PEDM: Pre/Post Ensemble Decision Making for Malware Identification

Seyed Mehdi Hazrati Fard † **and Sattar Hashemi** ††,

Department of Computer Science and Engineering, Shiraz University, Shiraz, Iran

Summary
Machine learning approaches use a variety of features such as opcodes, bytecodes, and system-calls to achieve accurate malware detection. Each of these feature sets provides a unique semantic view, while, considering the effect of all together is more reliable to detect attacks. A malware can disguise itself in some views, but disguising in all views will be much more difficult. By this motivation, multi-view learning (MVL) considers multiple views of a problem to improve the overall performance. In this paper, two approaches are proposed to incorporate some various feature sets and exploit complementary information to identify the category of a file. To alleviate the complexity of the problem sparse representation is employed to make the base classifier. Then, two ways are introduced to combine the effect of multi-view. At first, the consensus of multiple views are used to minimize the overall error of a classifier and as the second, some independent classifiers are learned and weighted voting is used for the final decision. To show the generalization power of the proposed method, several datasets are investigated. Experimental results indicate that in addition to simplicity and high performance, regarding the selected base classifier, the proposed methods are able to handle imbalanced datasets.

Keywords:
multi-view learning, sparse representation, ensemble learning, malware identification.

1. Introduction

Malware stands for Malicious Software, which is a program designed for malicious purposes [1]. They include families such as virus, worm, Trojan, backdoor, rootkit, DoS, and exploit. As the complexity and concealment tricks of malware are improved progressively, employing the machine learning findings can improve the model performance against zero-day attacks [2]. Considering some distinct attributes of data helps to combine several aspects of information that may bring performance improvement. This led to propose a new approach in machine learning named multi-view learning (MVL) [3]. For example in image processing, color and texture information are two different features, which can be regarded as two-views data. As well, in malware identification, static and dynamic features can be considered as two main aspects of the problem. Static view includes bytecode, opcode and format features as three basic views [2], while dynamic features contain the system calls and the other behavioral features of the files [1].

To avoid detection, a malware file wants to appear as normal as possible, similar to the activities of other files. So, given a single aspect of the files, malicious activities may appear normal, while by analyzing multiple views there is more chance to reveal malfunctions [4]. Different feature views can provide complementary information about the actual payload of an executable file that maybe cannot be revealed by only one feature view [5].

A malware family members have several resemblances, e.g. using similar APIs for their functionality and maybe there are some changes in how to employ them. It can be a basic assumption in machine learning that the instances of each class have been scattered on a subspace and each sample can be reconstructed with the rest of them [6]. Consequently, an unknown sample can be considered as a linear combination of the other samples in that class. By this motivation, Wright et al. introduced a simple but powerful classifier named Sparse Representation based Classifier (SRC) [6]. This idea has received a lot of attention in the realm of face recognition [7]–[10]. As the behavior of malicious files is also similar, it can be assumed that they also have lied on a subspace and a new file can be reconstructed with a linear combination of the files on that subspace. As SRC takes the advantages of sparsity, only a few samples participate in reconstructing an out-of-sample and the rest of them can be neglected. Since most of the real security databases are imbalance, SRC is a very good choice to use as the base classifier.

In this research inspired by MVL, two variations of SRC are proposed using several aspects of a file for malware identification. Fig. 1 demonstrates the stages of proposed algorithms in a simple diagram. In both of them, multi-view features are employed in SRC to reconstruct an unknown sample and eventually, the final decision .

Manuscript received January 5, 2026
Manuscript revised January 20, 2026
https://doi.org/10.22937/IJCSNS.2026.26.1.23

will be taken through an ensemble method.

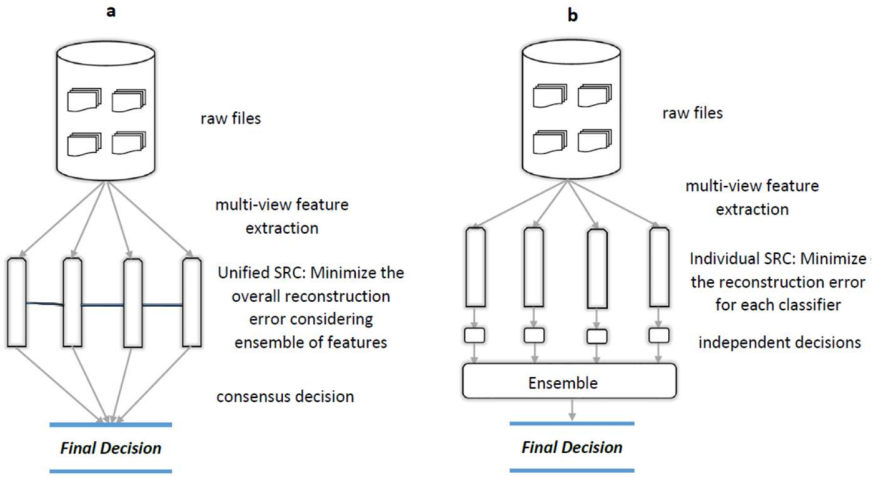

Figure. 1 The stages of two proposed algorithms 1.a) Pre-EDM that use pre-ensemble in the first stage of learning classifiers 1.b) Post-EDM that build some independent classifiers individually and combine the results, finally.

Figure. 1.a shows Pre Ensemble Decision Making (Pre-EDM) schema and Fig. 1.b is due to Post Ensemble Decision Making (Post-EDM). The main difference is how to learn the classifiers; in Pre-EDM, a unified classifier is learned based on the consensus of reconstruction errors while in Post-EDM, some SRCs decide independently by each feature set and the combination phase is postponed to the final stage by through a weighted majority voting. As the proposed algorithms are in a similar trend, we have named them Pre/Post Ensemble Decision Making (PEDM).

