Deep Learning for Malicious Flow Detection

Yun-Chun Chen[1] Yu-Jhe Li[1] Aragorn Tseng[1] Tsungnan Lin[1,2]

1: National Taiwan University 2: Institute for Information Industry

Agenda

- Introduction
- Gradient Dilution
- Previous Methods
- Methodology
- Experiment
- Extensions

Agenda

- Introduction
- Gradient Dilution
- Previous Methods
- Methodology
- Experiment
- Extensions

Malicious Flow Detection

Traditional Methods (Intrusion-Detection System)

- **Signature Oriented**
  - Rule-based.
  - Pattern matching.
- **Disadvantages**
  - Limited in perceiving new variants of the flow of the malware.
  - If flow is encrypted, pattern matching will lose effectiveness.

Adopted Approach (Machine Learning-based)

- **Behavior-based**
  - Feature extraction.
  - Learn pattern.
- **Advantages**
  - Able to perceive new variants of the flow of the malware.
  - Still effective even if the flow is encrypted.

Previous Work

Cisco [Anderson *et al.*]

- **Model**
  - Linear model.
  - Logistic model.
  - Support vector machine.
- **Experimental Results**
  - Accuracy = 90.3%.

Deep Learning

Description

- **Powerful Method**
  - Succeeded in many aspects of applications.
  - Handles problems with complexity.
- **Generalization Performance**
  - Outperform other machine learning methods in our experiment.
  - Partial flow experiment.
  - Zero-shot experiment.

Proposed Method

- **Adopt Deep Learning Approaches**
- **Design a New Neural Network Architecture**
- **Adjust Training Mechanism**
  - Imbalanced data distribution.
  - Result in the difficulty of training a neural network.

Agenda

- Introduction
- Gradient Dilution
- Previous Methods
- Methodology
- Experiment
- Extensions

Gradient Dilution

Gradient-based Training Mechanism

- Deep learning is a **gradient-based** training mechanism.
- Each epoch computes gradient with respect to each parameter θ.

$$\frac{\partial Loss}{\partial \theta} = \sum_{i=1}^{N} \frac{\partial Loss_i}{\partial \theta} \tag{1}$$

- *i* represents the index of the training data.
- *N* refers to the cardinality of the training data set.

Imbalanced Data Set

- **Disparity Between Classes**
  - Gradients contributed by majority class dominates the total gradient in terms of quantity.
  - The model will be insensitive to the minority class.
  - Gradients contributed by the minority class are diluted.

Agenda

- Introduction
- Gradient Dilution
- Previous Methods
- Methodology
- Experiment
- Extensions

Previous Methods

Data Manipulating Techniques

- **Oversampling** [Pears *et al.*]
  - Re-sample data from the minority class.
  - Duplicated data is contained in the training data set.
- **Undersampling** [Liu *et al.*]
  - Eliminate data from the majority class at random.
- Both methods aim at providing a balanced data distribution.

Training Mechanism Adjustment

- **Incremental Learning** [Kulkarni *et al.*]
  - Feasible method to tackle imbalanced data issue.
  - Learn from the newly introduced data without forgetting past memories.
  - Train on a small data set and increase the size of training data set gradually.

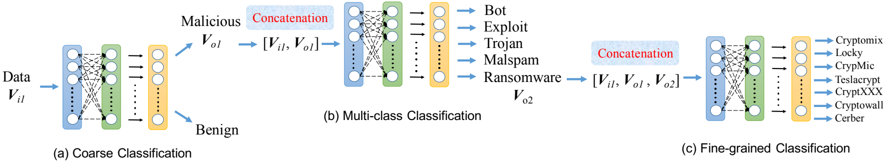
Agenda

- Introduction
- Gradient Dilution
- Previous Methods
- Methodology
- Experiment
- Extensions

Proposed Model

Tree-Shaped Deep Neural Network

- Mitigate the imbalanced data issue implicitly.
- Classify data in a layer-wise manner.
- Can be trained **end-to-end**.
- Loss will propagate from the output-end to the input-end.
- Learn global information and partial knowledge simultaneously.
- **Generate features** to enhance classification performance.



