Atlantis Highlights in Computer Sciences, volume 4
Proceedings of the 3rd International Conference on Integrated Intelligent Computing Communication & Security (ICIIC 2021)

A New Combined Model with Reduced Label Dependency for Malware Classification

Prishita Ray[1,*], Tanmayi Nandan[2], Lahari Anne[3], Kakelli Anil Kumar[4]

[1,2,3,4] *School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India*
[*] *Corresponding author: prishita.r85@gmail.com*

ABSTRACT
With the technological advancements in recent times, security threats caused by malware are increasing with no bounds. The first step performed by security analysts for the detection and mitigation of malware is its classification. This paper aims to classify network intrusion malware using new-age machine learning techniques with reduced label dependency and identifies the most effective combination of feature selection and classification technique for this purpose. The proposed model, L2 Regularized Autoencoder Enabled Ladder Networks Classifier (RAELN-Classifier), is developed based on a combinatory analysis of various feature selection techniques like FSFC, variants of autoencoders and semi-supervised classification techniques such as ladder networks. The model is trained and tested over UNSW-NB15 and benchmark NSL-KDD datasets for accurate real time model performance evaluation using overall accuracy as well as per-class accuracy and was found to result in higher accuracy compared to similar baseline and state-of-the-art models.

Keywords: *Autoencoders, Computer security, Feature selection, FSFC, Ladder networks, Machine learning, Multi-class classification, Network intrusion malware.*

1. INTRODUCTION

Malware, defined as a software that is added or modified in a system to intentionally cause harm, is a generic term that encloses all Trojans, viruses, spyware etc. [1]. Though most of the malwares have network activity while performing attacks, the malwares that corrupt the network systems and exploit network protocols are termed as network malwares. The damage caused by these malwares to the network system, resources, and node systems point out the need for early detection, which includes identification, analysis and mitigation mechanisms [2-3].

Network intrusion malwares can be broadly classified based on their attack types such as illegal access to root user credentials, denial of system services, etc. [4]. However, manually observing and labelling each program running on a system when generating a large dataset, with its corresponding subclass is a highly tedious task involving inaccuracies. Therefore, it is practical to devise an approach that can automate this classification task [5].

The proposed model, L2 Regularized Autoencoder Enabled Ladder Networks Classifier (RAELN-Classifier), aims to facilitate the classification phase of network intrusion malware attacks mitigation process [6]. It identifies the various classes of the malware files using new age machine learning techniques, having reduced label dependency, with promised accuracy in real time situations [7-8]. It comprises of a novel combined unsupervised feature selection and semi-supervised multi-class classification technique that is trained and tested on the UNSW-NB15 and NSL-KDD benchmark datasets. The performance of the combination is compared with other fully-supervised baseline and state-of-the-art classification algorithms, with and without prior feature selection [9]. Overall and per-class classification accuracies are compared to validate the best combination of feature selection and classification model. The novel combination in the proposed model is chosen after a series of experiments performed using combinations of feature selection algorithms like FSFC, autoencoder based techniques with supervised and semi-supervised classification techniques such as ladder networks [10-12].

1.1. Supervised, Unsupervised and Semi-Supervised Learning

Supervised learning refers to the branch of machine learning algorithms that find patterns and relations among the features in a labelled dataset. Unsupervised learning includes clustering algorithms that learn from the unlabelled datasets. A slight modification of unsupervised learning is semi-supervised learning, where only some samples in the input data are labelled. The algorithms of this branch find patterns within the features on the labelled samples to produce suitable output labels on the unlabelled samples. This approach is useful when there is a dearth of labelled data available for training and the model performance is expected to be at par with supervised learning.

1.2. Undercomplete Autoencoders

An autoencoder [13] is an unsupervised machine learning model based on artificial neural networks that is commonly used for dimensionality reduction purposes. The architecture of such networks consists of two parts: an encoder and a decoder. The encoder compresses the input *x* to a latent-space representation using an encoding function *h = f(x).* When the dimension of this representation is smaller than that of the input, it is said to be under complete. The decoder has a layer configuration opposite to that of the encoder and decodes this latent space representation to reconstruct the original input using the decoding function *x'= g(h). x'* here refers to the reconstructed input. The mean squared error of *x'* with the original input *x* is used to calculate the loss of the network. Autoencoders adjust the model hyperparameters to reduce this reconstruction loss, Equation (1), on the data where the innermost hidden layer contains a number of nodes equal to the desired number of features to obtain a suitable latent-space representation with reduced dimensionality.

$$loss = \sqrt{\frac{1}{N}\sum\left(x - \hat{x}\right)^2} \tag{1}$$

Thus, those features with the highest weights at the end of the training process reflect their importance in describing the data. They are the most relevant and must be chosen.

