Detection of DoS/DDoS attacks: the UBM and GMM approach

Jorge Steven Martinez Osorio*, Jaime Alberto Vergara Tejada†, Juan Felipe Botero Vega‡
University of Antioquia, Medellín, Colombia
Email: [*steven.martinez, †jalberto.vergara, ‡juanf.botero]@udea.edu.co

***Abstract*—There are many different kinds of traditional techniques used for DoS/DDos attack detection, some of them include Artificial Intelligence, IDS, DPI, and most of them are well known and have remained unchanged during the last few years. In this work we implement two novel techniques called GMM and UBM, which are normally used in other scientific or engineering areas, to detect DoS/DDos cyberattacks using a real traffic dataset (CICIDS2017). Three experimental scenarios were implemented, including UBM, GMM, and a Random Forest alternative. The obtained results show new opportunities to use these novel methods in other approaches and explore new solutions to important problems like DoS/DDoS cyber-attack detection, which are closely related to a lot of services on the current Internet. The main goal of this work is to show the potential of UBM and GMM techniques in this simple problem and explore new applications in more complex scenarios.**

***Index Terms*—DoS, DDoS, cybersecurity, networking, machine learning, Gaussian mixture model, universal background model, random forest.**

I. Introduction

The Gaussian Mixture Model (GMM) and the Universal Background Model (UBM) are techniques that are widely used in speech recognition applications as generic probabilistic models for speaker detection or verification [1], [2]. Cybersecurity attacks represent a major issue in data centers, where sensitive information is stored and virtual services are executed; these attacks are commonly detected by IDS, using traditionaltechniques, sometimes based on Machine Learning techniques like Bayesian classifiers.

An important thing to have in mind when facing the problem of cyberattack detection using ML algorithms is to select the correct features of the network traffic. Some authors have explored the problem of attack detection by analyzing the features of the traffic collected from different cyber attacks. In [3], authors use the dataset CICIDS2017 and try to determine the best set of features to detect an attack using common ML methods like Naive Bayes, K Nearest Neighbors, and Random Forests. Comparison of different classification methods is a commonplace in recent works, like in [4] and [5], where traditional IDS techniques are evaluated in contrast to ML and Deep Learning (DL) techniques. There are more novel approaches like [6], where a technique based on a nonparametric cumulative sum is used in order to detect DoS attacks that expose different characteristics of traffic volume in the application layer.

978-3-903176-32-4

Most of reviewed works use traditional ML alternatives, and for this reason, our principal contribution is to propose a different approach that implements GMM and UBM to represent the real traffic of an attack and detect it. In this case, two experiments were performed using the UBM and GMM models. Subsequently, the obtained results were compared with a traditional random forest classifier. For the experiments, we used a dataset provided by the Canadian Institute for Cybersecurity from the University of Brunswick [7].

The rest of the paper is organized as follows: the next section introduces the GMM and UBM concepts and their implementation. Section III explains the characteristics of the dataset, the feature extraction, and the preprocessing applied to the captured traffic. Then, section IV explains the proposed experimental scenarios. Finally, section V shows the comparison between the implemented classification methods and conclusions are depicted in section VI.

II. Background

A. Gaussian Mixture Model (GMM)

A Gaussian Mixture Model is a probabilistic model aiming to depict data as of a Gaussian distribution or a combination of multiple Gaussian distributions [8]. A random variable X follows a Gaussian probability density function (PDF) in one dimension:

$$f(x) \;=\; \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{x-\mu^2}{2\sigma^2}) \doteq N(x;\mu,\sigma^2) \qquad (1)$$

Where μ is the mean and σ is the standard deviation of $\boldsymbol{X}$. This distribution is widely used in different areas as science or engineering and can be used to generalize problems with different nature [9]. The multivariate case of the Gaussian distribution follows the next equation:

$$f(\boldsymbol{x}) \;=\; \frac{1}{\boldsymbol{\Sigma}^{\frac{1}{2}}\;(2\pi)^{\frac{D}{2}}} exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})) \doteq N(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) \qquad (2)$$

Where $\boldsymbol{\mu} \;\epsilon\; \mathbb{R}^D$ is the means vector and it is composed of the mean of each dimension of the random variable $\boldsymbol{X}$. Also, $\boldsymbol{\Sigma} \;\epsilon\; \mathbb{R}^{D\times D}$ is the covariance matrix and represents the

variance of the random variable in a higher dimension.

