# Understanding Machine Learning Model Updates in Malware Detection Systems Based on Feature Attribution Changes

大阪大学　大学院情報科学研究科
情報ネットワーク学専攻 村田研究室
雲　帆（YUN FAN）

# Background

- In a **malware detection system**, the statistical characteristics of malware change over time, causing the detection performance degrades
- The classification models in malware detection systems need **updates** to improve the detection performance
    - update: add new data to the training dataset and re-train the model
- After updates, the new model needs to be **validated**
    - accuracy
    - the area under the curve (AUC)
    - …

# Purpose

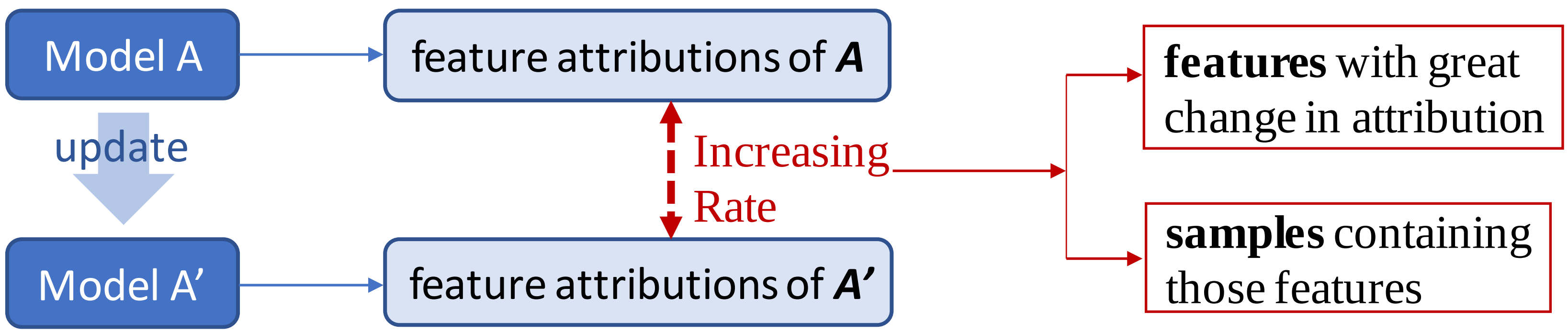
- Common validation methods only calculate the detection accuracy or AUC scores
- When the detection performance is not satisfying after model update, we need more information to determine the cause
  - *why performance changed ?*
  - *what changes in the update affect performance?*

**Purpose:**

Get detailed information about model changes to understand the model updates in malware detection systems.

# Proposed Method

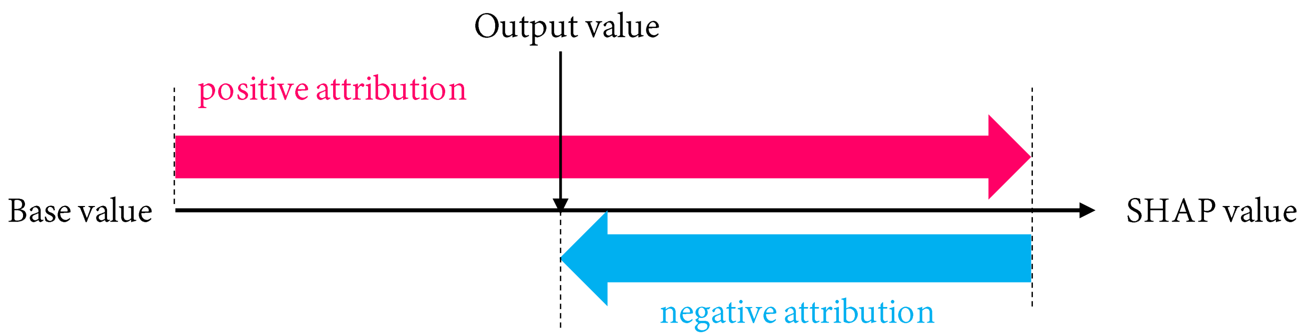
- Machine learning (ML) models are often used in malware detection systems, and **feature attributions** are typically used to explain the ML models
- We use the **feature attribution changes** to analyze model changes
- <u>Proposed method</u>




- By identifying the features and samples with great changes, we can analyze what changes affect the detection performance during updates

# Feature Attribution

- We use **Shapley additive explanations (SHAP)** to calculate the **feature attributions**
- SHAP is a **consistent** feature attribution method
    - *When the model has changed and a feature has higher impact on the model, the importance of that feature <u>cannot</u> be lower*
- SHAP explains the output as a sum of the effects of each feature