The main contributions of this paper are as follows:

- Proposing two ensemble methods to combine the results of multi-views in two ways: to minimize the overall error of a classifier at the initial level and to ensemble the final results of individual classifiers.
- Employing a lazy algorithm with sparsity virtue, i.e. SRC, that is not sensitive to imbalance data and is proper for real problems of malware detection and identification.

The main advantages of PEDMs are their simplicity and parsimony while good accuracy. Furthermore, as the reconstruction stage in a sparse environment is only based on a few in-hand samples, they can manage the imbalanced datasets. Furthermore, due to the selected base classifier, these methods can handle multi-class problems.

To assess the proposed methods, they have been tested on various datasets in the fields of Windows, Linux, and Android files. In most of them, the accuracies were more than 97\% and only for the most complicated of them it was more than 94\%. The mentioned approaches can be extended to any other problem with some aspects of features.

The rest of this paper is organized as follows: Section 2 reviews the related work on MVL and its applications in several malware detection tasks. In Sect. 3, two proposed methods are investigated. Several datasets are introduced in Sect. 4 and according to each one, experimental results are presented and compared with the rival methods. Sect. 5 concludes the paper with a summary of the proposed work and discussions.

2. Related Work

In this section, firstly, the concepts of learning a model using multi-views are investigated and then, some worthy works in the realm of malware detection and identification are introduced.

2.1 Multi-View Learning (MVL)

Recently, many learning methods regarding the diversity of different views of data have been proposed. The motivation of MVL is to solve the problem of machine learning with data represented by multiple distinct feature

sets [11]. It is significant to make proper use of information from several views. Considering a problem from various standpoints is a common task in real-world applications and may lead to better recognition of the problem [3].

The simplest solution for MVL is to concatenate all multiple views into a unified vector and apply a common learning algorithm directly [3]. The drawback of this approach is the curse of dimensionality and often leads to overfitting. In this case employing some dimensionality reduction algorithms is useful, but still, the specific statistical property of each view is ignored [3].

In the recent literature, MVL methods are divided into three major categories: co-training, co-regularization, and margin consistency [3]. In Co-training learners are trained alternately on distinct views. Co-EM [12], Co-testing [13], and Robust Co-training [14] are representatives of this family.

For Co-regularization algorithms, the disagreement between the discriminant functions of two views is considered as a regularization term in the optimization function. Sparse multi-view SVMs [15], multi-view TSVMs [16], multi-view Laplacian SVMs [17] and multi-view Laplacian TSVMs [18] are some major examples of this category. Another work proposed by Taheri et. al. is used several times in the security fields and to detect ransomware [19].

Margin-consistency algorithms use the latent consistency of classification results from multiple views [20]–[23]. This family is newer than the others and recently has been employed in the security tasks. Multi-view ensemble learning is the most famous method in this realm. In ensemble learning, multiple models such as classifiers or experts are combined to solve the problem. A significant application of ensemble learning is data fusion that improves the confidence of the decision made by the model [24]. Several studies have employed ensemble learning for malware detection tasks [5], [25]–[27].

2.2. Previous Work

A key point of prosperity in any machine learning task is the available features. Several useful features can be extracted from samples to use for discerning the nature of them. Opcode n-grams are extracted from the code section of portable executable (PE) files and reveal significant information to detect unknown malware [28]–[31]. Bytecode n-grams are the entire binary executable program that has no explicit semantic information, but the sequence of bytecodes contains useful features for unknown malware detection [32]–[34]. Format features are other important traits that have explicit semantic information extracted from PE header, section header, import, and resource section. They have been used in several malware detection projects [35]–[38].

Each aforementioned feature contains some details about the nature of a file, but all together provide complementary information for better diagnosis. Bai and Wang used multi-view ensemble learning for unknown malware detection and employed all three mentioned features together in their proposed method [5].

Other useful features that have been used several times in malware detection tasks are function-based. These features can be extracted from behaviors of an executable file that is running in an isolated environment [1]. Homayoun et al. proposed an efficient method to classify ransomware families based on some system call features [39]. Moreover, Menahem et al. employed function-based features besides PE features and bytes n-grams, then used different feature extraction parameters to construct five datasets. They construct five classifiers: C4.5 Decision Tree, Naïve Bayes, KNN, VFI, and OneR based on aforementioned datasets respectively. Consequently, they combined them for a final decision. The combination methods include majority voting, performance weighting, distribution summation, Bayesian combination, Naïve Bayes, stacking and Troika. Experimental results showed that using multi-view features and ensemble methods improved the accuracy and outperformed each of the mentioned single view classifiers [26].

It is commonplace to extract several features from a view, e.g. opcode sequence, but these are the derivations of a single-view and cannot boost each other significantly for MVL proposes. Landage et al. made three different kinds of opcode sequence representations of the instances. Then, constructed three base classifiers with each representation and combined them with the majority and veto-based voting [40]. Final results did not show a significant improvement than the individual classifiers that can approve the advantages of combining some classifiers based on some distinct view.

API calls are another feature view of the files that can be extracted while a file is running in a dedicated environment, so, it is a dynamic view. Sheen et al. extracted features from PE header moreover the API calls and ran different learning algorithms to construct a set of classifiers. Then selected the best subset of classifiers and combined them, and reached out better results than individual classifiers or other ensemble learning algorithms [27]. Caruana et. al. stated that combining a proper subset of base classifiers to constitute an ensemble may work better than using all of them [25]. They called this method Ensemble Selection (ES), which can achieve strong generalization performance with small sized base classifiers.

Android malware is another family of malicious files that have been mentioned in past years. Ozdemir et al. extracted some different static and dynamic features from APK files and employ multiple learning algorithms to construct diverse base classifiers. Then a subset of base classifiers was selected using a simple heuristic algorithm and combined them by majority voting. Experimental results show the superiority of this ensemble over the rival methods [41]. There were other proposed works in this realm that employ MVL in API calls of android executable files beside the other features and improve detection rate [42].

