2021 IEEE 6th International Conference on Computing, Communication and Automation (ICCCA)
Aurel Vlaicu University of Arad, Romania. Dec 17-19, 2021

Analysis of Different Dimensionality Reduction Techniques and Machine Learning Algorithms for an Intrusion Detection System

T N Varunram
Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
tn.varunram@gmail.com

Shivaprasad M B
Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
smbenal99@gmail.com

Aishwarya K H
Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
aishwaryakh4@gmail.com

Anush Balraj
Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
anushbalrajkk@gmail.com

Savish S V
Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
savishsv123@gmail.com

Ullas S
Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
s_ullas@blr.amrita.edu

2021 IEEE 6th International Conference on Computing, Communication and Automation (ICCCA) | 978-1-6654-1473-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCCA52192.2021.9666265

***Abstract*—Intrusion Detection Systems are of paramount importance in the present-day network security environment. By analyzing and predicting the behavior of incoming data, IDS helps in classifying it as either a benign or dangerous activity. In this paper, implementation of multiple Intrusion Detection Systems is done using different machine learning algorithms like KNN, AdaBoost, SVM, Logistic Regression, Random Forest, Artificial Neural Networks and Naive Bayes. The class imbalance in the dataset, caused by innate mannerisms of network analysis, is corrected using Synthetic Minority Oversampling Technique (SMOTE). The dataset is then reduced using three different Dimensionality Reduction Techniques namely PCA, t-SNE, and UMAP. The three datasets produced by these techniques are then used to build the Intrusion Detection System, using the machine learning algorithms. The models will implement binary classification to classify the incoming attacks between benign activity and DDoS Attacks.**

Keywords—PCA, t-SNE, UMAP.

I. INTRODUCTION

Over the past decade, the increase of users on the internet and related networks has also led to an increase in the number of network security issues. Vulnerable systems can fall prey to malicious attacks which result in unauthorized access to data, disruption of network traffic and the damaging of functioning devices on the network. These attacks take on various forms, primarily- denying access to services by flooding the network with redundant requests (DDOS) Ref. 20, sending multiple requests to ports to find a vulnerable active port (Port Scan), using malicious software to gain access to a computer (Infiltration), and attacks that are launched through the internet (Web Attacks). In binary classification, the two classes of intrusion are Distributed Denial of Service (DDoS) and Benign.

Intrusions into a network are usually combated by an Intrusion Detection System (IDS). It monitors and analyses all incoming network activity. On discovering a suspicious activity, it alerts the administrator or central units like a Security Information and Event Management system (SIEM) that acts on them immediately. Machine learning is ideal for intrusion detection systems, considering its speed of execution once trained, and its ability to classify multiple classes at a stretch [10][11]. However, visualizing and working on training sets becomes harder when the number of features is higher [9][15]. Therefore, it is necessary to reduce the number of input variables of a dataset through Dimensionality Reduction, which also reduces the execution time and memory space required. The dataset being used is the CICIDS (2017) Friday Working Hours DDoS Attack Dataset which contains real-world data obtained from network traffic analysis

An approach for IDS using AdaBoost and a combination of Feature Selection and Dimensionality Reduction was proposed by Arf Yulianto et. al [1]. They used the CICIDS 2017 dataset for Binary Classification between Benign and DDoS. The dataset was balanced and then the top 25 features were taken as inputs to a PCA model. PCA reduced the dimensionality to 16 and the dataset was used to train an AdaBoost model. Abebe Tesfahun et. al built a Random Forest model and an Information Gain feature selection algorithm for an IDS system built on the KDD CUP '99 dataset [3]. A comparative analysis of IDS systems based on different generic machine learning models was built by T Saranya et. al which again used the KDD CUP '99 dataset [4]. and also, by Sini Raj P et. al [12].

Deep Learning models of different depths were built and analyzed for Binary as well as Multiclass Classification by R Vinaykumar et. al. They concluded that Deep Neural Networks performed much better than Machine Learning models on classical IDS datasets like KDD CUP '99, CICIDS etc. [2]. Vasileios, Nektaria and Georgios made a comparative study between PCA and t-SNE for Dimensionality Reduction technique in Android Malware Detection system and concluded that t-SNE outperformed PCA in terms of preserving data and overall performance [5]. Yuta et.al compared UMAP, PCA and t-SNE in combination with K-Means clustering for large scale SARS Cov2 mutation datasets. They found that a combination of UMAP + K-Means outperformed the other two combinations, inferring that UMAP is better than t-SNE and PCA [6].

