

## CS6264 Project 7: Exploring Modeling & Attacking PE Models via MLSploit

### Task 1: Training DL Models

#### Approach

To classify malware families, I trained three DL models LSTM, CNN, and RNN on API call sequences using the MLSploit platform. The training dataset, ‘pe.train.txt’, consisted of labeled PE files categorized as benign or malicious. Evaluation was conducted on ‘pe.eval.txt’ to validate model performance.

#### Findings

1. Model Selection: LSTM achieved the highest accuracy due to its ability to capture long-term dependencies in sequences. RNN was effective for identifying localized patterns, while CNN performed moderately but struggled with vanishing gradient issues.
2. Hyperparameters:
  - Sequence Window Size: Set to 51 to capture sufficient context for accurate predictions.
3. Accuracy: The updated log files revealed accuracies of 95% (LSTM), 90% (RNN), and 88% (CNN).

#### Observations

Longer sequences improved classification accuracy but increased training time. LSTM’s ability to capture sequential dependencies made it superior for API call sequence data.

### Task 2: Attacking DL Models

#### Approach

For this task, I enabled the sequence parameter, set the window size to 51, and activated sequence\_lstm, sequence\_rnn, and sequence\_cnn modes in the Mimicry Attack configuration. A Mimicry Attack was implemented using the ‘pe.benign.txt’ and ‘pe.target.txt’ datasets. The PE Transformer module rewrote malicious binaries to mimic benign API call sequences.

#### Findings

1. Attack Success:
  - LSTM: Classification accuracy dropped to 48.67%
  - CNN: Classification accuracy dropped to 0%
  - RNN: Classification accuracy dropped to 2.33%
2. Performance Impact: Models struggled with samples exhibiting long sequences of benign-like calls, confirming the attack’s effectiveness.

## Observations

Enabling sequence-based processing exposed vulnerabilities in DL models to mimicry attacks. While models like LSTM resisted attacks to some degree, CNN and RNN models showed significant degradation, highlighting the importance of sequence robustness in model design.

## Task 4: Training Classical ML Models

### Approach

I trained ML models on additional API-related features (existence, frequency, and arguments) using MLSploit.

- Existence: Enabled existence and existence\_knn with existence\_knn\_k set to 7.
- Frequency: Enabled frequency and frequency\_rf with frequency\_rf\_trees set to 10.
- Arguments: Enabled arguments and arguments\_nb.
- During the ensemble evaluation, all three features (existence, frequency, arguments) were enabled.

### Findings

#### 1. Feature Comparisons:

- Existence features yielded validation accuracy of 95.2%.
- Frequency features achieved validation accuracy of 100%.
- Arguments features resulted in validation accuracy of 86.3%.

#### 2. Ensemble Performance:

- Ensemble accuracy for first evaluation was 96.4%.
- Ensemble accuracy for second evaluation was 97.6%.
- Ensemble accuracy for third evaluation was 90.8%.

## Observations

Combining multiple API-related features in the ensemble improved overall classification performance. Frequency features, supported by Random Forest, demonstrated exceptional predictive ability, while existence and arguments added complementary strengths.

## Task 5: Attack Transfer to ML Models

### Approach

For this task, I used the MalwareLab module to submit and execute 10 attack malware samples. The PE Ensemble-Evaluate module was configured with the following parameters:

- Sequence: Enabled.
- Window Size: Set to 51.

### Findings

1. MalwareLab Execution:
  - 10 samples successfully submitted and executed.
  - Samples included various attack binaries with unique SHA256 hashes:
    - Example: attack-0.exe with hash  
6d8793cd0c14e28835463b6b8ce3ee9707206ed2e7c47379afe8860fcdab45dd.
2. Model Evaluation:
  - LSTM Accuracy: 58.86%
  - RNN Accuracy: 2.57%
  - CNN Accuracy: 0%

### Observations

The LSTM model demonstrated moderate robustness against the attack samples, while CNN and RNN models showed severe degradation in performance, indicating susceptibility to the crafted attacks. These results highlight the importance of designing resilient architectures and using ensemble techniques for better defense.

### Bonus Task:

#### Approach

##### Ember Model Training

##### Training Configuration:

- Enabled sequence and set the window size to 51.
- Enabled ember during the ensemble training process.