3. Proposed Method

Ensemble methods combine some classifiers to obtain better results than could be reached from any of them [43]. The major strength of ensemble methods depends on the diversity of base classifiers. There are several techniques to reach such classifiers e.g. resampling the training set, using various learning algorithms or same learning algorithm with different parameters [44]. Using independent features to learn the classifiers is a good choice to reach diversity; e.g. employing MVL [3].

The generalization of an ensemble method is usually stronger than base learners. In the earliest methods, it was common to use some weak learners as the individual classifiers, but many researches state that it is not necessary to generate weak classifiers [45]. Some prevalent ensemble methods employ decision tree and neural network as base classifiers. Most ensemble methods use a single base algorithm to produce homogeneous base learners, but selecting the most suitable algorithm according to the nature of existing samples is a crucial task. SRC is an applicable classifier [6] that has attracted much attention in the past decay. There are several improvements on this method that has been proposed recently [7]–[10].

The seminal work was proposed by John Wright [6]. The goal is to represent an out of sample as a linear combination of some selected instances. If there are sufficient training examples from each category, it is feasible to represent the test sample as a linear combination of just those samples from the same category [6]. To reconstruct a sample based on other instances the simple objective is:

$$\|s - Xw\| \qquad (1)$$

where s is a new sample, X contains the available instances and w is the vector of coefficients that should be sparse enough to choose a few samples for reconstruction. Minimizing the cardinality by adding l_0-*norm* of w, forces many coefficients to be zero, but as the problem is *NP-hard* and intractable in general case, according to the convex envelope the constraint can be approximated with l_1-*norm* [46].

$$w^* = \underset{w}{argmin}\|s - Xw\|^2 + \lambda\|w\|_1 \qquad (2)$$

λ is the regularization parameter to specify the sparsity level and can be defined due to the size of the dictionary [47]. There are several toolboxes, such as NESTA [48] and SPAMS [49] to solve this function in polynomial time, using coordinate descent [50].

In the field of malware identification, suppose X is the matrix of labeled samples, each one is presented with d features in the rows. Consider n samples in the columns, n_1 belong to class *1*, n_2 belong to class *2* and so on. Fig. 2 depicts this structure. So the obtained vector w contains n elements pertain to all available samples, n_1 coefficients according to samples of class *1*, n_2 coefficients for samples of class *2* and so on. Most of the coefficients are negligible and only a few ones are valuable.

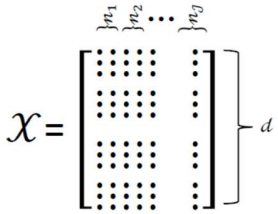

Figure. 2 Each column of the matrix shows a sample in d dimensions. The first n_1 samples are according to class *1*, n_2 samples belong to class *2* and so on.

Consequently, the test sample can be reconstructed with the samples of each class separately. As it was a preassumption that the samples of each class have lied on a subspace, the class with the minimum reconstruction error can define the class of the test sample [6].

$$j^* = \underset{j\in\{1,2,\ldots,J\}}{arg\,min}\left\|s - X_j w_j^*\right\| \qquad (3)$$

j shows the label of each class through J classes and j^* is the label of the class with the minimum reconstruction error. Eq. 2 and 3 show the seminal method of SRC [6].

Figure. 3 can schematically depict the motivation of SRC. Here, since the samples of a class lie on a subspace, the new sample can be reconstructed with a linear combination of its neighborhoods on the path of that class.

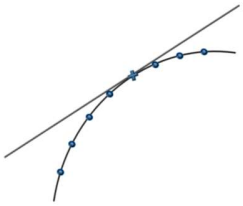

Figure. 3 The spanned space of a class samples can be considered locally linear and each sample can be reconstructed with the linear combination of its adjacent on this path.

Inspired by SRC and the idea behind MVL to consider the problem from several aspects, the first algorithm was proposed. So to make diverse classifiers, several feature sets are employed in SRC. The new file is reconstructed with a linear combination of available samples based on each feature view. The objective is to minimize the reconstruction error according to each view. Considering k views, the best coefficient vector w^* is the one who minimizes the sum of errors according to all views. The modified function can be written as follows:

$$w^* = \underset{w}{argmin} \sum_{k=1}^{K} \|s^k - X^k w^k\|^2 + \lambda \|w^k\|_1 \qquad (4)$$

Due to the nature of each problem and the count of available samples, the best value for λ can be obtained through cross-validation. As the number of features in all views are not the same, it is vital to exert a normalization factor on each reconstruction error term and divide each one to the number of features related to that view:

$$w^* = \underset{w}{argmin} \sum_{k=1}^{K} \frac{\|s^k - X^k w^k\|^2}{d^k} + \lambda \|w^k\|_1 \qquad (5)$$

where d^k is the dimension of the kth view. w^* is the best weight vector include the coefficients of training samples according to all views. Finally, the instances of the class that lead to the minimum sum of reconstruction errors regarding all feature sets, indicate the label of new sample:

$$j^* = arg \min_{j \in \{1,2,\dots,J\}} \sum_{k=1}^{K} \frac{\|s^k - X_j^k w_j^{k*}\|}{d^k} \qquad (6)$$

As the aim of this optimization function is to minimize the sum of reconstruction errors, all views participate in the initial stage to find w^* as a shared vector. Thus, this algorithm is named Pre Ensemble Decision Making (Pre-EDM). Algorithm 1 shows the stages of Pre-EDM to label an unknown file.

Algorithm 1: **Pre-EDM**

Input: a raw file

output: the selected class

1. Extract features of several views from all files.
2. Find the best coefficient vector of w^* that leads to minimum reconstruction error based on all features.
3. Choose the class which best describe the out of sample through all views.