The GMM can be used to represent data as a linear combination of a number of Gaussian distributions, where, for example, each distribution could represent a subpopulation of the dataset, and by joining all these distributions,a good representation of the whole population can be obtained. The GMM can be generalized from Eq. 2:

$$f(\boldsymbol{x}) = \sum_{m=1}^{M} \frac{c_m}{|\boldsymbol{\Sigma_m}|^{\frac{1}{2}} (2\pi)^{\frac{D}{2}}} exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_m})^T {\boldsymbol{\Sigma_m}}^{-1}(\mathbf{x}-\boldsymbol{\mu_m})) \tag{3}$$

$$f(\boldsymbol{x}) \;=\; \sum_{m=1}^{M} \; c_m \; N(\boldsymbol{x}; \boldsymbol{\mu_m}, \boldsymbol{\Sigma_m}) \tag{4}$$

$$\sum_{m=1}^{M} \; c_m \;=\; 1 \tag{5}$$

Eq. 3 shows that a GMM is the sum of M Gaussian distributions (or multivariate Gaussian distributions), each one of them weighted by the parameter c_m. This parameter must be less or equal than 1 and the sum over all c_m must be equal to 1. The number of Gaussian distributions can be determined in different ways. One alternative is running a grid search, where different GMM are compared according to accuracy results obtained in the training and test steps. On the other side, an Information criterion like the Bayesian Information Criterion (BIC) or the Akaike Information Criterion can be used to understand how well the model represents the data [10].

In order to estimate the parameters $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, and c_m of the distribution, the GMM algorithm uses the Expectation Maximization Algorithm (EM). This algorithm iterates between two steps to calculate and refine the parameters that increase the model's likelihood. In the first step, called the expectation step or E step, the parameters to determine can be randomly initialized or a clustering algorithm can be used, after this probability of a sample belonging to a particular Gaussian m in the iteration j is calculated. Then, in the second step called the maximization step or M step, a new set of parameters $\boldsymbol{\Lambda}(\boldsymbol{\mu_m}, \boldsymbol{\Sigma_m}, \boldsymbol{c_m})$ that maximizes the aforementioned probability is chosen. These two steps will be repeated until the probability difference between two iterations is less than a certain threshold [11], [12].

B. Universal Background Model (UBM)

In a general form, the UBM is a large GMM containing a relatively big set of population features. There are different ways to obtain the final model. One of them, and probably the easiest one, is to generate a large GMM using all the data of the training step. The classes must be balanced to avoid biases over the dominant class. Another way to obtain the final model is to train an individual UBM for each class in the training dataset and then mix all of these subpopulation models. The latter approach has the possibility of using unbalanced data and controlling the composition of the final model [1].

After the UBM is generated, an adaptation method must be used to fit the trained model to the new data and to generate the final model, taking a part of the training data and the trained model. This is done by executing the Maximum *A Posteriori* (MAP) estimation. MAP will be applied over the UBM to estimate a class dependent model [13]. MAP provides an alternative to maximum likelihood estimation for machine learning. Also, MAP tries to estimate the updated distribution and the parameters $\boldsymbol{\Lambda}$ that explain in a better way the population maximizing the *a posteriori* probability without the need of calculating it. According to the Bayes theorem the *a posteriori* probability can be calculated by:

$$P(A|B) = \frac{P(B|A)\; P(A)}{P(B)} \tag{6}$$

Commonly this equation can be referred to as the *a posteriori* probability of A given B and the $P(A)$ is the *a priori* probability of A. $P(B)$ is used to normalize the result, it can be removed and the *a posteriori* now is proportional to the probability of $P(B|A)$ multiplied by $P(A)$.

$$P(A|B) \propto P(B|A)\; P(A) \tag{7}$$

MAP estimation is not interested in calculating the exact value or the *a posteriori* probability, instead, MAP is interested in optimize $P(B|A)\; P(A)$, aiming to estimate the distribution and the parameters $\boldsymbol{\Lambda}$ which could explain in the best way the phenomenon.

