International Journal of Engineering & Technology, 7 (3.33) (2018) 15-22

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

The Performance Comparison of the Classifiers According to Binary Bow, Count Bow and Tf-Idf Feature Vectors for Malware Detection

Young-Man Kwon[1], So-Hee Jun[2], Won-Mo Gal[3] and Myung-Jae Lim[4*]

[1]*Department of Medical IT, College of Healthy Industry, Eulji University, Korea*
[2]*Department of Medical IT, College of Healthy Industry, Eulji University, Korea*
[3]*Department of Environmental Health and Safety, College of Healthy Industry, Eulji University, Korea*
[4]*Department of Medical IT, College of Healthy Industry, Eulji University, Korea*
**Corresponding author E-mail: lk04@eulji.ac.kr*

Abstract

In this paper, we compared the performance of the classifiers according to feature vectors with Binary BOW, Count BOW and TF-IDF for malware detection. We used the feature of Opcode that extracted from PE file. For performance comparison, we measured the AUC score for the classifiers those are DT, KNN, MLP, MNB and SVM. As a result, we recommend neural network (MLP) and instance-based model (KNN) because they show the high AUC score and accuracy regardless of the unbalanced dataset and the feature vector. If you use classical classifiers, we recommend DT because it guarantees high AUC score and accuracy regardless of the same condition as the above. If you use SVM, you have to do Robust scaling to resolved outlier and unbalanced dataset. If you use MNB, you need to use N-gram technique to improve AUC score.

Keywords: *Malware Detection; Feature Selection; Machine Learning; BOW (Bag of words); TF-IDF*

1. Introduction

In the fourth industrial revolution, threat of cyber security is serious problem. One of these threats is the malware called malicious software. This is the programs that invade and damage other computer, server or network on purpose.

There are two classical methods of analyzing and detecting the malware [1]. The signature-based method detects the malware with the database of signatures that collected from the past. However, using stored signatures is hard to deal with transformed or unknown malware. The other method is behavior-based method that detects the malware with monitoring program's behavior by executing it in the virtual environment. Recently, many other additional techniques such as machine learning are applying to detect malware. Kolter get good performance with boosted decision tree using n-grams of bytes codes as feature [2]. Gilbert used CNN for detecting malware with gray scale image that malware file was converted to bytes and used word embedding too [3].

In machine learning tasks, the performance of classifier can be influenced by feature vector that feed to. In this paper, we compare the performance of the classifiers according to the feature vectors which are Binary BOW (Bag of Words), Count BOW and TF-IDF (Text Frequency-Inverse Document Frequency). At first, we extracted the Opcode from PE file as feature and made feature vector with above three methods. Then, we measured the AUC and accuracy score for several classifiers, DT (Decision Tree), KNN (K-Nearest Neighbor), MLP (Multi-Layer Perceptron), MNB (Multinomial Naïve Bayes) and SVM (Support Vector Machine). After experiments, we do ANOVA (Analysis of Variance) analysis to show difference between each classifier's performances according to the feature vectors. We also made additional experiments. The first was made by the scaling feature vectors to see the influence of scale. The second was made by feature vectors for the n-gram of Opcode to see the influence of performance according to the dimension size of input data.

2. Related Works

2.1. Feature Vectors

NLP (Natural Language Processing) is the computational technique for analysing and representing naturally occurring texts [4]. NLP is used in many domains such as machine translations, automatic summarization, sentiment analysis, text classification and so on. In machine learning tasks, the model requires the numeric value as input and output. Therefore, text data need to be represented as numerical value for the input of model. For this reason, many tools of NLP such as BOW (Bag of Words), TF-IDF (Text Frequency-Inverse Document Frequency), Word2Vec are applied. Also, in malware detection domain, Trung Kien Tran used three NLP techniques, TF-IDF, PV-DBOW (Paragraph Vector with Distributed Bag of Words) and PV-DM (Paragraph Vector with Distributed Memory) to represent API sequences and got quite good performance [5]. Gilbert used Word2Vec to represent disassembled Opcode [3]. In this paper, we use Binary BOW, Count BOW and TF-IDF to represent disassembled Opcodes extracted from PE file.

Binary BOW represents the text of document to Boolean value that signify whether the word of vocabulary is presence (1) or not

(0) in each document. Count BOW represents text to numerical value that count word's frequency of occurrence in each document. TF-IDF represents text as weighted numeric value by multiplying TF (Text Frequency) and IDF (Inverse Document Frequency). TF is the count of terms' occurrence and IDF is inversed of document frequency which is degree of how much the word contributes to distinguish the document with others. IDF gets low score when the term occurs in almost every document like stop words. In contrast, if the term occurs in a few documents, it gets high IDF score. Therefore, TF-IDF feature vector reflects the term's importance through calculating occurrences of the term in each document and whole documents. There are many equations for calculating IDF. In our experiment, we used following equation [6].

$$\text{idf}(d, t) = \log\left[\frac{(1+n)}{(1+\text{df}(d,t))}\right] + 1 \quad (1)$$

Where, n is the total number of documents and dt(d, t) is the document frequency that contains term t. Those three methods create the TDM (Term Document Matrix), matrix with terms in vocabulary as the rows and documents names as the columns [7]. The size of feature vectors is number of documents by vocabulary size that extracted from all documents. These feature vectors have no information of words' order in each document.

