Improved MalGAN: Avoiding Malware Detector by Learning Cleanware Features

Masataka Kawai, Kaoru Ota, and Mianxiong Dong

Department of Information and Electronic Engineering,
Muroran Institute of Technology, Japan

室蘭工業大学
MURORAN INSTITUTE OF TECHNOLOGY

Contents

- Introduction
- Avoidance method by MalGAN
- Improved MalGAN
  - Experimental methodology
  - Result and Discussion
- Countermeasures for MalGAN
- Conclusions
- Future work

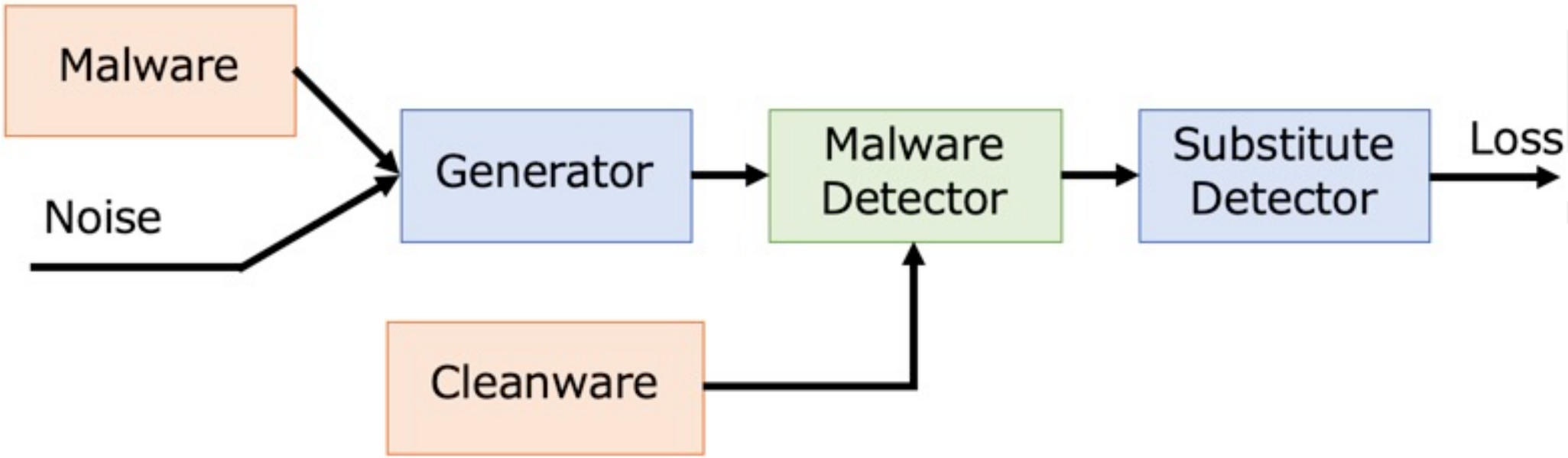
Introduction

- Researches on malware detection using machine learning are increasing

- At the same time, researches on avoidance method against these detection are also increasing

Previous Researches

- Hu et al. generated feature quantities of malware to avoid detection by MalGAN



Configuration of MalGAN

Issues and Approaches

1) Directly import the malware detector into MalGAN, training, and predict

→ Execute the malware detector externally, and import results

2) Reduce the feature quantities in MalGAN to 128 dimensions using the variable importance of RF

→ Use all APIs in the malware

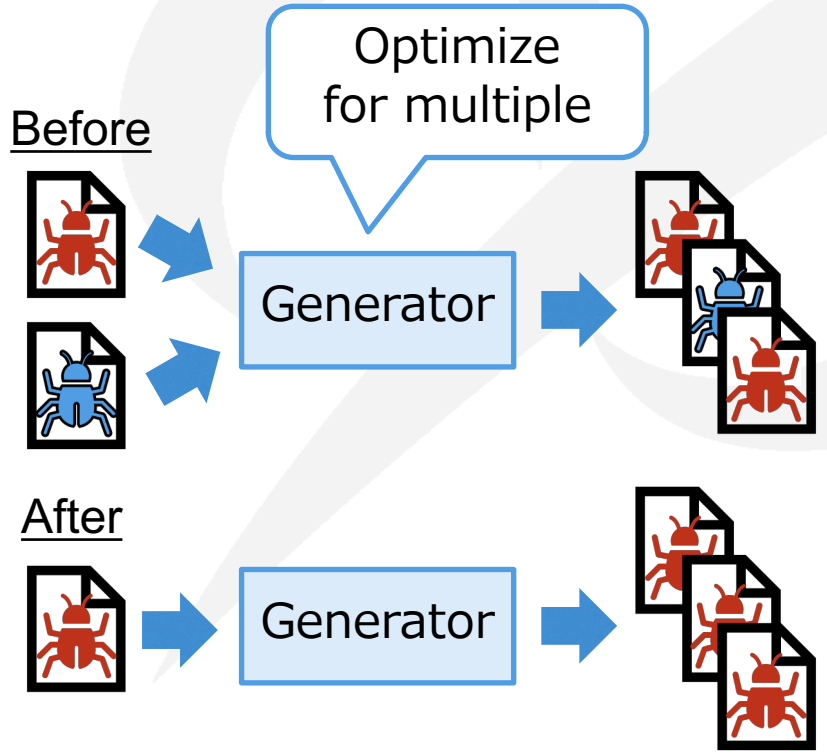
Issues and Approaches

3) Use the same API list as feature quantities, for both MalGAN and the malware detector

