Android Malware Detection Utilizing Translucent Deep Neural Network

Samaneh Mahdavifar, Ali A. Ghorbani

Canadian Institute for Cybersecurity (CIC), Faculty of Computer Science, University of New Brunswick (UNB)

Problem

The success of neural network, which most of the deep learning algorithms are built upon, is influenced by its major shortcoming, i.e., **it cannot justify the inference it makes**. Adding an explanation feature to a neural network would enhance its trustworthiness and learning capability. This add-on feature, usually represented by **if-then rules**, could be employed in safety-critical systems. The explanation capability of a neural network would improve its generalization in the classification or help the system to supplement additional data to the incomplete dataset based on the values of the condition and action parts.

Motivation

- A major concern in the area of **cybersecurity** is to give a clear explanation of the internal logic of the system and obtain an insight into the problem.
- The main **goal** of this paper is to extract refined rules from a trained Deep Neural Network (DNN) to substitute the deep learning model for classifying unseen Android malware samples.

Matrix Controlled Inference Engine (MACIE)

MACIE is a medical diagnostic expert system developed in the mid-1980s.

$$y_i.w_{i,j}.x_j \geq 0 \; and \; |w_{i,j}| \; is \; maximized$$

$$|current| > unknown$$

Final Rule:

$$If \; x_6 = Unknown \; and \; x_3 = False \; and \; x_5 = True$$

$$Then \; Conclude \; y_1 = True$$

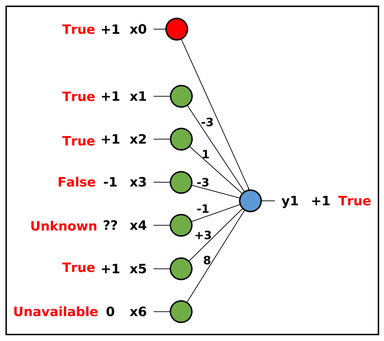


Figure: An example of how MACIE's rule extraction algorithm works

Algorithm 1: MACIE's Rule Extraction Algorithm (MRE)

Input:
x: input sample
y_i: output neuron with value ± 1
b_i: bias for y_i
W_i: matrix contains the weights
Output:
r_i: generated rule for input x and output y_i

$current := b_i$
$vars\_unused := \{j | w_{i,j} \in W_i \neq 0\}$
=set of nodes connected to y_i that are not used in the rule.
$unknown := \sum_{j \in vars\_unused} |w_{i,j}|$
=max value that $current$ can change.
$r_i := \emptyset$
while $True$ **do**
  if $|current| > unknown$ **then**
    break;
    clauses generated so far give a valid justification that is maximally general.
  end if
  Find an input $j \in vars\_unused$ such that $y_i.w_{i,j}.x_j \geq 0$ and $|w_{i,j}|$ is maximized
  Output rule condition $c_{i,j}$ using $x_j \in x$ and its activation value.
  $r_i := append(r_i, c_{i,j})$
  $current := current + w_{i,j}.x_j$
  $unknown := unknown - |w_{i,j}|$
  $vars\_unused := vars\_unused - \{j\}$
end while
return r_i

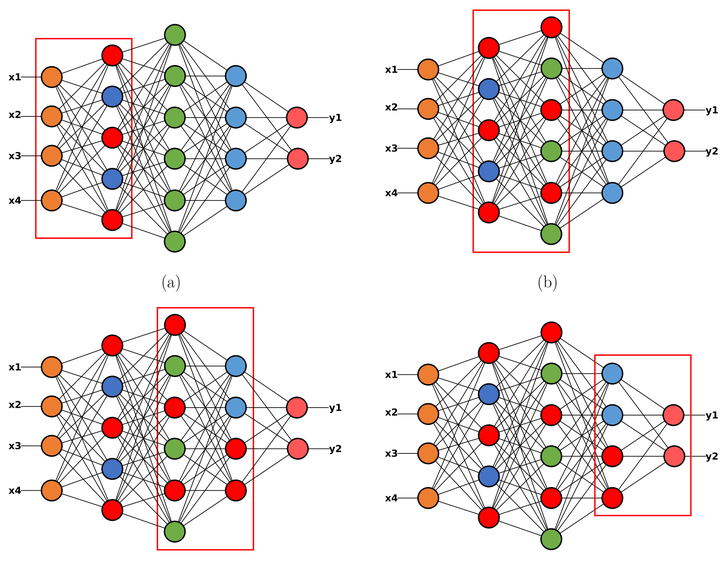


Figure: Applying DMRE on a DNN with three hidden layers.