##### Results:

- Training Accuracy: 100%
- Validation Accuracy: 99.32%

#### Ember Model Attack

##### Attack Configuration:

- Used the Ember Attack module (pe.module.ember-attack) with the hashes generated in Task 5.
- Model (pe.model.zip) trained in Task 4 was utilized for this attack.

Findings:

#### Attack Execution:

- 242 samples successfully processed using the EMBER Attack module.
- Each sample had unique hash modifications post-attack.

#### Observations

The Ember model exhibited high robustness during initial testing but was significantly impacted by advanced mimicry attacks. Static features provided strong detection capabilities; however, adversarial perturbations targeting these features reduced the model's effectiveness. This highlights the necessity for combining static and dynamic features to build hybrid defense mechanisms.

#### Appendix

##### Screenshots

###### - Training DL Models

The screenshot shows the MLsploit web application interface. On the left, there is a sidebar titled "Research Modules" containing three items: "Foolbox" (Fool neural networks!), "MalwareLab Module" (This is the MalwareLab module), and "PE Module" (This is the PE module). The main area is titled "DL Training" and contains two buttons: "Ensemble-Train" (highlighted in red) and "Ensemble-Evaluate". To the right, a detailed configuration panel is shown for the "Task: Ensemble-Train". It includes several input fields and dropdown menus. The parameters listed are:

- sequence: true
- Sequence - Enable
- sequence\_window: 51
- Sequence - Window Size
- sequence\_type: multi\_classification
- Sequence - Classification Type
- sequence\_lstm: true
- Sequence - LSTM - Enable
- sequence\_rnn: true
- Sequence - RNN - Enable
- sequence\_cnn: true
- Sequence - CNN - Enable
- existence: false
- Existence - Enable
- existence\_rf: false
- Existence - Random Forest - Enable
- existence\_rf\_trees: 10
- Existence - Random Forest - Number of Trees
- existence\_nb

**MLsploit** Experiment with AI Security in your browser!

New Pipeline Files Settings

### Research Modules

- Foolbox** Fool neural networks!
- MalwareLab Module** This is the MalwareLab module
- PE Module** This is the PE module

### DL Training

Ensemble-Train → Ensemble-Evaluate

### Task: Ensemble-Evaluate

Parameters:

- sequence: true
- sequence\_window: 51
- sequence\_type: multi\_classification
- existence: false
- frequency: false
- arguments: false
- ember: false

**MLsploit** Experiment with AI Security in your browser!

New Pipeline Files Settings

### Research Modules

- Foolbox** Fool neural networks!
- MalwareLab Module** This is the MalwareLab module
- PE Module** This is the PE module

### DL Training

Ensemble-Train → Ensemble-Evaluate

### Pipeline: DL Training

2 days ago FINISHED

**Output Files**

- pe-Ensemble-Evaluate.log.txt
- pe-Ensemble-Evaluate.log\_err.txt
- prediction.zip
- pe.model.zip
- pe-Ensemble-Train.log.txt
- pe-Ensemble-Train.log\_err.txt

## - Attacking DL Models

The screenshot shows the MLsploit interface with the following components:

- Research Modules** sidebar:
  - Foolbox: Fool neural networks!
  - MalwareLab Module: This is the MalwareLab module.
  - PE Module: This is the PE module.
  - Buttons: Ensemble-Train, Ensemble-Evaluate, Mimicry-Attack, PE-Transformer, Detect-Trampoline, Ember-Attack.
- Mimicry Attack** panel:

```
Mimicry-Attack → PE-Transformer
```
- DL Training** panel:

```
Ensemble-Train → Ensemble-Evaluate
```
- Task: Mimicry-Attack** configuration panel:

Parameters:

  - sequence: true
  - sequence\_window: 61
  - sequence\_type: multi\_classification
  - sequence\_lstm: true
  - sequence\_rnn: true
  - sequence\_cnn: true
  - generations: 10

The screenshot shows the MLsploit interface with the following components:

- Research Modules** sidebar:
  - Foolbox: Fool neural networks!
  - MalwareLab Module: This is the MalwareLab module.
  - PE Module: This is the PE module.
  - Buttons: Ensemble-Train, Ensemble-Evaluate, Mimicry-Attack, PE-Transformer, Detect-Trampoline, Ember-Attack.
- Mimicry Attack** panel:

```
Mimicry-Attack → PE-Transformer
```
- DL Training** panel:

```
Ensemble-Train → Ensemble-Evaluate
```
- Task: PE-Transformer** configuration panel:

