#dfist24

devfest
Istanbul 2024

# Adversarial Attacks to Next Generation Antivirus Models

**Fatih ERDOĞAN**

**Sr. Blue Team Engineer, Picus Security**

# $ whoami

- **Fatih ERDOĞAN**
- Sr. Security Engineer @**Picus Security**
- Working in **Blue Team**
- DFIR, malware analysis, threat hunting, security r&d
- Vulnerability research
  - CVE-2024-8011, CVE-2024-36287, CVE-2024-2537, CVE-2023-7224
- Speaker
  - The H@ck Summit, Hacktrick, IEEE CSCON, Linux Yaz Kampı, Akademik Bilişim
- Formerly: Trendyol, Zemana, STM, Prodaft


**@FeCassie**

**fatiherdogan1**

PICUS

## Outline

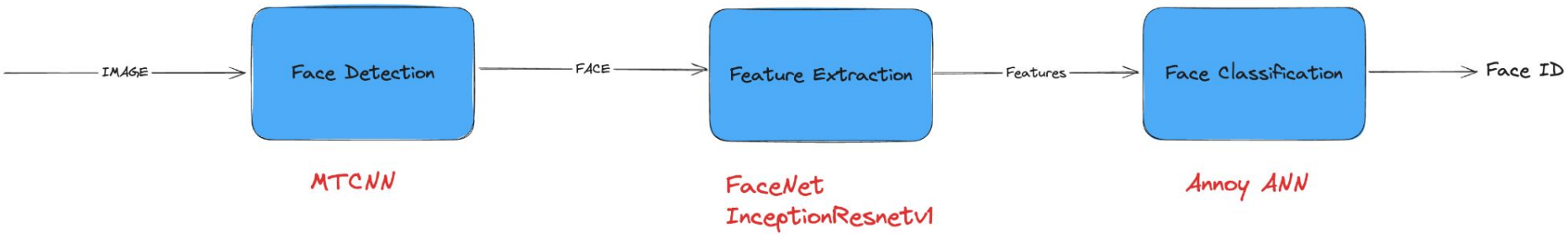
- ❏ Why this research?
- ❏ Object/Face Detection
- ❏ Adversarial Attacks
- ❏ Antivirus Concept
- ❏ AI-Based Malware Detection
- ❏ Designing to Adversarial Attacks
- ❏ Conclusion

# Why this research?

- ❏ Object/Face Detection
- ❏ Prediction accuracy
- ❏ Adversarial Attacks
- ❏ AI-Based Antivirus Models
- ❏ Malware detection using Deep Learning
- ❏ How can be bypassed??