1.3 FSFC

Feature-Selection based Feature Clustering (FSFC), is an unsupervised feature clustering algorithm that groups features based on their similarity and importance [14]. In order to find the similarity between two features F_i and F_j, their Symmetric Uncertainty (*SU*) is calculated using the formula $SU(F_i, F_j) = 2 \cdot \frac{IG(F_i,F_j)}{H(F_i)+H(F_j)}$, where *IG* (Equation (2)) is the information gain of feature F_i when F_j is observed, and *H(F)* (Equation (3)) is the entropy value of the feature *F*.

$$IG(F_i, F_j) = H(F_i) - H\left(\frac{F_i}{F_j}\right) \tag{2}$$

$$H(F) = -\sum prob(f) \cdot \log_2 prob(f), \forall\, f \in F \tag{3}$$

To reflect the degree of redundancy of feature F_i with its k nearest neighbours, the density of *kNN* (*DkNN)* is calculated by $DkNN = \frac{\sum SU(F_i,F)}{k}\ \forall\, F \in kNN(F_i)$. Among all features that are cluster centres (in set F_c), the pair having the minimum distance between them or alternatively having the highest symmetric uncertainty gives the minimum cluster centre distance *maxSU*(F_c).

Symmetric Uncertainty (*SU*) is calculated for each pair of features and the feature set is ordered based on decreasing values of *DkNN*. The feature with the highest *DkNN* is selected as the first cluster centre. If a new feature *F* is not in the *kNN* of a cluster centre and vice versa, the feature is selected as a new cluster centre and the *maxSU*(F_c) is updated as the maximum of the *SU* between the feature *F* and the existing cluster centres if greater than the existing *SU* value. Next, the redundant features not in F_c are distributed into clusters by comparing their *SU* with the *maxSU* of each of the other cluster centres. This leads to formation of the clusters. Finally, to find the representative feature of each cluster, the average redundancy of each feature within a cluster *C* is calculated as $AR(F, C) = \frac{\sum_{i=1}^{n} SU(F,F_i)}{n}$, where *n* is the number of features in the cluster. The feature with the highest *AR* in cluster *C* will contain the most information and will serve as the representative.

1.4. Ladder Networks

Ladder Networks [15] is a popular deep learning architecture that is used to identify and represent more relevant and abstract hierarchical structures that are present within the inputs but cannot be identified directly. They learn by training the latent variables using both inferences and posterior probabilities, thus combining supervised and unsupervised learning in a principled way. They consist of encoder and decoder networks similar to that in autoencoders. However, the networks have skip connections between them, that is, every layer in the encoder is connected to its corresponding layer in the decoder directly as well as through the other layers in between. The skip connections ensure no data is lost during representation. The ladder networks are designed in a way that two different pathways handle the labelled and unlabelled samples for supervised learning and unsupervised learning respectively. These pathways are called Labelled Samples pathway and Unlabelled samples pathway.

2. RELATED WORK

Recent methods proposed for malware classification include [16] and [17] that use new-age image based deep learning techniques. The models are tested over benchmark dataset, Microsoft Malware Classification Challenge (BIG-2015). However, the model learns from highly labelled datasets, with no feature selection prior to classification.

Combination of feature selection and classification techniques is proposed in [18], with deep convolutional neural network-based feature extractor and Support Vector Machine based classifier. A combination of various feature selection techniques with supervised classification models are compared in [19]. However, all of these methods rely on the actual classes of the samples to find patterns within the features whereas in real life it is a highly tedious task to manually observe and label each of the running programs on a system.

A semi-supervised transfer learning model for cloud malware classification with no feature selection was proposed in [20]. The model was trained and tested on Kaggle datasets, resulting in highly accurate malware classification. It was trained on limited labelled training samples with no feature extraction, gave high accuracy, thereby rendering semi-supervised learning a better option for malware classification.

State-of-the-art supervised classifiers such as Random Forest, Naïve Bayes have given good average accuracy on multi-class classification use cases, with highly imbalanced datasets and extensive number of features. The network intrusion malware datasets used for the experiments in this paper have similar dataset structure with those used in [22], having high feature dependency and multiple classes, thus these state-of-the-art classifiers will be compared with the proposed RAELN-Classifier model.

Multi class classification using ladder networks on the KDD dataset was performed in [21] with minimal classes. This semi-supervised approach was compared against standard supervised learning techniques such as Random Forest and Support Vector Machine models. However, experiments were done using a 50% labelled dataset having fewer samples (around 5000 samples) without any prior feature selection. Dataset classes were modified based on number of training samples considered notwithstanding the class imbalance problem, with no supervised baseline ladder networks used for comparison to state how reduced label dependency is an advantage over its supervised counterpart.

Similar effort is seen in [23] which proposed a ladder network-based intrusion detection system. However, there is no comparison with supervised counterparts and no mention of feature selection algorithms used. The wireless network intrusion malware dataset used in the paper was also not imbalanced. The combination of semi-supervised learning and deep neural networks with pseudo labels and ladder networks is tested in [24]. Their proposed combined model was tested on benchmark NSL-KDD to check the efficacy of the model. However, feature selection was not performed prior to the classification, and pseudo-labelling is not always reliable as it causes confirmation-bias [25].

Therefore, a combined unsupervised feature selection and semi-supervised classification model having reduced label dependency, as proposed in this paper, can effectively classify network intrusion malware even with fewer labelled samples, on a larger training and testing set (around 1,50,000 samples) with a higher accuracy.