2.2. Classifier

We measured the performance of classifier according to feature vector to find and utilize the best combination of feature vector and classifier. In our experiment, we used three model-based classifiers, those are DT (Decision Tree), MNB (Multinomial Naïve Bayes) and SVM (Support Vector Machine). We also used instance-based classifier, KNN (K-Nearest Neighbors), and the simplest neural network classifier, MLP (Multi-Layer Perceptron).

2.2.1. Decision Tree

DT is the tree model widely used for classification and regression. It learns the simple decision rules that split the node to predict the target value from features. CART (Classification and Regression Tree) algorithm is used for training DT that split the node in two subsets through a single feature k and its threshold t_k [8]. Following equation is the cost function that CART algorithm tries to minimize.

$$J(k, t_k) = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right} \quad (2)$$

Where, $G_{left/right}$ is the impurity of left and right subset and $m_{left/right}$ is the number of instances in the left and right subset. There are several methods for measuring impurity such as Gini and Cross-Entropy.

2.2.2. K-Nearest Neighbors

KNN is the instance-based model that simply arranges the training data. It classifies the new data with majority vote by number of K-neighbors which are nearest to it. In KNN, there are two things to notice. First, selecting the way to calculate the distance is important because each k instance needs to calculate distance with new data which caused high computational costs. Second, using appropriate value of k depends on situation. If the k is too small, classification can be influenced by outlier. In contrast, if k is too big, it can make low classification performance.

2.2.3. Multi-Layer Perceptron

MLP, that is also called fully connected network, is composed with one input layer, more than one hidden layer and one output layer. Each layer except output layer has bias neuron. Training MLP means training the weights and bias to make final output as the correct answer. It trains the weights in two steps, feed forward and back propagation. In feed forward step, training data is feeds to network and computes the output in each layer for prediction. After model prediction, back propagation step measures the error of output and computes each neuron's error contributions go by previous layers till reach to input layer. That is, it fine-tuned the weights to reduce the error. Each neuron in MLP is applied to nonlinear function such as sigmoid, ReLU (Rectified Linear Unit), hyperbolic tangent and so on. The nonlinear function makes MLP can train with more complex feature than linear model. MLP is quite sensitive with feature scale. It works better when the data have a mean of 0 and, a variance of 1[9]. In our experiments, we used one hidden layer with 100 neurons and ReLU activation function.

2.2.4. Multinomial Naïve Bayes

MNB is the Naïve Bayes classifier for discrete features. Naïve Bayes methods are based on Bayes' theorem with the assumption that all features are independent. It makes simple calculation and can avoid curse of dimensionality. Naïve Bayes train the parameter by combine the statistical probability on each class. The most common Bayesian model in text classification domain is multi-variate Bernoulli model that used binary features and multinomial model that used features such as count of words' occurrence. Depending on size of vocabulary, multi-variate Bernoulli model works better with small vocabulary, multinomial model works better in large vocabulary. However, multinomial model performs better at any vocabulary size in average [10].

2.2.5. Support Vector Machine

SVM is powerful model that can handle many tasks such as linear or nonlinear classification, regression and outlier detection. SVM trains to make maximum margin in decision boundary which is create by few instances of train data that location in between two classes' boundary. Those data points are called support vectors. SVM classifies new instance by calculating the distance between support vectors and new instance. Given a training set of instance-label pairs $(x_i, y_i), i = 1, \ldots, l$ where $x_i \in R^n$ and $y \in \{1, -1\}^i$, SVM require the solution of the following optimization problem [11].

$$\min_{w,b,\xi} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i$$
$$\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \ \xi_i \geq 0. \quad (3)$$

Where, function ϕ map training vector x_i into a higher dimension space. $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is called kernel function. There are many kernel functions such as linear, polynomial, RBF (Radial Basis Function) and sigmoid. We used RBF kernel that used most widely and equation is following.

$$K(x_i, x_j) = \exp(-\gamma \| x_i - x_j \|^2), \ \gamma > 0 \quad (4)$$

SVM is the classifier that sensitive with input data scaling. In SVM, kernel values usually depend on the inner products of feature vectors. Therefore, large scale values can cause numerical problem. Hsu recommend linearly scaling each attribute to the range [-1, +1] or [0, 1] [11].

2.3. Scaling

In machine learning tasks, input data is significant part. Therefore, pre-processing of data such as scaling can influence to the performance of classifier. With few exceptions, Machine learning algorithms do not perform well when the input numerical attributes

have very different scale. In our experiment, we measure the performance of classifier with scaled feature vectors by two different methods, Min-Max and Robust.
The Min-Max scaling, also called normalization, rescaled all value in between 0 to 1 by the following equation.

$$x_i' = \frac{x_i - \min(x)}{\max(x) - \min(x)} \quad (5)$$

Where, x_i is the data value that before scaling and x_i' is after scaling.
The Robust scaling is similar with Min-Max scaling but, it used interquartile range for rescaling that makes algorithm can robust to outliers. The equation is followed.

$$x_i' = \frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)} \quad (6)$$

Where, $Q_1(x)$ is the lower (first) quartile and $Q_3(x)$ is the upper (third) quartile.

2.4. N-Gram

N-gram is the sequence of given data that split into chunks of size N. N-gram is widely used in Natural Language Processing. Willeam used N-gram for text categorization tasks and got high accuracy 99.8% [12]. Also, N-gram is effectively used in malware detection domain. Mikhail used N-gram model to extract essential feature from operation code sequence and construct an N-gram frequency vector [13].

3. The Design of Experiment

We compare the performance of classifiers according to Binary BOW, Count BOW and TF-IDF feature vectors. The overall flowchart of our experiment are shown in fig. 1.

