Zero-day Ransomware Attack Detection using Deep Contractive Autoencoder and Voting based Ensemble Classifier

*Umme Zahoora[a], Muttukrishnan Rajarajan[b], Zahoqing Pan[c], and Asifullah Khan[*a,d,e]*

[a]Department of Computer and information Science, Pakistan Institute of Engineering and Applied Sciences, Nilore-45650, Islamabad
[b] School of Mathematics, Computer Science and Engineering, City University of London, London EC1V 0HB,UK
[c]School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China
[d]Center for Mathematical Sciences, Pakistan Institute of Engineering and Applied Sciences, Nilore-45650, Islamabad
[e] PIEAS Artificial Intelligence Center (PAIC), Pakistan Institute of Engineering & Applied Sciences, Nilore, Islamabad 45650, Pakistan

Corresponding Author: [*]Asifullah Khan, asif@pieas.edu.pk

A B S T R A C T

Ransomware attacks are hazardous cyber-attacks that use cryptographic methods to hold victims' data until the ransom is paid. Zero-day ransomware attacks try to exploit new vulnerabilities and are considered a severe threat to existing security solutions and internet resources. In the case of zero-day attacks, training data is not available before the attack takes place. Therefore, we exploit Zero-shot Learning (ZSL) capabilities that can effectively deal with unseen classes compared to the traditional machine learning techniques. ZSL is a two-stage process comprising of: attribute learning (AL) and Inference Stage (IS). In this regard, this work presents a new Deep Contractive Autoencoder based Attribute Learning (DCAE-ZSL) technique as well as an IS method based on Heterogeneous Voting Ensemble (DCAE-ZSL-HVE). In the proposed DCAE-ZSL approach, Contractive Autoencoder (CAE) is employed to extract core features of known and unknown ransomware. The regularization term of CAE helps in penalizing the classifier's sensitivity against the small dissimilarities in the latent space. On the other hand, in case of the IS, four combination rules Global Majority (GM), Local Majority (LM), Cumulative vote-against based Global Majority (CVAGM), Cumulative vote-for based Global Majority (CVFGM) are utilized to find the final prediction. It is empirically shown that in comparison to conventional machine learning techniques, models trained on contractive embedding show reasonable performance against zero-day attacks. Furthermore, it is shown that the exploitation of these core features through the proposed voting based ensemble (DCAE-ZSL-HVE) has demonstrated significant improvement in detecting zero-day attacks (recall=0.95) and reducing False Negative (FN= 6).

Keywords: Zero-shot Learning, Zero-day Attack, Ransomware, Deep Learning, Autoencoder, Ensemble Classification.

1. Introduction

Ransomware is a malware that possesses its special characteristics in addition to the standard features of a generic malware [1]. Ransomware, generally follows similar methods to evade, propagate, and attack its victims as other malwares do. However, it injects its peculiar actions in the form of processes into target programs, then extract the data, and establishes the connection with Command and Control server (C&CS). Its main function is to encrypt all the important files in the target system and demand ransom for recovery. These typical steps of ransomware lifecycle are described pictorially in Fig. 1. Due to its specific objective, it is considered easy to write and modify the existing ransomware that can result in an explosive generation of its variants [2]. Wannacry variants (executed in 2017) are said to be responsible for damaging various organizations that were running an old version of Microsoft Windows. These attacks were propagated by employing EternalBlue. Some of the other known propagation methods of ransomware attack include; delivering its payload to the victim using Malicious Emails, bypassing the typical access control (Bucbi Ransomware [3]), using Exploit Kits (EKs), Injecting redirect link in JavaScript, Drive-by Download, Waterhole Attacks, and Malvertising [4,5].

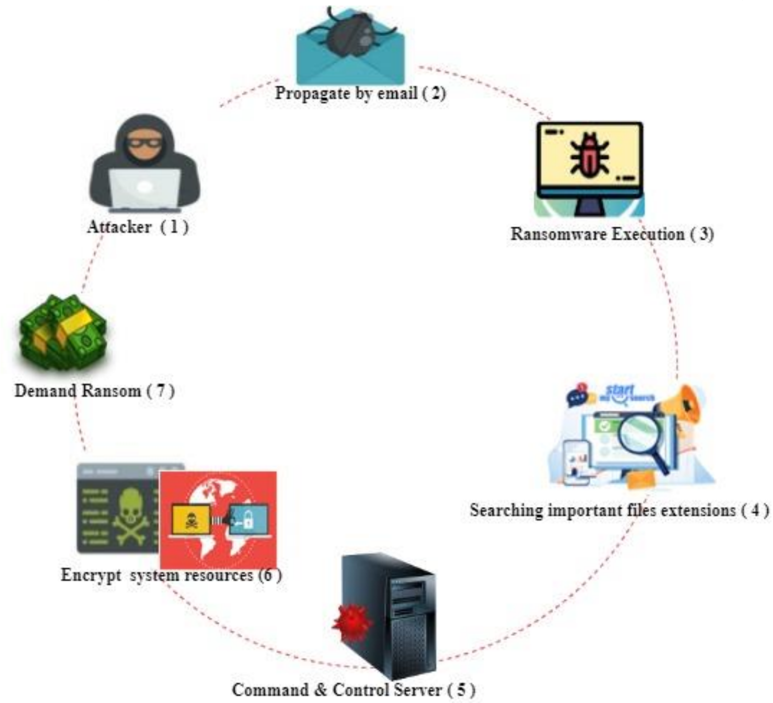

Fig 1: Typical steps of ransomware life cycle

The variants of ransomware can be broadly grouped under two categories: "Locker Ransomware" and "Crypto Ransomware". Locker Ransomware locks the user's system and thus restricts the access to the system files. Its common variants are: Winlock, DM-4 Locker, CTB Locker, Locky Ransomware, and Torren Locker. On the other hand, CryptoRansomware, instead of locking the whole system, encrypts the essential files of the system. Variants of CryptoRansomware include Pack Crypt, Crypt Locker, Dirty Decrypt, Crypto Wall, and Telsa Crypt. Earlier versions of the

Ransomware have used symmetric cryptographic algorithms [6]. However, lately some of its variants use asymmetric cryptographic methods. Most of the recent Ransomwares are using both symmetric and asymmetric cryptographic methods that are not easy to break.

Existing signature-based detection systems are not able to cope with the increasing number of unique ransomware variants [7]. Existing intrusion detection systems (IDS) are based on various analysis method to detect the ransomware. Static analysis based IDS rely on detecting the unique patterns [8, 9]. Commercially available IDS system adopts static analysis as they are fast and inexpensive. However, these static IDS normally fails in detecting the zero-day attack and the polymorphic variations of the attack. In contrast, behavioral based methods focus on the behavioral profile generated at run time. Behavioral based methods are superior to static analysis in detecting zero-day attacks and dealing with the polymorphic variations [10]. However, Behavioral based methods are comparatively slow and cannot detect the metamorphic variations. Anomaly detection based methods relies on modeling the normal connection behavior and detecting deviations [11, 12]. Such type of anomaly detection methods are better in detecting zero-day attack, but may yield high false alarm rate.

Recently, machine learning based IDS are increasingly used due to their excellent learning capabilities and their adaptive nature. Researchers exploits various supervised (Decision trees (C4.5), Support Vector Machines, K-Nearest Neighbor, Naïve Bayes) [13], unsupervised (Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)) [14], Deep Learning (Deep Belief Networks (DBN) [15]), and semi supervised methods [16] to develop intelligent IDS. Each of these methods has its advantages and disadvantages. However, it is not easy to achieve the effective IDS for dynamically changing environment with a single classifier and therefore an ensemble of classifier is more effective. The biggest challenge is to detect zero-day attacks. Zero-day attacks tries to exploit the new vulnerabilities of the victim's system hence nothing is known about them in advance. On the other hand, most of the current machine learning based solutions are dependent upon previous data for detecting future attacks, which is not available in the case of a zero-day attack.

This paper aims to develop a ransomware detection system that can generate an encoding (core features) based description of the zero-day attacks at run time. Furthermore, it can relate the derived description with the known attacks for detection purposes. Finally, to increase the generalization power of the classification system, we have also proposed an ensemble classifier focusing on the reduction of FN under nominal control of FP. In summary, the proposed technique is addressing the challenges mentioned above associated with zero-day attacks through the following contributions:

1. This work presents a novel zero-shot learning (ZSL) framework to detect the zero-day ransomware attack. Current approaches generally use an external source of information for attribute learning, which might be a time-consuming task and not viable for collecting data regarding zero-day vulnerabilities.

2. The proposed technique presents customized deep contractive autoencoder based attribute learning (DCAE-ZSL) for zero-day ransomware. For this purpose, an optimum loss function is learned in an unsupervised manner by optimizing the penalty term for achieving the invariant representation of the known and unknown ransomware. A lower-dimensional DCAE has been designed that forces the model to learn only the essential features of the input data. And compared to traditional ML approaches, it is empirically shown that DCAE based feature extraction effectively performs well against zero-day Ransomware.

3. An ensemble (DCAE-ZSL-HVE) is trained both on original features and derived attributes to address semantic loss for suppressing the intra-family variations and increasing the generalization ability of the classifier on unknown ransomware.

4. The proposed Inference Stage makes cost-sensitive inferences using four simple yet effective combination rules that provide a considerable compromise between FP and FN.

5. The performance of the proposed technique is compared with both the deep CNN models as well as shallow learning models.

The rest of the paper is structured as follows. In Section 2, the related work and its background is presented. The proposed Attribute Learning (AI) and IS (Inference Stage) phases are presented in Section 3. The implementation details of the experiments is presented in Section 4. Discussion and analysis is presented in Section 5. In Section 6, threat to validity of the proposed system is analyzed. Finally, the conclusion and future work are presented in Section 6.