$$\textit{maximize}\; P(\boldsymbol{\Lambda}|\boldsymbol{x}) = P(\boldsymbol{x}|\boldsymbol{\Lambda})\; P(\boldsymbol{\Lambda}) \tag{8}$$

C. Random Forest

Random Forest (RF) is a machine learning algorithm composed of predictive models called decision trees which are formed by binary rules where is possible to allocate the samples according to their attributes and predict the value of the response variable. Each decision tree is trained with a slightly different sample in comparison with the other decision trees. These samples with slight differences are obtained by a bootstrapping method that samples the training dataset and assign these samples to the decision trees defined by the random forest model [14], [15].

Random forest obtains a new prediction taking into account the predictions of all the decision trees. RF is capable of choosing the features from the dataset to use in the model automatically. RF is used in regression and classification problems and it does not require a lot of data pre-processing. On the other hand, RF can not be used to extrapolate results with different features in the training step. Also, the model's interpretability can be lost using multiple decision trees. RF will be used as a baseline to compare the performance of GMM and UBM in the attack classification task.

III. Data-set and Feature Extraction

This section explains how was build the dataset starting from the capture data and how was performed the feature extraction with the previous capture.

The dataset used in this work is **"Intrusion Detection Evaluation Dataset (CICIDS2017)"**, it contains benign network traffic and updated common security cyberattacks. The authors ensure that the dataset resembles the true real-world data, and CICFlowMeter was used to extract data from traffic flows [16]. They added 85 statistical features, but more can be included [17], [18], Table I shows some of these features.

Feature Name	Description
Flow Feduration	Duration of the flow in Microsecond
Total FWwd Packet	Total packets in the forward direction
Total Bwd packets	Total packets in the backward direction
...	...
Init Win bytes backward	The total number of bytes sent in initial window in the backward direction
Act data pkt forward	Count of packets with at least 1 byte of TCP data payload in the forward direction

Table I. Features generated by the CICFlowMeter tool.

The dataset was built on the behavior of 25 users generating HTTP, HTTPS, FTP, SSH, and email traffic, these flows are labeled as BENIGN. The dataset also includes attacks like Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet, and DDoS. This work uses only the data that contains DoS/DDoS and normal traffic included in a 12.5GB pcap file. The original dataset has 692703 samples with both traffic labels/classes and 85 features. These features were reduced to 68 deleting features that provide little information. The removed features helped to identify the traffic flow but do not provide information about the network phenomenon.

The dataset has 251712 samples with the label **"DDoS/DoS"** and 439683 samples with the label **"Benign"**. In order to use a balanced dataset to construct unbiased GMM/UBM, 251712 samples were randomly selected from the BENIGN traffic. Next, standardization and a correlation analysis were performed. Some of the features were correlated, so a dimension reduction via Principal Component Analysis (PCA) was conducted, ending with 18 components that contain 77.69% of the variance from the original dataset. The explained variance Analysis is depicted in Fig.1 where the sum over the first 18 components gives the aforementioned variance.

IV. Experimental scenarios

This section will show how was the dataset split in each experimental scenario and what technique was used in each experiment.

For the UBM approach, 50% of the dataset will be used to build the UBM. After this, 30% will be used for training, which includes the generation of GMMs for benign traffic and for attack traffic and the implementation of the MAP algorithm. Finally, the remaining 20% will be used for the testing phase. This is summarized in table II.

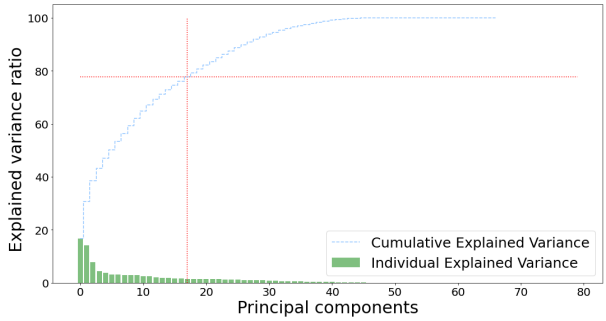


Figure 1. Cumulative Explained Variance applying PCA.