→ Create API list from different training data

4) Use multiple malware to learn in MalGAN

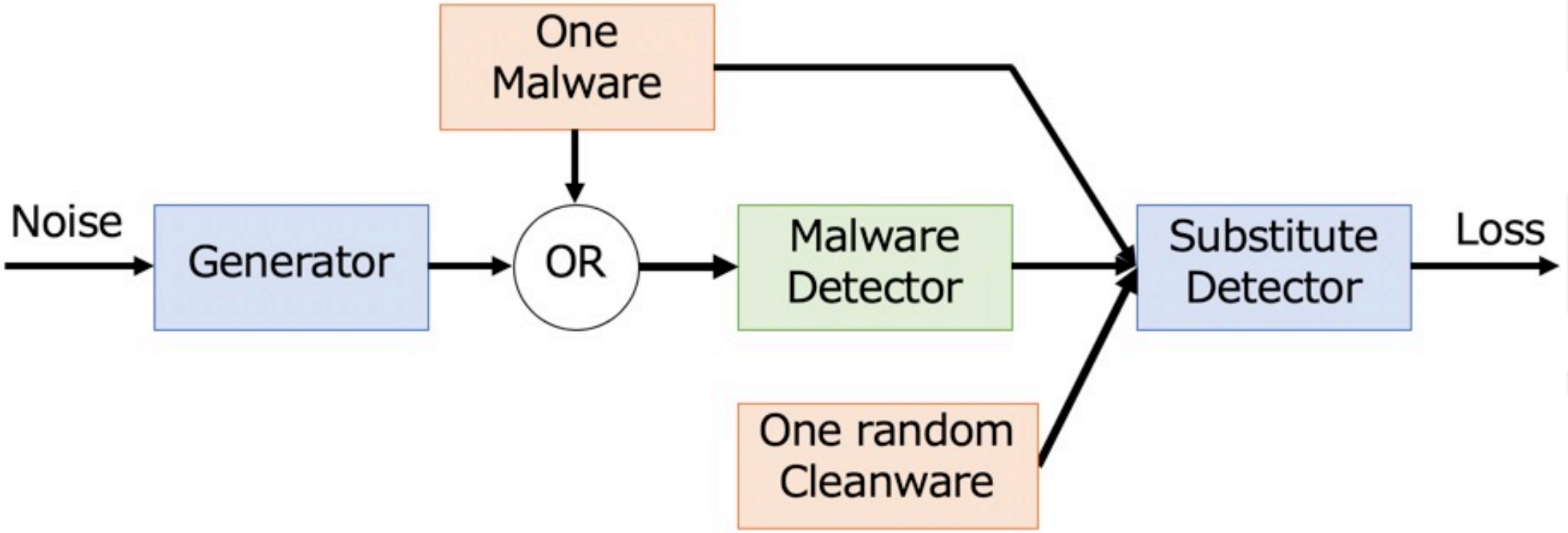
→ Only use one malware



Other Approaches

- Modify the network considerably, and change the range of data ([0,1]→[-1,1])
- Change optimizer parameters
- Change noise dimensions (20→all APIs in the training data)

Improved MalGAN



Configuration of Improved MalGAN

Experimental Settings

- Environment
  - Program language : Python 3.5.0
  - MalGAN : Tensorflow 1.11.0, Keras 2.2.4
  - Malware detector : Scikit-learn 0.20.0

- Dataset
  - FFRI Dataset 2018
    - Surface analysis data and hash value

Experimental Settings: Malware Detector

- Machine learning algorithm
  MLP (hidden layer size = (64,))
  RF (n estimators = 1000)

- Training data Malware 36265, Cleanware 26127
- Test data Malware 36047, Cleanware 26590

Experimental Settings: Malware Detector

- Create four patterns of API lists

Pattern	Malware	Cleanware	APIs
I	1104	826	16337
II	75	60	4045
III	1104	826	128
IV	75	60	128

Experimental Settings: Malware Detector

- The classification results of the test data

Pattern	RF		MLP	
	TPR(%)	FPR(%)	TPR(%)	FPR(%)
I	96.9	3.8	97.4	4.5
II	96.9	3.9	97.4	4.6
III	96.0	8.2	95.9	9.2
IV	95.6	8.5	95.5	8.9

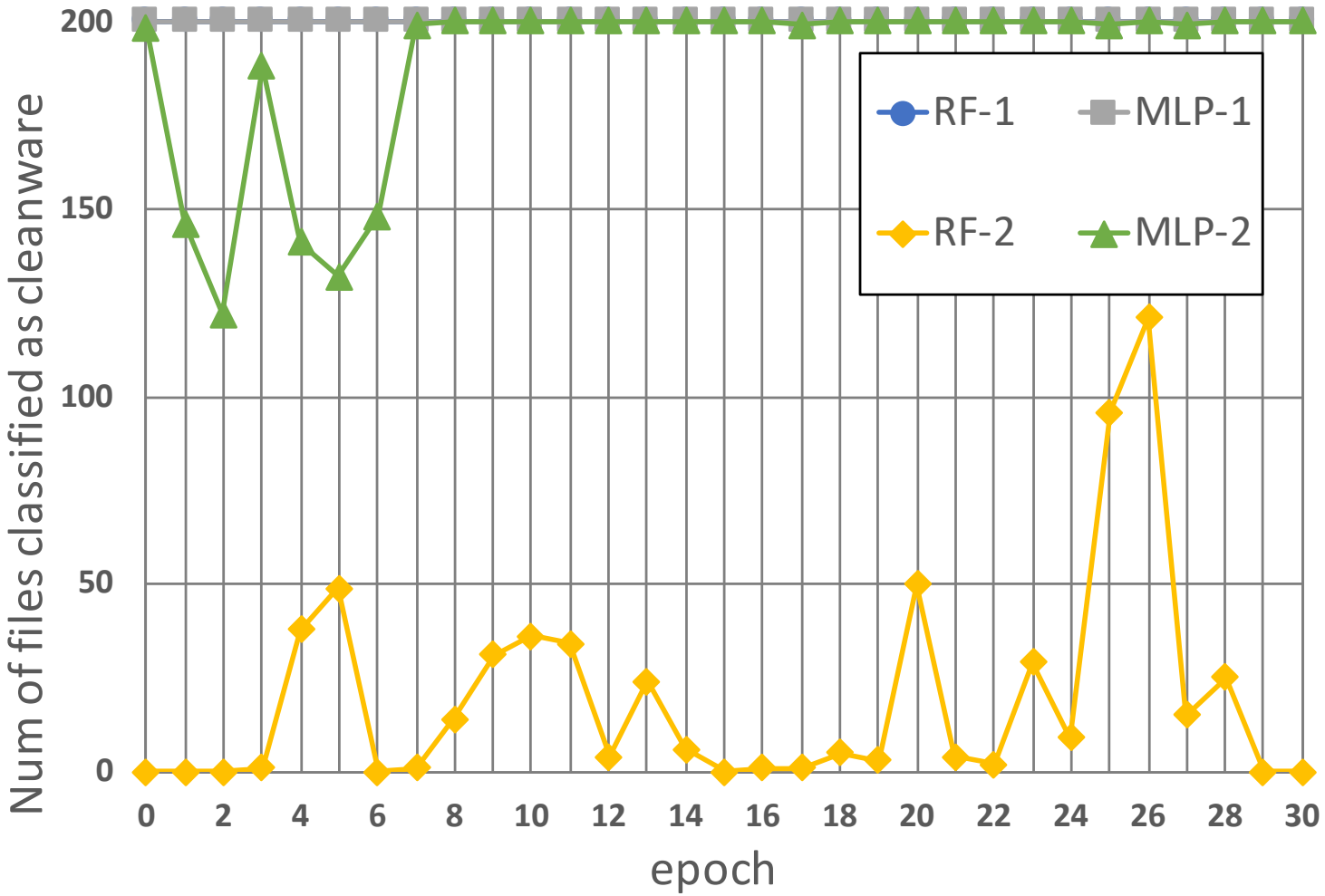
Experimental Settings: Improved MalGAN

- Training data　　Malware 1, Cleanware 4 ~ 44
- Create two patterns of API lists

Pattern	Malware	Cleanware	APIs
1	1	44	2452
2	1	4	774

- Generate 200 files for each epoch

Experimental Results: Improved MalGAN



In RF-1, MLP-1 and MLP-2, almost files are classified as cleanware

Avoiding is a success !