2. Related work and background

This section presents a review of recent ransomware detection techniques and their potential in detecting zero-day attacks. In general existing methods are based on static and dynamic analysis. Static analysis is performed without executing the malware to extract the structural features. Andronio et al. [17] presented a static analysis based HelDroid detection system specifically developed for mobile devices. The model uses only the encryption-based function to detect crypto and locker ransomware. Mercaldo et al. [18] presented a model-based technique for detecting mobile-based crypto and locker ransomware. This method analyzes the bytecode of potential files to inspect only those instructions that are involved in the infection phases. Das et al. [19] presented a model that performs semantic-based feature extraction by grouping API calls of the same resource. These API calls of the same resource are further represented as one feature set. The frequency of these feature sets helps to identify the repetitive actions that differentiates between benign and malware samples. Alsoghyer et al. [20] proposed an application programming interface (API)-based ransomware detection system (API-RDS) for android platform. The proposed method identifies the significant API calls from API packages and achieved 97% accuracy on a self-generated dataset. Although static methods are fast and have a high detection rate, their main focus is ransomware detection rather than the zero-day attack detection. Additionally, these methods are generally unable to detect polymorphic variations and packed families.

On the other hand, dynamic analysis is performed by running malware in a safe environment to extract behavioural features. Kharraz et al. proposed UNEVIL model based on dynamic analysis [21]. The system's main focus was on learning about the access pattern, system file activities, and the entropy of I/O data buffer. Song et al. [22] proposed a detection model that performs detection by monitoring the CPU consumption, memory utilization, files events and I/O usage. Andriono et al. [17] proposed a dynamic analysis to detect the threatening text. Dynamic analysis based methods are useful in detecting polymorphic variations. However, they are unable to detect metamorphic variations. In this regard, few researchers [23–25] have employed a hybrid feature analysis technique to improve the detection of polymorphic and metamorphic attacks. Alberto Ferrante et al. [24] proposed a hybrid method to detect mobile ransomware. Firstly, it examines the potential file using the static method before installation and then observes its runtime behaviour using dynamic analysis. During static analysis, it computes the frequency of opcodes to detect the ransomware attack. While during dynamic examination CPU usage, memory consumption, and network usage are explored to detect malware.

Conventional ML-based methods are useful in performing behavioural analysis. The EldeRan proposed by Sgandurra et al. [26] is based on the files' dynamic behaviour. It uses Mutual Information Gain to extract the most significant dynamic features and then feeds them to Logistic Regression Classifier. Hwang et al. [2] proposed, a two-stage mixed ransomware detection model, based on Markov and Random Forest models. Firstly, their technique builds a Markov model only on the Windows API call sequence pattern then, builds the Random Forest ML model on the remaining behavioral features to control FPR and FNR. However, typically the new attacks may not follow the training distribution to perform covert operations. A different ML-based solution to zero-day attack detection is an anomaly-based method that trains the model only on normal activities, and therefore, anything that is not normal is considered as malicious[27]. In [28], Al-ramy et al. presented a zero-day ransomware detection system using behavioural and data-centric features. Behavioural features are constructed using n-gram technique on the pre-encryption generated features. At the same time, data-centric features are generated by grouping the API calls of the same resources and forming frequency distribution of similar features. After, the feature generation step, important features are selected using information gain measure. Finally, the detection module is stacked using two types of classification. Firstly, it performs behavioural detection using SVM. If the sample is declared as malicious by behavioural detection module, then it is the final step. Otherwise, the decision is put forward to one class anomaly detection SVM for a final decision. However, the anomaly-based zero-day attack detection system may also generate a high false alarm rate.

Current IDS are mostly unable to cope with modifications in the attack's landscape and are also highly dependent on the training dataset. On the other hand, an attacker is always searching for new types of vulnerabilities and exploiting a new weakness in the system, leading to a zero-day attack. In this regard, we exploit a new paradigm of ML known as *zero-shot learning* (ZSL). ZSL-based models can detect unseen objects, in the absence of additional knowledge, by relating the attributes of seen and unseen classes. In literature, interesting techniques have adopted different methodologies [29] to find the semantic embeddings (core features) between seen and unseen classes. These embeddings are generally based on attributes that can be derived manually, discriminatively, using word vectors, mining knowledge from the web, or combining different kinds of embeddings. ZSL can be described as two-phase process. The first phase is the AL phase, and the second phase is the IS. In [30], the authors presented a new IS algorithm for Network IDS. Their main contribution involved an experimental set up for ZSL using of the KDD intrusion detection dataset. Then, in AL phase applied decision tree for extracting rules. In the IS phase, they derived the representation depending on point location in Grassmannian manifold, and explicit distance formula is utilized that finds the shortest distance between the unknown attack and the known attacks. On the other hand, in [31], XiaoZhang et al. presented regression model based ZSL method that fits the regression equation for each category. They then, calculated threshold for all respective categories. In their IS, the test samples' attributes are sequentially substituted to the equations of all categories. Finally, if the resulted calculation meets the criteria for all the corresponding thresholds, the attack is considered a known attack; otherwise, it is considered an unknown attack. Their reported technique can detect unknown intrusion types; such as Hydra-FTP and HydraSSH types of attacks. However, for a Java-Meterpreter, and Meterpreter type's other unknown attacks, its capability may not be satisfactory. In [32], Zhang et al., proposed sparse autoencoders based ZSL method for novel attack detection. It maps known feature space to semantic space, and try to restore the feature space using reconstruction error constraint. The need for detecting new ransomware attacks, together with the competence of CAE for change detection[33], and the competence of ZSL to classify unknown attacks on Zero-shot training data motivated us to use it for the AL stage. Moreover, high generalization ability of ensemble methods and effective combination rules encouraged us to use it for the IS.

3. Material and methods

This section presents the details of the proposed DCAE-ZSL and DCAE-ZSL-HVE methodologies. The DCAE-ZSL is the AL method based on finding robust embedding that can learn the semantic description of zero-day attacks in an unsupervised manner. To learn the context of a zero-day attack, CAE based feature representation and its effect on ZSL is explored using zero-day ransomware test data. Finally, the DCAE-ZSL-HVE based is proposed to improve the generalization power and to find a considerable compromise between FP and FN. The abstract working of the proposed ZSL model is described in Fig. 2.

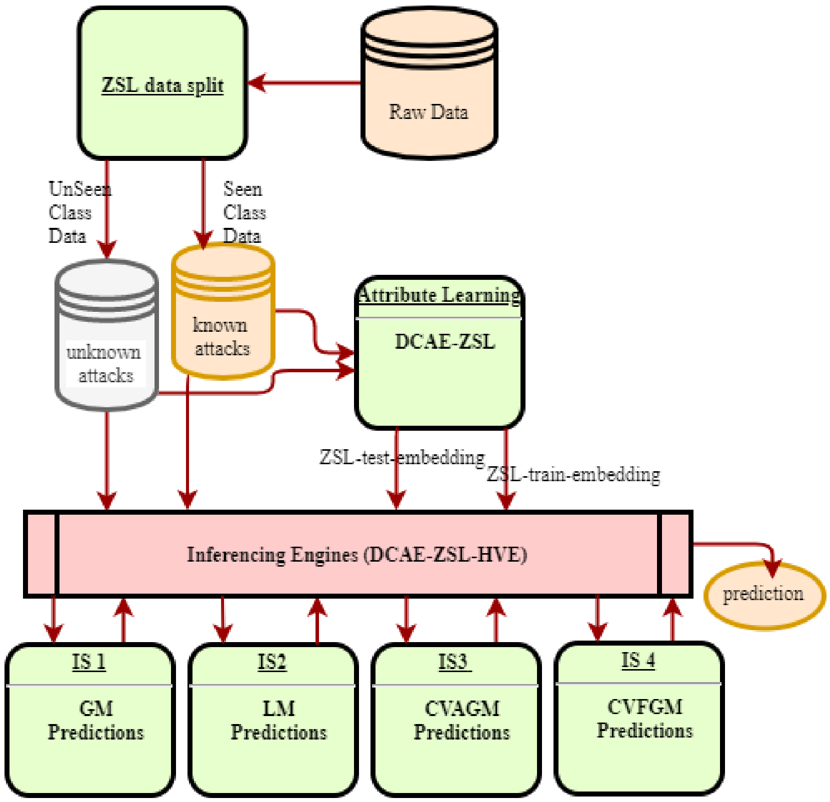

Fig. 2– Framework of the proposed ZSL architecture

3.1. Dataset

The ransomware dataset used in the proposed methodologies is accessed from the home page of Sgandurra et al. [26] using a given link(https://www.danielesgandurra.com/). The data samples were retrieved from the VirusShare4 site in the exe files format in February 2016. Later these samples were analyzed using Cuckoo Sandbox5 to trace the seven basic features in the runtime environment. The dataset consists of 582 ransomware samples and 942 goodware samples. Therefore, the available dataset is highly imbalance in nature. The major attributes of the data are given in Table 1. Collected samples were further manually categorized into 11 different established family's names. The detail of these families is reported in Table 2. The collected ransomware are the most popular variants and mostly are CryptoRansomware. The dataset of goodware is collected from trustworthy sources. Goodware application includes browsers, drivers, emulator, and file utilities like file search, word office tools, games and various other realistic applications of PC. Each sample is analyzed in a sandbox environment for 30 seconds. Although, the authors acquired the PCAP traces by connecting VM (Virtual Machines) with a network. However, only host-based features were collected.

3.1.1. Features and their visualization

The ransomware uses the API calls to write into the other processes to inject them with its peculiar actions or to use it to terminate the other processes. Usually, victim processes are *explorer.exe* or *svchost.exe*. On the other hand, the registry keys operation is important because to ensure the survivor of ransomware after each reboot. Moreover, it keeps track of every key to get the list of mounted devices for exploring more extensions of user files. In literature, ransomware detection is often implemented using API call sequences or registry key operations. Other features like DLL (Dynamic Link Library) are also important as ransomware often links to the DLL of Visual Basic or shell extensions to access the certificates of keys for encryption purpose. Additionally, dropped file features are important as ransomware used it for its notes. The important drooped files are html or RTF. There are in total of 16382 features collected. These features can be further divided into seven broad categories: 1) API represents API invocations, 2) Drop: represents the extensions of the dropped files, 3) REG: involves various registration key operations, 4) Files: include operation related to files, e.g., create or delete files, 5) FILES_EXT: is the extensions of all the files that are involved in dynamic analysis, 6) DIR: files directory activities, and 7) STR: denotes strings embedding. Unlike, the traditional methodologies, the focus of the proposed methods is not on a specific action. They are developed by using dynamic features like registry key operations, API invocation, files extension, file directory operation, drop files monitoring, files operations and embedded strings.