(a) Coarse Classification

(b) Multi-class Classification

(c) Fine-grained Classification

Algorithm

Backpropagation

$$\theta_i^{l+} = \theta_i^l - \eta \times \frac{\partial Loss}{\partial \theta_i^l} \tag{2}$$

- θ_i^l represents the i^{th} parameter at l^{th} layer.
- η stands for the learning rate.
- $\frac{\partial Loss}{\partial \theta_i^l}$ refers to the partial derivative of *Loss* with respect to θ_i^l.
- θ_i^{l+} refers to the updated parameter.

Drawback

- Treat each individual data fairly.
- Not able to resolve imbalanced data issue.

Proposed Algorithm

Quantity Dependent Backpropagation

$$\theta_i^{l^+} = \theta_i^l - \eta \times \boldsymbol{F} \times \nabla Loss \tag{3}$$

$$\boldsymbol{F} = \left[\frac{c_1}{n_1}, \frac{c_2}{n_2}, ..., \frac{c_N}{n_N}\right] \tag{4}$$

$$\nabla Loss = \left[\frac{\partial Loss_1}{\partial \theta_i}, \frac{\partial Loss_2}{\partial \theta_i}, ..., \frac{\partial Loss_N}{\partial \theta_i}\right]^T \tag{5}$$

- c_i is the pre-selected coefficient for the class that the i^{th} data belongs to.
- n_i represents the cardinality of the class that the i^{th} data belongs to.
- $\frac{\partial Loss_k}{\partial \theta_i}$ refers to the partial derivative of the *Loss* contributed by the k^{th} data with respect to θ_i.

Agenda

- Introduction
- Gradient Dilution
- Previous Methods
- Methodology
- Experiment
- Extensions

Experimental Settings

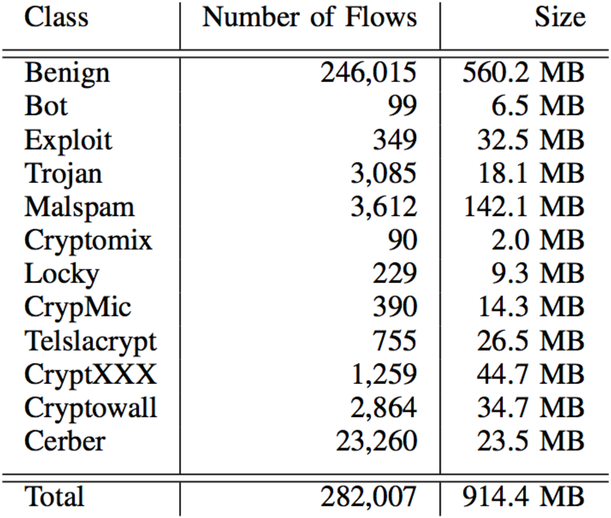
Real Collected Data

- **No** open data set available.
- Obtain the executable files of the malware from VirusTotal and record the network behavior under sandbox.
- From September 2016 to May 2017.

Data Description

- The malicious data is categorized into 5 different classes where each class represents different attack behaviors.
- Further label Ransomware into 7 different families.

Data Statistics



Class	Number of Flows	Size
Benign	246,015	560.2 MB
Bot	99	6.5 MB
Exploit	349	32.5 MB
Trojan	3,085	18.1 MB
Malspam	3,612	142.1 MB
Cryptomix	90	2.0 MB
Locky	229	9.3 MB
CrypMic	390	14.3 MB
Telslacrypt	755	26.5 MB
CryptXXX	1,259	44.7 MB
Cryptowall	2,864	34.7 MB
Cerber	23,260	23.5 MB
Total	282,007	914.4 MB

Feature Extraction

Connection Records

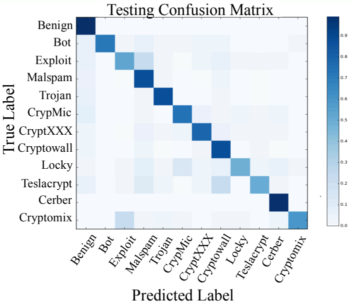
- **Internet Protocols and Connection Statistics** [Williams *et al.*]
    - Port, IP, and protocol information.
- **HTTP Requests** [Tseng *et al.*]
- **TLS Handshake** [Anderson *et al.*]
    - Byte distribution.
    - Spatial information.