Easy to avoid as the number of dimensions of MalGAN is increased

Result of Pattern II

Issues from Experimental Results

As the number of dimensions of MalGAN increases,
the number of APIs of generated data also increases

Original Malware	Pattern Ⅱ MLP-1	Pattern Ⅱ MLP-2
148	1270	451

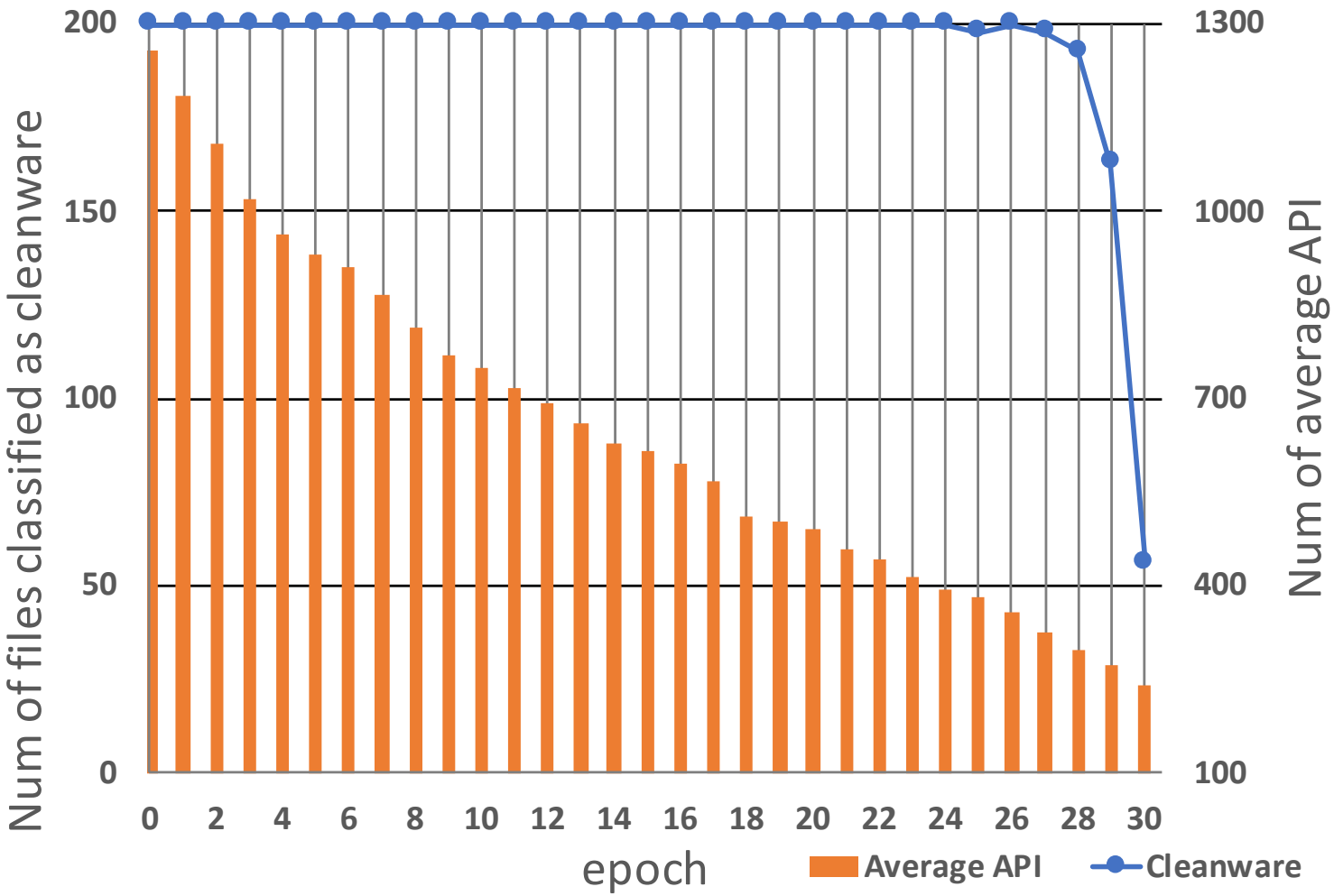
Solution

Add loss calculation Layer to Generator

$$customized\ RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{m}(\hat{x}_{ij} - x_{ij})^2}$$

Distance between the generated data $\hat{x}$ and the original malware x

Experimental Results: Applied loss calculation Layer



At 0 epoch about 1300 APIs are generated, but at 30 epochs reduced to about 250 APIs

Reducing is a success !

Result of Pattern II MLP-1

Countermeasures for MalGAN

MalGAN tries to avoid by adding feature quantities of cleanware as noise

- Be able to hinder by applying a learning method to make it resistant to noise
- Apply semi-supervised learning to the malware detector