Table 1: Attributes of the dataset

Parameters	Values
Data type	Binary data
Total malicious samples	582
Total benign samples	942
Total samples	1524
Total features	16382
Missing values	None

Table 2: Distribution of the different ransomware families

ID	Family Name	Data Distribution	ID	Family Name	Data Distribution
0	Goodware	942	6	Locker	98
1	Critroni	51	7	Matsnu	60
2	CryptLocker	108	8	Pgpcoder	5
3	CryptoWall	47	9	Reveton	91
4	Kollah	26	10	TeslaCrypt	7
5	Kovter	65	11	Trojan-Ransom	35

3.1.2. Data distribution and its visualisation using t-SNE

In this work, t-Distributed Stochastic Neighbor Embedding, known as t-SNE [34], is used to show the data distribution of various classes. t-SNE maps higher dimensional data to two or three-dimensional space. It is based on SNE optimization and Student's t-distribution. It finds the pairwise similarity matrix between the data points. The t-SNE optimization function maintains the maximum structure of the original mapping. Therefore, in the proposed DCAE-ZSL technique, t-SNE is used to visualize higher dimensional data into two dimensional lower space, as shown in Figures 3 and 4. It is useful to visualize the clusters present in malicious and benign data at various scales. Fig.3. shows the distribution of the ransomware data for the two-class problem. Where the 'cyan' colour is representing the ransomware samples, and the black colour is representing goodware samples. It can be observed that the two classes may overlap at some points, thus indicating the possible similarities between the two categories. There also exist some samples that are far away from their relative classes due to intra-class variations. Fig.4. shows the distribution of the multiple families of ransomware more clearly by using different colours.

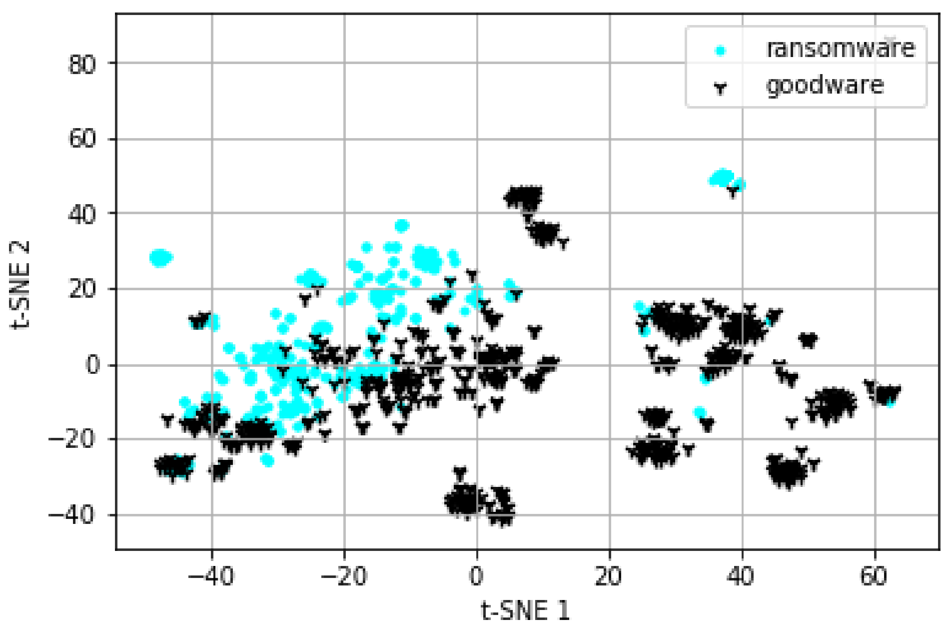

Fig. 3 Data distribution of ransomware vs goodware

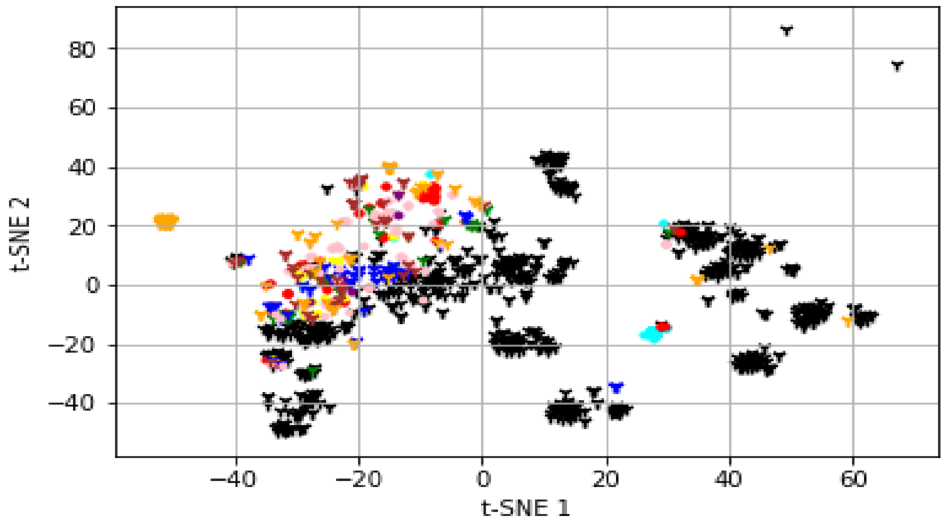

Fig. 4 Data distribution of ransomware families' vs. benign samples. Benign samples are shown in black color

3.2. Zero-shot Learning for zero day attack detection

In general, ZSL [29] deals with predicting classes, which were not part of the training. It is mostly applicable when the training data does not truly represent all the interested class categories. This situation is most viable for IDS due to the growing numbers of zero-day attacks. Correspondingly, in ZSL, classes of training and test instances are different from each other. Classes of training instances are known as seen classes, and test instances are known as unseen classes. ZSL can be described as a two-stage process: AL and IS. Attributes are learned using derived knowledge from some labelled training data or finding some intermediate information to relate the seen and unseen classes. In the IS, the derived knowledge is used to detect the unseen classes. Therein literature, different AL algorithms are presented in the context of IDS, e.g. Attribute Learning for Network Intrusion Detection (ALNID)[31], Graph Embeddings[32], Deep Attribute Prediction (DeepAP)[33], and Grassmannian [24]. ZSL can be classified as Inductive ZSL and Transductive ZSL subject to the available information. Inductive ZSL learned attributes using only seen class information, while Transductive ZSL uses both labelled unseen data and unlabeled unseen data. Based on a test set, ZSL is classified as conventional ZSL (CZSL) or Generalized ZSL (GZSL). CZSL evaluates its model on unseen classes, whereas, GZSL uses both seen and unseen classes in evaluation. The proposed DCAE-ZSL method is the transductive approach. It generalized the model on unknown attacks by utilizing both seen and unseen classes in an unsupervised manner and evaluated using the CZSL method.

3.3. ZSL-Data Split

We firstly performed data partitioning for training and evaluation of the ZSL in ransomware detection related tasks to achieve the desired goals. For this purpose, we split the original data 'D' into two disjoint sets of seen and unseen classes, named as ZSL-train data = $\{X_i^{tr}, Y_i^{tr}\}$ and, ZSL-test data $=\{X_i^{ts}, Y_i^{ts}\}$, respectively. Where 'X_i' is representing the attributes of the samples, and 'Y_i' is representing their respective labels. The proposed system presents the binary classification of ransomware vs goodware(G_i^n). Models are trained on Y^{tr} members and tested on Y^{ts} members. Respective members of Y^{tr} and Y^{ts} are described as sets in (Eq.1 and Eq.2 respectively), where $Y^{tr} \cap Y^{ts} = \varnothing$. In the context of ransomware detection seen classes are known attacks on which models are trained. However, unseen classes are unknown zero day ransomware on which models are tested.

$$Y^{tr} = \{G_i^k, \text{Critroni, CryptLocker, CryptoWall, Kollah, Kovter, Locker, Matsnu}\} \quad (1)$$

$$Y^{ts} = \{G_{k+1}^n, \text{Pgpcoder, Reveton, TeslaCrypt, Trojan-Ransom}\} \quad (2)$$

3.4. The proposed DCAE-ZSL attribute learning method

This module's objective is to generate the class independent description of seen and unseen (zero-day) attacks. Autoencoder can learn useful latent representations without class information. In essence, the learned representation should represent the core features by suppressing unnecessary variations. CAE is used to suppress unnecessary variations. Further, Deep Undercomplete Autoencoder is used to extract the most useful core representation of the data. To build DCAE-ZSL, we have trained ten hidden layers of CAE on ZSL-train data. Where the first five layers are encoding layers, and the remaining five are the decoding layers. However, the DCAE-ZSL technique selected the robust features from the fifth encoding layer comprised of 100 neurons. Overall, the proposed DCAE-ZSL methodology is schematically explained in Fig.5. Implementation details of designed topology are described in section 4.1. Whereas, Table 3 is illustrating the designed topology of the CAE.