Almseidin et.al built multiple machine learning algorithms on the KDD CUP '99 and found out that a J48 classifier works the best on this dataset. Ahmed Ahmim et.al built a multi-class classifier for the CICIDS 2017 dataset with

a hybrid model consisting of REP tree, JRip algorithm and a Random Forest Classifier. A Self Taught Learning Network was built by Quamar Niyaz et.al to classify the different classes present in the NSL KDD benchmark dataset. It involved two Deep Learning Networks, one which learnt about the features in the dataset, and another used for classification. Maleck et. al built a multi-layer feed forward network to classify the different classes in the UNSW-NB15 dataset and found that their model outperforms all existing Machine Learning models used for this dataset.

The rest of this paper is organised as follows, section i, related background speaks about all the different algorithms that have used as well as the dataset and evaluation metrics. this is followed by proposed system in section ii where the workflow is discussed in depth. experimental setup and results are discussed in section iii and conclusion in section iv.

The publications [22], [23] and [24] are analysed to understand the security needs for an IoT system.

II. RELATED BACKGROUND

A. Logistic Regression

Logistic Regression is a statistical learning model based on a logistic function that uses probability computation to infer the association between multiple independent variables, and a single unconditional dependent variable. This technique is borrowed by Machine Learning from the field of statistics. Logistic Regression always works with categorical variables or discrete values and hence is mainly used in Classification Problems. The probability predicted by the model has only two values, false and true. This probability determines the result of the dataset's dependent variable. Logistic regression can be of the type binomial, ordinal or multinomial. Logistic Regression gives rise to the sigmoid curve which uses the sigmoid function, which again takes in real input values resulting in a finite and differentiable function with a derivative greater than zero at each point, and a single inflection point. This function simply converts the independent variable into an expression of probability that ranges between 0 and 1 with respect to the dependent variable.

B. Support Vector Machine

Support Vector Machines (SVM) is a supervised machine learning model that is more frequently used in classification problems. It is also used in Regression analysis. The values of features are plotted as points in an N-dimensional area. The value of these features become the value of a particular coordinate. SVM performs classification based on the best hyper-plane that clearly demarcates the classes. This algorithm focuses on finding the hyper-plane with the highest margin that is one which has maximum distance between its data points [14]. Support Vectors are the vectors that define the position of the hyper-plane and these vectors are the ones that are close to the hyper-plane.

SVM uses a set of mathematical functions which are defined by a parameter called kernel. Kernels are used to transform the input data into a form that is easy for SVM to understand. Regularization is an important parameter for improving the speed, accuracy of convergence and reliability of the model.

C. Naive Bayes

Naive Bayes Model is a family of machine learning algorithms that are based on the Probability and Bayes' Theorems. It uses the principle of conditional probability, which states that the probability of an event depends on it having a few relationships with one or more events that have already occurred. This group of algorithms all makes use of one common assumption that every pair of features which is being classified is independent of each other and also contributes equally towards the outcome. It is usually used when the output variable is discrete in nature. The classifier assumes that all the features are not related to each other.

The membership probabilities are predicted for each class and checks if it belongs to a particular class, and the most likely class or the Maximum A Posteriori (MAP) is the one with the highest probability. The major advantage of Naive Bayes lies in the fact that it does not require much training data. It is capable of working with continuous and categorical data. This algorithm is highly scalable because it has many predictors and data points. It is fast and also can be used in making real-time predictions [13].

D. Artificial Neural Network

An Artificial neural network (ANN) is analogous to a human brain in design and processes the information just like a brain does with connected neurons. ANNs are made of neurons that are stacked up in different layers. These neurons receive inputs from multiple other neurons, which is then multiplied by their assigned weights and added. This sum is later passed onto one or more neurons. An Artificial neural network begins by undergoing a training phase wherein it learns to recognize patterns in data either visually, textually or aurally. Basic components of the neural network include the Input layer which focuses on inputting different types of data into the network. This is followed by several hidden layers that perform different computations to understand the data and finally an output layer used to classify the data. An activation function is often added onto every layer. The activation function is a function in order to help the neural network learn even complex patterns in the data well. ANN has been used widely in IDS in recent years [2][11].

The major catalyst for the ANN to perform accurately is to adjust weights to the right numbers. When more and more training examples are provided the neural network eventually adjusts the weights and tries to connect each input to the correct outputs. In this phase which is like supervised learning, the network actually compares output produced with the expected output. By means of back propagation, the difference that occurs between both these outcomes is adjusted.