3. PROPOSED MODEL

RAELN-Classifier is a combination of feature selection and classification model for classifying network intrusion malware, which is trained and tested on UNSW-NB15 and NSL-KDD datasets. This combination is selected after a series of experiments with new-age techniques, mentioned in Section 1, and comparing the results with baseline and state-of-art models based on overall and per-class classification.

3.1 Datasets

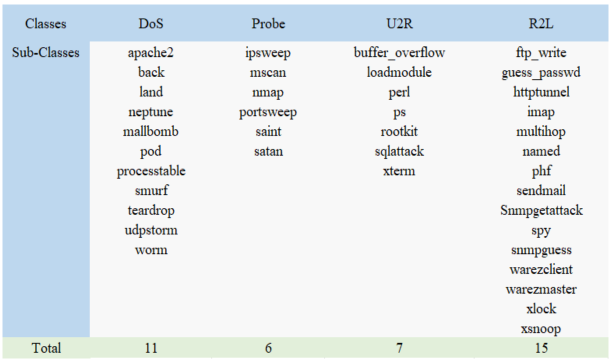
Experiments are performed on the following two network intrusion malware datasets:

3.1.1 UNSW-NB15 Dataset

UNSW-NB15 [26] was developed to efficiently evaluate the Network Intrusion Detection System against network behaviours. The dataset generation architecture consists of a traffic generator called IXIA, a pcap file capturer called Tcpdump tool, and a few feature extractors like Argus and Bro-IDS. These features are matched and are grouped as basic, content and time. Additional features are also generated with this matching process, that can help detect stealth and multi-flow attacks. Followed by the feature generation process, the features are associated with labels determining whether they are legitimate or malicious. The dataset is labelled into 9 classes, namely Fuzzers, Backdoor, Analysis, DoS, Exploit, generic, Reconnaissance, worms, and shellcode and 1 benign or normal class. UNSW-NB15 stands out from other datasets with clear metrics like realistic network configuration, labelled observations, full packet capture, etc. Comparison based on these metrics reveals that this dataset is better than KDD99, NSL-KDD, CAIDA, DEFCON, ISCX, DARPA 2009 [27].

3.1.2 NSL-KDD Benchmark Dataset

NSL-KDD [28] is a benchmark dataset for internet traffic, consisting of 4 subsets which include the training and testing sets. The records are a consolidated statistic of 43 features of which the last two are labels determining whether an input is malicious and if yes, what is its severity. The first 41 features can be broadly categorized into 4 groups namely, Intrinsic, Content, Host-based and Time-based. The dataset has a total of 39 sub-classes, each of which belongs to one of the 4 attack classes in Figure 1, DoS (Denial of Service) where a system is rendered unavailable for users by sending multiple requests, Probe which is the stealing of personal information, U2R (User to Root), a privilege escalation attack and R2L (Remote to Local) where local access is obtained over a remote system. The dataset is pre-processed to generalize these subclasses to reduce discrepancies in the prediction as the training set contains 22 sub-classes whereas the testing set contains 39 sub-classes.



Classes	DoS	Probe	U2R	R2L
Sub-Classes	apache2	ipsweep	buffer_overflow	ftp_write
	back	mscan	loadmodule	guess_passwd
	land	nmap	perl	httptunnel
	neptune	portsweep	ps	imap
	mailbomb	saint	rootkit	multihop
	pod	satan	sqlattack	named
	processtable		xterm	phf
	smurf			sendmail
	teardrop			Snmpgetattack
	udpstorm			spy
	worm			snmpguess
				warezclient
				warezmaster
				xlock
				xsnoop
Total	11	6	7	15

Figure 1 Sub-classes and the parent attack classes in the NSL-KDD dataset

3.2. Data Pre-processing

The datasets consist of categorical features which need to be converted to numeric values. To deal with the huge range in values, the features are normalized using their mean(μ) and standard deviation(σ). The labels are one-hot integer encoded (converted to binary-valued vectors with sizes equal to the total number of classes in the dataset, with each position containing 0 by default and 1 if the label belongs to a particular class) to make classification easier. Feature selection algorithm, FSFC [14] uses entropy to calculate the feature importance which needs discrete values whereas the normalized feature values are continuous. To address this discrepancy, each of the feature values are discretized into six integer values based on the ranges between $\mu \pm 1.5\sigma$, $\mu \pm \sigma$, $\mu \pm 0.5\sigma$ and μ of their respective features.

3.3. Unsupervised Feature Selection

Redundant features within the dataset reduces the model performance and may lead to overfitting during training as explained in [13]. Hence, Principal Component Analysis (PCA) is done to determine the optimal number of features required.

From Figure 2, the point in the PCA graph at which the variance curvature decreases and becomes almost linear gives the ideal number of features that can effectively classify the data as explained in [29]. As a result, in the UNSW-NB15 dataset (with 43 features) and the NSL-KDD dataset (with 41 features), the selection of 25-35 features can help preserve around 98-99% of the variance in the data.