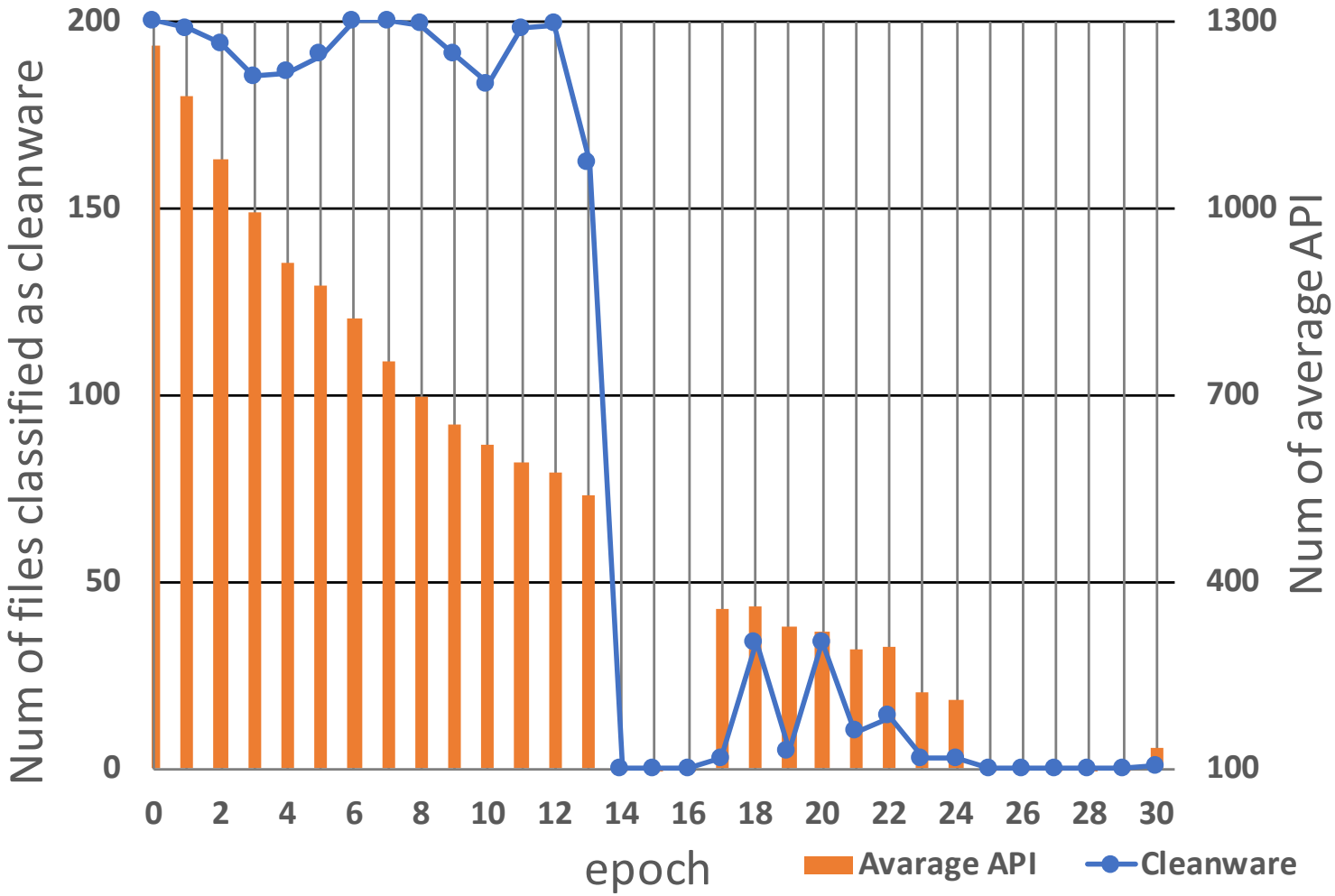
Self-Training

1. Create one classifier from a few labeled data
2. Classify non labeled data
3. Add high prediction probability data among them to the classifier
4. Repeat 2 to 3

In this experiment

- Labeled data:　Malware 500, Cleanware 500
- Non Labeled data :　Others
- Prediction probability:　More than 80%
- Repeat times:　Until 200
- Classification result (Pattern II MLP):　TPR 90.8%　FPR 11.1%

Experimental results: Applied Self-Training



After 13 epochs, the number of cleanware is extremely reduced

Before 13 epochs, there are no problems since the number of average API is large

Hindering is a success !

Result in Pattern II MLP-1

Conclusions

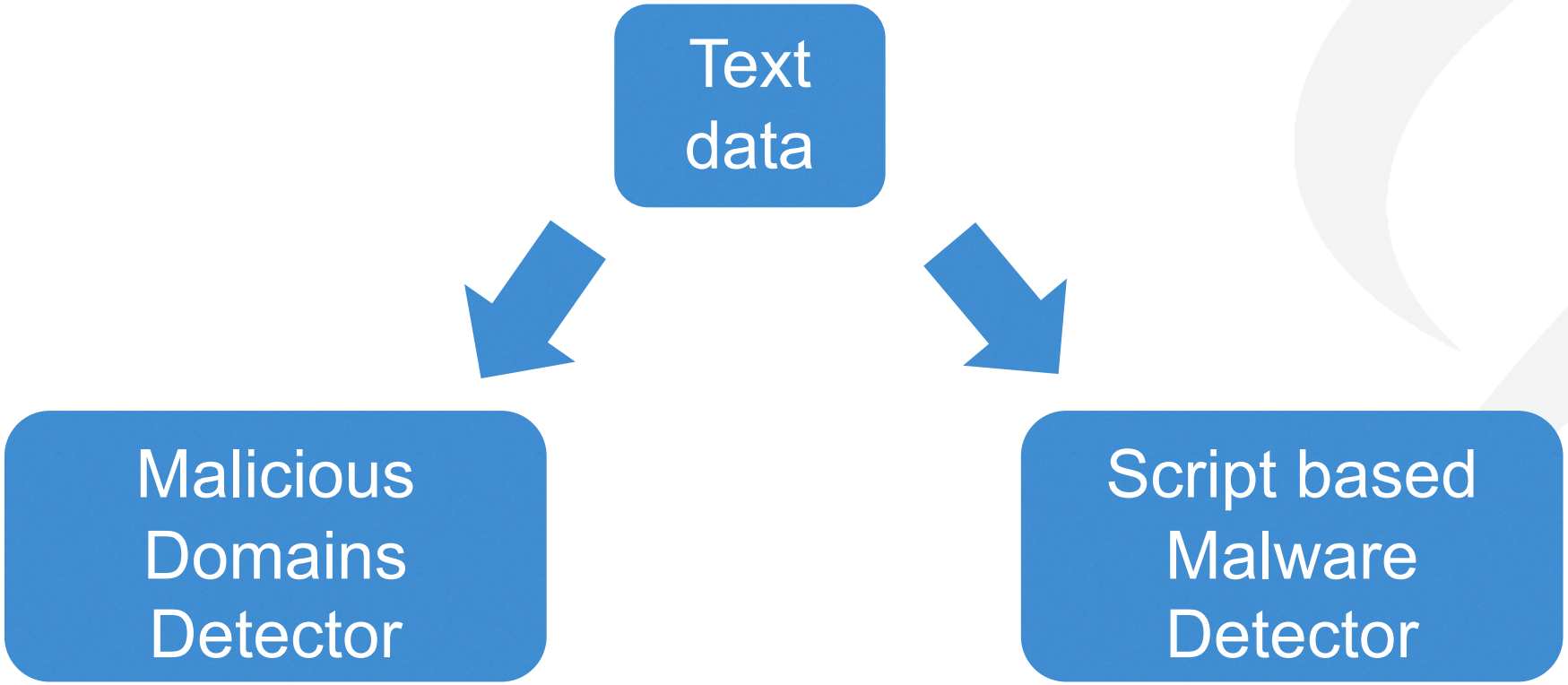
・We improve the issues of MalGAN and generate feature quantities to avoid detection

・We solve the issue of generating a huge number of API data by adding loss calculation Layer

・We propose self-training as a learning method to hinder avoidance, and confirm it is feasible