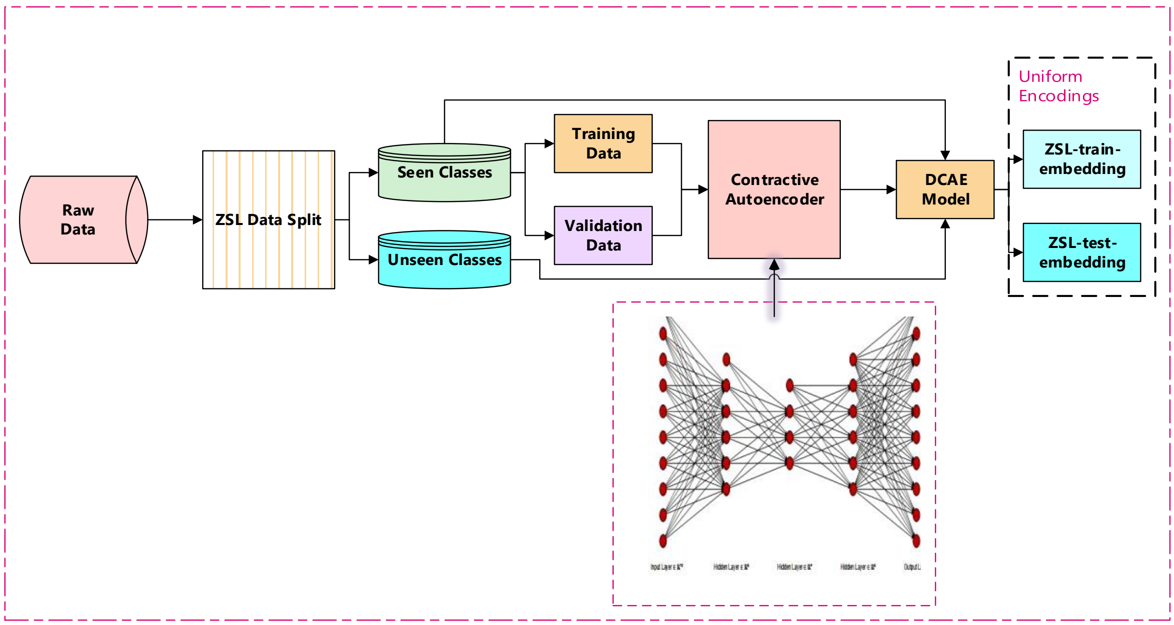

Fig. 5– Overview of the proposed Attribute Learning (AL) architecture (DCAE-ZSL)

Table 3: Number of neurons in encoding and decoding layers of Autoencoder

	E1	E2	E3	E4	E5	D6	D7	D8	D9	D10
Input Unit	16382	4000	2000	1200	600	100	600	1200	2000	4000
Output Unit	4000	2000	1200	600	100	600	1200	2000	4000	16382
Activation Function	Relu	Relu	Relu	Relu	Relu	Relu	Relu	Relu	Relu	Sigmoid

3.4.1. Autoencoder

Autoencoder is an unsupervised neural network that backpropagates by setting its input as its target value [35]. A simple Autoencoder is composed of an input layer, a hidden layer, and an output layer. The Autoencoder's objective is to learn useful hidden representations with minimum reconstruction loss.

$$L = \|x - x'\|^2 \tag{3}$$

Autoencoder based machine learning algorithm is a two-step process that involves pre-training and fine-tuning [36]. Pre-training is unsupervised learning that backpropagates error in a greedy layer-wise manner to reconstruct its input with minimum loss. Pre-training consists of the encoding and decoding layers. In encoding, each layer learns important features from previous information; subsequently, these essential features are the original input's encoded form. The mathematical description of the encoding process thus becomes:

$$Encoding = x' = \partial\left(W_e x + b_e\right) \tag{4}$$

While in decoding layers, each encoded layer is decoded to reconstruct the original features. The process of decoding is mathematically described in Eq. (5):

$$Decoding = x = \partial\left(W_d x' + b_d\right) \tag{5}$$

where, x and x' are representing the original and encoded signals, respectively, W_e is the weight matrix of encoding layers and W_d is the weight matrix for the decoding layers, b_e and b_d are their respective biases, and ∂ is the activation function to add the nonlinearity.

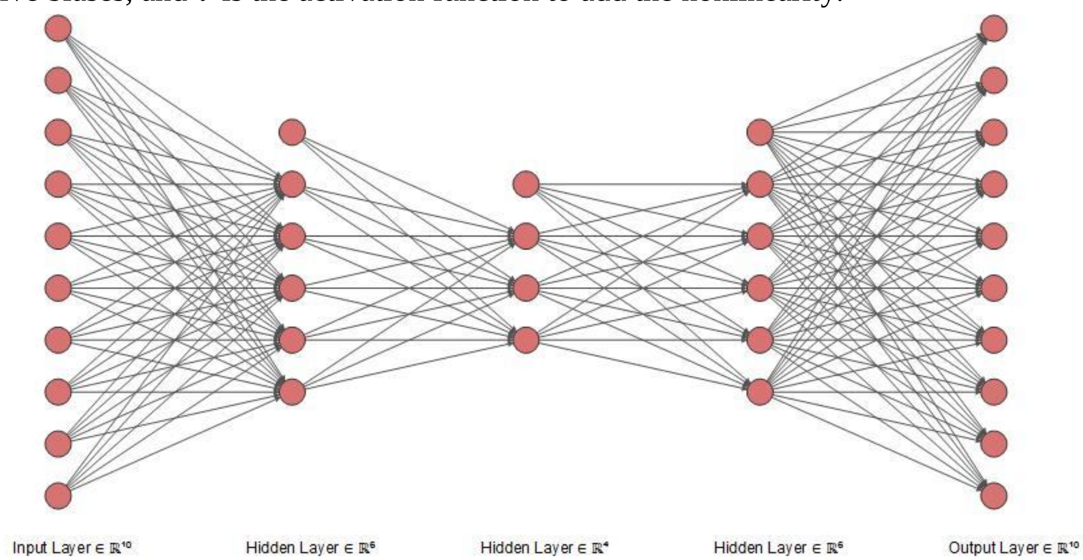

Fig. 6 Undercomplete Autoencoder

3.4.2. Contractive Autoencoder (CAE)

CAE is an unsupervised regularized autoencoders variant[37]. The CAE proposed by Rafie et al. is the extension of the Denoising (DE) autoencoder having the same motivation to produce a robust representation. It allows small perturbation around its training data using an additional penalization term ($\|J_h(\mathrm{x})\|_F^2$) in its generic loss function. This penalization term penalizes the large derivative using a lambda parameter that controls the extent of change in input, w.r.t the learned representation. This penalization term forces the model to learn the uniform representation of the concept (as described in Eqs. (6) and (7)).

$$L = \|x - x'\|^2 + \lambda \|J_h(\mathrm{x})\|_F^2 \tag{6}$$

$$\|J_h(\mathrm{x})\|_F^2 = \sum_{ij} \left(\frac{\partial h_j(\mathrm{x})}{\partial x_i} \right)^2 \tag{7}$$

CAE can learn the non-linear manifold. The common variations present in the data correspond to the manifold's local dimensions. Whereas, the variations that are of rare type will correspond to the orthogonal dimension. The penalization term ensures invariant feature along all dimensions. At the same time, the reconstruction error term offers to reconstruct the input faithfully. In case of directions having strong contractive pressure are the ones, where the input density is sparse. In comparison, the directions with weak contractive pressure raise the input density. Hence, training set direction can resist this contractive pressure.

Alternatively, Contractive mapping can be achieved by weight decaying in the linear case. For non-linearity, it can be achieved by motivating the hidden units to their saturated regime. Sparse autoencoders achieve it by keeping most of its components close to zero. Thus, maintains tiny derivatives in its Jacobian term. Similarly, DE autoencoders indirectly incorporate the robustness in its reconstructions phase. They achieved scholastically by generating the corrupted input to attain identity function, while reconstructing the clean version. Whereas, CAE's are the ones that explicitly encourage robustness in its encoding phase, which is more vital as it penalizes the magnitude of the first derivatives at training data. This property makes it as a suitable choice for feature extraction purposes as compared to others.

3.5. Inference Stage (IS)

In this stage, classes are inferred from the extracted attributes. Therein literature has three well-known IS methods [38]: Probabilistic frameworks [39], energy function [38, 40], and the K-Nearest Neighbor (K–NN). The probabilistic framework's most general form combines the derived attributes with original attributes to find the target class. However, it has two variants: (i) Directed Attribute Prediction (DAP) and Indirected Attribute Prediction (IAP). DAP method builds the learning model for each attribute of AL stage. These learning models are then used in IS to predict new classes using attribute signature. IAP creates the learning model for each training class. At the evaluation stage, the prediction from each training class tempt labelling of the attribute layer, which helps to infer the test class labels[41]. The K-Nearest Neighbour (K–NN) based methods rely on distance measures to find the closest match of input instance to the attribute instance derived in AL stage. The proposed IS method learns various models e.g.: Random Forest (RF), Logistic Regression (LR), Support Vector Machine (SVM), Gaussian Naïve Bayes (GNB) on attributes learned in DCAE-ZSL stage and on original attributes. At test time, the two feature representation schemes' inferences are combined using GM, LM, CVAGM, and CVFGM on unseen classes of attack. The proposed (DCAE-ZSL-HVE) ensemble scheme with its combination rules is described below.

3.5.1. Proposed ensemble (DCAE-ZSL-HVE) based Inference Stage

In this paper, we applied different voting-based ensemble approaches to infer unknown ransomware. Firstly, the Global Majority (GM) based method selects the majority votes from all the decision models considering them as individual decision spaces. Algorithm 1 describe the GM procedure. Besides, two decision spaces are distinctly defined based on features representation scheme in the Local Majority case (Algorithm 2 show the LM procedure). With this voting-based mechanism, a winner class label is selected based on the local majority of any single decision space. It is experimentally observed that multiple base learners can achieve greater generalizability. However, not all the base learners can yield effective classification performance in a practical pattern recognition task regarding the diversity or pseudo-independent nature of base learners. Some base learners may have adverse effects on the ensemble learner. Therefore, we carried out a model selection by finding the cumulative vote-against value for each model. Top *'n'* models with the minimum cumulative vote against them are selected to form a single decision space in case of global majority voting (CVAGM). Similarly, in the case of the proposed CVFGM cumulative vote for value is calculated for each model. Then select the top 'n' models with maximum cumulative vote-for value. Detailed procedure of the combination schemes CVFGM and CVAGM is described in Algorithm 3. The aim was to reduce the FN by increasing the generalization of ensemble and giving more weightage to FN. Overview of the proposed (DCAE-ZSL-HVE) architecture is described in Fig.7.