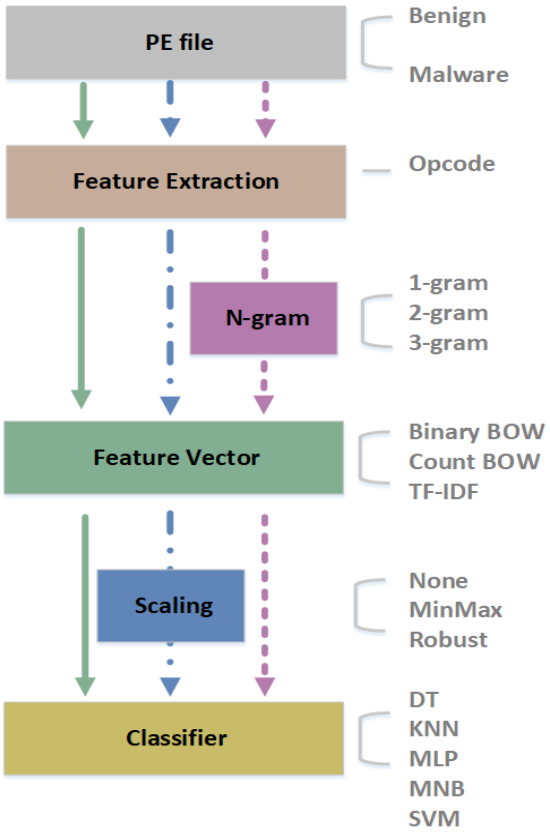

Fig. 1: Overall flowchart of our experiment

In this paper, we used total 1224 PE files as dataset. The benign is 445 files that was extracted from Window file directory. The malware is 779 files got from the Web sites, "Virusshare [14]", "malwareurls jpxeankoret [15]", "malc0de [16]" and "malware-blacklist [17]". The malware files are consisted with 418 Trojan, 176 PUP, 58 Virus, 34 Backdoor, 29 Adware, 21 Downloader, 13 Spyware and so on.
At first, we used pefile module to extract features from PE file and capstone module to extract disassembled Opcode. Then, we made feature vectors with Binary BOW, Count BOW and TF-IDF. Those are used for input of classifiers directly or indirectly with scaling and N-gram technique. Feature vectors are split into train data and test data with ratio of 70 to 30.
We used five classifiers for measuring the performance. Those are DT, KNN, MLP, MNB and SVM classifiers that offer from Scikit-learn library in python. DT, MNB and SVM are model-based classifiers. But, the KNN is instance-based classifier and MLP is the simplest neural network classifier. These two classifiers are selected to compare with model-based classifiers. During experiment, we used k=3 in KNN classifier. In the case of MLP, we used one hidden layer with 100 neurons.
In our experiment, we measured AUC (Area Under the Curve) score and test accuracy for each classifier. First, we measured the AUC score of classifiers. The AUC score is summary of the ROC (Receiver Operating Characteristic) curve which is TPR (True Positive Rate) against FPR (False Positive Rate). When the ROC curve is close to the top of graph, the AUC score got higher which is ideal score because it means that classifier got low FPR and high TPR. The AUC score is between 0 to 1 and 1 is the best score. No matter how the dataset is unbalanced, the AUC score is 0.5 with random prediction [18]. In many cases, accuracy score is used as criteria of classifier's performance. However, Ling showed that AUC score which concerned probability estimation of the classification is statistically consistent and more discriminating measure than accuracy score for balanced or unbalanced dataset, as a measure for learning algorithms [19]. Therefore, we will focus on AUC score for classifier's performance comparing according to the feature vectors. Then, in addition, we measured the accuracy score with test data that classifiers never seen.
For each AUC and accuracy measurement, we made experiments with different random seed 30 times to get reliable result. After that, we visualized the results with the box and whisker plot and calculated the average and standard deviation. The box and whisker plot, also called box plot is used for displaying the distribution of data. It shows the minimum, first quartile, median, third quartile and maximum value of data. The minimum and maximum are represented below and above the box, the median is represented inside of box as line. From minimum to maximum shows the full range of variation. From first quartile to third quartile shows the IQR (Interquartile Range). The suspected outlier and the outlier are represented unfilled circle and filled circle at the outside of minimum and maximum value [20].
By using the result, we did ANOVA (Analysis of Variance) analysis to know whether the difference of the performance is or not. ANOVA is the statistic method for the mean comparison of multiple groups. In the case of significant result of ANOVA, we did Post-Hoc test. Through those statistical analysis, we conduct the analysis of result.
Second, we also did additional experiment by scaling feature vectors. Because it is known that SVM and MLP classifier are sensitive with input data scaling. In our experiment, we used scaled feature vectors by Min-Max scaling and Robust scaling.
Third, we also did additional experiment N-gram feature vectors to see the performance according to size of input dimension. We did experiment with size of N that is 2 and 3.
It is important to denote the three different experiments by different shape of arrow in the flow chart of experimental design that is shown in fig. 1.

4. Experiment and Results

In this paper, we made three different experiments to compare the performance of classifier according to Binary BOW, Count BOW and TF-IDF feature vectors. The first experiment was carried with just basic feature vectors, second experiment was done with scaled feature vectors and third experiment with N-gram feature vectors.
We run the 30 times experiment for the all case to analyze the result statistically. After that, we measure the average and standard deviation for each classifier's AUC score and accuracy score. We also show the box plot of them.