Future work

- Apply other feature quantities to MalGAN



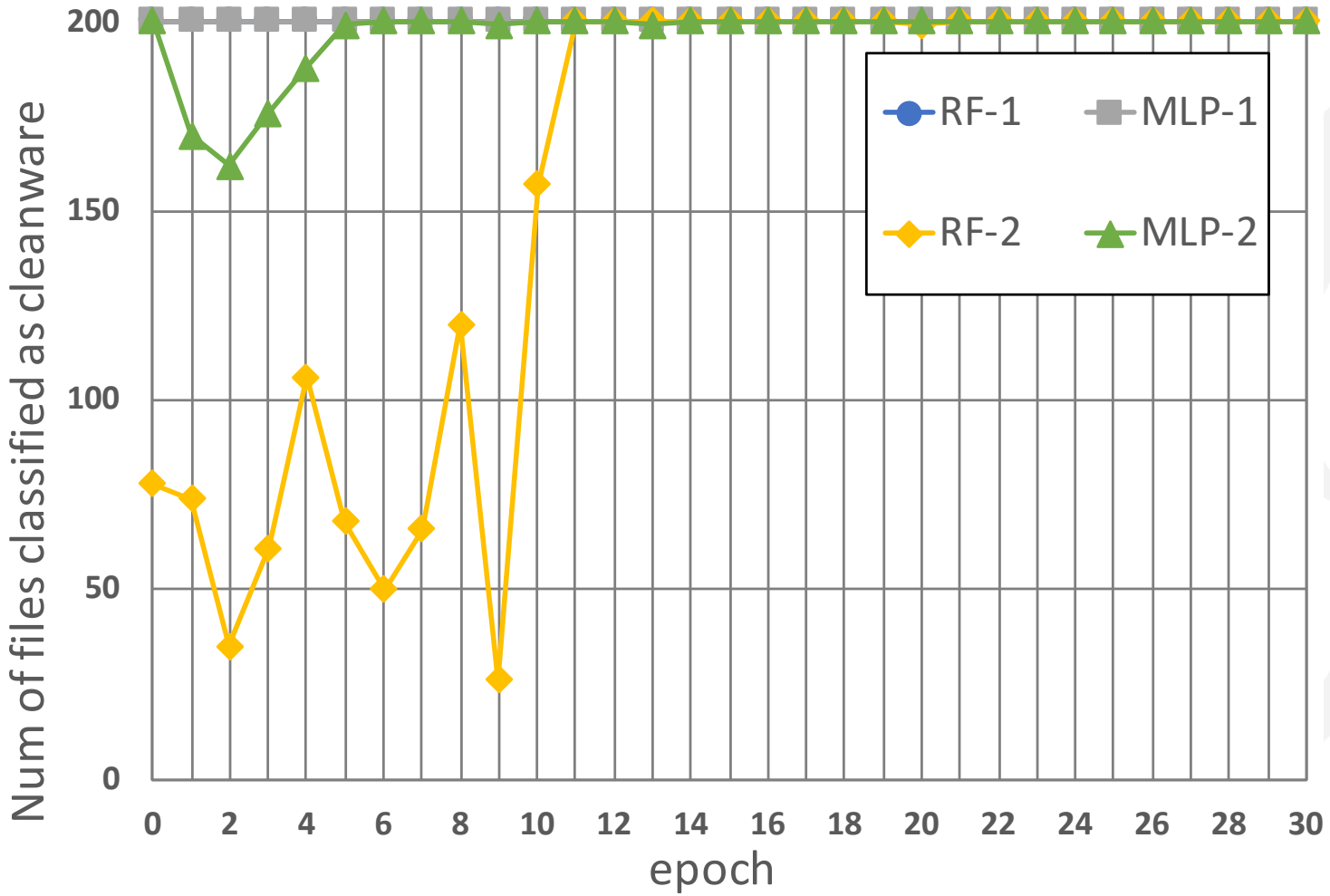
Q&A

Thanks !