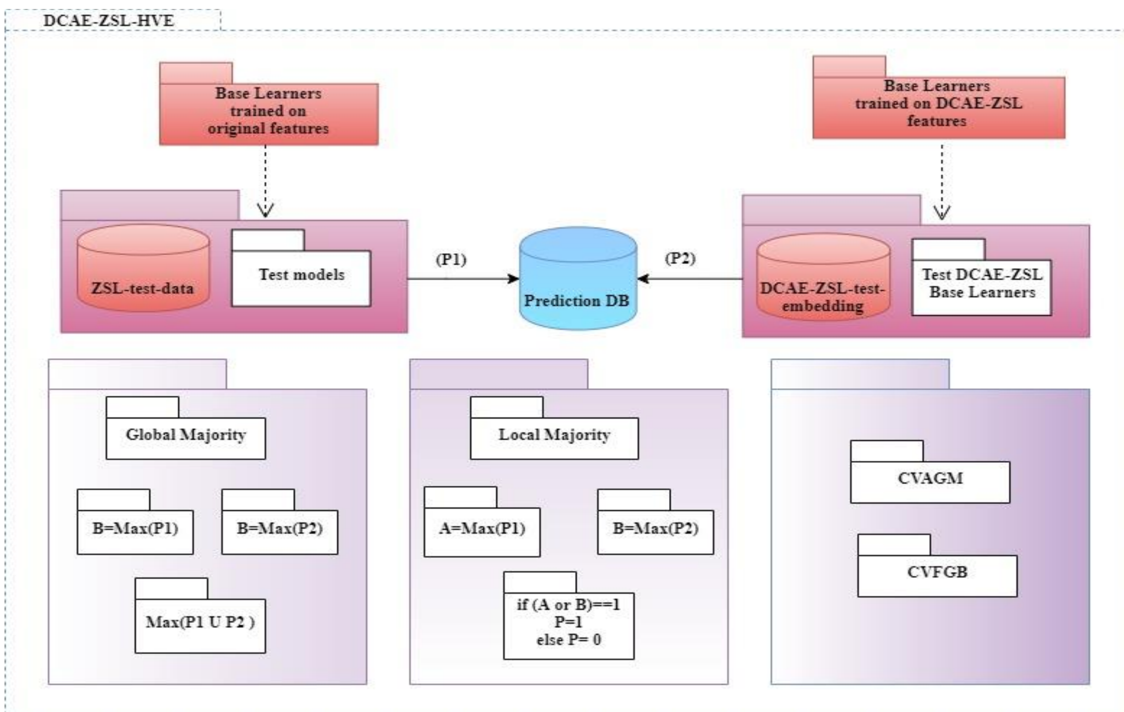

Fig. 7– Overview of the proposed (DCAE-ZSL-HVE) Inference Stage (IS) architecture

Algorithm 1 : Local Majority based Combination scheme

Input:Local-DB1:*m x n dimensional, prediction matrix of base estimators trained on original features*

Local-DB2:*m x n dimensional, prediction matrix of base estimators trained on contractive features*

Output : *Final-Prediction*

1. *n ← no-of-estimators;*
2. *m ← test-sample-size;*
3. *Final-prediction ← [];*
4. ***for*** *r in* ***range****(m):*
5. **A=Max(**Local-DB1[r, :]) ;
6. B=**Max(**Local-DB1[r, :]) ;
7. ***if*** (A==1 **or** B==1)***;***
8. *Final-prediction[r] ← 1;*
9. ***else***
10. *Final-prediction[i] ← 0;*
11. ***end for***
12. ***return*** *Final-Prediction;*

Algorithm 2 : Global Majority based Combination scheme

Input:Local-DB1:*m x n dimensional, prediction matrix of base estimators trained on original features*

Local-DB2:*m x n dimensional, prediction matrix of base estimators trained on contractive features*

Output : *Final-Prediction*

1. *n ← no-of-estimators;*
2. *m ← test-sample-size;*
3. *Final-prediction ← [];*
4. Global-DB =Local-DB1 **U** Local-DB2;
5. ***for*** *r in* ***range****(m):*
6. *Final-prediction[r,:] ←* **Max[**Global-DB [r,:]]***;***
7. ***end for***
8. ***return*** *Final-Prediction;*

Algorithm 3 : Commutative Vote For & Cumulative Vote Against based Global Majority Combination Schemes

Input:Local-DB1: *m x n dimensional, prediction matrix of base estimators trained on original features // The prediction matrix is in the form of zero and one ,where '1' is representing ransomware and '0' is representing goodware.*

Local-DB2: *m x n dimensional, prediction matrix of base estimators trained on contractive features*

Output : *Final-Prediction*

```
n ←no-of-estimators;
m ←test-sample-size;
k←No-of-top-classifiers;
Global-DB =Local-DB1 U Local-DB2;
for r in range(m): // Traverse Global-DB
    cumulative_vote_for ←[];
    cumulative_vote_against ←[];
    for i in range(n):
        No_of_Vote_for [r ,i] ←Count no of predictions similar to the ith base estimator;
        No_of_Vote_against [r ,i] ← Count predictions dissimilar to the ith base estimator;
    end for
end for
for j in range(m):
    for t in range(n):
        cumulative_vote_for[t]= sum( No_of_Vote_for [j,:]) ;
        cumulative_vote_against[t]= sum( No_of_ Vote_against [j,:]);
    end for
end for
Select top k base learners from cumulative_vote_for matrix;
Apply majority voting on selected base learners;
return Final-Prediction;// predictions by CVFGM;
Select top k base learners from cumulative_vote_against matrix;
Apply majority voting on selected base learners;
return Final-Prediction;// predictions by CVAGM;
```

3.6.Performance comparison with state of the art Deep Models

Deep learning (DL) techniques are becoming famous for enhancing the performance of network intrusion detection systems (NIDS)[42]. However, DL based detection models are highly dependent upon large amounts of labelled training data, often unavailable. In this comparison, we investigate the potential of transfer learning (TL) in detecting zero-day ransomware. TL is an approach that enables the transferring of learned features from the source domain to the target domain, especially in cases where the target domain samples are less in number. We compare the performance of the proposed model with several deep models trained using TL, e.g. ResNet50[43], GoogleNet (Inception-V1)[43], and Inception-V3[44]. We used the pre-trained model trained on the imageNet dataset and then fine-tuned the model on ZSL-train data. The implementation details of these architectures are provided in the section 4.2. These TL based deep CNN architectures are customized to make them applicable for ransomware image dataset by adjusting new input layer as per the dimensions of the targeted ransomware image data (224X224X3). Similarly, the last fully connected layer of the standard architecture is replaced with two neurons to classify the goodware and ransomware samples using softmax function at the last layer

On the other hand, the remaining convolutional blocks are kept unchanged. These models' weight space is optimized using the backpropagation algorithm to minimize the cross-entropy-based loss function. We show how effective are the TL based detection models in detecting zero-day attacks, when there is no information available, compared to the proposed and shallow learning methods. GoogleNet is a Deep CNN architecture that employs inception block to transforms the image representation at multiple scales using multi-resolution filters. However, Inception-V3 replaces these multi-resolution filters with asymmetric filters to make them computationally efficient. ResNet improves the optimization strategy for fast convergence using skip connections.

4. Implementation details

All the experiments were carried out using Microsoft Window7 professional, 16.0 GB memory, X64-based PC, 64-bit operating system. The core coding modules are implemented using python version 3.6. However, the base learners SVM, RF, GNB and LR were developed using Scikit library. Proposed CAE topology is designed using Keras library. In addition, pillow and t-SNE libraries were also used.

4.1. Parameter settings of baseline models and Deep Contractive Autoencoder

Parameter setting involves the parameters of the DCAE and different models, including RF, GNB, SVM, and LRC. To extract DCAE bottleneck features, CAE is trained on the ZSL-train set. To optimize the parameters, 15% of the total data is reserved as validation data. Table 4 shows the values of the parameters, which are set during the training phase of DCAE. CAE is trained using a contractive-loss function that takes lambda =0.000001. Different conventional learning model’s optimization is carried out using 5-cross validation on precision and recall due to the data's imbalanced nature.

Table 4: Parameter setting of Autoencoder

Parameters	Values
Total encoded layers	4 + 1
Total decoded layers	5
Batch size	2
Encoding dimension	100
Total layers	10
Lambda	0.000001

4.2. Parameter settings of TL based CNN models

TL-based CNN models' parameter setting involves its hyperparameters settings like Epochs: 100, learning rate: 0.0001, batch size: 5, and momentum: 0.95. Furthermore, the train, test and validation data are kept the same for comparison purpose both for the baseline models and the proposed technique. TL-based Deep CNNs models' training is optimized using SGD optimizer to minimize the cross-entropy loss.

4.3 Performance evaluation

To evaluate the proposed DCAE-ZSL technique's effectiveness, the different performance measures used in various experiments are recall, accuracy, precision-recall curve (PR Curve), and AUC-PR. PR Curve is a plot between precision and recall by varying threshold. The mathematical description of the used performance measures is given below in Eqs. (8-10).

$$DetectionRate = \text{Precision} = \frac{TP}{TP+FP} \tag{8}$$

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \tag{9}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{10}$$

The positive (P) and negative (N) represent two classes, in the current problem, the positive class is the ransomware, and the negative is goodware. The objective is to increase TP value and decrease FN value, as the malicious class detection is more important in this scenario.

5. Results and discussion

5.1. Effectiveness of the proposed DCAE-ZSL attribute learning method

The objective of this experiment is to show that the robust features extracted through the bottleneck layer of Undercomplete CAE can work well against zero-day attacks. Table 5 shows the result of zero-day attack detection using a full set of original features. In addition, the results of zero-day attack detection using robust and reduced features extracted through the proposed DCAE-ZSL technique. It illustrates the obtained results in terms of the recall and accuracy. DCAE-ZSL-RF has shown significant improvement in recall (0.85) and accuracy (92.8) as compared to recall (0.79) and accuracy (81.0) of the baseline RF model that was trained on the original features. Similarly, DCAE-ZSL-GNB and DCAE-ZSL-SVM have shown considerable improvement in terms of the recall and accuracy, as compared to the baseline GNB and SVM model. DCAE-ZSL-LR performance has also shown improvement in terms of the recall and is more sensitive in detecting the positive class than the baseline LR model. Overall it can be observed that features extracted through the proposed technique can enhance the ZSL capabilities of conventional learning algorithms.

Table 5: Different classifiers' result on original and transformed feature space (proposed DCAE-ZSL) on test set.