Parameters:

  - hash: rbot-original.exe

The screenshot shows the MLsploit web application interface. On the left, there's a sidebar titled "Research Modules" containing icons and descriptions for "Foolbox", "MalwareLab Module", and "PE Module". Below these are buttons for "Ensemble-Train", "Ensemble-Evaluate", "Mimicry-Attack", "PE-Transformer", "Detect-Trampoline", and "Ember-Attack". The main area has a title "Mimicry Attack" with a flowchart icon showing "Mimicry-Attack" connected to "PE-Transformer". To the right, a "Pipeline: Mimicry Attack" section shows it was "10 minutes ago" and is "FINISHED". A "DL Training" section is also visible, with a "Output Files" dialog box showing a list of generated files:

- pe-PE-Transformer.log.txt
- pe-PE-Transformer.log\_err.txt
- attack.exe.zip
- pe-Mimicry-Attack.log.txt
- pe-Mimicry-Attack.log\_err.txt
- attack-feature.zip
- attack-prediction.zip
- attack.cfg.zip

## - Training Classic Models

This screenshot shows the "Task: Ensemble-Train" configuration for the "Classical ML" section. The sidebar on the left is identical to the previous screenshot. The main area has a "Classical ML" section with a flowchart icon showing "Ensemble-Train" connected to "Ensemble-Evaluate". Below it is a "MalwareLab" section with a "Submit" button. To the right is a detailed configuration panel for "Task: Ensemble-Train" with various parameters:

- existence**: true
- existence\_rf**: false
- existence\_rf\_trees**: 10
- existence\_nb**: false
- existence\_knn**: true
- existence\_knn\_k**: 7
- existence\_sgd**: false
- existence\_mlp**: false
- frequency**: true
- frequency\_rf**: true

Below these are sections for "Existence - Random Forest", "Existence - Naive Bayes", "Existence - k-Nearest Neighbors", "Existence - Stochastic Gradient Descent", and "Existence - Multi-Layer Perceptron".

**MLsploit** Experiment with AI Security in your browser!

New Pipeline Files Settings

### Research Modules

- Foolbox** Fool neural networks!
- MalwareLab Module** This is the MalwareLab module
- PE Module** This is the PE module

**Classical ML**

Ensemble-Train → Ensemble-Evaluate

**MalwareLab**

Submit

**Task: Ensemble-Evaluate**

Parameters:

- sequence: false
- Sequence - Enable
- sequence\_window: 32
- Sequence - Window Size
- sequence\_type: multi\_classification
- Sequence - Classification Type
- existence: true
- Existence - Enable
- frequency: true
- Frequency - Enable
- arguments: true
- Arguments - Enable
- ember: false
- Ember - Enable

**MLsploit** Experiment with AI Security in your browser!

New Pipeline Files Settings

### Research Modules

- Foolbox** Fool neural networks!
- MalwareLab Module** This is the MalwareLab module
- PE Module** This is the PE module

**Classical ML**

Ensemble-Train → Ensemble-Evaluate

**MalwareLab**

Submit

**Pipeline: Classical ML**

3 days ago FINISHED

**Output Files**

- pe-Ensemble-Evaluate.log.txt
- pe-Ensemble-Evaluate.log\_err.txt
- prediction.zip
- pe.model.zip
- pe-Ensemble-Train.log.txt
- pe-Ensemble-Train.log\_err.txt

## - Attack Transfer to ML Models

The screenshot shows the MLsploit interface. On the left, there's a sidebar titled "Research Modules" containing icons and descriptions for "Foolbox" ( Fool neural networks!), "MalwareLab Module" (This is the MalwareLab module), and "PE Module" (This is the PE module). Below these are buttons for "Ensemble-Train", "Ensemble-Evaluate", "Mimicry-Attack", "PE-Transformer", "Detect-Trampoline", and "Ember-Attack". In the center, there are four main sections: "Ensemble-Evaluate", "Ember Model Attack", "Ember Model Training", and "Mimicry Attack". Each section has a corresponding button or link. To the right, a "Pipeline: MalwareLab" summary is shown with the status "FINISHED" and a timestamp "3 days ago".