Masataka Kawai
15024306@mmm.muroran-it.ac.jp

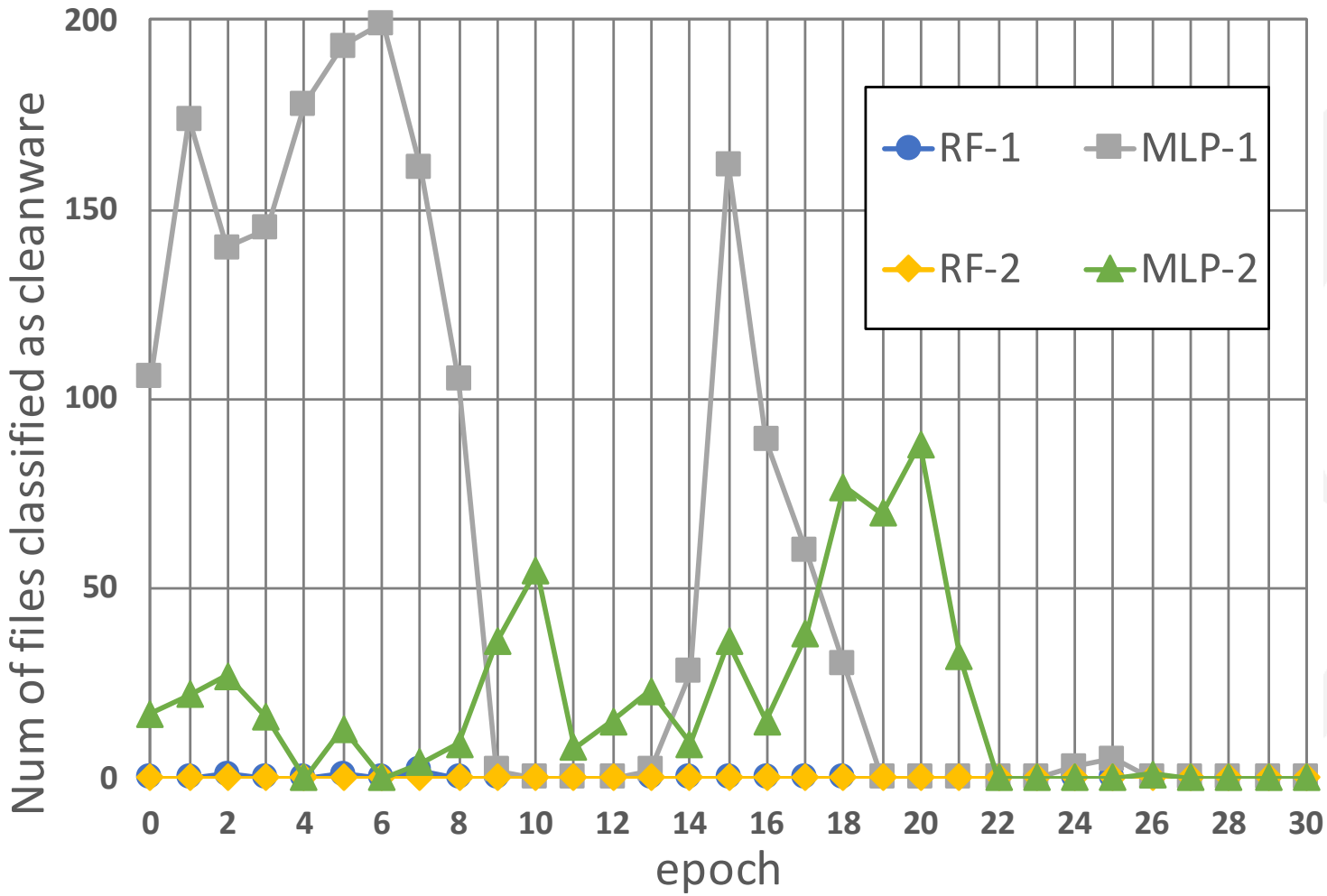
Backup slides

Experimental results: Improved MalGAN



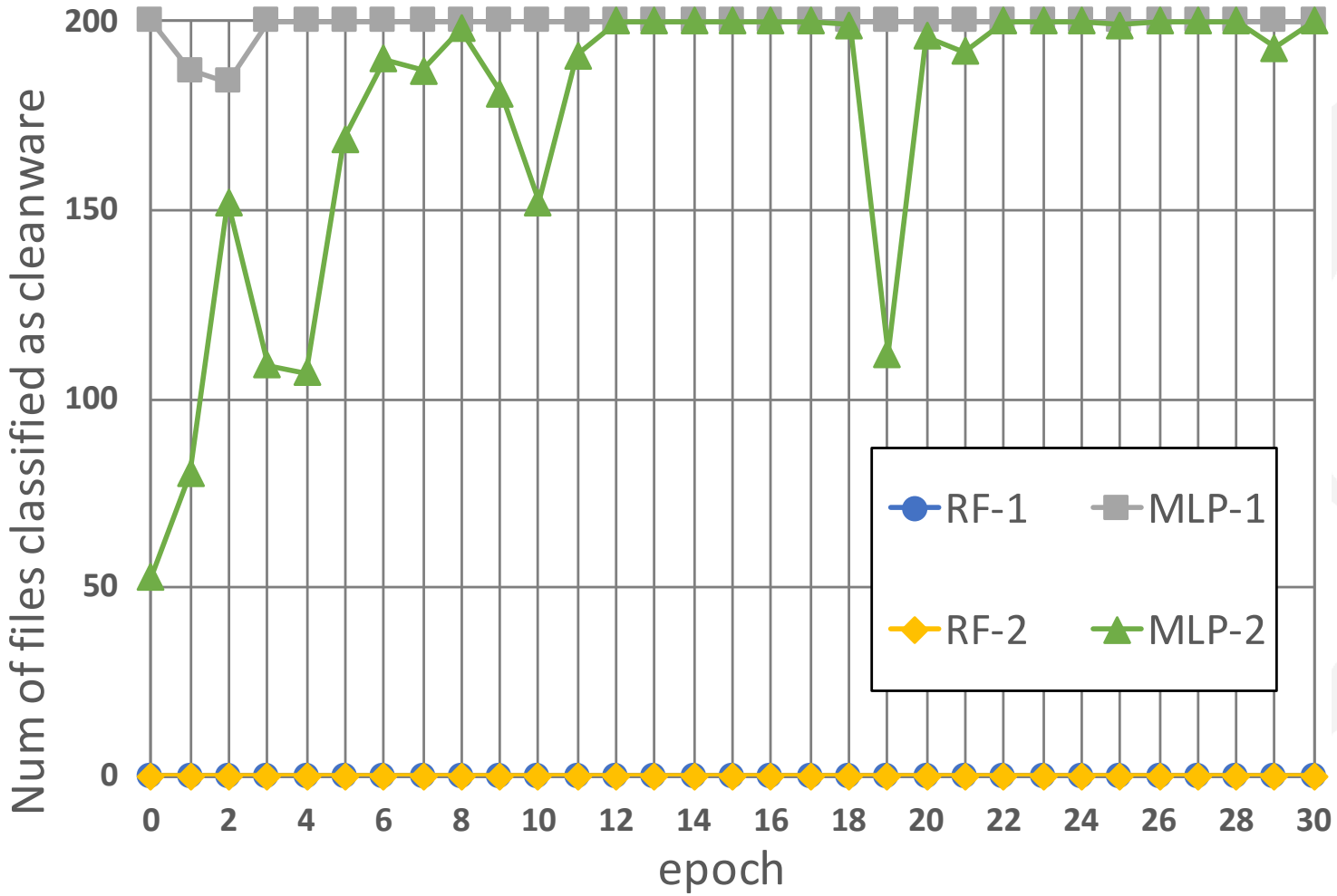
Result of Pattern I

Experimental results: Improved MalGAN



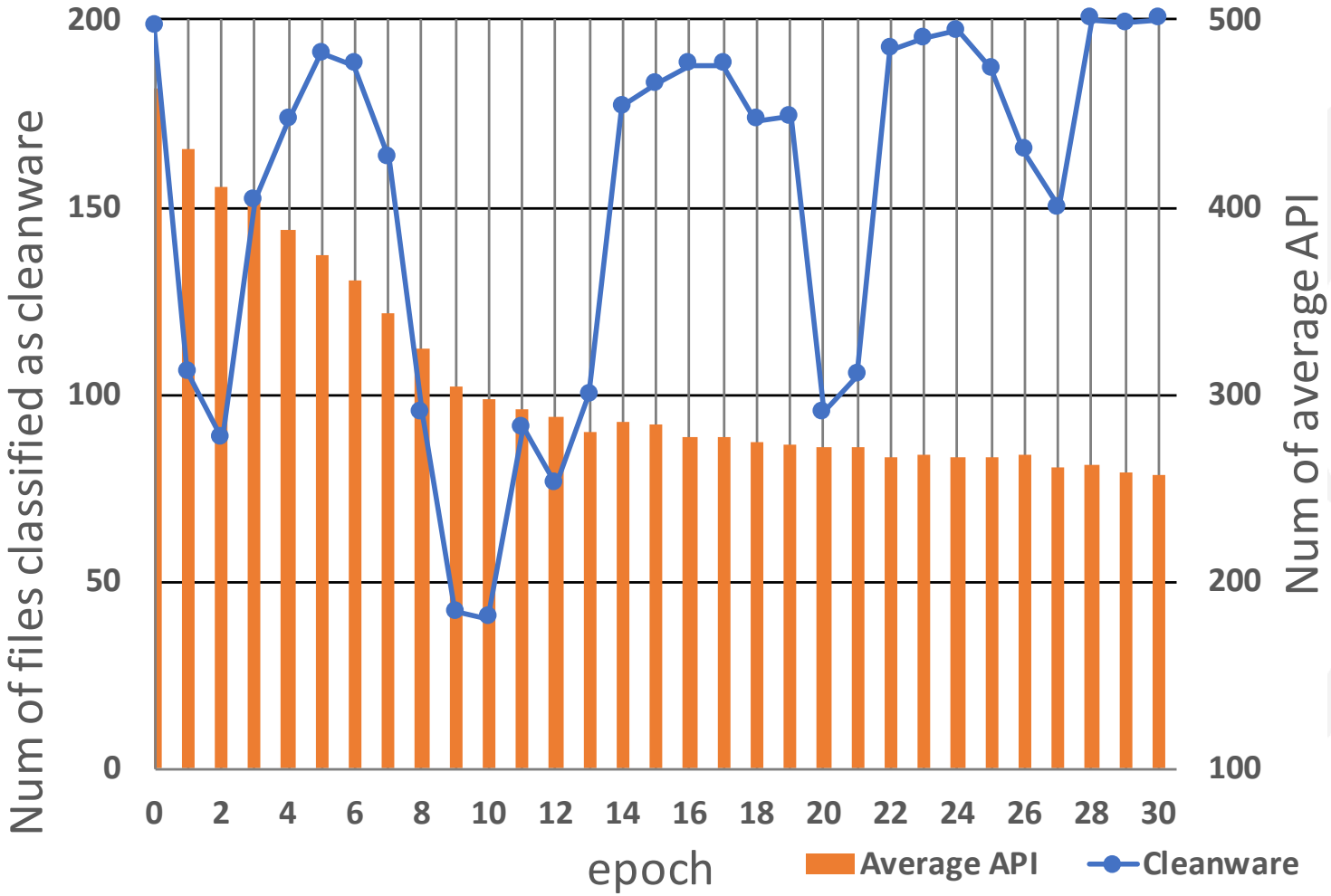