Attributes Type	Learning Models	TN	FP	FN	TP	Recall	Accuracy(%)
Original	**RF**	112	21	28	106	0.79	81.0
Proposed DCAE-ZSL		133	0	20	115	**0.85**	**92.8**
Original	**GNB**	105	28	47	87	0.64	71.9
Proposed DCAE-ZSL		84	49	11	123	**0.91**	**77.5**
Original	**SVM**	130	3	28	106	0.79	88.4
Proposed DCAE-ZSL		120	13	13	121	**0.90**	**90.3**
Original	**LR**	132	1	25	109	0.81	90.4
Proposed DCAE-ZSL		121	12	14	120	**0.90**	**90.3**

5.2 Effectiveness of the proposed ensemble (DCAE-ZSL-HVE) based Inference Stage

We present the extensive experimental results for different ensemble configurations using different learning models (RF, DCAE-ZSL-RF, SVM, DCAE-ZSL-SVM, LR, DCAE-ZSL-LR, DCAE-ZSL-GNB and GNB) trained distinctly on two feature representations in Table 6. The bold values indicate the best values for the each method used. Table 6 shows the performance of diverse voting approaches that define the different combination rules on the dynamic behavior dataset. The highest recall performance is obtained by using the local majority (i-e 0.95). This combination rule and ensemble scheme provide a 92.8% accuracy rate. Besides, the global majority based combination yields the second greatest results in terms of recall (i-e. 0.91). So, we conclude that the local majority can gain a recall measure for ransomware detection. The performance of the four different proposed ensemble is evaluated using recall. FP, FN, accuracy and F1 show that each inferencing method brings improvements than the individual learners.

Table 6: Comparison of DCAE-ZSL-HVE with four different proposed combination rules in terms of Recall, Accuracy, and F1 on test data

IS Methods	Combination Schemes	TN	FP	FN	TP	Recall	Accuracy (%)	F1
Proposed Method 1	GM	129	4	11	123	0.91	**94.3**	**0.94**
Proposed Method II	LM	120	13	**6**	**128**	**0.95**	92.8	0.93
Proposed Method III	CVAGM	**131**	**2**	21	113	0.84	91.0	0.91
Proposed Method IV	CVFGM	120	13	11	123	0.91	91.0	0.91

5.3 Performance analysis using recall

Fig.8. shows the recall measure comparison of, baseline models, proposed DCAE-ZSL based models and the proposed ensemble methods. From the figure, it can be seen that the proposed DCAE-ZSL transformation brings improvement in terms of recall for all learning models. Moreover, the proposed ensemble scheme further improves the recall rate. The results of all combination rules except Method III are better than the base learners trained on the two representation schemes. However, the best results (0.95) in terms of recall are obtained by using the local majority as a combination rule.

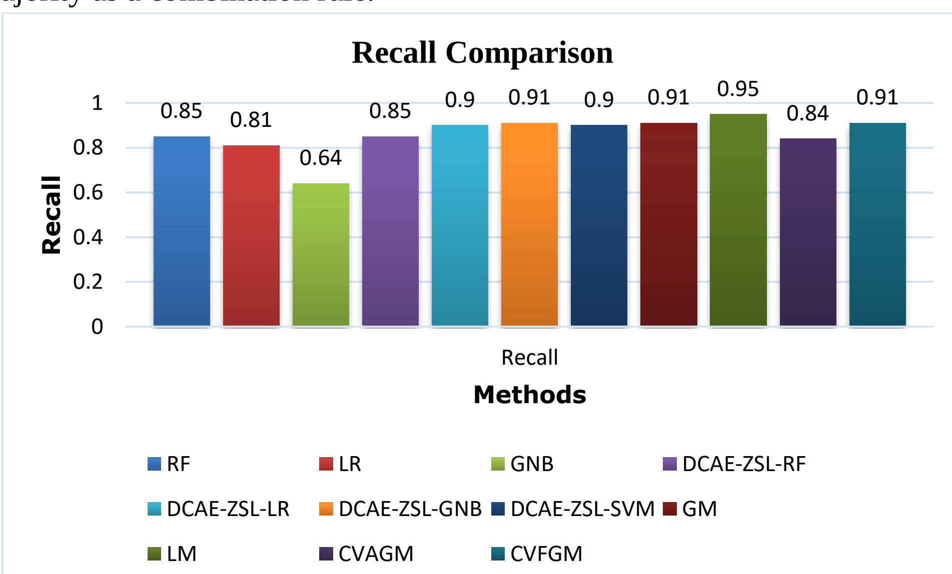

Fig. 8– Performance comparison in terms of recall

5.4 Performance analysis using FP and FN

In this section, we further evaluate the FP and FN error measures. Fig. 9 shows the True Positive (TP), and the FP values, of the classifiers trained on original features, proposed DCAE-ZSL transformation and trained on proposed ensemble scheme with four different combination rules. The figure shows that the lowest FP is achieved by the proposed DCAE-ZSL-RF model (FP=0). As this research work aims to decrease the FN value, it can be observed that it is decreasing with each proposed module. The lowest FN is achieved using the proposed ensemble scheme with the local majority combination rule (i.e., FN =6). FN values of all the classifiers that are trained on the roposed transformation are better than the baseline models which are trained on original features. When applied the proposed ensemble, it further decreasing the FN values.

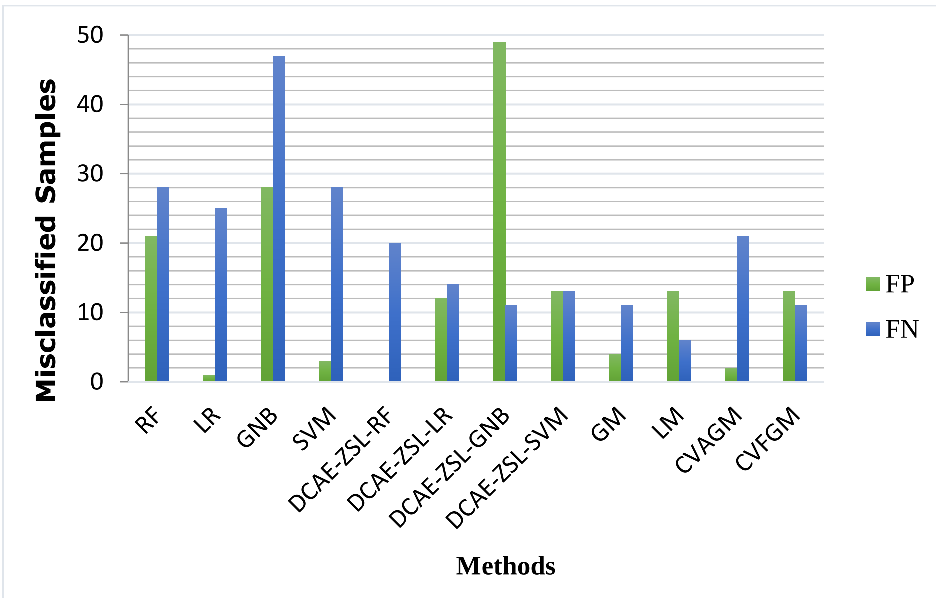

Fig. 9– Performance comparison in terms of FP and FN values

5.5 Detection rate analysis at multiple threshold levels

The ransomware attack is one of the most dangerous cyber-attacks due to its common effects on internet resources. Therefore, false alarms pose less cost as compared to a FN. Hence, we optimized the recall by compromising precision, as this compromise is desirable. Fig.10. (a) and (b) are showing the PR curves of baseline and proposed DCAE-ZSL techniques. It can be observed that when the recall threshold is less than 0.8, the baseline is performing better. However, as the baseline approaches 0.8, the proposed techniques outperform both in terms of precision and recall. Even below 0.8 recall threshold values, results are comparable.