4.1. Basic Feature Vectors

First, we made experiment with Binary BOW feature vector which is made with Boolean values that represent presence of each feature. The results of experiment with Binary BOW feature vector are shown in table 1 and fig. 2.

Table 1: The average and standard deviation of AUC score and accuracy in Binary BOW feature vectors

Feature Vector	Classifier	AUC		Accuracy (%)	
		AVG	STD	AVG	STD
Binary BOW	DT	0.942	0.011	94.755	1.048
	KNN	0.972	0.006	95.851	0.658
	MLP	0.989	0.005	96.232	1.045
	MNB	0.947	0.012	77.699	2.953
	SVM	0.968	0.010	93.306	0.988

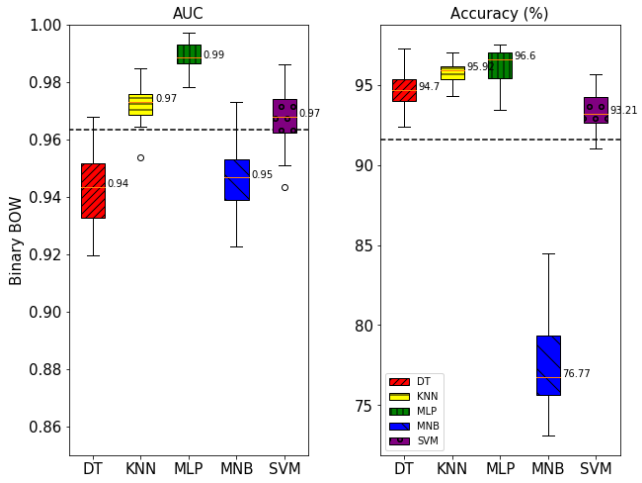

Fig. 2: Box plot of AUC score and accuracy with Binary BOW feature vector

In fig. 2, the horizontal dotted line in each plot represents the average value of all five classifiers' performance. And the annotated value in each box plot is median value of 30 results. That is different with average value of table.
In table 1 and fig. 2, the descending rank of AUC score is like MLP, KNN, SVM, MNB and DT. We analyze the AUC score with ANOVA and Post-Hoc Analysis by TuKey's HSD (Honestly Significant Difference) test with 0.05 significant level. As a result, we found 3 group. The first group is MLP. The second is KNN and SVM. The third is DT and MNB. MLP got the best AUC score.
Generally, it is known that the neural network is outperformed than traditional classifiers because it controls the capacity to avoid overfitting with largest capacity such as minimize generalization error while training [21]. In accuracy score, MLP still got the best score and MNB got the lowest score with striking gap against other classifiers. We can see from this result, that the neural network (MLP) is the best and instance-based classifier (KNN) is good as neural network. In the case of classic classifier, SVM is relatively better than DT and MNB.

In the case of DT, we can see the low AUC score and large variance of it. Tree-based algorithm just split the node of it to 2 subsets by threshold value calculated from dataset. Therefore, decision rule can be different with each experiment because the train and test data are changed with different random seed. This means tree-based algorithm is influenced sensitively by the dataset. To reduce this high variance, there are ensemble algorithms such as random forests that used several decision trees.
In the case of MNB, it got quite high AUC score but low accuracy score with similar variance. It means that MNB classified positive data as good but does not good in ratio of whole classification. We can guess the reason that our dataset is quite biased to negative class dataset.
Next, we made experiments with Count BOW and TF-IDF feature vector for comparison with Binary BOW feature vector's result. The results are shown in table 2 and fig. 3.

Table 2: The average and standard deviation of AUC score and accuracy in Count BOW and TF-IDF feature vector

Feature Vector	Classifier	AUC		Accuracy (%)	
		AVG	STD	AVG	STD
Count BOW	DT	0.960	0.012	96.467	0.890
	KNN	0.976	0.010	96.476	0.976
	MLP	0.964	0.009	90.897	2.776
	MNB	0.911	0.021	75.833	7.632
	SVM	0.962	0.006	82.681	3.296
TF-IDF	DT	0.953	0.011	95.634	0.933
	KNN	0.978	0.007	96.984	0.727
	MLP	0.984	0.006	96.757	0.749
	MNB	0.948	0.015	92.527	0.959
	SVM	0.918	0.018	63.641	2.302

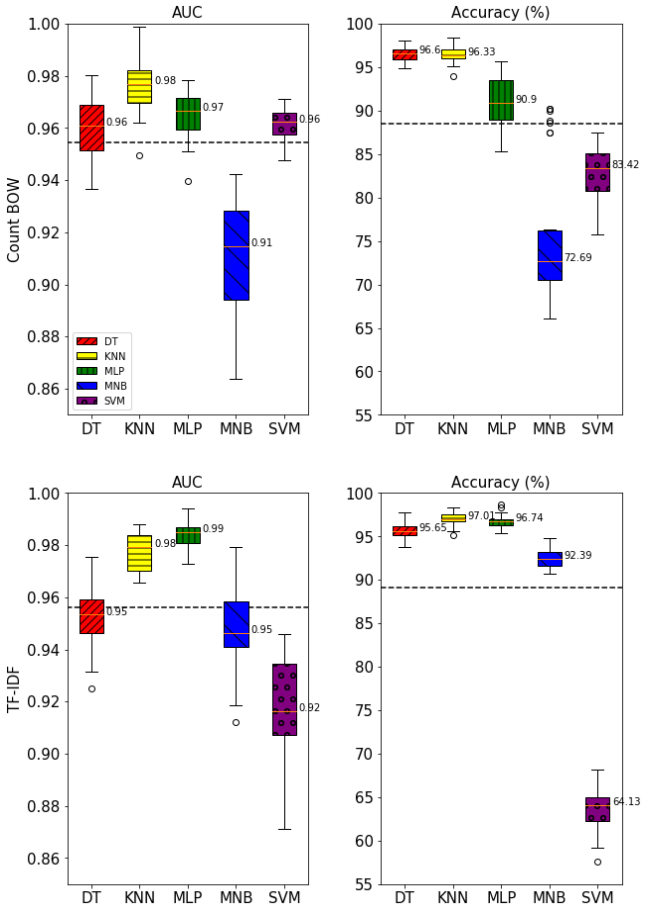

Fig. 3: Box plot of AUC score and accuracy score with Count BOW and TF-IDF feature vector