DeepMACIE

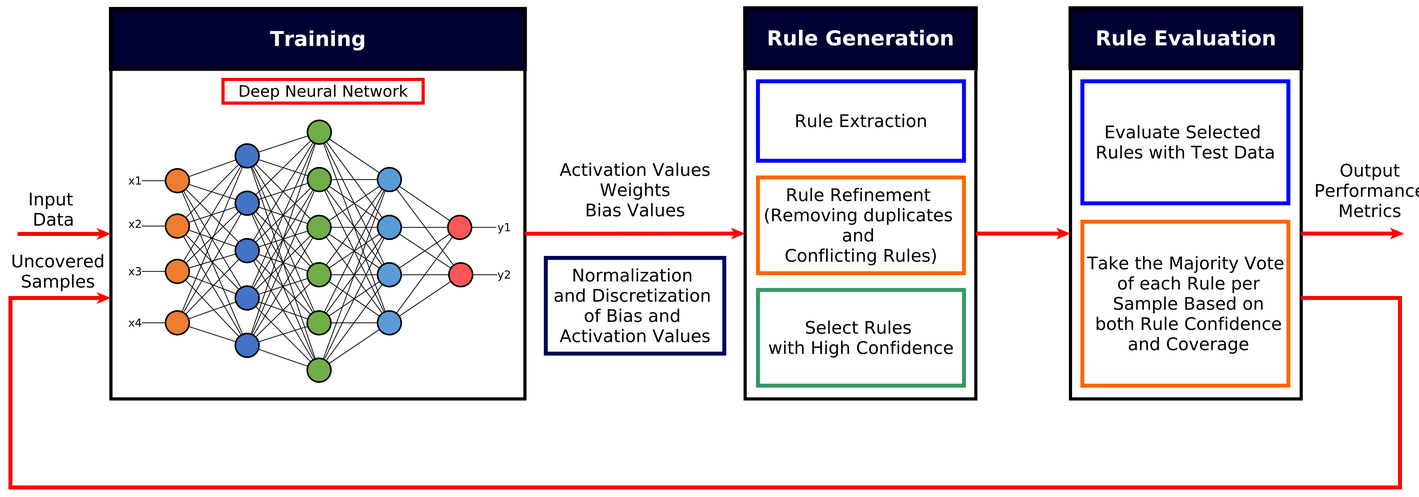


Figure: An overview of the proposed DeepMACIE system framework

Algorithm 2: DeepMACIE's Rule Extraction Algorithm (DMRE)

Input:
h: number of DNN hidden layers
X: input samples of h layers
Output:
$Ruleset$: set of output rules

$Ruleset := \emptyset$
for $k := 1$ to h **do**
  $X_k := \{x \in X \mid layer(x) = k\}$
  for x in X_k **do**
    $R := \emptyset$
    $y := output(x)$
    for y_i in y **do**
      $b_i =$ bias value for node y_i
      $W_i =$ weights matrix for node y_i
      $r_i := MRE(x, y_i, b_i, W_i)$
      if $r_i \neq \emptyset$ **then**
        $R := append(R, r_i)$
      else
        $y := y - \{y_i\}$
      end if
    end for
    $Ruleset := append(Ruleset, R)$
  end for
  Update $Ruleset$ so that the conditions are based on input nodes only
end for
return $Ruleset$

Experimental Results

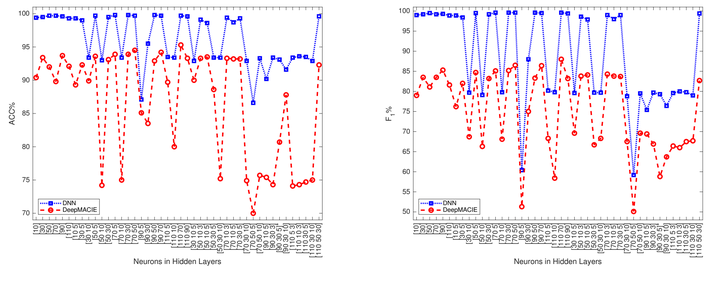
- Android Malware Dataset
  - Collected more than 5,500 samples (4,000 benign, 1,500 malicious)
  - 2,909 samples were run successfully in **Copperdroid**
  - Four categories: banking, adware, and SMS malware, and benign
- 134 distinct **system calls** as feature vector
- Adam Optimization Algorithm (parameters fine-tuned)
- Cross-Entropy as the loss function
- Sigmoid as the activation function
- Stratified 5-folds cross-validation