Original Attributes(Full Set) **Proposed DCAE-ZSL Attributes(100 Features)**

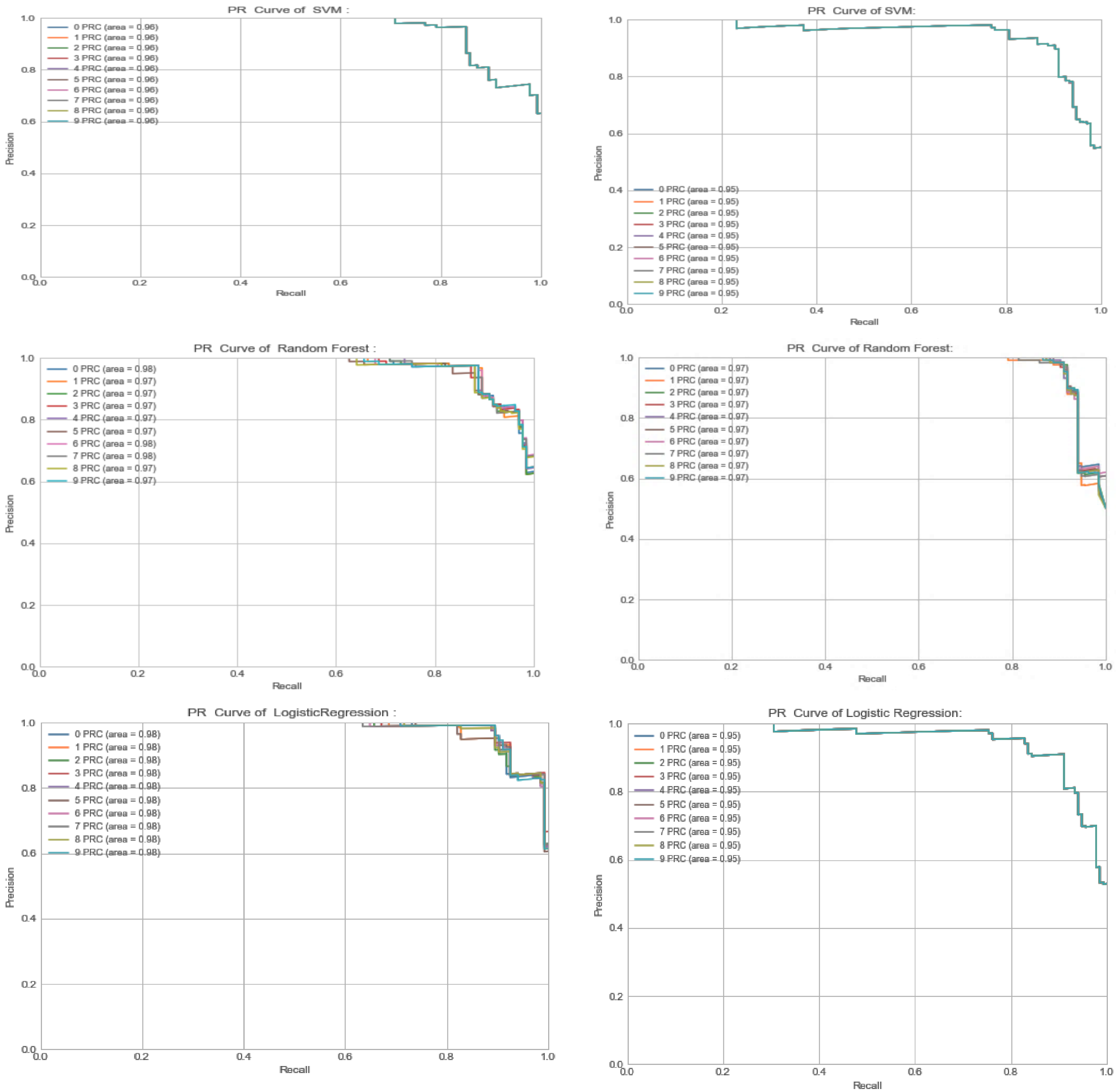

Fig. 10– PR curve of baseline vs proposed technique

5.6 Performance Comparison with state of the art techniques

Table 7 shows the comparison between the proposed method's performances and fine-tuned CNN models on ZSL-test data. The performance of the proposed DCAE-ZSL-HVE is compared with three classification models (GoogLeNet, Inception- V3, ResNet-50) on the same data using the same train test distribution. These classification models are now in trend to solve a complex problem like intrusion detection[45]. The Performance analysis of TP, FN and recall values suggests that proposed DCAE-ZSL-HVE models learn the ransomware specific feature better than GoogLeNet, Inception- V3, ResNet-50. In comparison, the fine-tuned CNN models show slightly better performance in learning goodware specific features but at the cost of a high false alarm rate (minimum 23 by Inception- V3). However, the proposed DCAE-ZSL-HVE generates only 13 false alarm. This may be because GoogLeNet, Inception- V3, ResNet-50 are data-hungry classifiers and are biased towards majority class samples.

Fig.11. depicts the Bar Chart graph comparison among best performers of baseline models (i-e LR), Deep CNN models (i-e ResNet50), the proposed IS (i-e LM) in term of error measures (FP and FN), recall value and number of misclassified samples (MCS). The number of MCS is calculated by summing the error measures FP and FN. It can be observed that the values of all metrics using the proposed ensemble-based IS (LM) is better than all the other reported results, except FP for baseline LR. However, LR yields high MCS than the proposed IS (LM). This indicates that the proposed technique achieves a considerable compromise between FP and FN. Moreover, it attains the lowest number of MCS. Therefore the proposed approach is well designed to detect zero-day ransomware.

Table 7: Comparison of the proposed DCAE-ZSL-HVE with Deep Learning models in terms of Recall, and, Accuracy on zero-day attacks based test data

Techniques	**TN**	**FP**	**FN**	**TP**	**Recall**	**Accuracy (%)**
GoogleNet[46]	127	33	**6**	101	0.81	85.3
Inception V3[44]	127	23	6	111	0.84	89.1
ResNet50[43]	126	25	7	109	0.91	88.0
Proposed DCAE-ZSL-HVE (LM)	120	**13**	**6**	**128**	**0.95**	**92.8**

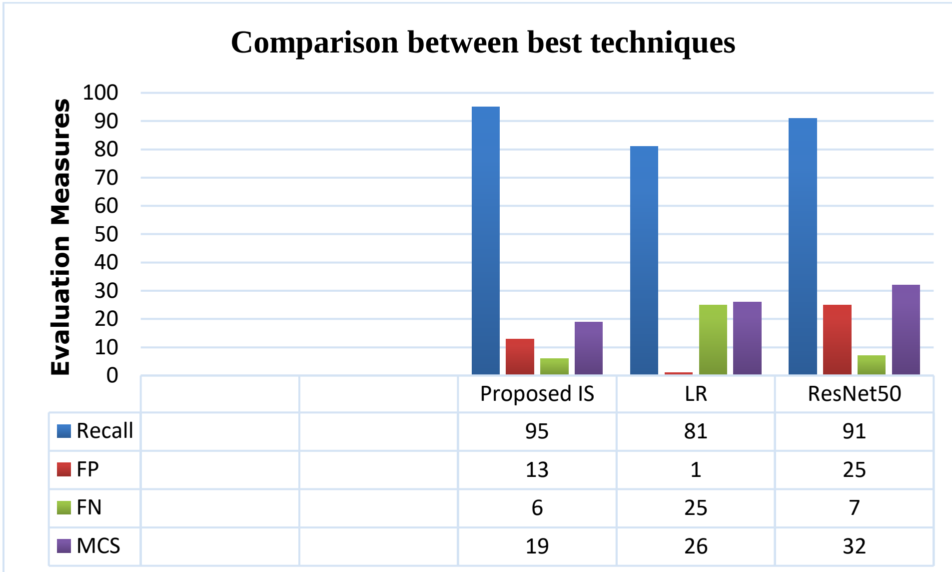

Fig. 11– Comparison of the proposed method with best performing state of the art method

To evaluate the performance of the proposed methods against other detection systems, we investigate different approaches from the literature for ransomware detection. Table 8 shows the comparison between DCAE-ZSL and EldeRan [26], evaluated using ZSL-test-data given in section 3.3. Both techniques are compared in terms of the detection rate on 100 features. However, the proposed method is using less training data as compared to the existing technique. The proposed DCAE-ZSL technique does not include the remaining test classes in training while evaluating one family from ZSL-test data. The results show that the proposed technique is better in detecting Pgpcoder, Trojan-Ransom and Reveton than the existing method.

Table 8: Detection rate comparison with existing EldeRan[26] on test data

Family	No of ransomware samples	Detection Rate (100 features) EldeRan	Detection Rate (100 features) DCAE-ZSL-HVE
Pgpcoder	4	0.75	**1**
Reveton	90	0.88	**0.96**
TeslaCrypt	6	1.00	0.92
Trojan-Ransom	34	0.94	**0.95**

Table 9 shows the comparison with the existing techniques in term of recall measure. We compared the recall performance of proposed DCAE-ZSL_HVE (0.95) with some of the existing methods namely , C4.5(0.52)[13], KNN(0.70)[13], DNA Sequencing Engine(0.82) [47]and Anomaly(0.89) [27]. DNA Sequencing Engine is using the same dataset, but train test split may not be the same.

However, its testing is carried out on the known type of ransomware. Similarly, methods such as C4.5[13] and KNN[13] are tested on known types of ransomware. Results show that the proposed method is outperforming even on unknown test data compared to the existing methods that show performance on known attacks. These results show that intra-family variations of ransomware can be suppressed effectively using the proposed DCAE-ZSL technique. It also shows that by efficiently integrating the decision of base learners trained on two different representation can perform well than the single base learner.

Table 9: Recall comparison with existing techniques on test data

Techniques	Type	Recall	Test data
Proposed DCAE-ZSL-HVE (Local Majority)	Heterogenouas Ensemble	**0.95**	Unknown attacks
Saleh et al. [13]	KNN	0.70	Known attacks
Saleh et al. [13]	C4.5	0.52	Known attacks
Al-rimy et al.[27]	Anomaly	0.89	Unknown attacks
Sgandurra et al. [26]	EldeRAN	**0.93**	UnKnown attacks

Table 10. shows the qualitative comparison using some significant characteristics between the proposed method and the most recent ransomware detection methodologies. The quantitative comparison is difficult to perform due to lack of availability of any benchmark dataset, different test environments, and due to the difference in objectives of developing the detection systems. The proposed method uses full set dynamic features to capture all malicious API sequences, file access, registry access, drop files, file extension search, and threatened text instead of focusing on single event. Further, we have extracted the core semantic embeddings that utilizes the subset of these events. Further we objectively trained and tested the proposed ensemble model to detect zero-day ransomware.

Table 10: Comparison of the proposed method with recent Ransomware detection techniques

Techniques	year	Ensemble	Features Type	Zero-day Attack Detection	Cost Sensitive Approach	Classification Type
Proposed	2021	yes	Entire dynamic features	Yes	yes	Binary
B.Zhang *et al.*[48]	2020	No	N-gram opcodes	No	No	Family
Al-rimy *et al.*[49]	2019	yes	Dynamic API Calls	No	No	Binary

R.Vinayakumar *et al.*[50]	2019	No	User centric data	No	No	Family
S.Maniath *et al.*[51]	2017	No	Dynamic API Calls	No	No	Binary
S.Homayoun[52]	2019	yes	Sequences of events	Yes	No	Family

6. Threats to validity

While developing the framework for zero-day ransomware detection, the proposed framework has some threats to validity which are defined below:

Construct validity: In the present research, the developed framework for ransomware detection only discriminate between the goodware and ransomware of local machine, but does not include the verification of network traffic with server. Moreover, family classification of ransomware is not performed.

Internal validity: Another threat is the homogeneity of the data used. However, in this current work, the data has been collected from diverse sources and thus it is largely hetrogenous.

External Validity: In this work, we considered 11 different ransomware families, while for training we used seven families and the remaining four families are used to evaluate the model for zero-day ransomware detection. Additionally, the work can be extended to train the framework with more ransomware families, which then can become more proficient to detect the real world ransomware. Another external threat to validity is timely detection, however dynamic analysis is a time taking process.

7. Conclusion and future work

This paper presents a two-stage ransomware detection system using concepts of ZSL, DCAE, and Ensemble learning. The first stage is an AL (Attribute Learning) phase, whereby, a novel DCAE-ZSL technique is proposed to learn the uniform latent semantic embeddings both for known and unknown attacks. The learned latent space also penalizes the input for little changes and can focus on major similarities between known and unknown classes. Thus, it is able to learn only the core transformation that results in performance improvement against zero-day attack detection. Through different experiments, it is observed that DCAE-ZSL based transformed representation outperforms the traditional machine learning approaches against zero-day attacks. Then, base learners are trained on the original features and on features extracted through the proposed DCAE-ZSL method. Finally, the second stage of the proposed framework consists of the IS (Inference Stage) that implements four combination rules to obtain the final prediction results. The prediction results by K-NN, DNA Sequencing Engine and Anomaly based methods shows that our method can provide good recall score as compared to these methods. Further, the proposed method shows better performance in detecting Pgpcoder, Trojan-Ransom, and Reveton families than the existing method [20] in terms of precision. Moreover, results show that the proposed DCAE-ZSL-HVE methods achieve a considerable compromise between FP and FN as compared to the conventional baseline models. Hence, we conclude that the invariant and reduced feature representation of the original features can efficiently detect new classes at the test time.