Class	Samples by class	UBM-data samples	Training samples	Test samples
Attack	251712	125954	75337	50422
Benign	251712	125758	75690	50264

Table II. Dataset segmentation for UBM approach.

For the GMM approach, 70% of the dataset is used for training and estimating the parameters of each combination of Gaussians, and 30% is used in the test step, as shown in table III. Both scenarios will rely on the scoring metric included in the sklearn library for classification tasks. The analysis was performed by testing with models that had from 1 to 19 Gaussians in order to compare the results of different combinations of distributions.

Class	Samples by class	Training samples	Test samples
Attack	251712	201399	50313
Benign	251712	201340	50372

Table III. Dataset segmentation for GMM approach.

For the Random Forest experiment, the data is split in the same manner as the GMM experiment. In this case, a search grid is used to determine the best model, changing parameters like n estimators (number of trees in the model), max features (the number of features to consider when looking for the best split), max depth (the maximum depth of the tree), and criterion (the function to measure the quality of a split) [19], giving a total of 144 different models. These parameter will take values as follows:

- n_estimators = {15, 20, 25}
- max_features = {3, 5, 7, 9}
- max_depth = {None, 6, 7, 8, 9, 10}
- criterion = {"gini", "entropy"}

Taking into account the explanation about the three experimental scenarios; the following section shows the performance of each experiment and some conclusions about this work.

V. Performance Evaluation

This section will show the performance for each experiment, mentioning the advantages or shortcomings found in the

implementation of each algorithm, starting with the UBM experiment, followed by the GMM experiment, and finishing with the RF approach. As preliminary conclusions, we found the RF approach worked very well in this classification problem showing better performance than GMM and UBM. GMM performed better than UBM. Finally, the following results show an important first step to use these novel techniques in the networking area.

A. UBM scenario

For this experiment, the accuracy is shown in table IV and fig 2a. After 9 Gaussians overfitting over the attack class appears, whilst with less Gaussians, the accuracy goes from 60% to 77.5%. Table V shows a summary of this experiment, including the precision and sensitivity of the scenario.

Gaussians	1	2	3	4	5
ACC	0.617	0.707	0.655	0.696	0.602
Gaussians	6	7	8	9	10
ACC	0.775	0.400	0.663	0.537	0.494
Gaussians	11	12	13	14	15
ACC	0.544	0.501	0.501	0.501	0.5001
Gaussians	16	17	18	19	
ACC	0.501	0.501	0.501	0.501	

Table IV. Accuracy by gaussian in UBM scenario. The accuracy average is 56.3%.

Gaussian	**Precision %**		**sensitivity %**	
-	Attack	Benign	Attack	Benign
1	57	85	95	28
2	64	87	93	49
3	79	61	42	89
4	76	66	57	82
5	59	62	69	51
6	81	75	72	83
7	35	43	23	57
8	65	68	72	61
9	52	55	67	41
10	50	49	50	49
11	55	54	52	57
12	50	0	100	0
...	...	...	...	...
19	50	0	100	0

Table V. Summary results for UBM.

Additionally, the Confusion matrix for 2, 3, 4, and 6 Gaussians are the best result found.

	Confusion Matrix - 2		Confusion Matrix - 3	
	Attack	Benign	Attack	Benign
Attack	46723	3698	21420	29001
Benign	25820	24444	5758	44496

	Confusion Matrix - 4		Confusion Matrix - 6	
	Attack	Benign	Attack	Benign
Attack	28914	21507	36447	13974
Benign	9106	41158	8666	41598

B. GMM scenario

For this experiment, the accuracy is shown in table VI and fig. 2c. This model shows a good classification of the test dataset in the different combinations from 1 to 19 Gaussians; the optimal point in this experiment could be from 3 to 9 Gaussians, where the accuracy values are good and the model is not too complex or too simple to represent the phenomenon. Table VII shows a summary of this experiment.