E. Adaboost

Adaboost, or Adaptive Boosting, is a machine learning algorithm that utilizes the boosting ensemble method which merges several weak classifiers into a single, strong classifier. Adaboost starts off by assigning equal weights to all data points in the dataset. For a dataset of N observations, the initial weight, w, is calculated as (1/N). Random samples are then classified using the decision tree stumps. After the first classifier is trained, the incorrectly classified records are tallied to calculate the error rate ε_t, which is used to calculate the classifier's output weight α_t.

The output weight is then provided as input to the machine learning model, and is used to identify the wrong classified data points. The weights of wrongly classified data

points are increased, while the ones of correctly classified ones are decreased. The training weights are then updated as a vector of weights using the output of weak classifiers ($h_t(x)$). The weights are calculated and updated over several iterations, so that in the successive models the weak data points are given higher priority. The final classifier output H(x) is then calculated using the output weight and the output of weak classifiers.

F. Random Forest

Random Forest is a supervised learning model based on the bootstrap aggregating ensemble method to compute the result. It works by building an ensemble of uncorrelated decision trees and consolidating the results of these trees based on several features and conditions to arrive at the final result. The uncorrelatedness of the decision trees is an important factor in the model, as diverse trees allow errors of one tree to be overlapped by another. It uses the bagging ensemble method and feature randomness to increase the diversity of the decision trees, allowing them to sample from the dataset randomly with replacement, and constraining the tree to choose from a selected subset of features

G. K Nearest Neighbor.

KNN is the simplest among all supervised machine learning algorithms. It searches for k nearest neighbors or data points and assigns the majority class to the newly added data point. The measure used to calculate the distances, either Manhattan, Euclidean, Minkowski or Hamming distance.

The optimal k value chosen should yield maximum accuracy and least error rate, and it has to be found before building the model. The larger the k value, the more time needed for classification. The datasets that are synthesized by the dimensionality reduction techniques have an optimal k value of two or three. This says that KNN is more compatible for classification.

KNN is a non-parametric model which means it does not rely on the patterns of the dataset, therefore the training time is quick and hence it is a lazy learning method. In training time, the only thing that needs to be done is fitting the training data to the model and in prediction time, the new data is tested among all the training data so it takes time for prediction. The number of records in a dataset matters as larger size leads to more computations and comparisons.

H. Principal Component Analysis

PCA is a linear machine learning algorithm which is widely used for dimensionality reduction of datasets. Reducing dimensions may affect accuracy of the model to some extent but it reduces computational complexity and machine learning algorithms run faster. PCA technique is more sensitive to variance present in the dataset so standardization is done in the first step to keep the values in the range -1 to 1. This has mean as 0 and standard deviation as 1. Covariance matrix is then calculated to give the similarity between two dimensions. A Positive value indicates that the two columns are directly proportional and similar, whereas negative depicts are inversely proportional and the columns are less similar. PCA with dimensionality reduction is to find descriptive features that are more similar and reduce them to a single descriptive feature else it leads to redundancy in the dataset. Eigenvectors are computed from the covariance matrix. From those vectors, Eigenvalues are found out. K highest values of eigenvectors are chosen. They contain most variance and it holds most information from the dataset. The number of Eigenvectors chosen makeup the dimension of the new dataset. A projection matrix is drawn that holds important descriptive features. These values are drawn onto the subspace where principal components come in. More variance is found in the first principal component compared to other components.

I. t-Distributed Stochastic Neighbor Embedding

t -Distributed Stochastic Neighbour Embedding is a Machine Learning algorithm that is used for visualization as well as dimensionality reduction. It is a dimensionality reduction technique which is non-linear and well suited for embedding high dimensional data to a low dimension, usually two or three. It was introduced in 2008 by Laurend van der Maaten [7][8]. The model works by taking data points from the high dimensional data to a lower dimension by packing similar objects together with higher probability while objects that are not similar are packed at a distance with lower probability. The model first computes a probability distribution over pairs of high dimensional data and plots it on a low dimensional graph. This is done by calculating the Euclidean distance between pairs of points and giving higher probability to the points that are nearby and lower probability to the points that are far away. The second part involves creating another probability distribution over the low dimensional graph to decrease the distance between similar objects so as to cluster them more tightly and increase the distance between dissimilar objects so as to differentiate between the different clusters. This is achieved by decreasing the Kullback -Leibler (KL) divergence between all the different probability distributions in the lower dimension. Here we represent the probabilities as a t-distribution instead of normal distribution.