Result of Pattern III

Experimental results: Improved MalGAN



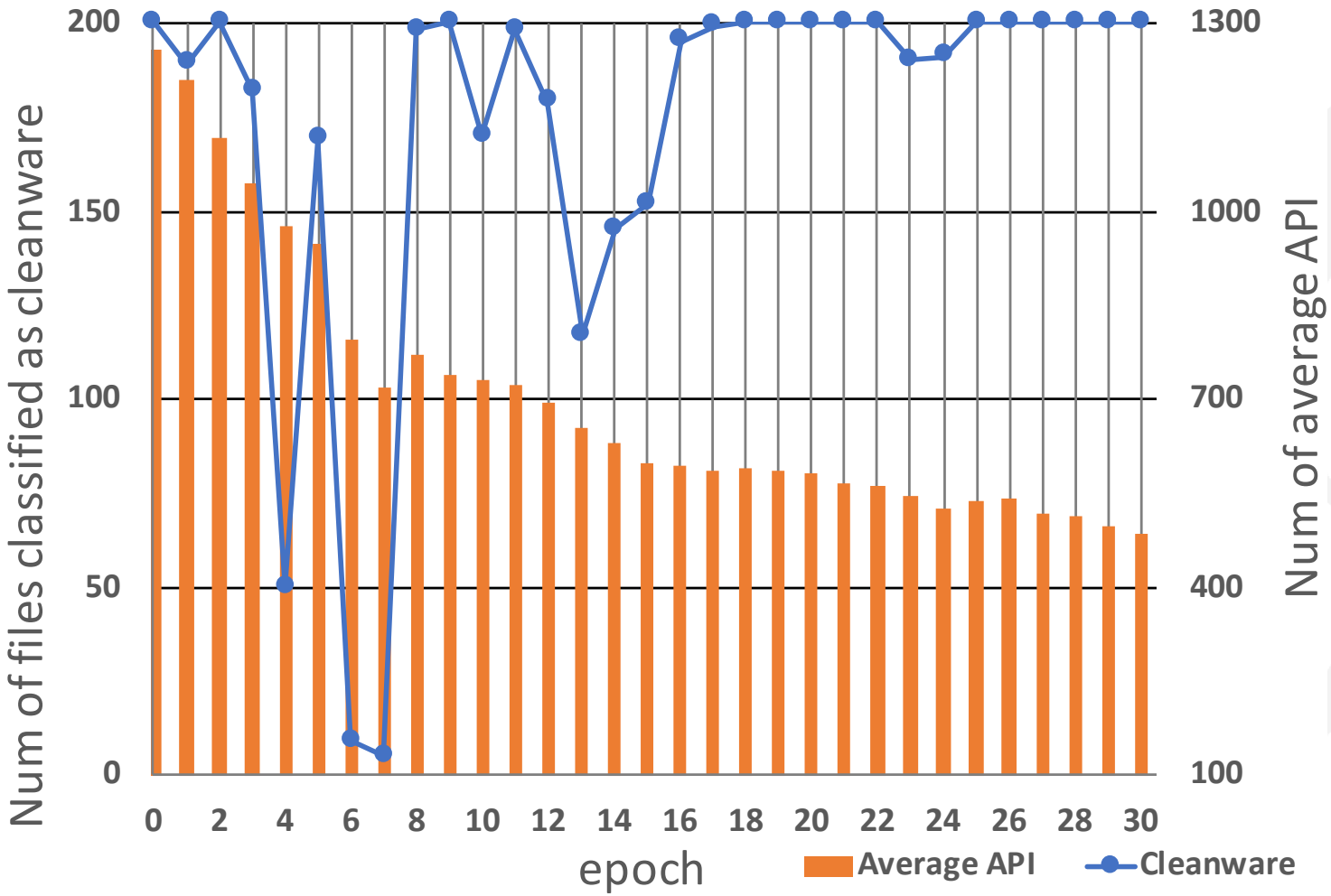
Result of Pattern IV

Experimental results: Applied loss calculation Layer



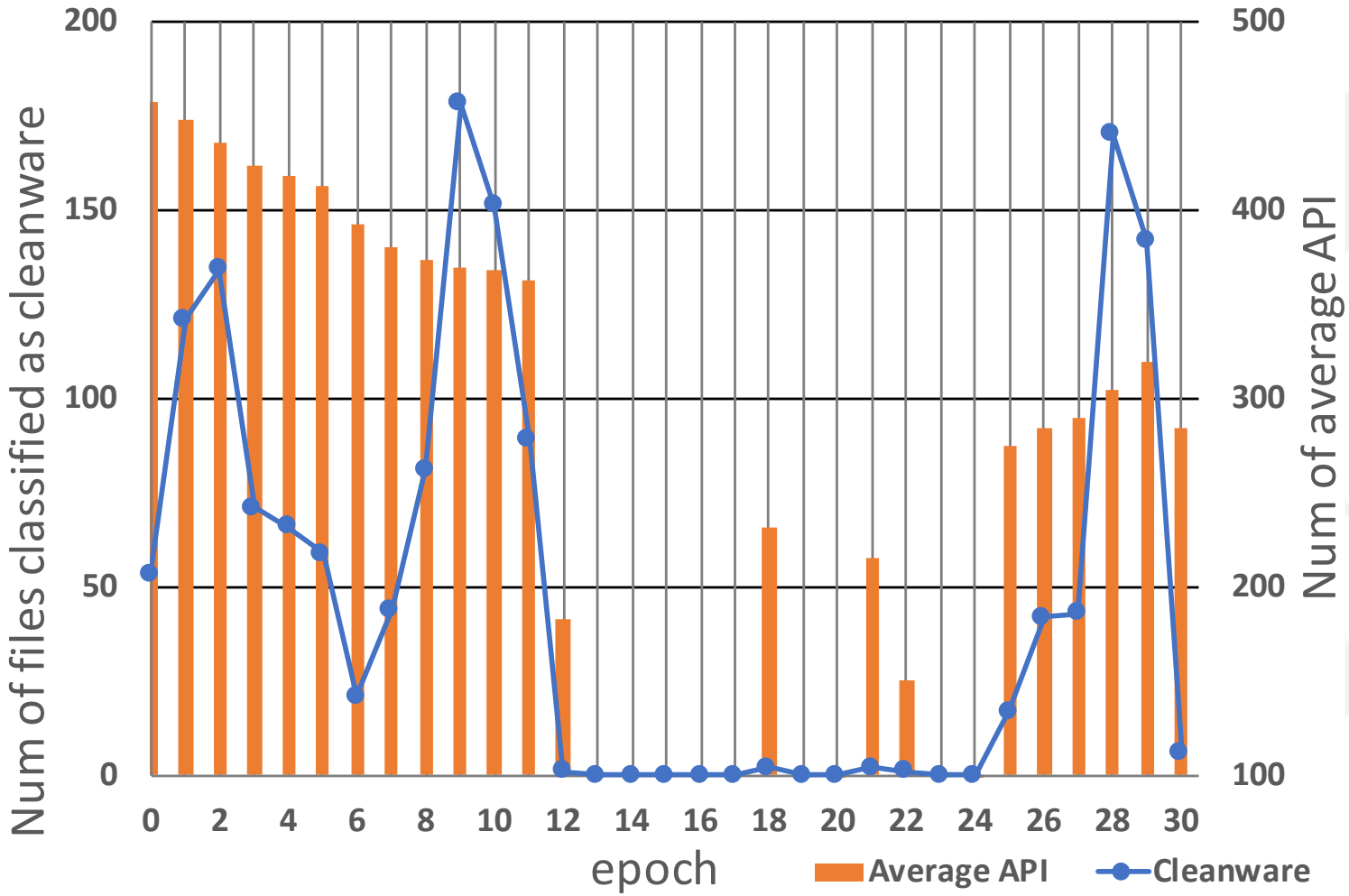
Result of Pattern II MLP-2

Experimental results: Applied loss calculation Layer