Therefore, approximately 25-35 most relevant features are selected by utilizing the two unsupervised feature selection approaches, that are autoencoder-based [13] and a relatively new clustering algorithm known as FSFC [14] mentioned in Section 1.2 and 1.3 respectively. To obtain the most relevant features to ensure higher accuracy, these algorithms find correlations between the features in a way that is independent of the class labels.

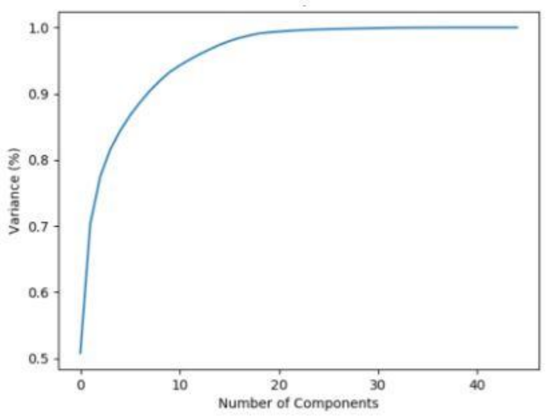


Figure 2 PCA based estimation of the number of components required for relevant feature selection in the UNSW-NB15 and NSL-KDD Datasets.

3.3.1 Autoencoder-based Feature Selection

Undercomplete autoencoders, as explained in Section 1.2 can be used for unsupervised feature selection. The performance of 4 variants of autoencoders, based on their reconstruction loss and variance in weights are compared. The Autoencoder AE, only minimizes reconstruction loss which might lead to overfitting [23] during classification. To address this problem, regularization terms are introduced, that provide additional information in the calculation of the loss function. These terms are calculated based on the L1 and L2 norms of the weight vectors between the hidden layers of the autoencoder.

The L1 regularization, also known as lasso regularization is the linear norm of the weight vectors, and the L2 regularization, known as ridge regularization is the Euclidean norm of the weight vectors, as explained in the Equations (4) and (5) below.

$$L_1\ norm:\ ||w||_1 = |w_1| + |w_2| + \ldots + |w_n| \quad (4)$$
$$L_2\ norm: ||w||_2 = w_1^2 + w_2^2 + \ldots + w_n^2 \quad (5)$$

Thus, two new variants of the autoencoders AE_L1 and AE_L2 are created that minimize their respective loss functions as follows:

$$loss_{AE-L1} = loss + ||w||_1 \quad (6)$$
$$loss_{AE-L2} = loss + ||w||_2 \quad (7)$$

Table 1. Performance Comparison of the 4 Autoencoder variants on the Dataset features

Autoencoder	Reconstruction Loss	Variance of Calculated Feature Importance	Convergence
AE	0.0386	0.3907	Good
AE_L1	0.568	0.3909	Bad
AE_L2	**0.0785**	**0.2699**	**Moderate**
AE_Dropout	0.1455	0.4606	Moderate

**AE_L2 refers to L2 Regularized Autoencoder that is used in the proposed model, RAELN-Classifier.*

The fourth autoencoder variant (AE_Dropout) makes use of dropout regularization, that does not consider weight values in a vector that fall below a threshold probability value (0.8) in the calculation of the loss. This variation is included in the weight vector leading to the innermost hidden layer. This loss is denoted as $loss_{dropout}$.

All the 4 variants were trained on both the datasets and the observations are recorded in Table 1. From the table, AE has the least reconstruction loss with a good convergence and can find feature importance within a small variation range. AE_L2 has a slightly higher reconstruction loss, which implies that it is able to generalize over the input data better using the regularization term and can reduce overfitting. Therefore, in the experiments, autoencoder variants (AE and AE_L2) will be considered for feature selection.

3.3.2 FSFC-based Feature Selection

FSFC algorithm is used to compare the performance of the two autoencoder variants in combination with a specific classifier on the final multi-class classification task. This algorithm performs unsupervised feature selection based on KNN clustering as mentioned in Section 1.3.

3.4. Overview of the Training Approach

As part of the experiments, models with the above feature selection algorithms (AE, AE_L2 and FSFC) in combination with semi-supervised ladder networks (discussed in Section 1.4) will be trained and tested, of which AE_L2 with semi-supervised ladder networks will form our proposed RAELN-Classifier model. Also, the performance of ladder networks without prior feature selection is observed. Each model will input data from the features that are selected using one of the above feature selection algorithms which will then be passed to the ladder network along with the available sample labels for classification. Within the ladder network, samples will be passed to labelled or unlabelled pathways depending on the availability of target labels. As the network contains both the encoder and decoder parts of the architecture, it will adjust its hyperparameter values to minimize the total labelled and unlabelled cost and ultimately obtain a good accuracy on both training and testing sets. Baseline approaches are also designed, with fully-supervised ladder networks, with and without prior feature selection.

The aim of the baseline models will only be to minimize the labelled cost in their ladder networks by adjusting their hyperparameter values, so they will only contain the encoder for their labelled pathway. The accuracy obtained by these baseline models on the training and testing sets will be used to compare the performance and decide on the best combined feature selection and classification model suitable for the intended task. Additionally, the performance of the Random Forest and Naïve Bayes state-of-the-art models with and without prior feature selection is compared to highlight the advantage offered by the proposed RAELN-Classifier combined model, in terms of better accuracy obtained.