# Object/Face Detection

# Face Recognition

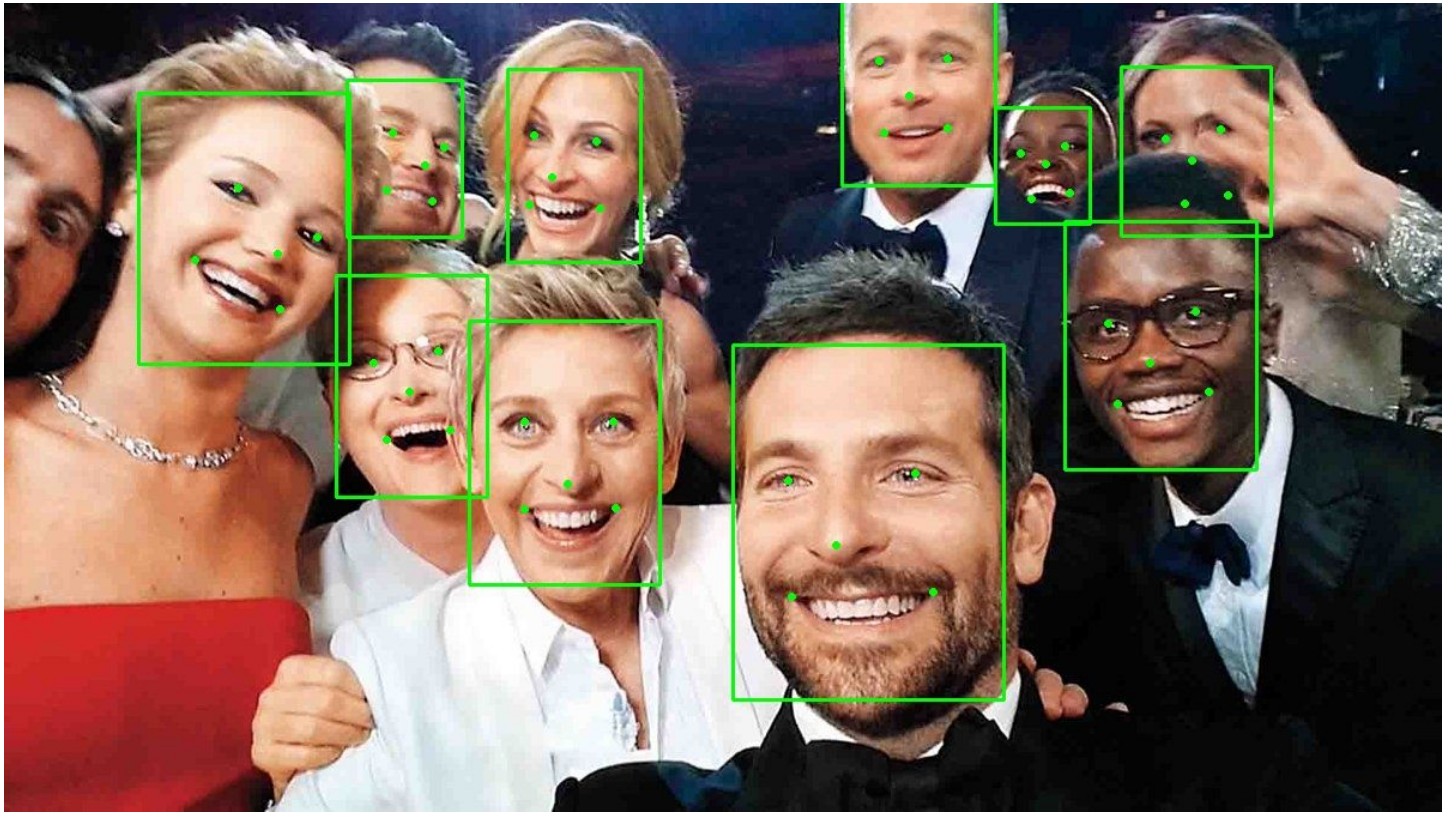



PyTorch

kaggle

https://github.com/ipazc/mtcnn
https://github.com/timesler/facenet-pytorch
https://github.com/spotify/annoy
https://www.kaggle.com/code/timesler/guide-to-mtcnn-in-facenet-pytorch