Table: Classification metrics (%) of DNN for different values of α and β_1

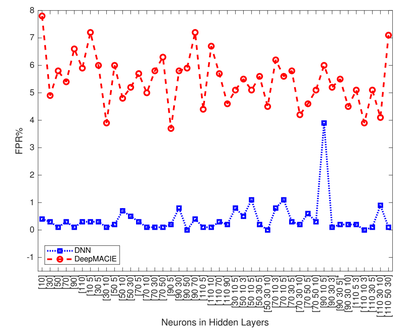
	β_1														
	0.1			0.3			0.5			0.7			0.9		
α	F1	ACC	FP	F1	ACC	FP	F1	ACC	FP	F1	ACC	FP	F1	ACC	FP
1	20.6	53.7	40.1	11.3	61	20	24.8	53.9	33.3	4.6	61.3	6.7	30.8	64.5	25.3
0.5	27.9	64.1	24.4	31.2	46.7	60	36.6	62.3	33.3	56.6	68.7	33.4	40.4	80	0.1
10^{-1}	99.2	99.5	0.3	99.3	99.6	0.3	99.4	99.6	0.1	99.3	99.5	0.6	**99.5**	**99.7**	**0.1**
10^{-2}	97.1	98.2	1.3	97.4	98.3	1	97.3	98.2	1.3	98	98.7	0.8	98.6	99.1	0.7
0.05	98.5	99	0.4	98.4	99	0.5	98	98.7	0.7	98.7	99.1	0.6	98.6	99.1	0.5
10^{-3}	93.3	95.7	2.9	93.2	95.7	2.9	91.6	94.7	3.4	94.3	96.3	2.6	93.1	95.6	3.1
10^{-4}	45.7	71	15.2	61.9	71.1	27.1	41	67.6	20	53.8	64	35.7	51.2	72.1	15.9

Table: Classification metrics (%) of DeepMACIE for different values of α and β_1

	β_1														
	0.1			0.3			0.5			0.7			0.9		
α	F1	ACC	FP	F1	ACC	FP	F1	ACC	FP	F1	ACC	FP	F1	ACC	FP
1	31.1	54.9	15.9	15.1	28.3	6	27.5	43.5	19.4	1.1	68.1	0	31.9	44.5	10.7
0.5	30.9	58.1	10.8	37.2	50.6	31	36.6	60.6	17.5	50	59	19.8	33	65.9	2.4
10^{-1}	79.3	87.5	6.6	79.8	87.8	6.3	78.5	87.1	6.4	77.7	86.6	6.8	**81**	**89.8**	**5.9**
10^{-2}	78.6	86.9	7.5	78.3	86.9	6.8	77.9	87	5.8	79.6	87.5	7.3	78.1	86.9	6.7
0.05	78.1	87	6.1	77.7	86.8	6.1	76.6	86.2	6.5	78.2	86.8	7	79.8	87.4	8
10^{-3}	78.6	87	7.1	76.2	85.7	7.6	76.5	86.2	6.4	77.6	86.6	6.5	78.1	86.8	7
10^{-4}	44.9	75.2	6.1	59.9	75.1	15.9	47.2	62.1	7.4	57.6	74.1	17.4	51.5	76.9	5.7



(a) Accuracy (b) F_1



(c) False Positive Rate

Figure: Classification metrics for different architectures of the deep network.

The best network architecture→[134 110 70 2].

Table: Evaluation metrics on the training bins of the Android malware dataset.

	PR	RC	$\mathbf{F_1}$	ACC	FPR	FNR
DNN	0.99	0.997	0.993	0.99	0.025	0.003
DeepMACIE	0.974	0.957	0.966	0.951	0.065	0.043

Table: Comparison of the average ACC, FPR, and F_1 of DNN with DeepMACIE and J48

Android Malware Dataset	ACC (%)	FPR (%)	$\mathbf{F_1}$ (%)
DNN	100	0.3	100
DeepMACIE (retrain uncovered samples)	95	5.7	88
Decision Tree (J48)	89	11	89

Conclusion

- In this paper, we introduced **DeepMACIE** for extracting refined rules from a trained multilayer DNN, which enhances its utility by adding an explanation capability through a rule extraction process.
- We evaluated our proposed framework on an Android malware dataset comprising more than 5,500 samples from several sources, such as VirusTotal service, Contagio security blog, and previous researchers.
- We did a series of experiments to optimize the parameters of the deep network.
- The outstanding performance of DeepMACIE on the training data bins proves that our approach is promising enough to be applied to real cybersecurity datasets.
- The comparison results of DeepMACIE with DNN and J48 show that DeepMACIE achieves acceptable accuracy of 95%, and false positive rate of 5.7%, about 5% lower than that of the Decision Tree (DT).
- DeepMACIE is capable of extracting rules even when data is unknown or missing.

UNB EST. 1785 UNIVERSITY OF NEW BRUNSWICK | Canadian Institute for Cybersecurity