3.5. Training Model Settings

The model is trained on the UNSW-NB15 dataset and NSL-KDD benchmark dataset separately. In the UNSW-NB15 dataset, 75% (1,50,000 samples) of the data is used for training and 25% (50,000 samples) is used for testing. A similar train-test split ratio is used with the NSL-KDD dataset where 1,20,000 samples and 40,000 samples are used for training and testing respectively. In the semi-supervised setting, only the class labels of the first 5000 samples in the training set are considered for calculating the labelled cost and the rest of the samples are used to calculate the unlabelled cost, whereas, in the supervised setting, the class labels of all samples in the training set are considered to calculate only the labelled cost.

For the UNSW-NB15 dataset having 43 features and 10 classes, the AE and AE_L2 autoencoders are trained with layer sizes of (43, 38, 30, 38, 43) and a batch size of 10000, for 250 epochs to choose the top 30 features suitable for classification. From the FSFC feature selector, the top 35 non-redundant features are selected. The data of these chosen features are given as input to each of the 4 classifiers (ladder networks, supervised ladder networks, random forest and naive bayes). The model framework of the ladder networks includes 4 layers with sizes (n_f, 25, 15, 10), where n_f is the number of selected features. They are trained with a batch size of 1000 for 250 epochs. The max tree depth of the random forest classifier is set to 10.

For the NSL-KDD dataset having 41 features and 5 classes, only the layer sizes of the autoencoders are changed to (41, 35, 30, 35, 41) with all training parameters kept the same to choose the top 30 features. The FSFC feature selector returns the top 34 non-redundant features based on the feature clusters formed. Data of these features are passed to the classifiers. The hidden layer sizes of the ladder network architecture are varied to (n_f, 25, 10, 5), which are trained with a batch size of 1000 for 250 epochs. The random forest tree depth is set to 10 as before.

The experiments were performed on Windows 10 64-bit Intel(R) Core(TM) i5-7200U CPU (2.71GHz) with 8GB RAM and an NVIDIA GeForce 940MX GPU with 6GB memory. The NumPy, scikit-learn, pandas and Matplotlib Python libraries were commonly used for all the above settings. TensorFlow Python library [30] was used to implement the autoencoders and ladder network models using the GPU. To avoid training bias over the same set of samples, training data is shuffled randomly. The autoencoders are trained with an initial learning rate of 0.01 that decays by 30% in every 25 epochs and all layers use the ELU activation function. For the ladder networks, initial learning rate is set to be 0.005 with a decay of 4% for every 50 epochs. All layers except the last use the ELU for activation, the last layer uses the softmax function.

4. RESULTS AND DISCUSSION

The performance of the models having combinations of unsupervised feature selection algorithms with semi-supervised ladder networks for multi-class classification are compared with that of baseline models of fully-supervised ladder networks and state-of-the-art supervised ML classification algorithms such as Random Forest and Naive Bayes Classifiers [21,22,31,32] with and without prior feature selection for the classification task (Refer Section 3.4 for details). This is done so that a conclusion can be drawn about the best feature selector and classification model combination suitable for the multi-class classification.

The performance metrics used for comparing the models are average sample accuracy and average per-class accuracy on the training and testing sets. Average sample accuracy is the ratio of correctly predicted samples to the total number of samples. Per-class accuracy is important to evaluate the model performance in terms of generalization capability over an imbalanced dataset.

From the results in Table 2, the proposed combined model of the AE_L2 Regularized Autoencoder feature selector with semi-supervised Ladder Networks (RAELN-Classifier) gives the best average testing accuracy of 81.06% on the UNSW-NB15 dataset, having 10 classes, ensuring minimum model overfitting among all other combinations. It performs better than the corresponding supervised baseline model in all cases (80.676% on the UNSW-NB15 dataset), and when compared with the state-of-the-art Random Forest (74.744%) and Naïve Bayes (53.784%) classifier models. The performance of the proposed RAELN-classifier model is also better than semi-supervised Ladder Networks without prior feature selection (64.592%), highlighting the efficacy of the model for the required classification task.

The average testing accuracy of RAELN-Classifier is comparable to that of semi-supervised Ladder Networks classifier without prior feature selection in Table 3, on the benchmark NSL-KDD dataset having 5 classes (74.709% and 76.227% respectively). This can be attributed to the fact that fewer classes in a relatively balanced dataset such as in NSL-KDD [33] make it easier for the model to identify patterns among the features whereas with more classes, this task becomes more difficult, hence describing why feature selection prior to classification does not provide a major advantage with the NSL-KDD benchmark dataset. To generalize over a larger number of classes, even when they have unequal or imbalanced representations, such as in the UNSW-NB15 dataset, the patterns identified over the dataset features by the model would need to be more specific and accurate, hence prior feature selection in RAELN-Classifier proves to be beneficial [34].