Another perspective is to seek an independent sparse coefficient vector according to each view. Motivated by this idea, the second proposed method emerged. To do so, the objective function according to the k^{th} view will be:

$$w^{k*} = \underset{w^k}{argmin} \|s^k - X^k w^k\|^2 + \lambda \|w^k\|_1 \qquad (7)$$

Here, the vector of w^* is obtained independently regarding each view, and then the out of sample can be reconstructed based on each w^{k*} of a view:

$$j^* = arg \min_{j \in \{1,2,\dots,J\}} \|s^k - X_j^k w_j^{k*}\| \qquad (8)$$

Finally, we have the verdict of individual classifiers according to each feature view and taking a majority vote over all of them, can determine the final decision. To agglomerate the votes of individual classifiers, we used weighted vote according to their accuracy. In this method, each classifier makes a decision independently based on a different coefficient vector in the middle stage and finally an ensemble on all decisions determine the class label, so, we named it Post Ensemble Decision Making (Post-EDM). Algorithm 2 shows the stages of Post-EDM consequently.

Algorithm 2: **Post-EDM**

Input: a raw file

output: the selected class

1. Extract features of several views from all files.
2. Find the best coefficient vector of w^* that leads to minimum reconstruction error according to each set of features individually.
3. Choose the class which best describe the out of sample through each view.
4. Use a weighted vote to determine the final class label.

The main difference between the two methods is how to combine the results of several views. In Pre-EDM, a unified SRC decides based on the consensus of all views and find the class with the minimum overall reconstruction error regarding all features, but Post-EDM at first makes some independent classifiers based on each view and eventually combine the independent opinions to determine

the final class label. In conclusion, both of them decide through a multi-view ensemble manner; Pre-EDM use multi-view and ensemble together in a common classifier, whilst, Post-EDM use multi-view at the first level of learning classifiers and employ ensemble for a final decision.

As the main trend of the two proposed algorithms is the same, we named them PEDM. Another advantage of PEDM besides the precision is the insensitivity to imbalance data. As to reconstruct a new sample a linear combination of some in-hand instances are needed, the reconstruction can be considered locally linear and the rest of the samples in the intended class are not required. So PEDM is not dependent on all samples of a class and only needs the samples that lie on the subspace near the test sample.

Table 1 The introduced datasets and their specifications. To evaluate the methods two binary class datasets include malware and benign files and two multi-class databases consist of some malware types are employed.

dataset	*environment*	*# samples*	*# classes*	*feature set 1*	*feature set 2*	*feature set 3*	*feature set 4*
IoT dataset	Linux	551	2	opcodes TF-IDF	opcode 2-grams	bytecode TF-IDF	bytecode 2-grams
VXHeaven	Windows	330	2	opcodes TF-IDF	opcode 2-grams	bytecode TF-IDF	bytecode 2-grams
Ransomware dataset	Windows	1627	4	system call	opcodes TF-IDF	opcode 2-grams	
Microsoft malware	Windows	10825	9	opcodes TF-IDF	opcode 2-grams	opcode 3-grams	

4. Experiments

We have conducted several groups of experiments to evaluate the effectiveness of PEDM for malware classification. In this section, several datasets on various platforms, e.g. Android and Windows, are explained. Then, the capability of feature combination is illustrated and compared with the individual feature set and other rival methods. The used datasets and their specifications are introduced briefly in Table 1. Whereas PEDM does not need many samples from each class for reconstruction, firstly, some samples from each category are selected and then, the required features are extracted.

To show the merit of PEDM, the results of them are compared with the SRC using each in-hand feature set and the concatenation of all feature sets. Employing supervector of all features was the seminal idea of MVL that was used in the previous work. The algorithms have been implemented in Matlab 2016b, and have run on a personal desktop equipped with a Core i7-3770 CPU and 32GB of memory. To evaluate the methods some metrics are required. Accuracy is a useful metric that has been used widely in the machine learning assessments and considers the rate of true predictions with respect to all. If TP and TN are the malicious and benign files that have been classified true and N is the number of all samples accuracy will be:

$$accuracy = \frac{TP + TN}{N} \qquad (9)$$

However, in the case of imbalanced data, a supplementary evaluation is needed. Matthews Correlation Coefficient (MCC) [51] is another measures of quality exactly to evaluate the performance of classifiers in case of imbalanced datasets [52]. The MCC value is between -1 and +1 that high values are intended in a classifier. Assume S is the rate of positive samples and P is the rate of samples that classified as positive. Then, MCC can be shown as:

$$MCC = \frac{\frac{TP}{N} - S \times N}{\sqrt{PS(1-P)(1-S)}} \qquad (10)$$

Receiver Operating Characteristic (ROC) is another metric that initially has been used in the medicine and security tasks, while, it is adequate for any evaluation including imbalanced data [53]. ROC investigates a relationship between sensitivity and specificity of a binary classifier, while, sensitivity or true positive rate (TPR) measures the proportion of positives correctly classified. Specificity or true negative rate (TNR) measures a proportion of negatives correctly classified, and one minus TNR indicates false positive rate (FPR). In ROC, TPR is plotted against FPR and Area Under ROC Curve (AUC) is the most important statistic associated with ROC curve [53].

4.1. Internet of Things (IoT) Dataset

Nowadays, IoT devices become more and more prevalent and consequently many malware developers target IoT devices [54]. Thus, one of the selected datasets to challenge our method includes IoT files. In this dataset, 280 malware samples were collected from 32-bit ARM-based malware in the Virus Total Threat Intelligence platform. For compatibility of the malware and benign files, 271 common files of Linux Debian package repository were chosen [55]. The files were unpacked by the Debian installer bundle and Object-Dump tool was employed to decompile all samples. Consequently, the sequence of opcodes in each sample was obtained.

