann, "Evaluating standard feature sets towards increased generalisability and explainability of ML-based network intrusion detection," 2021, *arXiv:2104.07183*.

[16] F. Bronzino, P. Schmitt, S. Ayoubi, H. Kim, R. Teixeira, and N. Feamster, "Traffic refinery: Cost-aware data representation for machine learning on network traffic," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 5, no. 3, pp. 1–24, 2021.

[17] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, vol. 1, Jan. 2018, pp. 108–116.

[18] A. Gharib, I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "An evaluation framework for intrusion detection dataset," in *Proc. Int. Conf. Inf. Sci. Secur. (ICISS)*, Dec. 2016, pp. 1–6.

[19] V. Hnamte and J. Hussain, "An extensive survey on intrusion detection systems: Datasets and challenges for modern scenario," in *Proc. 3rd Int. Conf. Electr., Control Instrum. Eng. (ICECIE)*, Nov. 2021, pp. 1–10.

[20] S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas, "Data preprocessing for supervised leaning," *Int. J. Comput. Sci.*, vol. 1, no. 2, pp. 111–117, 2006.

[21] S. Bhattacharya, P. K. R. Maddikunta, R. Kaluri, S. Singh, T. R. Gadekallu, M. Alazab, and U. Tariq, "A novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU," *Electronics*, vol. 9, no. 2, p. 219, Jan. 2020.

[22] J. Carrasco, S. García, M. M. Rueda, S. Das, and F. Herrera, "Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review," *Swarm Evol. Comput.*, vol. 54, May 2020, Art. no. 100665.

[23] J. Stewart, *Calculus*. Boston, MA, USA: Cengage Learning, 2015.

[24] K. Banachewicz and L. Massaron, *The Kaggle Book: Data Analysis and Machine Learning for Competitive Data Science*. Birmingham, U.K.: Packt, 2022. [Online]. Available: https://cir.nii.ac.jp/crid/1130854870125643045

[25] V. K. Ayyadevara, *Neural Networks With Keras Cookbook: Over 70 Recipes Leveraging Deep Learning Techniques Across Image, Text, Audio, and Game Bots*. Birmingham, U.K.: Packt, 2019.

[26] Q.-S. Xu and Y.-Z. Liang, "Monte Carlo cross validation," *Chemometric Intell. Lab. Syst.*, vol. 56, no. 1, pp. 1–11, Apr. 2001.

[27] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

RICARDO FLORES MOYANO (Member, IEEE) received the degree in electronic engineering from Universidad Politécnica Salesiana (UPS), in 2006, and the M.S. and Ph.D. degrees in telematics systems engineering from Universidad Politécnica de Madrid (UPM), in 2013 and 2018, respectively. As a Ph.D. candidate, he was a part of the Networking and Virtualization of Communications Services Research Group (GIROS) and participated in Spanish research projects, such as CloudTrust, GREDOS, and ElasticNetworks. He is currently a full-time Professor with the Department of Computer Science Engineering, Universidad San Francisco de Quito (USFQ). His research interests include applied machine learning, software defined networks (SDN), network functions virtualization (NFV), future internet, cloud networking, and 5G networks.

ALEJANDRO DUQUE (Member, IEEE) is currently pursuing the degree in computer science with Universidad San Francisco de Quito (USFQ). Since 2022, he has been a part-time Research Assistant with the Department of Computer Science, USFQ. His research interests include neural network architectures, applied machine learning, and pattern recognition. He won a Scholarship to pursue his degree from USFQ.

DANIEL RIOFRÍO (Member, IEEE) received the M.Sc. and Ph.D. degrees in computer science from The University of New Mexico, USA, in 2012 and 2015, respectively. He is currently a full-time Professor with the Computer Science Engineering Program, Universidad San Francisco de Quito (USFQ), where he is also the Coordinator of the program. His research interests include the novel applications of machine learning, computer vision, security, cybersecurity, and optimization techniques for radiotherapy and radiosurgery. He has supported the Open Technology Fund (OTF) as a fellow of Senior Information Controls, in 2017, and holds a U.S. patent (9,844,684/10,850,122), in December 2017 and December 2020.

NOEL PÉREZ (Senior Member, IEEE) received the B.Eng. degree in computer science engineering and the M.Sc. degree in applied computer science and in digital image processing, pattern recognition, and machine learning from Universidad de Ciego de Avila, Cuba, in 2005 and 2007, respectively, and the Ph.D. degree in data mining from the University of Porto, Portugal, in 2015. From 2008 to 2015, he was a Research Fellow with the Faculty of Engineering, Institute of Mechanical Engineering and Industrial Management (INEGI), University of Porto, where he was a Postdoctoral Fellow with the Faculty of Sciences, Instituto de Telecomunicações (IT), from 2015 to 2017. Since 2017, he has been a full-time Professor with Universidad San Francisco de Quito, Ecuador, where he co-founded the Applied Signal Processing and Machine Learning Research Group. His research interests include digital image processing, data mining, pattern recognition, and machine learning.

DIEGO BENÍTEZ (Senior Member, IEEE) received the B.Eng. degree in electrical engineering from Escuela Politécnica Nacional, Quito, Ecuador, in 1994, and the M.Sc. and Ph.D. degrees in electrical engineering from the Institute of Science and Technology, The University of Manchester, Manchester, U.K., in 1997 and 2001, respectively. From 2005 to 2007, he was a Postdoctoral Research Associate with the Sensing, Imaging, and Signal Processing Research Group, School of Electrical and Electronic Engineering, The University of Manchester. From 2007 to 2012, he was a Senior Research Engineer with the Bosch Research and Technology Center, Pittsburgh, PA, USA, where he was also an Academic Visitor with the Institute for Complex Engineered Systems and the INFER Laboratory, Carnegie Mellon University. From 2012 to 2014, he was a Visiting Research Scholar with Universidad de las Fuerzas Armadas ESPE under the ''Prometeo Program'' of SENESCYT, Ecuador. He is currently a full-time Faculty Member of Universidad San Francisco de Quito (USFQ), Quito, where he co-founded and led the Applied Signal Processing and Machine Learning Research Group. He has authored more than 150 refereed journals and conference papers on these topics. He holds 26 granted and pending U.S. and overseas patents. His research interests include signal and image processing, pattern recognition and machine learning, intelligent instrumentation and measurement systems for medical, energy management, security, and smart building applications. He has served as the IEEE Ecuador Section Chair, from 2018 to 2019.

MARIA BALDEON-CALISTO (Member, IEEE) received the B.Eng. degree in industrial engineering from Universidad San Francisco de Quito, Ecuador, and the M.Sc. and Ph.D. degrees in industrial engineering from the University of South Florida, USA, in 2018 and 2020, respectively. Since 2021, she has been a full-time Professor with Universidad San Francisco de Quito, teaching undergraduate and graduate courses with the Department of Industrial Engineering. She is also the Director of the CATENA Research Institute. Her research interests include digital image processing, artificial intelligence in medical diagnosis, neural architecture search, and hyperparameter optimization.

DAVID FERNÁNDEZ received the M.S. degree in telecommunications engineering and the Ph.D. degree in telematics engineering from Universidad Politécnica de Madrid (UPM), Spain, in 1988 and 1993, respectively. Since 1995, he has been an Associate Professor with the Department of Telematics Systems Engineering (DIT), UPM. His current research interests include software-defined networks, network virtualization, cloud computing datacenters technologies, and network security.