Flow Behavior

- **Inter-Arrival Time** [Moore *et al.*]
    - Proposed nearly 250 discriminators to classify flow records.
- **Markov Matrix** [McGrew *et al.*]
    - Stores the relationship between sequential packets.
    - Temporal information.

Experimental Results

Method	Accuracy	Precision
DNN + Backpropagation	59.08%	8.33%
DNN + Oversampling (10000 samples/class) [7]	85.18%	65.9%
DNN + Undersampling (45 samples/class) [8]	68.89%	49.45%
DNN + Incremental Learning [9]	78.84%	71.23%
DNN + QDBP	84.56%	62.3%
SVM (RBF)	83.87%	38.8%
Random Forest	98.9%	68.25%
TSDNN + QDBP	**99.63%**	**85.4%**



$$Precision = \frac{1}{n}\sum_{i=1}^{n}\frac{TP_i}{TP_i + FP_i} \qquad (6)$$

TP : True Positive

FP : False Positive

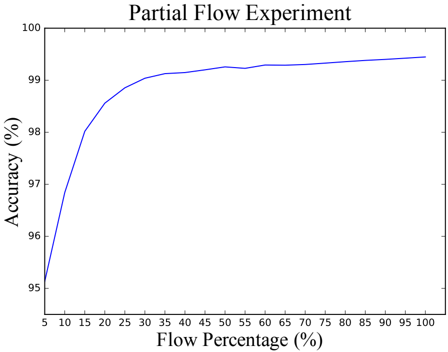
Agenda

- Introduction
- Gradient Dilution
- Previous Methods
- Methodology
- Experiment
- Extensions

Partial Flow Experiment

Description

- **Real-time Detection**
- Divide each flow into fractions.
- Consider only a portion of data.
- Test the potentiality of being a malicious flow.



Zero-Shot Experiment

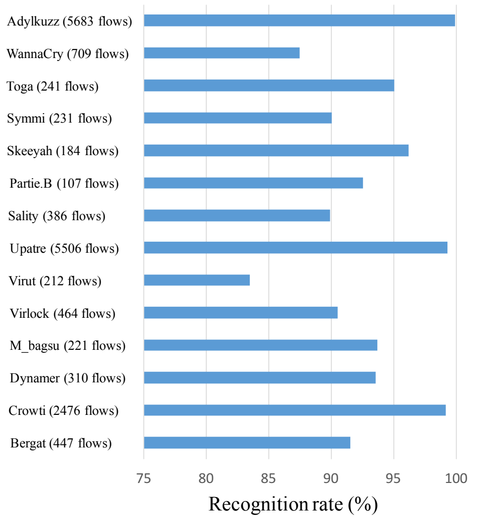
Description

- Generalization performance of deep learning.
- Examine the ability of TSDNN to identify some malware that has never been trained before.
- Justify that a behavior-based approach is a better way compared to the traditional signature-oriented methods.

Data

- We obtain 14 different kinds of malicious flows.

Experimental Results


Adylkuzz (5683 flows)
WannaCry (709 flows)
Toga (241 flows)
Symmi (231 flows)
Skeeyah (184 flows)
Partie.B (107 flows)
Sality (386 flows)
Upatre (5506 flows)
Virut (212 flows)
Virlock (464 flows)
M_bagsu (221 flows)
Dynamer (310 flows)
Crowti (2476 flows)
Bergat (447 flows)
Recognition rate (%)