- **Consistency** enables comparison of attribution values <u>across models</u>

# SHAP Value Changes

- We calculate an **increasing rate of SHAP values (*I*)** to measure a feature's attribution change in an update

$$I_{x_i} = \frac{v2_{x_i} - v1_{x_i} + c_1}{\min(|v1_{x_i}|, |v2_{x_i}|) + c_2}, \quad where\ c_2 > 0, c_1 = \begin{cases} c_2, & when\ v2_{x_i} - v1_{x_i} \geq 0, \\ -c_2, & when v2_{x_i} - v1_{x_i} < 0. \end{cases}$$

$\boxed{I > 0}$ → Feature attribution is higher
Samples are more likely to be classified as **positive**

$\boxed{I < 0}$ → Feature attribution is lower
Samples are more likely to be classified as **negative**

- When $|I| \approx 0$, the feature's effect to the model update is very low
- Identify **features** with high increasing rate by $|I| \geq k$ and analyze **samples** containing those features

# Experimental Setup

- Dataset
  - Android application files: *AndroZoo**
  - 9 dataset with different size (containing 10% malicious samples)
- Threshold: $k$=3

| | Malicious | Benign |
|---|---|---|
| Model 1 | 101 | 816 |
| Model 2 | 151 | 1,224 |
| Model 3 | 201 | 1,631 |
| Model 4 | 251 | 2,039 |
| Model 5 | 301 | 2,447 |
| Model 6 | 351 | 2,854 |
| Model 7 | 401 | 3,262 |
| Model 8 | 451 | 3,670 |
| Model 9 | 501 | 4,077 |

| | AUC |
|---|---|
| Model 1 | 0.9389 |
| Model 2 | 0.9588 |
| Model 3 | 0.9607 |
| Model 4 | 0.9664 |
| Model 5 | 0.9695 |
| Model 6 | 0.9709 |
| Model 7 | 0.9740 |
| Model 8 | 0.9735 |
| Model 9 | 0.9745 |

The improvement became small after Model 4&5

**AndroZoo: Allix, K, etc.: Androzoo: Collecting millions of android apps for the research community.(2016)*

# Experimental Results

- The number of samples that contain features with high increasing rate in each update

The number of samples with increasing rate $|I| \geq 3$

The percentage of samples with increasing rate $|I| \geq 3$ in the training dataset

**shows the extent of model change**

more likely to be detected as malicious (caused by adding malicious data)

more likely to be detected as benign (caused by adding benign data)

| | | Malicious | Benign | % |
|---|---|---|---|---|
| Models 1 & 2 | $I \geq 0$ | 22 | 38 | 6.5 |
| | $I < 0$ | 56 | 36 | 10.0 |
| Models 2 & 3 | $I \geq 0$ | 44 | 10 | 3.9 |
| | $I < 0$ | 12 | 19 | 2.3 |
| Models 3 & 4 | $I \geq 0$ | 29 | 46 | 4.1 |
| | $I < 0$ | 0 | 8 | 0.4 |
| Models 4 & 5 | $I \geq 0$ | 9 | 16 | 1.1 |
| | $I < 0$ | 0 | 4 | 0.2 |
| Models 5 & 6 | $I \geq 0$ | 3 | 26 | 1.1 |
| | $I < 0$ | 0 | 2 | 0.1 |
| Models 6 & 7 | $I \geq 0$ | 6 | 29 | 1.1 |
| | $I < 0$ | 25 | 8 | 1.0 |
| Models 7 & 8 | $I \geq 0$ | 0 | 19 | 0.5 |
| | $I < 0$ | 1 | 9 | 0.3 |
| Models 8 & 9 | $I \geq 0$ | 3 | 10 | 0.3 |
| | $I < 0$ | 0 | 1 | 0.0 |

# Evaluation

- The proposed method can explain how new data affected performance change
- The proposed method can analyze the effects of adding malicious and benign samples respectively
- For example:
  - The improvement was mainly caused by adding malicious data
  - The percentage of selected samples is larger in models 6&7 ⟶ **Case study**