J. Uniform Manifold Approximation and Projection

Uniform Manifold Approximation and Projection was a Dimensionality Reduction technique based on graphs, introduced in 2018 by Leland McInnes. UMAP starts off by finding simplices to capture the local structure of the dataset. Simplices are a simple way to build a k-dimensional geometric object. Any k dimensional object is called a k simplex if it is made by connecting k+1 points. Hence, a 0-simplex is a point, 1-simplex is a line, 2-simplex is a triangle and so on. Once we find these simplices in our dataset, we can then start gluing them along their faces to form a simplicial complex. Initially we treat each and every point in our dataset as a 0-simplex. UMAP constructs more complicated simplexes by extending a radius outward of each 0-simplex. When two such radii overlap, these two points are then treated as neighbors and an edge is drawn between them which indicates the probability that they are connected. UMAP does not arbitrarily define any radius but instead lets the user choose it by having a parameter called n-neighbors. The radius chosen around each point is nothing but the distance between that point and its nth neighbor. In this way, it creates a graph in the high dimensional space, and by using the simplicial complex theory it ensures that the structure of the dataset is well captured. The use of simplicial complexes to build the high dimensional graph gives us something called a topological space. The model then proceeds to reproduce this high dimensional topological space into a lower

dimension while keeping the structure of the data intact using cross entropy.

L. Evaluation Metrics

Evaluation is necessary for checking performance of the model. The performance of the model refers to the accuracy, which indicates how much the model has learnt from all the patterns and mappings in the train dataset. There are several evaluation metrics for describing the performance of the models. The prominent among them being the confusion matrix, it includes several parameters such as True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN). From these parameters, three important measures can be derived- precision, recall and accuracy. In the binary class dataset, there are classes which are classified as positive and negative.

Assume that "yes" is a positive class and "no" is a negative class. True Positive is when the predicted class is "yes" and it is the same as the target class for that tuple in the actual dataset. True Negative is when the predicted class is "no" and it is the same as the target class for that tuple in the actual dataset. False Positive is when the predicted class is "yes" but the target class for that tuple in the actual dataset is "no". False Negative is when the predicted class is "no" but the target class for that tuple in the actual dataset is "yes". The inference from these parameters is that false positive is more crucial than false negative and the model has to reduce false positive value in order to get better performance. Precision tells how much the model has correctly classified positive classes according to total predictions done by it for the positive class. Recall tells how much the model has correctly classified positive classes and the actual class if positive according to the dataset. Accuracy tells how much the model has correctly classified the tuples to its actual class given all predictions done by the model.F1 Score measures the distribution of target classes in the scale of 0 to 1. Value close to 1 indicates there is an equal distribution of positive and negative classes. It is calculated using precision and recall. The metrics used for the models were Accuracy, Precision, Recall an F1 score. The amount of time taken to train each of the machine learning models as well as the time taken to reduce the original dataset using PCA, t-SNE and UMAP were recorded.

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \quad (1)$$

$$F1\ Score = \frac{2*Precision*Recall}{Precision+Recall} \quad (2)$$

III. PROPOSED SOLUTION

For the model, the Friday Afternoon DDoS attack dataset is used which contains 225746 records having 97718 instances of benign traffic and 128027 instances of DDoS attack traffic. The dataset contains 78 descriptive features and 1 target feature. The data type in this dataset is continuous which is easier for the models to work with.

The Dataset is cleaned first by removing all the NA and INF values across the dataset. These values are filled with the mean of that column. The imbalance in the dataset is solved using a library called SMOTE present in the imblearn library of Python. It balances the dataset by up sampling the class that has lower records. After running SMOTE on the dataset, there is equal distribution of classes.