Table 2. UNSW-NB15 Dataset Accuracy (Training and Testing) in % (10 classes)

Classification Algorithm	Random Forest (supervised)		Naïve Bayes (supervised)		**Ladder Networks (semi-supervised)**		Ladder Networks (supervised)	
Feature Selection Algorithm	Train	Test	Train	Test	Train	Test	Train	Test
Regularized Autoencoder (AE_L2)	81.302	74.744	47.202	41.04	**79.36**	**81.06**	78.2	80.676
FSFC	83.743	65.628	52.663	53.784	82.54	39.552	81.38	79.732

Autoencoder (AE)	83.77	38.506	30.168	41.04	82.72	71.006	80.270	73.526
No Feature Selection	84.865	55.56	45.932	41.04	84.36	64.592	81.672	79.778

**The proposed model, RAELN-Classifier performance results are highlighted in bold*

Table 3. NSL-KDD Dataset Accuracy (Training and Testing) in % - Benchmark Dataset (5 classes)

Classification Algorithm	Random Forest (supervised)		Naïve Bayes (supervised)		**Ladder Networks (semi-supervised)**		Ladder Networks (supervised)	
Feature Selection Algorithm	Train	Test	Train	Test	Train	Test	Train	Test
Regularized Autoencoder (AE_L2)	99.216	70.662	48.700	43.013	**99.906**	**74.709**	98.96	74.678
FSFC	99.60	71.18	63.39	42.991	99.23	74.882	99.531	74.661
Autoencoder (AE)	99.75	71.417	43.713	43.004	99.7	73.106	99.071	74.361
No Feature Selection	99.716	57.72	46.539	43.009	100	76.227	99.729	75.576

**The proposed model, RAELN-Classifier performance results are highlighted in bold*

Table 4. Average Per Class Classification Accuracy on the UNSW-NB15 Dataset (in %) with the AE_L2 Feature Selection Algorithm

Malware Class	RAELN-Classifier (Train)	RAELN-Classifier (Test)	Ladder Networks Supervised (Train)	Ladder Networks Supervised (Test)
Analysis	8.6	0.0	8.0	0.0
Backdoor	0.0	0.0	0.0	0.0
DoS	11.8	22.8	0.7	0.89
Exploits	89.0	76.4	91.9	91.81
Fuzzers	66.4	68.0	77.2	78.8
Generic	97.9	96	97.8	96.7
Benign	87.8	88.0	84.9	88.4
Reconnaissance	69.1	76.3	52.17	58.1
Shellcode	13.4	18.2	12.17	11.62
Worms	0.0	0.0	0.0	0.0
Average	44.328	44.57	42.48	42.632

Moreover, with malware evolving every day, classifying them into broad classes, such as in NSL-KDD, cannot be of much help in malware mitigation process.

The average per class training and testing accuracies obtained by the proposed RAELN-Classifier and baseline fully-supervised ladder network models (with AE_L2 feature selector) on the UNSW-NB15 dataset are presented in Table 4. It can be observed that RAELN-Classifier gives better average per class accuracies (44.328% on training and 44.57% on testing) than the baseline supervised model (42.48% on training and 42.632% on testing). The backdoor and worm classes have 0.0% accuracy due to their minimal representation in the dataset (around 0.01% of the samples) [35-37], which reflected during training, thus explaining why the model was not able to sufficiently generalize over these classes with less representation as well.

Therefore, based on evaluations using both the performance metrics, it can be observed that the proposed combined model of the regularized AE_L2 autoencoder in combination with semi-supervised Ladder Networks

(RAELN-Classifier) gives the best results by ensuring minimum overfitting when compared with other supervised baseline and state-of-the-art classification algorithms both with and without prior feature selection. The associated code and results of the project can be accessed at the following link: https://github.com/PRISHIta123/RAELN

5. CONCLUSION

Malware programs are increasing at an alarming rate that leads to massive monetary and data losses to user bases. It, therefore, becomes imperative to analyse and classify such malware at the earliest. To address the problem, a novel combined model with reduced label dependency is proposed to perform multi-class network intrusion malware classification. From the results obtained on the experiments, it is concluded that the proposed RAELN Classifier, gave better results than state-of-the-art and baseline supervised classification algorithms implemented for the same task, with and without prior feature selection. As future work, other possible semi-supervised approaches such as spectral graph transducer and Gaussian fields for classification, in combination with prior feature selection will be explored and evaluated based on same performance metrics.

AUTHORS' CONTRIBUTIONS

Conceptualization: Prishita Ray, Lahari Anne, Tanmayi Nandan

Methodology: Prishita Ray

Formal analysis and investigation: Prishita Ray, Lahari Anne, Tanmayi Nandan

Writing - original draft preparation: Prishita Ray

Writing - review and editing: Lahari Anne, Tanmayi Nandan, Dr. Kakelli Anil Kumar

Supervision: Dr. Kakelli Anil Kumar

ACKNOWLEDGMENTS

We would like to thank the management and administrative members of Vellore Institute of Technology for giving us the opportunity and required resources to implement the project and prepare this research article.