Gaussians	1	2	3	4	5
ACC	0.605	0.735	0.770	0.771	0.769
Gaussians	6	7	8	9	10
ACC	0.814	0.826	0.828	0.823	0.833
Gaussians	11	12	13	14	15
ACC	0.823	0.825	0.824	0.828	0.828
Gaussians	16	17	18	19	
ACC	0.833	0.838	0.839	0.843	

Table VI. Accuracy by gaussian in GMM scenario. The accuracy average is 80.3%.

Gaussian	**Precision %**		**sensitivity %**	
-	Attack	Benign	Attack	Benign
1	56	83	95	26
2	77	71	68	79
3	92	70	59	95
4	93	70	59	95
5	93	70	58	95
6	90	76	70	92
7	94	76	70	96
8	94	76	70	95
9	92	76	70	94
10	95	76	70	97
11	93	76	70	95
12	94	76	70	95
13	94	76	70	95
14	95	69	76	96
15	95	69	76	96
16	95	69	76	96
17	96	77	70	97
18	96	77	70	97
19	97	77	71	98

Table VII. Summary results for GMM.

The best confusion matrix for 6, 7, 8, and 9 Gaussians are:

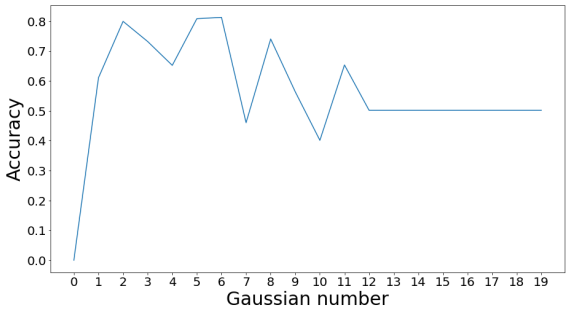
	Confusion Matrix - 6		Confusion Matrix - 7	
	Attack	Benign	Attack	Benign
Attack	35355	14958	34997	15316
Benign	3785	46587	2201	48171

	Confusion Matrix - 8		Confusion Matrix - 9	
	Attack	Benign	Attack	Benign
Attack	35330	14983	35436	14877
Benign	2353	48019	2899	47473

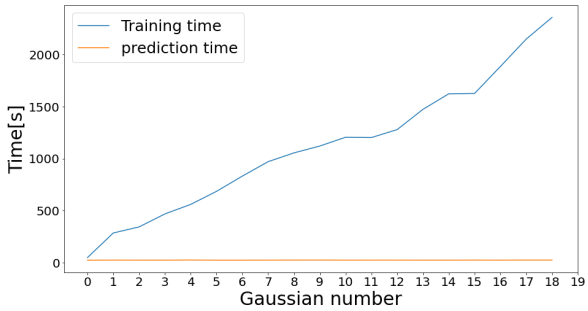
Additionally, figures 2b and 2d show the training time and prediction time on each experiment, this can be useful to understand how much time is required to train and use a model with different number of Gaussians. Actually training time is relatively low, considering the amount of samples, having around 40-500 seconds for the UBM experiment and 200-450 for the GMM experiment.

C. RF scenario

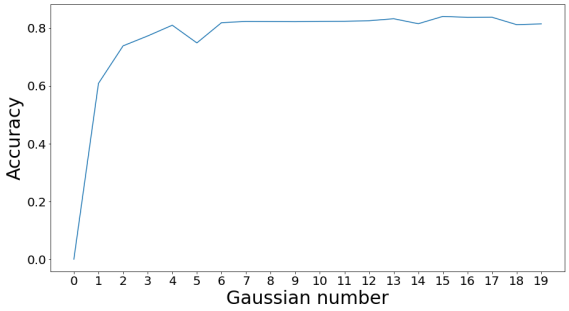
As stated before, in this case, 144 models with a different set of parameters were evaluated. In fig. 2e, the accuracy went from 85.13% up to 96.7% showing better results in comparison with the GMM and UBM scenarios. Table VIII shows 10 of 144 results where the second column is the