In this study, the developed system for zero-day ransomware detection only detects either an exe file is a goodware or ransomware. In future, this work can be extended to identify the family of the ransomware. Similarly, in future CAE can also be used to extract the pre- encryption based discriminant features for an early detection.

Acknowledgment

The authors would like to thank Pattern Recognition lab at Pakistan Institute of Engineering and Applied Sciences for providing them healthy research environment and computational facilities.

References

1. Al-rimy BAS, Maarof MA, Shaid SZM (2018) Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions. Comput. Secur.
2. Hwang J, Kim J, Lee S, Kim K (2020) Two-Stage Ransomware Detection Using Dynamic Analysis and Machine Learning Techniques. Wirel Pers Commun. https://doi.org/10.1007/s11277-020-07166-9
3. Bilge L, Dumitras T (2012) Before we knew it: An empirical study of zero-day attacks in the real world. In: Proceedings of the ACM Conference on Computer and Communications Security
4. Sood AK, Enbody RJ (2011) Malvertising - Exploiting web advertising. Comput Fraud Secur. https://doi.org/10.1016/S1361-3723(11)70041-0
5. McDowell K (2006) Now that we are all so well-educated about spyware, can we put the bad guys out of business? In: Proceedings of the 34th Annual ACM SIGUCCS Fall 2006 Conference, SIGUCCS '06
6. Kharraz A, Robertson W, Balzarotti D, et al (2015) Cutting the gordian knot: A look under the hood of ransomware attacks. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)
7. Masdari M, Khezri H (2020) A survey and taxonomy of the fuzzy signature-based Intrusion Detection Systems. Appl. Soft Comput. J.
8. Medhat M, Gaber S, Abdelbaki N (2018) A new static-based framework for ransomware detection. In: Proceedings - IEEE 16th International Conference on Dependable, Autonomic and Secure Computing, IEEE 16th International Conference on Pervasive Intelligence and Computing, IEEE 4th International Conference on Big Data Intelligence and Computing and IEEE 3
9. Jung S, Won Y (2018) Ransomware detection method based on context-aware entropy analysis. Soft Comput. https://doi.org/10.1007/s00500-018-3257-z
10. Zakaria WZA, Abdollah MF, Mohd O, Ariffin AFM (2017) The rise of ransomware. In: ACM International Conference Proceeding Series
11. Lin SW, Ying KC, Lee CY, Lee ZJ (2012) An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. Appl Soft Comput J. https://doi.org/10.1016/j.asoc.2012.05.004
12. Tajoddin A, Abadi M (2019) RAMD: registry-based anomaly malware

detection using one-class ensemble classifiers. Appl Intell. https://doi.org/10.1007/s10489-018-01405-0
13. Saleh MA (2020) Evaluation of Supervised Machine Learning Classifiers for Detecting Ransomware based on Naïve Bayes, SVM, KNN, C 4.5, andRandom Forest Algorithms
14. Ashit D (2016) Detection of Malware and Malicious Executables Using E-Birch Algorithm. Int J Adv Comput Sci Appl. https://doi.org/10.14569/ijacsa.2016.070118
15. Tian Q, Han D, Li KC, et al (2020) An intrusion detection approach based on improved deep belief network. Appl Intell. https://doi.org/10.1007/s10489-020-01694-4
16. Idhammad M, Afdel K, Belouch M (2018) Semi-supervised machine learning approach for DDoS detection. Appl Intell. https://doi.org/10.1007/s10489-018-1141-2
17. Andronio N, Zanero S, Maggi F (2015) HELDROID: Dissecting and detecting mobile ransomware. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)
18. Mercaldo F, Nardone V, Santone A, Visaggio CA (2016) Ransomware steals your phone. Formal methods rescue it. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). https://doi.org/10.1007/978-3-319-39570-8_14
19. Das S, Liu Y, Zhang W, Chandramohan M (2016) Semantics-based online malware detection: Towards efficient real-time protection against malware. IEEE Trans Inf Forensics Secur. https://doi.org/10.1109/TIFS.2015.2491300
20. Alsoghyer S, Almomani I (2019) Ransomware detection system for android applications. Electron. https://doi.org/10.3390/electronics8080868
21. Kharraz A, Arshad S, Mulliner C, et al (2016) Unveil: A large-scale, automated approach to detecting ransomware. In: Proceedings of the 25th USENIX Security Symposium
22. Song S, Kim B, Lee S (2016) The Effective Ransomware Prevention Technique Using Process Monitoring on Android Platform. Mob Inf Syst. https://doi.org/10.1155/2016/2946735
23. Aydin MA, Zaim AH, Ceylan KG (2009) A hybrid intrusion detection system design for computer network security. Comput Electr Eng.

https://doi.org/10.1016/j.compeleceng.2008.12.005
24. Ferrante A, Malek M, Martinelli F, et al (2018) Extinguishing ransomware - a hybrid approach to android ransomware detection. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)
25. Netto DF, Shony KM, Lalson ER (2018) An Integrated Approach for Detecting Ransomware Using Static and Dynamic Analysis. In: 2018 International CET Conference on Control, Communication, and Computing, IC4 2018
26. Sgandurra D, Muñoz-González L, Mohsen R, Lupu EC (2016) Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection
27. Al-rimy BAS, Maarof MA, Prasetyo YA, et al (2018) Zero-Day Aware Decision Fusion-Based Model for Crypto-Ransomware Early Detection. Int J Integr Eng 10:. https://doi.org/10.30880/ijie.2018.10.06.011
28. Al-rimy BAS, Maarof MA, Shaid SZM (2018) A 0-Day Aware Crypto-Ransomware Early Behavioral Detection Framework
29. Sun X, Gu J, Sun H (2020) Research progress of zero-shot learning. Appl Intell
30. Rivero J, Ribeiro B, Chen N, Leite FS (2017) A grassmannian approach to zero-shot learning for network intrusion detection. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)
31. Zhang X, Gao L, Jiang Y, et al (2019) A zero-shot intrusion detection method based on regression model. Proc - 2019 7th Int Conf Adv Cloud Big Data, CBD 2019 186–191. https://doi.org/10.1109/CBD.2019.00042
32. Li Z, Qin Z, Shen P, Jiang L (2019) Zero-shot learning for intrusion detection via attribute representation. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)
33. Lv N, Chen C, Qiu T, Sangaiah AK (2018) Deep Learning and Superpixel Feature Extraction Based on Contractive Autoencoder for Change Detection in SAR Images. IEEE Trans Ind Informatics. https://doi.org/10.1109/TII.2018.2873492
34. Van Der Maaten L, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res

35. Baldi P (2012) Autoencoders, Unsupervised Learning, and Deep Architectures. ICML Unsupervised Transf Learn. https://doi.org/10.1561/2200000006
36. Qureshi AS, Khan A, Shamim N, Durad MH (2019) Intrusion detection using deep sparse auto-encoder and self-taught learning. Neural Comput Appl. https://doi.org/10.1007/s00521-019-04152-6
37. Rifai S, Vincent P, Muller X, et al (2011) Contractive auto-encoders: Explicit invariance during feature extraction. In: Proceedings of the 28th International Conference on Machine Learning, ICML 2011
38. Romera-Paredes B, Torr PHS (2015) An embarrassingly simple approach to zero-shot learning. In: 32nd International Conference on Machine Learning, ICML 2015
39. Lampert CH, Nickisch H, Harmeling S (2014) Attribute-based classification for zero-shot visual object categorizationa. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2013.140
40. Akata Z, Perronnin F, Harchaoui Z, Schmid C (2013) Label-embedding for attribute-based classification. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition
41. Lampert CH, Nickisch H, Harmeling S (2010) Learning to detect unseen object classes by between-class attribute transfer
42. Binbusayyis A, Vaiyapuri T (2021) Unsupervised deep learning approach for network intrusion detection combining convolutional autoencoder and one-class SVM. Appl Intell
43. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition
44. Szegedy C, Vanhoucke V, Ioffe S, et al (2016) Rethinking the Inception Architecture for Computer Vision. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition
45. Vasan D, Alazab M, Wassan S, et al (2020) Image-Based malware classification using ensemble of CNN architectures (IMCEC). Comput Secur. https://doi.org/10.1016/j.cose.2020.101748
46. Zeng G, He Y, Yu Z, et al (2016) InceptionNet/GoogLeNet - Going Deeper with Convolutions. Cvpr
47. Khan F, Ncube C, Ramasamy LK, et al (2020) A Digital DNA Sequencing

Engine for Ransomware Detection Using Machine Learning. IEEE Access. https://doi.org/10.1109/ACCESS.2020.3003785

48. Zhang B, Xiao W, Xiao X, et al (2020) Ransomware classification using patch-based CNN and self-attention network on embedded N-grams of opcodes. Futur Gener Comput Syst. https://doi.org/10.1016/j.future.2019.09.025
49. Al-rimy BAS, Maarof MA, Shaid SZM (2019) Crypto-ransomware early detection model using novel incremental bagging with enhanced semi-random subspace selection. Futur Gener Comput Syst. https://doi.org/10.1016/j.future.2019.06.005
50. Vinayakumar V, Alazab M, Jolfaei A, et al (2019) Ransomware triage using deep learning: Twitter as a case study. In: Proceedings - 2019 Cybersecurity and Cyberforensics Conference, CCC 2019
51. Maniath S, Ashok A, Poornachandran P, et al (2018) Deep learning LSTM based ransomware detection. In: 2017 Recent Developments in Control, Automation and Power Engineering, RDCAPE 2017
52. Homayoun S, Dehghantanha A, Ahmadzadeh M, et al (2019) DRTHIS: Deep ransomware threat hunting and intelligence system at the fog layer. Futur Gener Comput Syst. https://doi.org/10.1016/j.future.2018.07.045

Conflicts of interest

Authors declare no conflict of interest.

Availability of data and material

All the datasets used in this work are publicly available, whereas datasets that are generated during simulations are available from the corresponding author on reasonable request.