REFERENCES

[1] Or-Meir, O., Nissim, N., Elovici, Y., & Rokach, L. (2019). Dynamic malware analysis in the modern era—A state of the art survey. *ACM Computing Surveys (CSUR)*, *52*(5), 1-48.DOI: https://doi.org/10.1145/3329786

[2] Piskozub, M., Spolaor, R., & Martinovic, I. (2019). Malalert: Detecting malware in large-scale network traffic using statistical features. *ACM SIGMETRICS Performance Evaluation Review*, *46*(3), 151-154. DOI : https://doi.org/10.1145/3308897.3308961

[3] L. Tan, K. Yu, F. Ming, X. Cheng, G. Srivastava, "Secure and Resilient Artificial Intelligence of Things: a HoneyNet Approach for Threat Detection and Situational Awareness", IEEE Consumer Electronics Magazine, 2021, doi: 10.1109/MCE.2021.3081874.

[4] Seyhan, Kübra, Tu N. Nguyen, Sedat Akleylek, Korhan Cengiz, and SK Hafızul Islam. "Bi-GISIS KE: Modified key exchange protocol with reusable keys for IoT security." *Journal of Information Security and Applications* 58 (2021): 102788.

[5] L. Tan, N. Shi, K. Yu, M. Aloqaily, Y. Jararweh, "A Blockchain-Empowered Access Control Framework for Smart Devices in Green Internet of Things", ACM Transactions on Internet Technology, vol. 21, no. 3, pp. 1-20, 2021,https://doi.org/10.1145/3433542.

[6] Nguyen, Tu N., Bing-Hong Liu, Nam P. Nguyen, and Jung-Te Chou. "Cyber security of smart grid: attacks and defenses." In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1-6. IEEE, 2020.

[7] Pham, Dung V., Giang L. Nguyen, Tu N. Nguyen, Canh V. Pham, and Anh V. Nguyen. "Multi-topic misinformation blocking with budget constraint on online social networks." *IEEE Access* 8 (2020): 78879-78889.

[8] Z. Guo, A. K. Bashir, K. Yu, J. C. Lin, Y. Shen, "Graph Embedding-based Intelligent Industrial Decision for Complex Sewage Treatment Processes", International Journal of Intelligent Systems，2021, doi: 10.1002/int.22540.

[9] H. Li, K. Yu, B. Liu, C. Feng, Z. Qin and G. Srivastava, "An Efficient Ciphertext-Policy Weighted Attribute-Based Encryption for the Internet of Health Things," IEEE Journal of Biomedical and Health Informatics, 2021, doi: 10.1109/JBHI.2021.3075995.

[10] L. Zhen, A. K. Bashir, K. Yu, Y. D. Al-Otaibi, C. H. Foh, and P. Xiao, "Energy-Efficient Random Access for LEO Satellite-Assisted 6G Internet of Remote Things", IEEE Internet of Things Journal, doi: 10.1109/JIOT.2020.3030856.

[11] L. Zhen, Y. Zhang, K. Yu, N. Kumar, A. Barnawi and Y. Xie, "Early Collision Detection for Massive Random Access in Satellite-Based Internet of Things," IEEE Transactions on Vehicular

Technology, vol. 70, no. 5, pp. 5184-5189, May 2021, doi: 10.1109/TVT.2021.3076015.

[12] Z. Guo, K. Yu, A. Jolfaei, A. K. Bashir, A. O. Almagrabi, and N. Kumar, "A Fuzzy Detection System for Rumors through Explainable Adaptive Learning", IEEE Transactions on Fuzzy Systems, doi: 10.1109/TFUZZ.2021.3052109.

[13] Shi, Y., Lei, M., Ma, R., & Niu, L. (2019). Learning robust auto-encoders with regularizer for linearity and sparsity. *IEEE Access*, *7*, 17195-17206. DOI: 10.1109/ACCESS.2019.2895884

[14] Zhu, X., Wang, Y., Li, Y., Tan, Y., Wang, G., & Song, Q. (2019). A new unsupervised feature selection algorithm using similarity-based feature clustering. *Computational Intelligence*, *35*(1), 2-22. DOI: https://doi.org/10.1111/coin.12183

[15] Rasmus, A., Valpola, H., Honkala, M., Berglund, M., & Raiko, T. (2015). Semi-supervised learning with ladder networks. *arXiv preprint arXiv:1507.02672*. DOI: https://dl.acm.org/doi/10.5555/2969442.2969635

[16] Jain, M., Andreopoulos, W., & Stamp, M. (2020). Convolutional neural networks and extreme learning machines for malware classification. *Journal of Computer Virology and Hacking Techniques*, *16*(3), 229-244. DOI: https://doi.org/10.1007/s11416-020-00354-y

[17] Safa, H., Nassar, M., & Al Orabi, W. A. R. (2019, June). Benchmarking convolutional and recurrent neural networks for malware classification. In *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)* (pp. 561-566). IEEE. DOI: https://doi.org/10.1109/iwcmc.2019.8766515

[18] Xiao, G., Li, J., Chen, Y., & Li, K. (2020). MalFCS: An effective malware classification framework with automated feature extraction based on deep convolutional neural networks. *Journal of Parallel and Distributed Computing*, *141*, 49-58. DOI: https://doi.org/10.1016/j.jpdc.2020.03.012