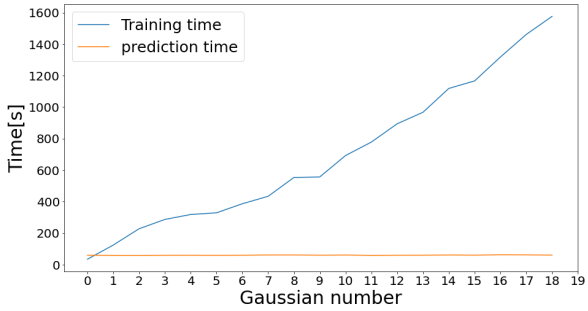
(a) Accuracy results for UBM scenario



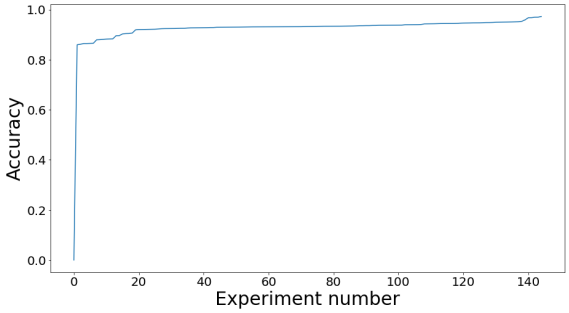
(b) Training time and prediction time for UBM scenario



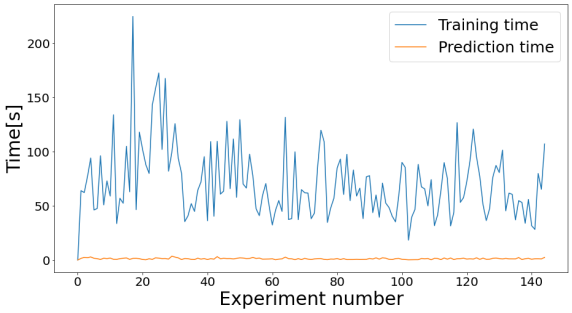
(c) Accuracy result for GMM scenario



(d) Training time and prediction time for GMM scenario



(e) Accuracy results for RF scenario



(f) Training time and prediction time for RF scenario

Figure 2. Accuracies, Training time and Prediction time for experimental scenarios.

accuracy of each result and the next columns depict the chosen parameters to generate each model.

The training time by each model in fig. 2f depends only on the parameters chosen. For the last three models, the training time is around 35-51 seconds, and the prediction time is around 0.79-1 seconds for the RF experiment.

The best three results have the following confusion matrices where the accuracy by each one is 96.1%, 96.2%, amd 96.7% respectively:

	Confusion Matrix - 17		Confusion Matrix - 18	
	Attack	Benign	Attack	Benign
Attack	45315	4998	46772	3541
Benign	196	50176	340	50032

	Confusion Matrix - 19	
	Attack	Benign
Attack	46788	3525
Benign	266	50106

The results of the experiments show that the RF algorithm works very well, obtaining until 96% of accuracy in this classification problem which is not that complicated and the simplicity of this problem may be one reason why the methods GMM and UBM did not work better than RF, but the first steps with the others two methods are really good because this shows an opportunity to improve the GMM and UBM methods and enrich the model to perform more complex classification experiments due to the GMM and UBM techniques worked very well in multiclass classification or multiclass verification problems in areas as speech recognition according to [1], [8]. These methods could show a better performance in problems where the goal is to classify different kinds of cyberattacks or classify the different strategies to deploy the same type of

attack; there the robustness of the GMM or the UBM can be used. Additionally, these results are a good first approximation to improve the experiments and begin to work with these techniques in other domains. The GMM and UBM could work better in more complex scenarios, since these techniques are models that could generalize better a phenomenon. A further work could be a study that includes other databases including different features in order to compare the performance of the GMM and UBM against traditional ML techniques. Training times for UBM and GMM are longer than the RF alternative. These times are not a huge time considering the amount of samples in the dataset and these models could be retrained to improve the model information and its classification without generating long delays in the operation time in comparison with other techniques that could take hours or days in specific cases.

N	**accuracy**	**criterion**	**max depth**	**max features**	**n estimators**
1	0.851	gini	10.0	9	20
2	0.852	gini	10.0	9	25
3	0.853	gini	6.0	9	15
4	0.853	gini	6.0	9	20
5	0.854	gini	6.0	9	25
6	0.948	gini	6.0	5	15
7	0.948	gini	6.0	5	20
8	0.961	gini	7.0	3	15
9	0.962	entropy	7.0	5	15
10	0.967	entropy	6.0	5	15

Table VIII. Summary for 19 results for RF.