In table 2 and fig. 3, the descending rank of AUC score is like KNN, MLP, DT, SVM and MNB. We found 3 group as the result of ANOVA and Post-Hoc Analysis. The first group is KNN. The

second group is DT, MLP and SVM. The third group is MNB. From this result, we can see the instance-based classifier (KNN) is the best. MLP still got good performance. In the case of classical classifier, KNN and SVM are good as neural network. If we consider this result with accuracy, KNN is only good. MNB is the worst.

In the case of this experiment, MLP got the lower ranking in the descending rank of AUC score than previous experiment. Also, we can see the variance of AUC score is visibly larger if we compare to the variance of the Binary BOW feature vector. Generally, it is known that MLP is sensitive with input data scale as referred earlier. To see it, we did additional experiment with scaled feature vector in following section 4.2. In the case of DT, it got better performance than Binary BOW feature vector but still got high variance. In the case of MNB, it got low AUC score and high variance.

Now, let us see the result of TF-IDF feature vector. TF-IDF feature is to give high weight to any term that appears often in a particular document, but not in many documents in the corpus. The descending rank of AUC score is like MLP, KNN DT, MNB and SVM. We found 3 groups as the result of ANOVA and Post-Hoc Analysis. The first group is KNN and MLP. The second is DT and MNB. The third is SVM.

From this result, we can see the neural network (MLP) and instance-based classifier (KNN) is the best. Because the TF-IDF value is the lower min-max value than Count BOW, we think this experiment was done with a little scaling effect of input data. So, we found that MLP is better than the KNN in contrast with Count BOW feature vector. In the case of classical classifier, DT and MNB are quite good and SVM is the worst. Especially, SVM got the lowest AUC score and accuracy with large gap against other classifiers.

According to the results of experiment with three feature vectors, we can see that MLP and KNN generally got good performance over all three feature vectors. Even though MLP is very simple, the performance of it is the best if we scale the input data. Throughout all feature vector the best average AUC score is 0.989 with Binary BOW feature vector in MLP. The best average AUC score is 0.978 in the case of instance-based classifier KNN with TF-IDF feature vector. In the case of model-based classifier, we recommend DT because it give us the good performance over three types of feature vectors.

4.2. Scaled Feature Vector

Through the first experiment, we found that scale of input data is very important because the performance of classifier can be influenced by it. We already said to this especially in the MLP and SVM. So, we did second experiment to see the tendency of performance according to scaling feature vectors.

In the real-life challenge, test data is unknown data for measuring the degree of how much classifier trained well. For this reason, we trained the scaler with train data and just applied it to test data for test data scaling. Therefore, test data is scaled with typical value which was set from train data.

In our experiment, we used two different methods, Min-Max and Robust. Those are from Scikit-learn library in python.

Let us see the experimental results with scaled Binary BOW feature vector are shown in the below.

Table 3: Average and standard deviation of AUC score and accuracy with scaled Binary BOW feature vector

Feature Vector		Classifier	Scaling	AUC			Accuracy (%)	
				AVG	STD		AVG	STD
Binary BOW		DT	M	0.942	0.011		94.755	1.048
			R	0.942	0.011		94.755	1.048
		KNN	M	0.972	0.006		95.851	0.658
			R	0.973	0.006		95.788	0.627
		MLP	M	0.989	0.005		96.250	1.025
			R	0.992	0.004		96.821	0.888
		MNB	M	0.947	0.012		77.690	2.948
			R	-	-		-	-
		SVM	M	0.968	0.010		93.306	0.988
			R	0.968	0.010		93.297	0.986

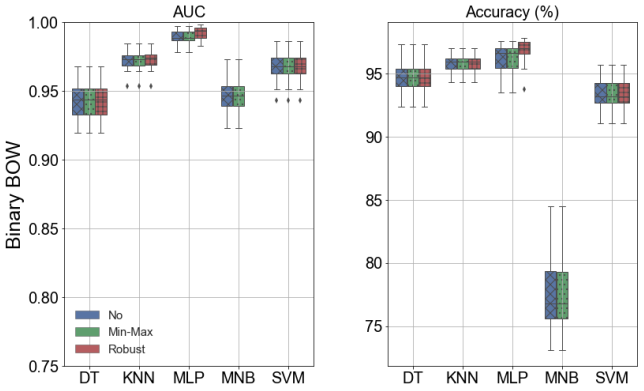

Fig. 4: Box plot of AUC score and accuracy with scaled Binary BOW feature vector

In table 3, each value in 'scaling' column that 'M' means Min-Max scaling and 'R' means Robust scaling. In fig. 4, each three different box plot represents the results that are applied to different scaling methods individually for each classifier. The first box plot within classifier used feature vector that is not scaled, the second used Min-Max scaling and the third used Robust scaling. Note that there is not Robust scaling in the case of MNB since MNB does not allow the negative value as input.