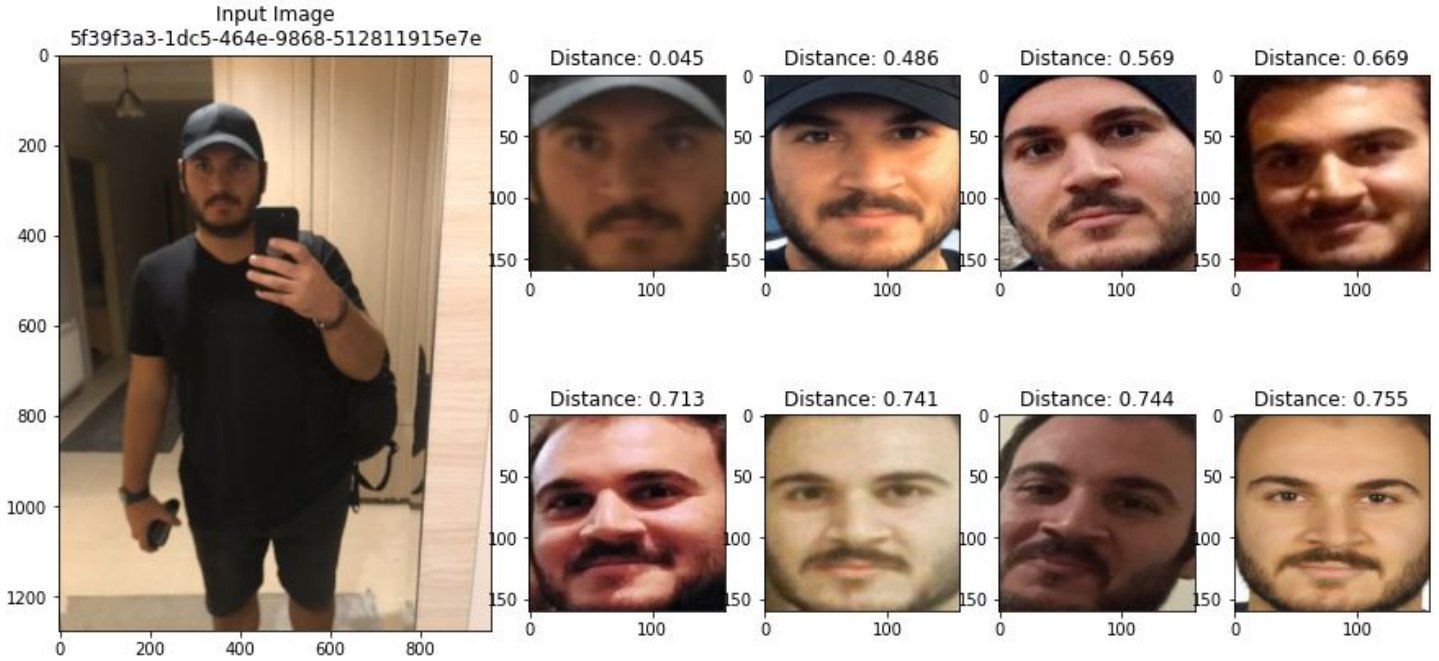
# Face Recognition



https://seeen.com/news/seeen-face-recognition-system

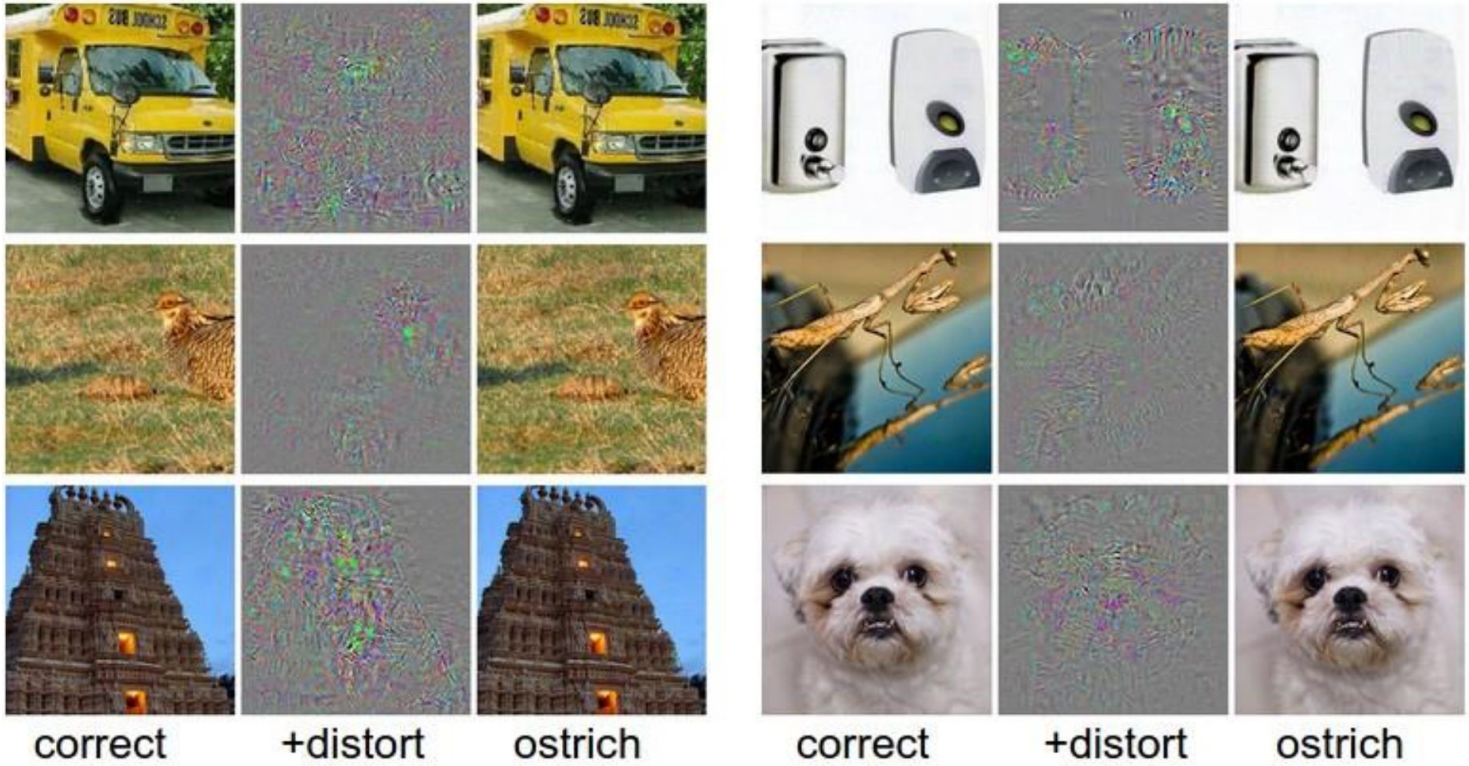
# Face Detection

```
FOUND CLASS 5f39f3a3-1dc5-464e-9868-512811915e7e

Face is found  5f39f3a3-1dc5-464e-9868-512811915e7e Min distance: 0.04509807378053665
```