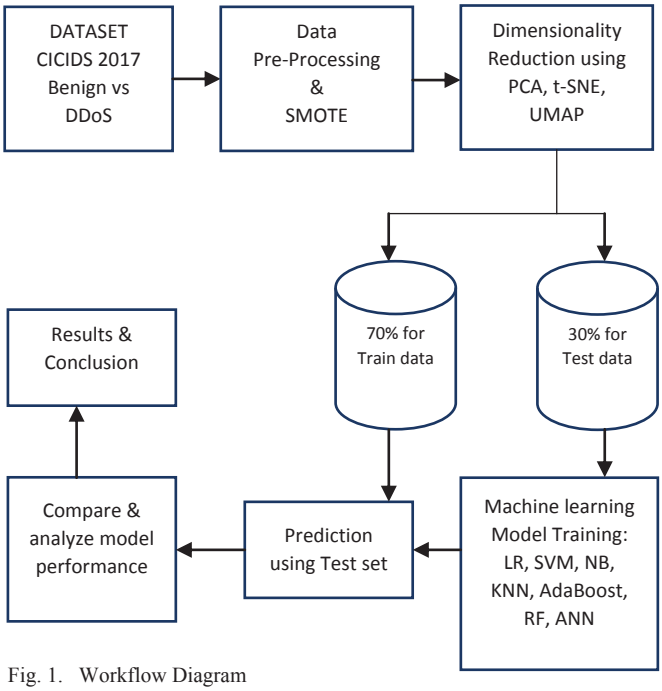

Fig. 1. Workflow Diagram

The third step in the design (refer Fig. 1) is Dimensionality Reduction. Three different datasets using the different reduction techniques namely, PCA, t-SNE and UMAP are generated. This step will end with splitting the reduced dataset into train and test data that effectively measures the performance of the model. The dataset is split in a manner where 70 percent is Train set and 30 percent for testing. The first set is used to train the model, i.e., the model learns how to classify using this set and hence it is called the Train set. Test data is used for evaluating the model's performance. Training is performed using Logistic Regression, SVM, Naïve Bayes, ANN, KNN, AdaBoost and Random Forest Models. Once the training phase with the dimensionality reduced datasets is over, the model is tested on how well it is able to work on the test set. This is done by making the model predict for the test set of data and comparing the predicted results with the ground truth values.

Data is categorized into the two target labels. The target labels in this dataset is Benign and DDoS wherein benign refers to the traffic being safe and free of an attack and DDoS label indicates that an attack has been made on the system. Finally, a comparative study based on the accuracy, F1 score, precision and recall is made where these algorithms are analyzed and tabulated to effectively construe which classifier will work best for Intrusion Detection Systems.

IV. EXPERIMENTAL SETUP AND RESULTS

The first step was to preprocess and normalize the data. Following this the dataset was balanced using the SMOTE algorithm which brought the counts of Benign and DDoS to 128027 each. For comparison and visualization reasons the dataset was reduced from the 78 feature initial dataset to 2 features using all the three dimensionality reduction algorithms.

The Dimensionality Reduction algorithms each ran for varying amounts of time based on the parameters and also the technique that they use. PCA using a matrix algebra based technique stood out as the fastest algorithm between the three, finishing its task at 2 minutes. UMAP and t-SNE were both much slower than PCA due to the huge number of calculations they make. UMAP took around 60 minutes to finish reducing the dataset whereas t-SNE took 90 minutes to finish.

The dataset is split into two sets one to train the model which is the train set and the other where the model must predict with unknown values, the test set. After splitting there were 89452 samples in the train and 38575 samples in the test set. The datasets were then passed into the different classification algorithms that have been proposed. The Precision, Recall, F1 score, Accuracy and the time taken by each of these models during training were recorded. In the table below the time taken column represents the overall time taken by the dimensionality reduction algorithms as well as the model's individual computation time. On an individual basis, Logistic Regression was the fastest algorithm to train, taking around 2 minutes overall to fit the data whereas AdaBoost was the slowest model taking around 150 minutes to finish its computations. KNN, Naive Bayes and Random Forest both took 3 minutes to train whereas Neural Networks and SVM took 10 minutes and 14 minutes respectively.

The results of the work based on the dimensionality reduction algorithms are split into separate tables 1, 2 and 3. Each table gives an account of the classification model's performance on the datasets reduced by specific algorithms.