Let us compare the results, those are not scaled, scaled with Min-Max and robustly scaled in table 3 and fig. 4. Then, we can see that there is no significant difference according to scaling input in the case of Min-Max scaling with Binary BOW feature vector. We think it is the reason why the value of Binary BOW feature vector is already 0 or 1. But, In the case of Robust scaling with Binary BOW feature vector, MLP only improved the AUC score with scaling.

We found that the overall AUC score and accuracy of neural network (MLP) and instance-based (KNN) are outperformed than model-based classifiers. Let us consider these conclusion again when we see the result of subsequent experiments.

Next, we measured the performance with scaled Count BOW and scaled TF-IDF feature vector. The results are shown in table 4 and fig. 5.

Table4: Average and standard deviation of AUC score and accuracy with scaled Count BOW and TF-IDF feature vector

Feature Vector		Classifier	Scaling	AUC			Accuracy (%)	
				AVG	STD		AVG	STD
Count BOW		DT	M	0.960	0.012		96.467	0.890
			R	0.960	0.012		96.476	0.894
		KNN	M	0.946	0.012		91.848	1.219
			R	0.969	0.009		93.986	1.131
		MLP	M	0.952	0.009		91.866	1.237
			R	0.963	0.009		94.665	0.886

		MNB	M	0.840	0.027		75.688	2.449
			R	-	-		-	-
		SVM	M	0.917	0.014		63.705	2.170
			R	0.971	0.008		90.761	1.029
TF-IDF		DT	M	0.953	0.011		95.634	0.933
			R	0.953	0.011		95.634	0.933
		KNN	M	0.982	0.006		97.355	0.673
			R	0.964	0.008		94.402	0.878
		MLP	M	0.977	0.009		96.141	0.872
			R	0.981	0.009		96.386	1.396
		MNB	M	0.962	0.012		85.707	1.806
			R	-	-		-	-
		SVM	M	0.951	0.010		80.389	1.790
			R	0.980	0.007		94.475	0.831

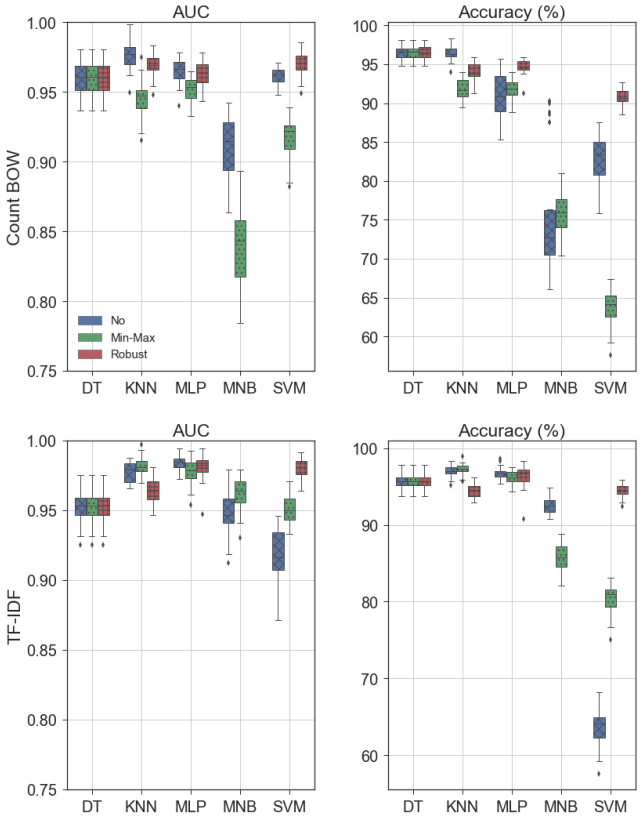

Fig. 5: Box plot of AUC score and accuracy with scaled Count BOW and TF-IDF feature vector

In the case of Min-Max scaling with Count BOW feature vector, the AUC score become lower than not scaled feature vector. So this is not recommended.
In the case of Robust scaling with Count BOW feature vector, the AUC score become lower than not scaled feature vector except SVM. We can guess the reason that SVM is sensitive with outlier but it can be dealt well with robust scaling.
In this case of scaling with Count BOW, the not scaled feature vector is better than the scaled feature vectors. The overall AUC score and accuracy about neural network (MLP) and instance-based (KNN) still are outperformed than model-based classifiers. If you must use SVM with count BOW, we recommend Robust Scaling
In the case of Min-Max scaling with TF-IDF feature vector, the AUC score is higher than not scaled feature vector except MLP. So this is recommended.
In the case of Robust scaling with TF-IDF feature vector, the AUC score is lower than not scaled feature vector except SVM as result with scaled Count BOW feature vector.
In the case of scaling with TF-IDF feature vector, the overall AUC score and accuracy about neural network (MLP) and instance-based (KNN) are outperformed than model-based classifiers as the conclusion of Count BOW feature vector.

4.3. N-Gram Feature Vector

There are other things that can influence to the classifier's performance such as dimension of input data. Therefore, we did second additional experiment to show difference of performance according to dimension of data in each feature vectors.
An N-gram is the continuous sequence of features that each sequence is composed with size N. In malware detection domain, N-gram is effective because malware has sequence pattern for executing invasion or damage. Asaf used N-gram pattern of Opcode for malware detection and got improved results [22]. However, when the size of N is increased, the dimension of features is also increased. High dimension input required high computational cost and can influenced to the performance of classifier. We do experiment for how the size of input dimension impacts to each classifier's performance according to the feature vectors.
First, we will see the result with 2-gram and 3-gram Binary BOW feature vector in table 5 and fig. 6.