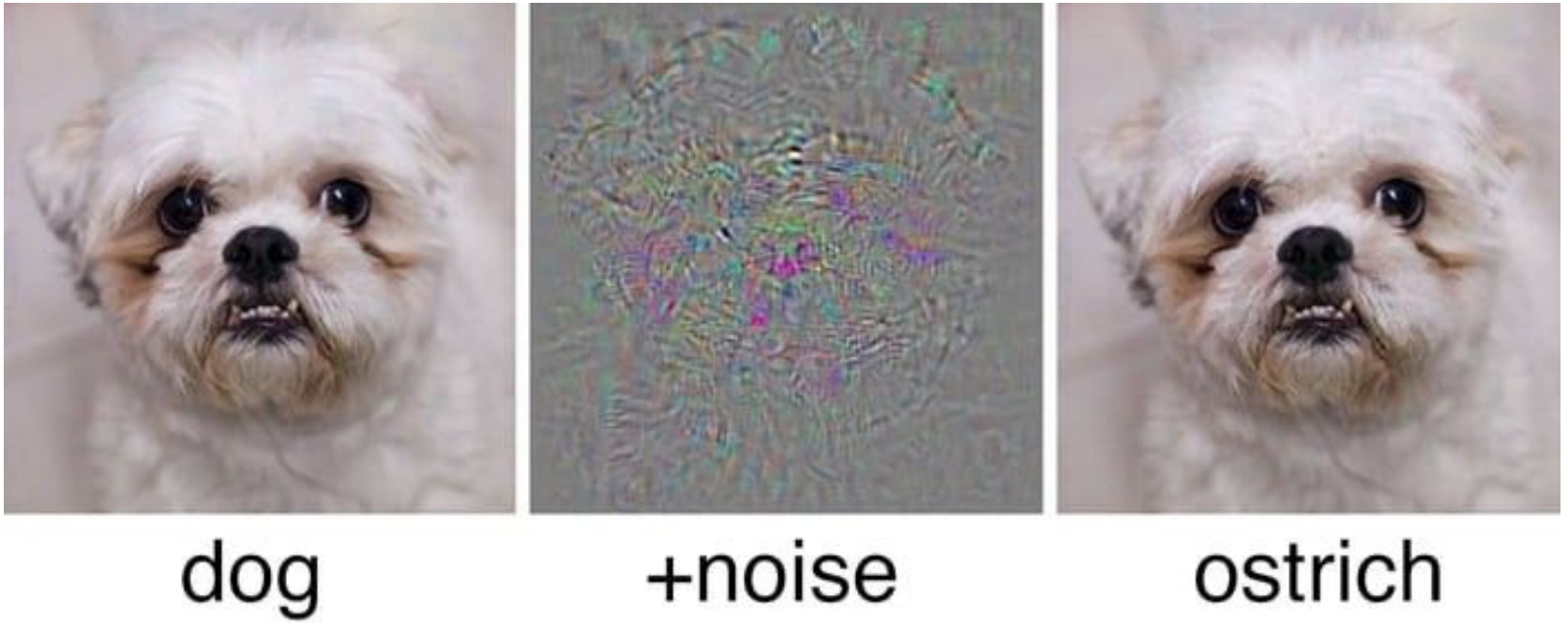
# Adversarial Attacks

# Adversarial Attacks




https://medium.com/sciforce/adversarial-attacks-explained-and-how-to-defend-ml-models-against-them-d76f7d013b18