VI. CONCLUSIONS

The UBM and GMM are powerful techniques that can be used in many different scenarios such as classification of security attacks in networking where the data to generate the models and probe the algorithms are cheaper. The GMM technique provides better results in the classification for the two-class problem (BENIGN and DDoS), where for 8 Gaussians the obtained accuracy is 82.8% in comparison to the UBM technique where 6 Gaussians led to an accuracy of 77.5%. We found that with more Gaussians the model starts to show overfitting. The accuracy result for the UBM technique can be related to an insufficient capability to explain the phenomenon of the extracted features used to generate the UBM. Is important to remark that the RF model performed better in terms of accuracy compared to the GMM and UBM, however, this work also tries to introduce techniques that work very well in other areas as speech recognition and implements them in the networking area, and more specifically in the cybersecurity area where the traditional techniques do not present relevant changes. These techniques could replace an Intrusion detection System or any other type of detection system in the future.

The training time for the best GMM and UBM models is around 15 minutes. This is a relatively short time taking into account that we are using around half million samples. Prediction time for both models took around 50 seconds to classify 50000 samples. The random forest generated took significatively less time due to RF paralyzed the training stage among 11 cores.

As future work, The GMM and UBM techniques could be implemented to classify more than two classes, e.g. trying to classify each kind of DoS/DDoS attack or trying to model a multi-class problem where the classes could be different kinds of cyberattacks.

ACKNOWLEDGMENT

This paper has been supported by the project “Red temática CYTED 519RT0580” funded by the Ibero-American Science and Technology Program CYTED.

REFERENCES

[1] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker verification using adapted gaussian mixture models,” *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[2] T. Arias-Vergara, J. C. Vásquez-Correa, J. R. Orozco-Arroyave, and E. Nöth, “Speaker models for monitoring parkinson’s disease progression considering different communication channels and acoustic conditions,” *Speech Communication*, vol. 101, pp. 11–25, 2018.

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization.” in *ICISSP*, 2018, pp. 108–116.

[4] A. A. Ghorbani, W. Lu, and M. Tavallaee, *Network intrusion detection and prevention: concepts and techniques*. Springer Science & Business Media, 2009, vol. 47.

[5] S.-N. Nguyen, V.-Q. Nguyen, J. Choi, and K. Kim, “Design and implementation of intrusion detection system using convolutional neural network for dos detection,” in *Proceedings of the 2nd international conference on machine learning and soft computing*, 2018, pp. 34–38.

[6] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, “Detecting http-based application layer dos attacks on web servers in the presence of sampling,” *Computer Networks*, vol. 121, pp. 25–36, 2017.

[7] “Cicids2017,” https://www.unb.ca/cic/datasets/ids-2017.html, accessed: 2021-01-04.

[8] D. A. Reynolds and R. C. Rose, “Robust text-independent speaker identification using gaussian mixture speaker models,” *IEEE transactions on speech and audio processing*, vol. 3, no. 1, pp. 72–83, 1995.

[9] D. Yu and L. Deng, *AUTOMATIC SPEECH RECOGNITION*. Springer, 2016.

[10] C. Rasmussen, “The infinite gaussian mixture model,” *Advances in neural information processing systems*, vol. 12, pp. 554–560, 1999.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[12] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[13] J.-L. Gauvain and C.-H. Lee, “Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains,” *IEEE transactions on speech and audio processing*, vol. 2, no. 2, pp. 291–298, 1994.

[14] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[15] M. Kuhn, K. Johnson *et al.*, *Applied predictive modeling*. Springer, 2013, vol. 26.

[16] “Cicflowmeter,” https://www.unb.ca/cic/research/applications.html, accessed: 2021-01-04.

[17] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, “Characterization of tor traffic using time based features.” in *ICISSP*, 2017, pp. 253–262.

[18] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, “Characterization of encrypted and vpn traffic using time-related,” in *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, 2016, pp. 407–414.

[19] “Random forest classifiers- sklearn,” https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html, accessed: 2021-01-04.