Table 5: Average and standard deviation of AUC score and accuracy with N-gram Binary BOW feature vector

Feature Vector		Classifier	N-gram	AUC			Accuracy (%)	
				AVG	STD		AVG	STD
Binary BOW		DT	2	0.952	0.017		95.861	1.396
			3	0.961	0.013		96.504	1.084
		KNN	2	0.983	0.007		97.364	0.837
			3	0.980	0.009		97.446	0.799
		MLP	2	0.994	0.003		98.315	0.508
			3	0.994	0.003		98.741	0.446
		MNB	2	0.982	0.005		88.659	1.559
			3	0.983	0.005		94.565	1.078
		SVM	2	0.983	0.005		93.542	0.919
			3	0.983	0.005		93.696	0.866

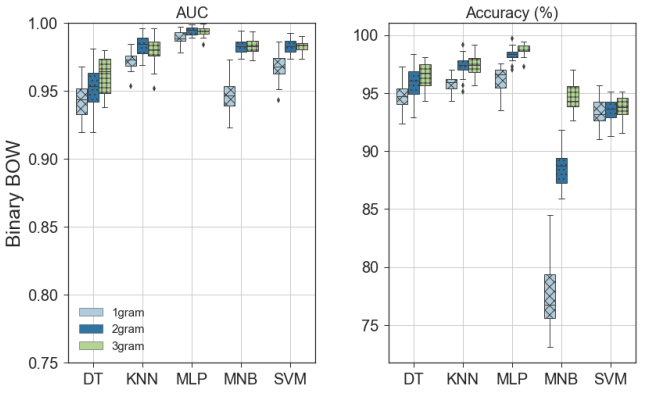

Fig. 6: Box plot of AUC score and accuracy with N-gram Binary BOW feature vector

In table 5, the value of column named 'N-gram' means the size of N-gram. In fig. 6, each three different box plot of each classifier represents the results that are used different size of N-gram. The first box plot of classifier used 1-gram which is basic feature vector, the second used 2-gram and the third used 3-gram.
In the case of 2-gram with Binary BOW feature vector, the AUC score is higher than 1-gram feature vector in every classifiers. In

the case of 3-gram with Binary BOW feature vector, the result is same with above case.
Also, when the size of N-gram is increased, the AUC score is also increased in almost every classifier. For MNB, it got outperformed results with large size of N because it works well in large size of vocabulary [10].
In this case of N-gram Binary BOW feature vector, the overall AUC score and accuracy about neural network (MLP) and instance-based (KNN) are outperformed than model-based classifiers with large size of N. This means that we can improve the performance with Binary BOW feature vector by applying N-gram technique.
We measured the performance with N-gram Count BOW feature vector and N-gram TF-IDF feature vector for comparison with Binary BOW's result. The results are shown in table 6 and fig. 7.

Table 6: Average and standard deviation of accuracy and AUC score with N-gram Count BOW and TF-IDF feature vector

Feature Vector	Classifier	N-gram	AUC		Accuracy (%)	
			AVG	STD	AVG	STD
Count BOW	DT	2	0.952	0.016	95.842	1.363
		3	0.962	0.011	96.667	0.890
	KNN	2	0.983	0.007	97.364	0.860
		3	0.980	0.009	97.446	0.799
	MLP	2	0.994	0.003	98.297	0.496
		3	0.994	0.003	98.732	0.417
	MNB	2	0.982	0.005	88.650	1.567
		3	0.983	0.005	94.565	1.078
	SVM	2	0.983	0.005	93.542	0.919
		3	0.983	0.005	93.696	0.866
TF-IDF	DT	2	0.957	0.011	96.268	0.882
		3	0.965	0.010	96.957	0.863
	KNN	2	0.982	0.005	97.237	0.842
		3	0.984	0.005	97.437	0.774
	MLP	2	0.995	0.002	98.333	0.528
		3	0.996	0.002	98.759	0.395
	MNB	2	0.985	0.004	94.031	1.030
		3	0.987	0.003	95.815	0.660
	SVM	2	0.986	0.005	63.632	2.164
		3	0.991	0.002	63.632	2.164

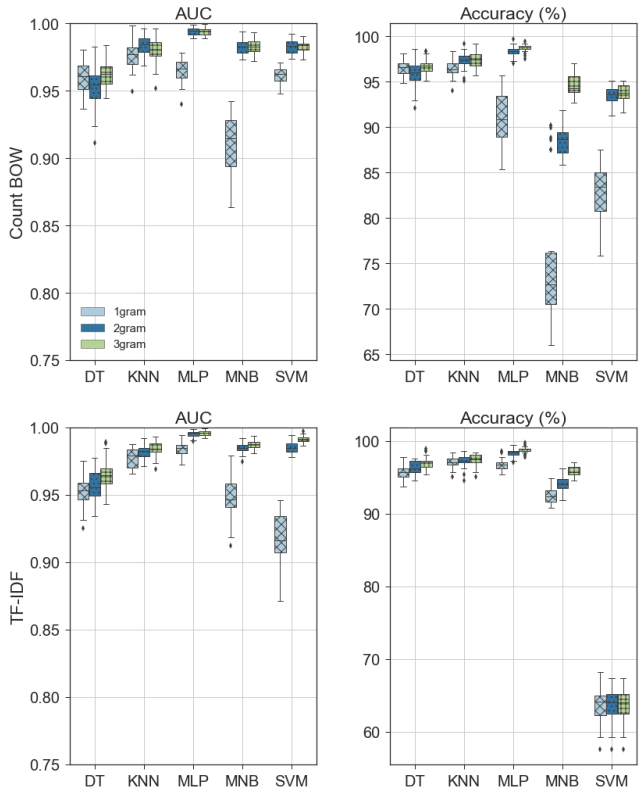

Fig. 7: Box plot of accuracy and AUC score with N-gram Count BOW and TF-IDF feature vector