# Adversarial Attacks

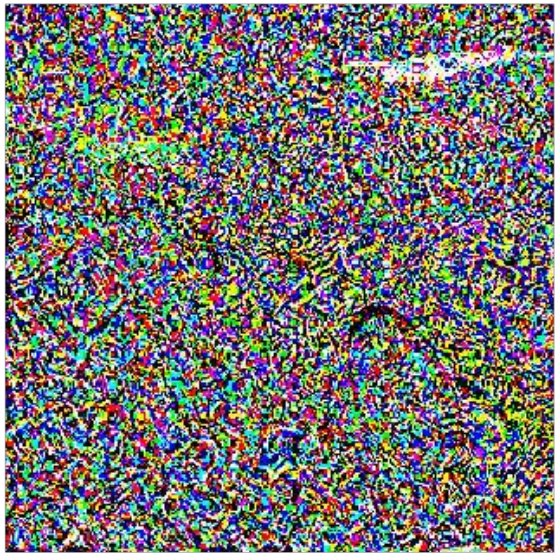


# Real Life Examples

# Sports Car

Original image: sports car

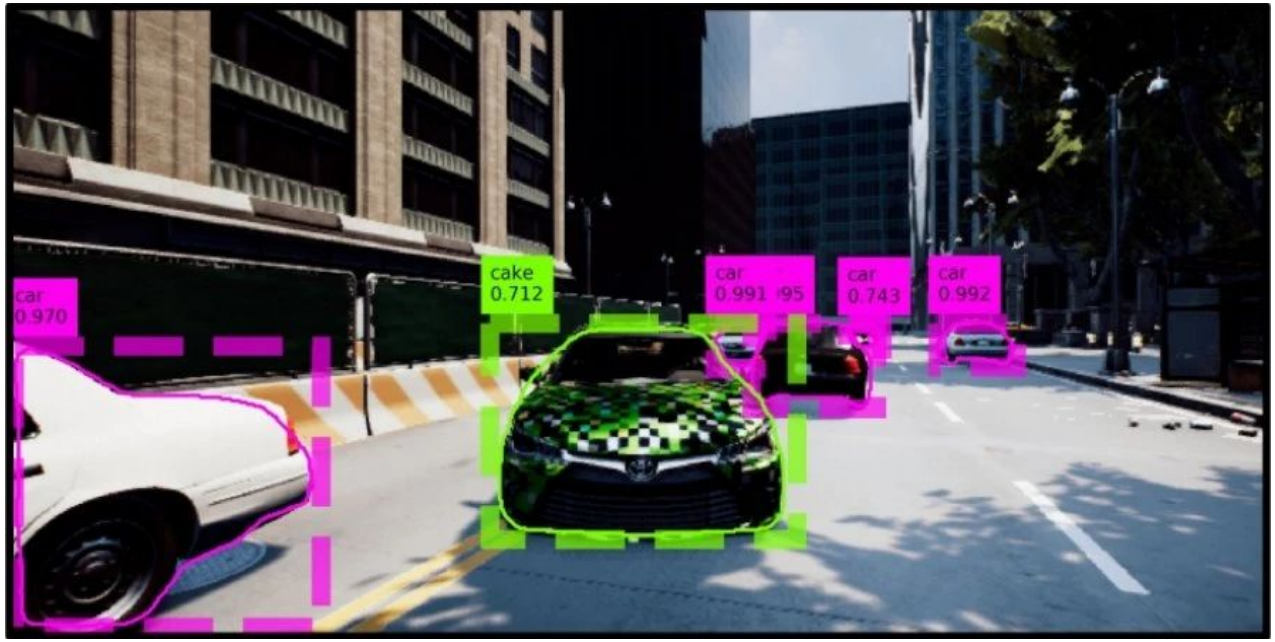
Attacking noise



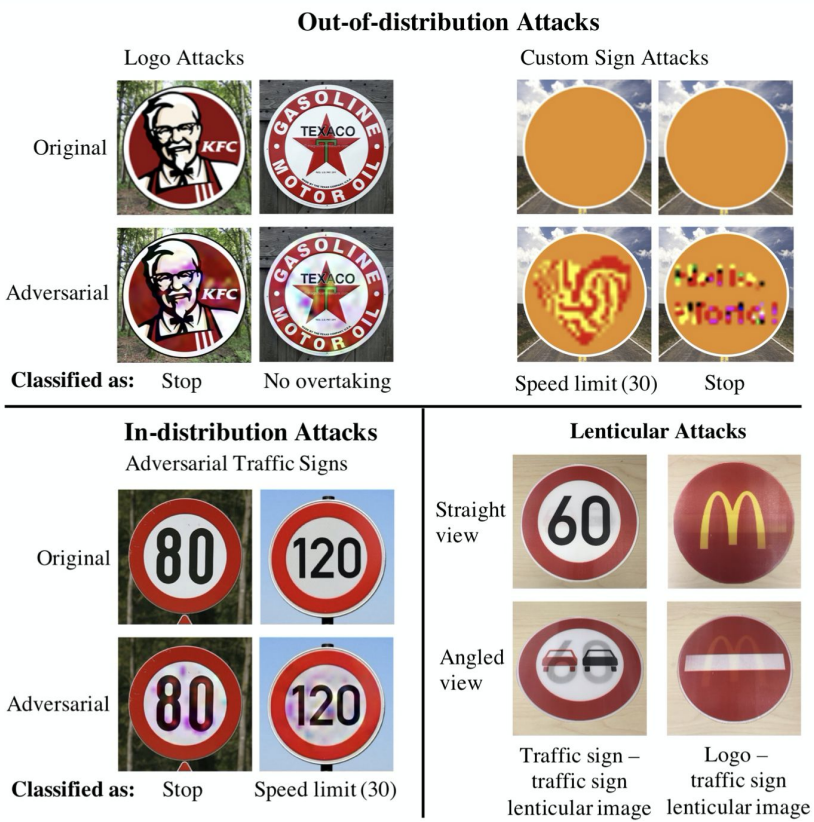