This screenshot is similar to the one above, but it shows a modal dialog titled "Output Files" overlaid on the "Ember Model Training" section. The dialog lists two files: "malwarelab-Submit.log.txt" and "malwarelab-Submit.log\_err.txt". The background sections and pipeline summary are visible but dimmed.

**MLsploit** Experiment with AI Security in your browser!

New Pipeline Files ⚙

### Research Modules

- Foolbox**  
Fool neural networks!
- MalwareLab Module**  
This is the MalwareLab module
- PE Module**  
This is the PE module

Ensemble-Train Ensemble-Evaluate

Mimicry-Attack PE-Transformer

Detect-Trampoline Ember-Attack

### Ensemble-Evaluate

Ensemble-Evaluate

### Ember Model Attack

Ember-Attack

### Ember Model Training

Ensemble-Train → Ensemble-Evaluate

### MalwareLab

Submit

### Mimicry Attack

Task: Ensemble-Evaluate

Parameters:

- sequence: true
- Sequence - Enable
- sequence\_window: 51
- Sequence - Window Size
- sequence\_type: multi\_classification
- Sequence - Classification Type
- existence: false
- Existence - Enable
- frequency: false
- Frequency - Enable
- arguments: false
- Arguments - Enable
- ember: false
- Ember - Enable

**MLsploit** Experiment with AI Security in your browser!

New Pipeline Files ⚙

### Research Modules

- Foolbox**  
Fool neural networks!
- MalwareLab Module**  
This is the MalwareLab module
- PE Module**  
This is the PE module

Ensemble-Train Ensemble-Evaluate

Mimicry-Attack PE-Transformer

Detect-Trampoline Ember-Attack

### Ensemble-Evaluate

Ensemble-Evaluate

### Ember Model Attack

Ember-Attack

### Ember Model Training

Ensemble-Train →

### MalwareLab

Submit

### Mimicry Attack

Pipeline: Ensemble-Evaluate

2 days ago FINISHED

Output Files

- pe-Ensemble-Evaluate.log.txt
- pe-Ensemble-Evaluate.log\_err.txt
- prediction.zip

## - Bonus Task

**MLsploit** Experiment with AI Security in your browser!

**Task: Ensemble-Train**

frequency - Multi-Layer Perceptron - Enable  
arguments  
false  
Arguments - Enable  
arguments\_rf  
false  
Arguments - Random Forest - Enable  
arguments\_rf\_trees  
10  
Arguments - Random Forest - Number of Trees  
arguments\_nb  
false  
Arguments - Naive Bayes - Enable  
arguments\_knn  
false  
Arguments - k-Nearest Neighbors - Enable  
arguments\_knn\_k  
5  
Arguments - k-Nearest Neighbors - k-value  
arguments\_sgd  
false  
Arguments - Stochastic Gradient Descent - Enable  
arguments\_mlp  
false  
Arguments - Multi-Layer Perceptron - Enable  
ember  
true  
Ember - Enable

**Research Modules**

- Foolbox  
Fool neural networks!
- MalwareLab Module  
This is the MalwareLab module
- PE Module  
This is the PE module

**Ensemble-Evaluate**

Ensemble-Evaluate

**Ember Model Attack**

Ember-Attack

**Ember Model Training**

Ensemble-Train → Ensemble-Evaluate

**MalwareLab**

Submit

**Mimicry Attack**

Mimicry-Attack → PE-Transformer

**DI Training**

**MLsploit** Experiment with AI Security in your browser!

**Task: Ensemble-Evaluate**

Parameters:  
sequence  
true  
Sequence - Enable  
sequence\_window  
51  
Sequence - Window Size  
sequence\_type  
multi\_classification  
Sequence - Classification Type  
existence  
false  
Existence - Enable  
frequency  
false  
Frequency - Enable  
arguments  
false  
Arguments - Enable  
ember  
true  
Ember - Enable

**Research Modules**

- Foolbox  
Fool neural networks!
- MalwareLab Module  
This is the MalwareLab module
- PE Module  
This is the PE module

**Ensemble-Evaluate**

Ensemble-Evaluate

**Ember Model Attack**

Ember-Attack

**Ember Model Training**

Ensemble-Train → Ensemble-Evaluate

**MalwareLab**

Submit

**Mimicry Attack**