In the case of 2-gram with Count BOW feature vector, almost every classifier got better performance than 1-gram with Count BOW feature vector except for DT. In the case of 3-gram with Count BOW feature vector, the AUC score is higher than 1-gram with Count BOW feature vector in every classifier. So, we recommend that N-gram technique be applied to the Count BOW feature vector.
For MNB, it still got outperformed performance when the size of N-gram is increased. This results are similar with N-gram Binary BOW feature vector.
In this case N-gram with Count BOW feature vector, the overall AUC score and accuracy about neural network (MLP) and instance-based (KNN) are outperformed than model-based classifiers.
In the case of 2-gram and 3-gram with TF-IDF feature vector, the AUC score become higher than 1-gram with TF-IDF feature vector.
Again, the overall AUC score and accuracy about neural network (MLP) and instance-based (KNN) are outperformed than model-based classifiers.

5. Conclusion

In this paper, we studied the performance comparison of the classifiers according to the Binary BOW, Count BOW and TF-IDF feature vectors. We measured the AUC score and accuracy with each feature vector.
We recommend neural network (MLP) and instance-based model (KNN) because they show the high AUC score and accuracy regardless of the unbalanced dataset and the feature vector forms (Binary BOW, Count BOW and TF-IDF). If you use classical classifier, we recommend decision tree because it guarantees high AUC score and accuracy regardless of the same condition as the above. If you use SVM, you have to do Robust scaling to resolve outlier and unbalanced dataset. If you use MNB, you need to use N-gram technique to improve AUC score.

Acknowledgement

This research was supported by R.O.K. National Research Foundation under grant NRF-2017R1D1A1B03036372. We thanks our colleagues from laboratory who provided insight and expertise that greatly assisted the research. And thanks to National Research Foundation who supported laboratory.

References

[1] Ashwini Mujumdar, Gayatri Masiwal, DR. B. B. Meshram, "Analysis of Signature-Based and Behavior-Based Anti-Malware Approaches," International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), Volume 2, Issue 6, June 2013
[2] J.Zico Kolter, Marcus A. Maloof, "Learning to Detect and Classify Malicious Executables in the Wild," The Journal of Machine Learning Research, Volume 7, December 2006, pp: 2721-2744
[3] Daniel Gilbert, "Convolutional Neural Networks for Malware Classification," October 2016
[4] Elizabeth D. Liddy, "Natural Language Processing," In Encyclopedia of Library and Information Science, Volume 2, NY.Marcel Decker, 2001
[5] Trung Kien Tran, Hiroshi Sato, "NLP-based Approaches for Malware Classification from API Sequences," Asia Pacific Symposium on Intelligent and Evolutionary Systems, 2017
[6] Python Library, scikit-learn, TfidfTransformer, http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer
[7] Manohar Swamynathan, *Mastering Machine Learning with Python in Six Steps*, Apress, 2017, pp: 268-272

[8] Aurelien Geron, *Hands-On Machine Learning with Scikit-Learn & TensorFlow*, O'REILLY, 2017, pp: 167-179
[9] Sarah Guido, Andreas Muller, *Introduction to Machine Learning with Python*, O'REILLY, 2016, pp: 104-119
[10] Andrew McCallum, Kamal Nigam, "A comparison of Event Models for Naïve Bayes Text Classification," AAAI Workshop, 1998, pp: 41-48
[11] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, "A Practical Guide to Support Vector Classification," 2016
[12] Willeam B.Carnar, John M.Trenkle, "N-Gram-Based Text Categorization," In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, 1994, pp: 161-175
[13] Mikhail Zolotukhin, Timo Hamalainen, "Detection of Zero-day Malware Based on the Analysis of Op-code Sequences," The 11TH Annual IEEE CCNC – Security, Privacy and Content Protection, 2014
[14] virusshare, https://virusshare.com
[15] joxeankoret, http://malwareurls.joxeankoret.com
[16] malc0de, http://malc0de.com
[17] malwareblacklist, http://www.malwareblacklist.com
[18] Sarah Guido, Andreas Muller, *Introduction to Machine Learning with Python*, O'REILLY, 2016, pp: 292-296
[19] Charles X. Ling, Jin Huang, Harry Zhang, "AUC: a Better Measure than Accuracy in Comparing Learning Algorithms," Part of the Lecture Notes in Computer Science book series (LNCS), Volumne 2671, May 2003
[20] Box Plot: Display of Distribution, http://www.physics.csbsju.edu/stats/box2.html
[21] Introductino to Multi-Layer Perceptrons (Feedforward Neural Networks), https://www.iro.umontreal.ca/~bengioy/ift6266/H12/html.old/mlp_en.html
[22] Asaf Shabtai, Robert Moskovitch, Clint Feher, Shlomi Dolev, Yuval Elovici, "Detecting unknown malicious code by applying classification techniques on OpCode patterns," Shabtai et al. Security Informatics, January 2012