[19] Sharma, S., Krishna, C. R., & Sahay, S. K. (2019). Detection of advanced malware by machine learning techniques. In *Soft Computing: Theories and Applications* (pp. 333-342). Springer, Singapore. DOI: https://doi.org/10.1007/978-981-13-0589-4_31

[20] Gao, X., Hu, C., Shan, C., Liu, B., Niu, Z., & Xie, H. (2020). Malware classification for the cloud via semi-supervised transfer learning. *Journal of Information Security and Applications*, *55*, 102661. DOI: https://doi.org/10.1016/j.jisa.2020.102661

[21] Nadeem, M., Marshall, O., Singh, S., Fang, X., & Yuan, X. (2016). Semi-supervised deep neural network for network intrusion detection. DOI: https://digitalcommons.kennesaw.edu/cgi/viewcontent.cgi?article=1022&context=ccerp

[22] Apao, N.J., Feliscuzo, L.S., Romana, C.L., & Tagaro, J. (2020). Multiclass Classification Using Random Forest Algorithm To Prognosticate The Level Of Activity Of Patients With Stroke. International Journal of Scientific & Technology Research, 9, 1233-1240. DOI: https://www.ijstr.org/final-print/apr2020/Multiclass-Classification-Using-Random-Forest-Algorithm-To-Prognosticate-The-Level-Of-Activity-Of-Patients-With-Stroke.pdf

[23] Ran, J., Ji, Y., & Tang, B. (2019, April). A Semi-Supervised learning approach to IEEE 802.11 network anomaly detection. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)* (pp. 1-5). IEEE. DOI: https://doi.org/10.1109/vtcspring.2019.8746576

[24] Zhang, S., & Du, C. (2020, October). Semi-supervised deep learning based network intrusion detection. In *2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)* (pp. 35-40). IEEE. DOI: 10.1109/CyberC49757.2020.00016

[25] Arazo, E., Ortego, D., Albert, P., O'Connor, N. E., & McGuinness, K. (2020, July). Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.DOI: 10.1109/IJCNN48605.2020.9207304

[26] Moustafa, N., & Slay, J. (2015, November). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 military communications and information systems conference (MilCIS)* (pp. 1-6). IEEE. DOI: 10.1109/MilCIS.2015.7348942

[27] Moustafa, N., & Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective*, *25*(1-3), 18-31. DOI: 10.1080/19393555.2015.1125974

[28] Dhanabal, L., & Shantharajah, S. P. (2015). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *International Journal of Advanced Research in Computer and Communication Engineering*, *4*(6), 446-452. DOI: https://doi.org/10.17148/ijarcce.2015.4214

[29] Deutsch, H. P., & Beinker, M. W. (2019). Principal component analysis. In *Derivatives and internal models* (pp. 793-804). Palgrave Macmillan, Cham. DOI: https://doi.org/10.1007/978-3-030-22899-6_34

[30] Chockwanich, N., & Visoottiviseth, V. (2019, February). Intrusion Detection by Deep Learning

with TensorFlow. In *2019 21st International Conference on Advanced Communication Technology (ICACT)* (pp. 654-659). IEEE. DOI: https://doi.org/10.23919/icact.2019.8701969

[31] Sleeman IV, W. C., & Krawczyk, B. (2021). Multi-class imbalanced big data classification on Spark. *Knowledge-Based Systems*, *212*, 106598. DOI: https://doi.org/10.1016/j.knosys.2020.106598

[32] Yang, Y., Zheng, K., Wu, C., & Yang, Y. (2019). Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network. *Sensors*, *19*(11), 2528. DOI:10.3390/s19112528

[33] Yang, Y., Zheng, K., Wu, C., Niu, X., & Yang, Y. (2019). Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks. *Applied Sciences*, *9*(2), 238. DOI: 10.3390/app9020238

[34] L. Tan, H. Xiao, K. Yu, M. Aloqaily, Y. Jararweh, "A Blockchain-empowered Crowdsourcing System for 5G-enabled Smart Cities", Computer Standards&Interfaces, https://doi.org/10.1016/j.csi.2021.103517.

[35] C. Feng et al., "Efficient and Secure Data Sharing for 5G Flying Drones: A Blockchain-Enabled Approach," IEEE Network, vol. 35, no. 1, pp. 130-137, January/February 2021, doi: 10.1109/MNET.011.2000223.

[36] Subramani, Prabu, K. Srinivas, R. Sujatha, and B. D. Parameshachari. "Prediction of muscular paralysis disease based on hybrid feature extraction with machine learning technique for COVID-19 and post-COVID-19 patients." *Personal and Ubiquitous Computing* (2021): 1-14.

[37] Bhuvaneswary, N., S. Prabu, K. Tamilselvan, and K. G. Parthiban. "Efficient Implementation of Multiply Accumulate Operation Unit Using an Interlaced Partition Multiplier." *Journal of Computational and Theoretical Nanoscience* 18, no. 4 (2021): 1321-1326.