After that, two feature sets are extracted from opcodes: TF-IDF and 2-grams. TF-IDF considers the repetition of each opcode in a file individually vs. the presence of this opcode in the other files, while 2-grams considers the importance of opcode sequences. Bytecode is another applicable view of the files. Then, two set of features according to bytecode TF-IDF and 2-grams are extracted. Eventually, we had 4 set of features to learn the classifiers.

For the first evaluation pace, 100 samples are selected randomly from each class and then leave-one-out is exerted to reconstruct each sample with the rest of 199. Table 2 compares the results of PEDMs to the original SRC according to each set of features individually and using supervector of all features. The results are due to the average of 10 times random selection of samples from the initial dataset. The first row of the table is according to Haddadpajouh et. al. due to this dataset [55].

Table 2 Comparing the results of PEDMs on IoT dataset with SRC (based on four feature sets and the concatenation of all). Furthermore, the result of Haddadpajouh et. al. [55] due to this dataset from their paper is prepared at the first row of the table.

IoT (100/100)	*TPR (%)*	*FPR (%)*	*accuracy (%)*	*MMC*
Haddadpajouh et. al. [55]	98.6	2.1	98.1	0.971
SRC opcode TF-IDF	96.1	3.2	95.3	0.931
SRC opcode 2-grams	92.5	4.4	94.4	0.919
SRC bytecode TF-IDF	91.3	5.2	92.6	0.908
SRC bytecode 2-grams	93.1	3.8	90.3	0.924
SRC supervector	97.9	2.8	96.7	0.967
Pre-EDM	100	0.0	100	1.000
Post-EDM	98.3	0.9	99.2	0.981

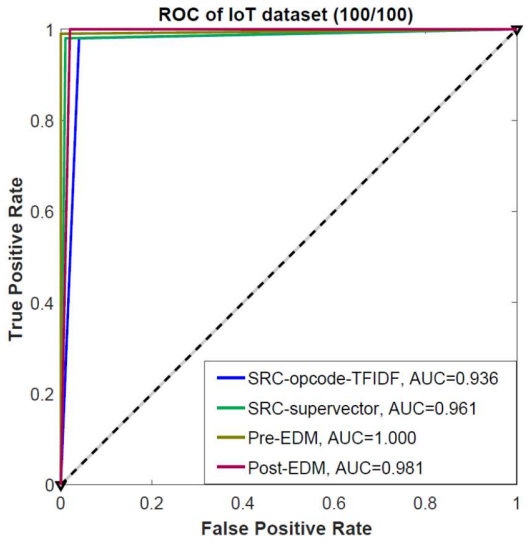

Figure. 4 Comparing AUC and ROC curve of PEDMs with the SRCs according to opcode TF-IDF and supervector of all features on 100 selected samples of each class in IoT dataset.

Fig. 4 shows the AUC and ROC curves according to the first run of some methods in Table 2. To prevent from the crowd diagram, the results of PEDMs have been compared with the best SRC according to a single feature set (opcode TF-IDF here) and supervector. According to the diagram, Pre-EDM has the perfect results and can detect all malicious IoT files.

To investigate the imbalance effect on the proposed methods, we selected a subset of 100 benign and 30 of malware samples. Table 3 shows the results according to 10 times random selection in the mentioned methods that show no major changes compared to the balanced condition. This can prove the ability of PEDMs to handle imbalanced conditions.

Table 3 The averages of results on the imbalanced dataset according to Pre/Post ensemble.

IoT (100/30)	*TPR (%)*	*FPR (%)*	*accuracy (%)*	*MCC*
Pre-EDM	99.1	1.1	98.4	0.967
Post-EDM	97.4	1.2	97.3	0.970

4.2. VXHeaven Dataset

VXHeaven is a benchmark dataset contains windows malware [56] and was used to evaluate several methods [57]–[59]. 1000 samples are picked randomly from the dataset and labeled as malware and benign. Then, TF-IDF and 2-grams according to opcodes and bytecodes of the selected files are extracted as 4 feature sets.

Table 4 The results of PEDMs on VXHeaven samples compared to SRC (based on four feature sets and the concatenation of all) and some recent methods.

VXHeaven (100/100)	*TPR (%)*	*FPR (%)*	*Accuracy (%)*	*MCC*
Farrokhmanesh et. al. [57]	91.3	7.9	90.4	0.906
Hashemi et. al. [58]	89.1	7.8	89.7	0.9254
Hashemi et. al. [59]	96.0	3.1	94.7	0.925
SRC opcode TF-IDF	86.6	12.8	84.5	0.885
SRC opcode 2-grams	85.5	14.4	87.4	0.890
SRC bytecode TF-IDF	88.3	7.6	89.6	0.833
SRC bytecode 2-grams	90.3	5.2	90.1	0.916
SRC supervector	92.9	5.5	93.6	0.927
Pre-EDM	96.2	2.1	97.3	0.961
Post-EDM	97.4	2.3	97.1	0.966

The mentioned algorithms are applied on samples 10 times; each time 100 samples are picked randomly from each class and the mean of results are presented in Table 4. Leave-one-out is exerted, while the runtime for each sample was less than a second. The first three rows of the table are due to some available methods that recently have worked on this dataset.

As we faced to a binary class dataset, we examined imbalanced condition for PEDMs again. To do so, a subset of 100 benign and 30 malware samples are selected randomly, 10 times. Table 5 implies the ability of the proposed methods to challenge the imbalanced conditions.

Table 5 The averages of results on the imbalanced dataset according to Pre/Post ensemble.

VXHeaven (100/30)	*TPR (%)*	*FPR (%)*	*accuracy (%)*	*MCC*
Pre-EDM	98.3	1.2	98.9	0989
Post-EDM	97.6	1.4	98.2	0.977

Figure. 5 shows the AUC and ROC curves according to the first run of PEDMs in compared with the best SRC according to single feature set (bytecode 2-grams here) and supervector that were in Table 6. As see, bytecode 2-grams is the best single feature set in this case and Pre-EDM shows the best results of all for 100 samples in each class.