TABLE I. PERFORMANCE OF MODELS ON PCA REDUCED DATASET

PCA					
Classification Algos	Precision	Recall	F1	Accuracy	Time
Logistic Regression	68.5	69	68.5	68.57	4 min
SVM + RBF	99.022	99.019	99.019	99.019	13 min
Naïve Bayes	70.5	68	68.5	68.37	5 min
ANN	98	98	98	97.96	12 min
AdaBoost	98.84	99.9	99.9	99.87	152 min
Random Forest	99.87	99.87	99.87	99.873	5 min
KNN	99.8	99.81	99.8	99.8	3 min

TABLE II. PERFORMANCE OF MODELS ON T-SNE REDUCED DATASET

t-SNE					
Classification Algos	Precision	Recall	F1	Accuracy	Time
Logistic Regression	55	53	53	53.13	92 min
SVM + RBF	99.971	99.971	99.97	99.971	81 min
Naïve Bayes	62.5	62.5	62	62.47	93 min
ANN	98	98	98	98.316	100min
AdaBoost	99.854	99.845	99.9	99.85	240min
Random Forest	99.96	99.96	99.96	99.975	93 min
KNN	99.98	99.98	99.98	99.98	93 min

TABLE III. PERFORMANCE OF MODELS ON UMAP REDUCED DATASET

UMAP					
Classification Algos	Precision	Recall	F1	Accuracy	Time
Logistic Regression	61	61	61	61.25	62 min
SVM + RBF	98.8218	98.826	98.826	98.826	98 min
Naïve Bayes	61	61	61	61.22	63 min
ANN	94	94	94	93.698	70 min
AdaBoost	99.93	99.91	99.919	99.9	210 min
Random Forest	99.92	99.93	99.93	99.923	63 min
KNN	99.95	99.95	99.95	99.957	63 min

It can be seen from the table that there is very minor performance variance between AdaBoost, Random Forest, KNN and SVM. Logistic Regression and Naive Bayes do not perform well at all in comparison with these four models. Focusing on AdaBoost, SVM, KNN and Random Forest were observed that the performance on each of these algorithms varies slightly between the different datasets as well. SVM, KNN and Random Forest perform the best on the dataset reduced by t-SNE whereas AdaBoost performs the best on the dataset reduced by UMAP. In terms of time taken for dimensionality reduction though, it is seen a clear winner in the PCA reduced dataset, as it takes only 5 minutes to run PCA as well as train a Random Forest classifier with a very good accuracy of 99.87%.

Going by raw accuracy it can be seen that the KNN takes the top spot at an accuracy of 99.98% on the t-SNE reduced data. In terms of time taken as well as the model's performance on the different datasets can be observed that KNN and Random Forest stand above the rest. Though AdaBoost has a performance similar to that of KNN and Random Forest, the time taken to train the AdaBoost is enormous (150 minutes) compared to the time taken by KNN and Random Forest, who each take only 3 minutes.

In the end, the Random Forest + t-SNE, KNN+UMAP and AdaBoost + PCA are considered and a comparison of the performances of these models and the models are done.

TABLE IV. PERFORMANCE COMPARISON WITH PRESENT SYSTEMS

Comparison with Present Systems					
Work	**Our work**	**Our work**	**Our work**	**Arif Yulianto et. al[1]**	**R Vinaykumar et. al[2]**
Method	**RF + t-SNE**	**KNN + UMAP**	**Adaboost + PCA**	**Adaboost + PCA**	**ANN**
Precision	99.96	99.95	99.84	81.69	90.8
Recall	99.96	99.95	99.9	95.76	97.3
F1	99.96	99.95	99.9	88.17	93.9
Accuracy	99.98	99.96	99.87	81.47	96.3

The above table 4 depicts the comparison between the performances of the models with the present system.

On comparing the work with the present systems, it can be seen that the models outperform the best models available for intrusion detection at the moment. Base papers [1] and [2] are used for comparing the work for binary classification and base paper [2] and a reference paper to compare present work for multi class classification.

For Binary Classification it is observed that the models chosen to be the best out of all the models have been built outperform the present systems. Particularly the research implementation of AdaBoost in combination with PCA proves to be better than base paper [1].

V. CONCLUSION

The research aims at finding out the best pair of dimensionality reduction and machine learning algorithms. Lot of pairs were found with very good performance. The outcome of the research work is quite surprised to notice that dimensionality reduction with PCA also gave a very well reduced dataset using classification models like Random Forest, KNN and AdaBoost. The other two Dimensionality Reduction algorithms on the other hand had a small margin of difference in terms of classification performance but the visualizations showed that t-SNE and UMAP can separate the

classes much better than PCA. In the classification front, Random Forest and KNN were the easiest and fastest to train as they train in under a minute. It has been confirmed that Random Forest combined with t-SNE has the best performance. The work can be continued by testing the same approach on a larger dataset for multi class classification as well as try out the same experiment with different newer datasets for intrusion detection that are present in the world today.