Result of Pattern IV MLP-1

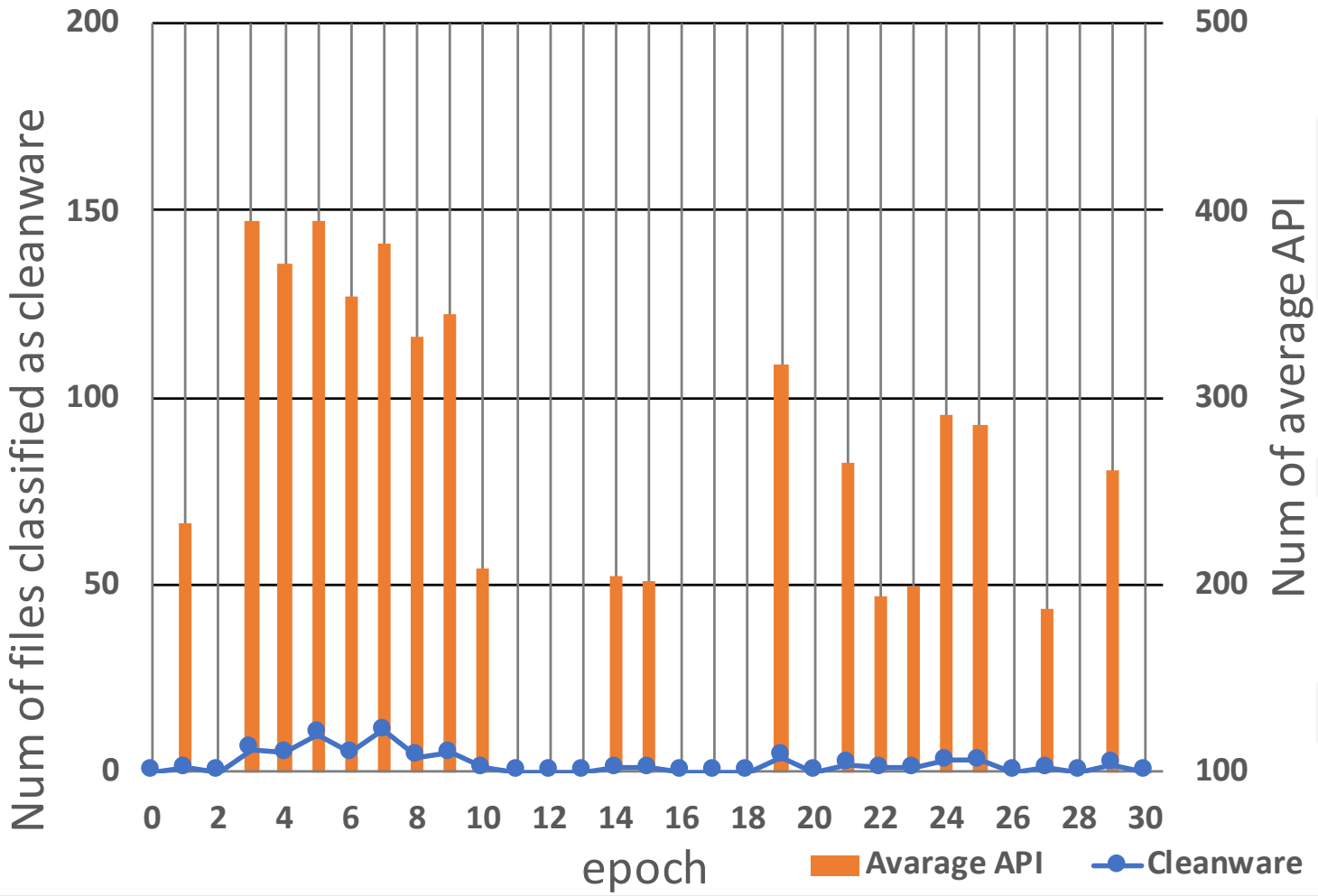
Experimental results: Applied loss calculation Layer



Result of Pattern IV MLP-2

室蘭工業大学
MURORAN INSTITUTE OF TECHNOLOGY

Experimental results: Applied Self-Training



Result of Pattern II MLP-2