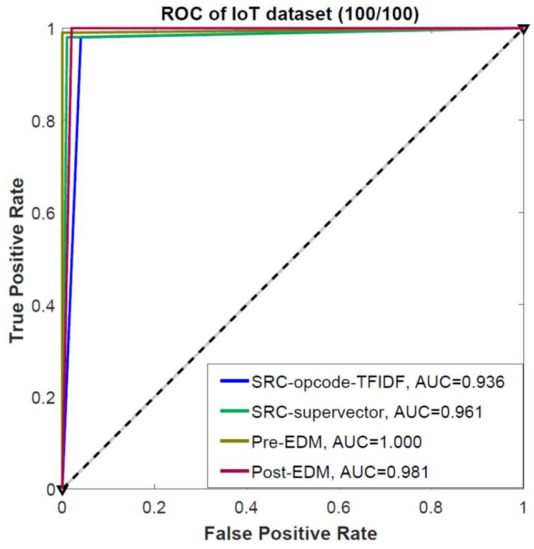

Figure. 5 Comparing AUC and ROC curve of PEDMs with the SRCs according to bytecode 2-grams and supervector of all features on 100 selected samples of each class in VXHeaven dataset.

4.3. Ransomware Dataset

As another evaluation, ransomware dataset has been employed [39]. The ransomware dataset contains sequences of activities according to some Windows Portable Executable (PE32) ransomware samples reported as malicious from Virustotal[1]. This dataset consists of three famous families of ransomware namely Locky, Cerber and TeslaCrypt. As ransomware samples are in form of PE files, portable applications available at the portableapps website[2] are considered as benign samples. Table 6 shows the number of samples in the dataset with three families of ransomware and a group of benign samples.

Table 6 The number of samples in each family of ransomware and benign.

Class	*number of samples*
Locky	450
Cerber	470
TeslaCrypt	507
Benign	200

All samples were launched in a special testbed to collect runtime behaviors of ransomware and normal

[1] https://www.virustotal.com

[2] https://portableapps.com/app

samples. The runtime behaviors were considered as system calls performed by process of a monitored sample that leads to the first view of the samples. Moreover, two sets of features were extracted from opcodes: TF-IDF and 2-grams.

Table 7 Comparing the results of PEDMs on ransomware with SRC (based on three feature sets and the concatenation of all) and Homayoon et. al. [39] who gathered this dataset.

ransomware (4 × 50)	*TPR (%)*	*FPR (%)*	*accuracy (%)*	*MCC*
Homayoun et. al. [39]	98.0	2.6	97.2	0.973
SRC System calls	86.3	12.1	85.2	0.881
SRC opcode TF-IDF	89.5	9.7	92.9	0.912
SRC opcode TF-IDF	91.5	7.0	90.3	0.919
SRC supervector	96.2	4.9	94.9	0.926
Pre-EDM	98.6	1.3	98.9	0.979
Post-EDM	98.9	0.8	98.7	0.983

As a few samples were adequate for the learning and a multi-class classification problem was ahead, we selected 50 samples of each class randomly and exerted leave-one-out for assessment. These steps were repeated 10 times and the results were summed up in table 7. The first row of the table is the results of Homayoon et. al. [39] that were reported in their paper according to the prepared database.

4.4. Microsoft Malware Dataset

Another Windows-based malicious dataset that has been used in our evaluation is Microsoft malware collection that was presented in the Microsoft malware classification challenge from the Kaggle website [60]. This dataset contains more than 10000 samples from 9 families of malware variants that have been analyzed statically to obtain their opcodes. Consequently, TF-IDF, 2-grams, and 3-grams of the opcodes from 900 files of all classes are extracted.

We selected 40 samples of each class randomly 10 times and repeat our experiments for each subset. Table 8 shows the average accuracies and the standard deviation according to 10 runs. Whereas all three feature sets are pertained to opcodes, in this case, the results of the ensemble methods are not significantly superior.

Table 8 Comparing the results of PEDMs on Microsoft malware dataset with SRC (based on three feature sets and the concatenation of all).

ransomware (4 × 50)	*TPR (%)*	*FPR (%)*	*Accuracy (%)*	*MCC*
SRC opcode TF-IDF	91.7	8.8	92.9	0.914
SRC opcode 2-gram	92.8	4.6	92.3	0.917
SRC opcode 3-gram	94.7	5.6	93.4	0.936
SRC supervector	93.2	6.1	94.5	0.926
Pre-EDM	96.1	2.9	94.9	0.940
Post-EDM	94.4	5.7	94.3	0.951

5. Conclusion and Future Work

In this study, inspired by multi-view learning (MVL) two ensemble methods have been proposed for malware identification and classification, namely Pre/Post Ensemble Decision Making (PEDM). The chosen base classifier was a variation of Sparse Representation based Classifier (SRC) that has been used widely in face recognition tasks. Whereas using multi-views of the files, e.g. opcode, bytecode, and system calls, help the classifiers to reveal the hidden dimensions of a malware file, we used them in each individual classifier and combine the results in two ways.

Actually, the difference of PEDMs is in the combination phase; Pre-EDM employs multi-views of the files to minimize the consensus error of a unified classifier at the first level, while Post-EDM learns some individual classifiers independently and postpone the combination of results to the final decision.

The proposed methods outperform any individual based classifiers trained on a single feature set and show elegant results on several datasets that have been investigated in the experimental results. Moreover, accuracy and MCC is better than the rival methods in this field. The advantages of PEDMs can be considered as follows:

- Combining the effect of several views of a file for discerning its class.
- The ability to handle multi-class classification tasks.
- The ability to deal with the imbalanced datasets.

As the future work, we suggest learning the combination phase of the algorithms intelligently. To do so, we can learn each classifier individually and then learn a model for the best combination of them. Another

suggestion is to extend these methods to the other machine learning tasks. There are various approaches in the real world that suffer from the nature of imbalanced data and can take the advantages of PEDMs.