| | | Malicious | % | Benign | % |
|---|---|---|---|---|---|
| Models 1 & 2 | $I \geq 0$ | 22 | 21.8 | 38 | 4.7 |
| | $I < 0$ | 56 | 55.4 | 36 | 4.4 |
| Models 2 & 3 | $I \geq 0$ | 44 | 29.1 | 10 | 0.8 |
| | $I < 0$ | 12 | 7.9 | 19 | 1.6 |
| Models 3 & 4 | $I \geq 0$ | 29 | 14.4 | 46 | 2.8 |
| | $I < 0$ | 0 | 0.0 | 8 | 0.5 |
| Models 4 & 5 | $I \geq 0$ | 9 | 3.6 | 16 | 0.8 |
| | $I < 0$ | 0 | 0.0 | 4 | 0.2 |
| Models 5 & 6 | $I \geq 0$ | 3 | 1.0 | 26 | 1.1 |
| | $I < 0$ | 0 | 0.0 | 2 | 0.1 |
| Models 6 & 7 | $I \geq 0$ | 6 | 1.7 | 29 | 1.0 |
| | $I < 0$ | 25 | 7.1 | 8 | 0.3 |
| Models 7 & 8 | $I \geq 0$ | 0 | 0.0 | 19 | 0.6 |
| | $I < 0$ | 1 | 0.2 | 9 | 0.3 |
| Models 8 & 9 | $I \geq 0$ | 3 | 0.7 | 10 | 0.3 |
| | $I < 0$ | 0 | 0.0 | 1 | 0.0 |

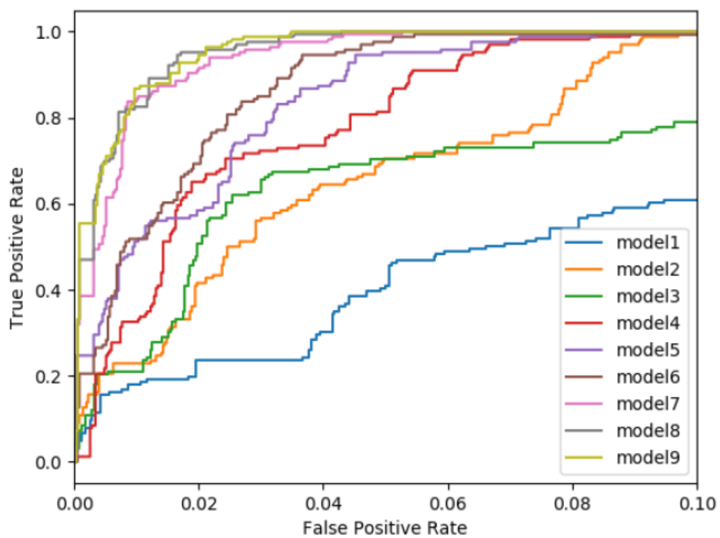
# Case Study

➢ The result in models 6&7 is caused by changes of **2 malware families**

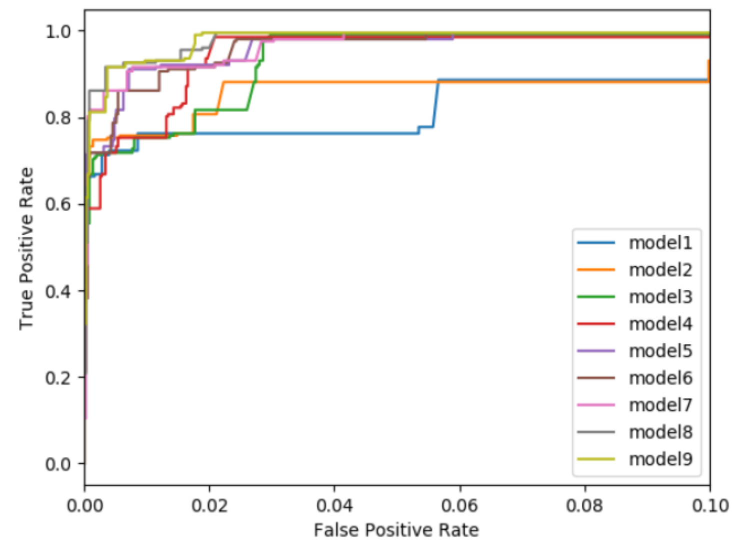
$I \geq 0$ : 6 samples ⟶ **"jiagu" family**

$I < 0$ : 25 samples ⟶ **"fakeapp" family**

| | | | |
|---|---|---|---|
| 5 & 6 | $I \geq 0$ | 3 | 26 |
| | $I < 0$ | 0 | 2 |
| 6 & 7 | $I \geq 0$ | 6 | 29 |
| | $I < 0$ | 25 | 8 |
| 7 & 8 | $I \geq 0$ | 0 | 19 |
| | $I < 0$ | 1 | 9 |




ROC of "jiagu"




ROC of "fakeapp"

- Performance on "jiagu" has improved even after model 4
- Changes in "fakeapp" has no negative effect on classification performance

# Conclusion and Future Works

- Conclusion
  - Our method can distinguish slight changes for a particular malware family.
  - Our method can identify the key features that related to the changes in model updates.
  - Our method can analyze the effects of adding malicious and benign samples respectively and the tendency of new predictions.
- Future works
  - Experiments for other systems to confirm the proposed method is available for all ML operations
  - Better solution for best choosing the thresholds
  - More analysis about the identified key features