REFERENCES

[1] Arif Yulianto , Parman Sukarno and Novian Anggis Suwastika, "Improving AdaBoost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset", The International Conference on Data and Information Science: Journal of Physics: Conf.Series 1192 (2019).

[2] R.Vinayakumar , Mamoun Alazab , K. P. Soman, Prabaharan,Poornachandran, Ameer Al-Nemrat and Sitalakshmi Venkatraman,"Deep Learning Approach for Intelligent Intrusion Detection System", IEEE Access (Volume:7), 10.1109/ACCESS.2019.2895334, April 2019.

[3] Abebe Tesfahun, D Lalitha Bhaskari, "Intrusion Detection using Random Forest Classifier with SMOTE and Feature Reduction", 2013 International Conference on Cloud and Ubiquitous Computing & Emerging Technologies.

[4] T.Saranya, S.Sridevi, C.Deisy, Tran Duc Chung, M.K.A.Ahamed Khan, "Performance Analysis of Machine Learning Models in Intrusion Detection System: A Review", Third International Conference on Computing and Network Communications(CoCoNet'19).

[5] Vasileios Kouliaridis, Nektaria Potha, Georgios Kambourakis, "Improving Android Malware Detection through Dimensionality Reduction Techniques", International Conference on Machine Learning for Networking, 2020

[6] Yuta Hozumi, Rui Wang, Changchuan Yin, Guo-Wei-Wei, "UMAP-assisted K-means clustering of large-scale SARS-CoV-2 Mutation Datasets", Computers in Biology and Medicine Volume 131, April 2021, 104264.

[7] Ranjit Panigrahi, Samarjeet Borah "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems, International Journal of Engineering and Technology.

[8] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour and H. Janicke, "A Novel Hierarchical Intrusion Detection System Based on Decision Tree and Rules-Based Models," 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), 2019, pp. 228-233, doi: 10.1109/DCOSS.2019.00059.

[9] M. Almseidin, M. Alzubi, S.Kovacs and M.Alkasassbeh, "Evaluation of Machine Learning Algorithms for intrusion detection system," 2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY), 2017, pp. 000277-000282, doi: 10.1109/SISY.2017.8080566

[10] Quamar Niyaz, Weiqing Sun, Ahmad Y Javaid, Mansoor Alam, "A Deep Learning Approach for Network Intrusion Detection System", 9th EAI International Conference on Bio-inspired Information and Communications Technologies, 2016.

[11] Malek Al-Zewairi, Sufyan Almajali, Arafat Awajan, "Experimental Evaluation of a Multi-Layer Feed-Forward Artificial Neural Network Classifier for Network Intrusion Detection System", The 2017 International Conference on New Trends in Computing Sciences doi:10.1109/ICTCS.2017.29

[12] [19] Sini Raj P., Divya M., "Comparative study of machine learning algorithms over big data sets", Journal of Advanced Research in Dynamical and Control Systems, Volume - 9, Issue - 11, Page No.72 - 79, Scopus - 2017.

[13] [20] Senthil Kumar .M, Sethuraman M., "Prevention of denial-of-service in next generation internet protocol mobility", Indonesian Journal of Electrical Engineering and Computer Science, Volume 12, Issue 1, 2018, Pages 137-146, Scopus-September -2018, ISSN: 25024752, Publisher: Institute of Advanced Engineering and Science, DOI: 10.11591/ijeecs.v12.i1.pp137-146, Impact Factor: Nil.

[14] [21] K. Mridha, et.al., "Phishing URL Classification Analysis Using ANN Algorithm," 2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON), 2021, pp. 1-7, doi: 10.1109/GUCON50781.2021.9573797.

[15] [22] Rajawat A.S., et.al. (2022) Efficient Deep Learning for Reforming Authentic Content Searching on Big Data. In: Bianchini M., Piuri V., Das S., Shaw R.N. (eds) Advanced Computing and Intelligent Technologies. Lecture Notes in Networks and Systems, vol 218. Springer, Singapore. https://doi.org/10.1007/978-981-16-2164-2_26

[16] [23] Ullas, S., and H. N. Vishwas. "Flow Management and Quality Monitoring of Water using Internet of Things." 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT). IEEE, 2019

[17] [24] Raj, Karre Harshith, and S. Ullas. "Real Time Turbidity Measurement in Sludge Processing Unit by using IoT." 2020 5th International Conference on Communication and Electronics Systems (ICCES). IEEE, 202