Acknowledgments

The authors would like to thank Homayoun, Hashemi, and Haddadpajouh that gave us their prepared datasets and results.

References

[1] Z. Bazrafshan, H. Hashemi, S. M. H. Fard, and A. Hamzeh, "A survey on heuristic malware detection techniques," in *IKT 2013 - 2013 5th Conference on Information and Knowledge Technology*, 2013, pp. 113–120.

[2] A. Shalaginov, S. Banin, A. Dehghantanha, and K. Franke, "Machine learning aided static malware analysis: A survey and tutorial," *Cyber Threat Intell.*, pp. 7–45, 2018.

[3] J. Zhao, X. Xie, X. Xu, and S. Sun, "Multi-view learning overview: Recent progress and new challenges," *Inf. Fusion*, vol. 38, pp. 43–54, 2017.

[4] A. Y. Liu and D. N. Lam, "Using consensus clustering for multi-view anomaly detection," in *Security and Privacy Workshops (SPW), 2012 IEEE Symposium on*, 2012, pp. 117–124.

[5] J. Bai and J. Wang, "Improving malware detection using multi-view ensemble learning," *Secur. Commun. Networks*, vol. 9, no. 17, pp. 4227–4241, 2016.

[6] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust Face Recognition via Sparse Representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[7] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma, "Toward a practical face recognition system: Robust alignment and illumination by sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 372–386, 2012.

[8] A. Y. Yang, S. S. Sastry, A. Ganesh, and Y. Ma, "Fast ℓ 1-minimization algorithms and an application in robust face recognition: A review," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, 2010, pp. 1849–1852.

[9] J. Ma, J. Zhao, Y. Ma, and J. Tian, "Non-rigid visible and infrared face registration via regularized Gaussian fields criterion," *Pattern Recognit.*, vol. 48, no. 3, pp. 772–784, 2015.

[10] Y. Gao, J. Ma, and A. L. Yuille, "Semi-supervised sparse representation based classification for face recognition with insufficient labeled samples," *IEEE Trans. Image Process.*, vol. 26, no. 5, pp. 2545–2560, 2017.

[11] S. Sun, "A survey of multi-view machine learning," *Neural Comput. Appl.*, vol. 23, no. 7–8, pp. 2031–2038, 2013.

[12] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *Proceedings of the ninth international conference on Information and knowledge management*, 2000, pp. 86–93.

[13] I. Muslea, S. Minton, and C. A. Knoblock, "Active learning with multiple views," *J. Artif. Intell. Res.*, vol. 27, pp. 203–233, 2006.

[14] S. Sun and F. Jin, "Robust co-training," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 25, no. 07, pp. 1113–1126, 2011.

[15] S. Sun and J. Shawe-Taylor, "Sparse semi-supervised learning using conjugate functions," *J. Mach. Learn. Res.*, vol. 11, no. Sep, pp. 2423–2455, 2010.

[16] X. Xie and S. Sun, "Multi-view twin support vector machines," *Intell. Data Anal.*, vol. 19, no. 4, pp. 701–712, 2015.

[17] S. Sun, "Multi-view Laplacian support vector machines," in *International Conference on Advanced Data Mining and Applications*, 2011, pp. 209–222.

[18] X. Xie and S. Sun, "Multi-view Laplacian twin support vector machines," *Appl. Intell.*, vol. 41, no. 4, pp. 1059–1068, 2014.

[19] M. Taheri, H. Azad, K. Ziarati, and R. Sanaye, "A QUADRATIC MARGIN-BASED MODEL FOR WEIGHTING FUZZY CLASSIFICATION RULES INSPIRED BY SUPPORT VECTOR MACHINES," *Iran. J. Fuzzy Syst.*, vol. 10, no. 4, pp. 41–55, 2013.

[20] G. Chao and S. Sun, "Alternative multiview maximum entropy discrimination," *IEEE Trans. neural networks Learn. Syst.*, vol. 27, no. 7, pp. 1445–1456, 2016.

[21] G. Chao and S. Sun, "Consensus and complementarity based maximum entropy discrimination for multi-view classification," *Inf. Sci. (Ny).*, vol. 367, pp. 296–310, 2016.

[22] L. Mao and S. Sun, "Soft Margin Consistency Based Scalable Multi-View Maximum Entropy Discrimination.," in *IJCAI*, 2016, pp. 1839–1845.

[23] S. Sun and G. Chao, "Multi-View Maximum Entropy Discrimination.," in *IJCAI*, 2013, pp. 1706–1712.

[24] R. Polikar, "{E}nsemble learning," *Scholarpedia*, vol. 4, no. 1, p. 2776, 2009.

[25] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 18.

[26] E. Menahem, A. Shabtai, L. Rokach, and Y. Elovici, "Improving malware detection by applying multi-inducer ensemble," *Comput. Stat. Data Anal.*, vol. 53, no. 4, pp. 1483–1494, 2009.

[27] S. Sheen, R. Anitha, and P. Sirisha, "Malware detection by pruning of parallel ensembles using harmony search," *Pattern Recognit. Lett.*, vol. 34, no. 14, pp. 1679–1686, 2013.

[28] D. Bilar, "Opcodes as predictor for malware," *Int. J. Electron. Secur. Digit. Forensics*, vol. 1, no. 2, pp. 156–168, 2007.

[29] R. Moskovitch *et al.*, "Unknown malcode detection using opcode representation," in *Intelligence and Security Informatics*, Springer, 2008, pp. 204–215.

[30] I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas, "Opcode sequences as representation of executables for data-mining-based unknown malware detection," *Inf. Sci. (Ny).*, vol. 231, pp. 64–82, 2013.

[31] A. Shabtai, R. Moskovitch, C. Feher, S. Dolev, and Y. Elovici, "Detecting unknown malicious code by applying classification techniques on opcode patterns," *Secur. Inform.*, vol. 1, no. 1, p. 1, 2012.

[32] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," *J. Mach. Learn. Res.*, vol. 7, no. Dec, pp. 2721–2744, 2006.

[33] D. K. S. Reddy and A. K. Pujari, "N-gram analysis for computer virus detection," *J. Comput. Virol.*, vol. 2, no. 3, pp. 231–239, 2006.

[34] I. Santos, Y. K. Penya, J. Devesa, and P. G. Bringas, "N-grams-based File Signatures for Malware Detection.," *ICEIS (2)*, vol. 9, pp. 317–320, 2009.

[35] J. Bai, J. Wang, and G. Zou, "A malware detection scheme based on mining format information," *Sci. World J.*, vol. 2014, 2014.

[36] M. Z. Shafiq, S. Tabish, and M. Farooq, "PE-probe: leveraging packer detection and structural information to detect malicious portable executables," in *Proceedings of the Virus Bulletin Conference (VB)*, 2009, vol. 8.

[37] M. Z. Shafiq, S. M. Tabish, F. Mirza, and M. Farooq, "Pe-miner: Mining structural information to detect malicious executables in realtime," in *International Workshop on Recent Advances in Intrusion Detection*, 2009, pp. 121–141.

[38] M. Zakeri, F. Faraji Daneshgar, and M. Abbaspour, "A static heuristic approach to detecting malware targets," *Secur.*

Commun. Networks, vol. 8, no. 17, pp. 3015–3027, 2015.
[39] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, and R. Khayami, "Know abnormal, find evil: Frequent pattern mining for ransomware threat hunting and intelligence," *IEEE Trans. Emerg. Top. Comput.*, 2017.
[40] J. Landage and M. P. Wankhade, "Malware detection with different voting schemes," *Compusoft*, vol. 3, no. 1, p. 450, 2014.
[41] M. Ozdemir and I. Sogukpinar, "An android malware detection architecture based on ensemble learning," *Trans. Mach. Learn. Artif. Intell.*, vol. 2, no. 3, pp. 90–106, 2014.
[42] S. Sheen, R. Anitha, and V. Natarajan, "Android based malware detection using a multifeature collaborative decision fusion approach," *Neurocomputing*, vol. 151, pp. 905–912, 2015.
[43] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, no. 1–2, pp. 1–39, 2010.
[44] L. I. Kuncheva, *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.
[45] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2012.
[46] C. Ramirez, V. Kreinovich, and M. Argaez, "Why l1 is a good approximation to l0: A geometric explanation," *J. Uncertain Syst.*, vol. 7, no. 3, pp. 203–207, 2013.
[47] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, vol. 1, no. 10. Springer series in statistics New York, NY, USA:, 2001.
[48] S. Becker, J. Bobin, and E. J. Candès, "NESTA: a fast and accurate first-order method for sparse recovery," *SIAM J. Imaging Sci.*, vol. 4, no. 1, pp. 1–39, 2011.
[49] J. Mairal, "SPAMS: a SPArse Modeling Software, v2. 5." 2014.
[50] S. J. Wright, "Coordinate descent algorithms," *Math. Program.*, vol. 151, no. 1, pp. 3–34, 2015.
[51] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochim. Biophys. Acta (BBA)-Protein Struct.*, vol. 405, no. 2, pp. 442–451, 1975.
[52] D. M. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," 2011.
[53] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, 2006.
[54] H. Naeem, B. Guo, and M. R. Naeem, "A light-weight malware static visual analysis for IoT infrastructure," in *2018 International Conference on Artificial Intelligence and Big Data (ICAIBD)*, 2018.
[55] H. HaddadPajouh, A. Dehghantanha, R. Khayami, and K.-K. R. Choo, "A deep Recurrent Neural Network based approach for Internet of Things malware threat hunting," *Futur. Gener. Comput. Syst.*, vol. 85, pp. 88–96, 2018.
[56] V. X. Heaven, "Computer virus collection," *2007-09-14)[2010-05-28]. http://vx. netlux. org/vl. php*. 2014.
[57] M. Farrokhmanesh and A. Hamzeh, "A novel method for malware detection using audio signal processing techniques," in *Artificial Intelligence and Robotics (IRANOPEN), 2016*, 2016, pp. 85–91.
[58] H. Hashemi and A. Hamzeh, "Visual malware detection using local malicious pattern," *J. Comput. Virol. Hacking Tech.*, pp. 1–14, 2018.
[59] H. Hashemi, A. Azmoodeh, A. Hamzeh, and S. Hashemi, "Graph embedding as a new approach for unknown malware detection," *J. Comput. Virol. Hacking Tech.*, vol. 13, no. 3, pp. 153–166, 2017.
[60] Microsoft, "Microsoft malware classification challenge," *[Online]*, no. Available: https://www.kaggle.com/c/malware-classification., 2015.

Seyed Mehdi Hazrati Fard received the M.Sc. degree in computer science from Shiraz University, Shiraz, Iran in 2012. Since that year he is an active member of Machine Learning Lab (MLL) and did several researches on deep networks and sparse representation. Furthermore, he has attended at Pishtazan Higher Educational Institute as lecturer. Currently, he is a Ph.D. student at Electrical and Computer Engineering School, Shiraz University, Shiraz, Iran. He has been at the University of Waterloo to spend sabbatical leave.

Sattar Hashemi received the Ph.D. degree in computer science from Iran University of Science and Technology in conjunction with Monash University, Australia, in 2008. Following academic appointments at Shiraz University, he is currently the associated professor of computer department at Electrical and Computer Engineering School, Shiraz University, Shiraz, Iran. He is recognized for contributions in the fields of machine learning and data mining. He has published many refereed papers and book chapters on data stream classification, social networks, database intrusion detection, and computer security.