Adversarial example: toaster

https://www.linkedin.com/pulse/using-adversarial-attacks-make-your-deep-learning-model-rodriquez/

# Autonomous Driving

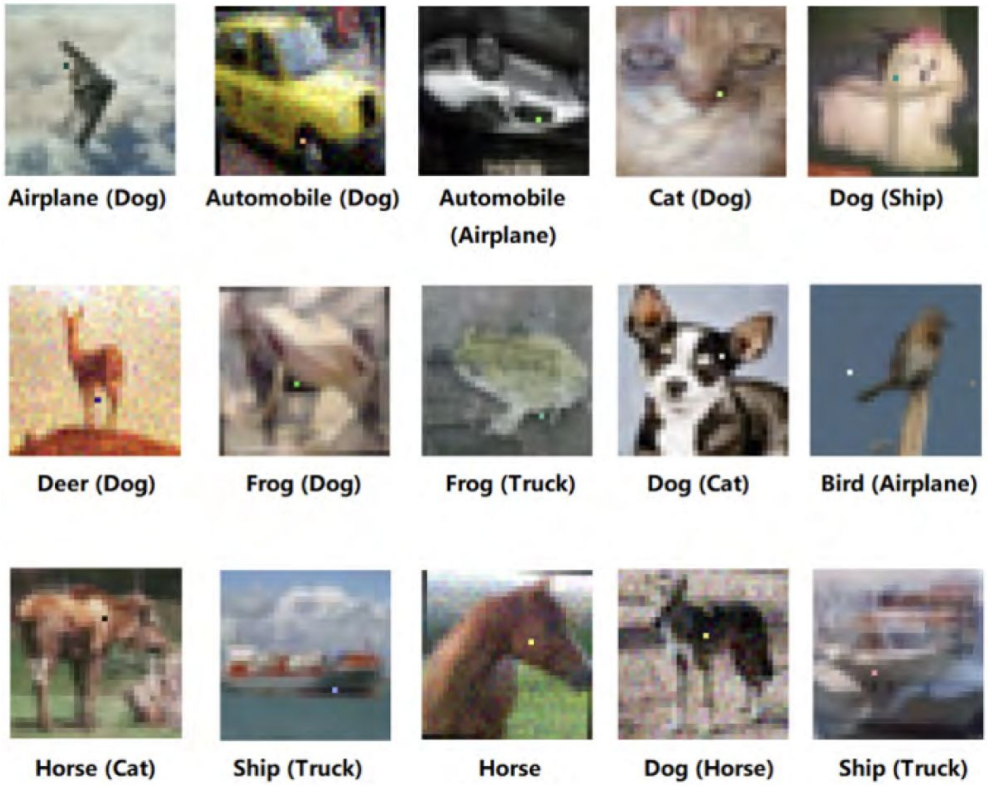


# Traffic Sign




https://arxiv.org/pdf/1802.06430

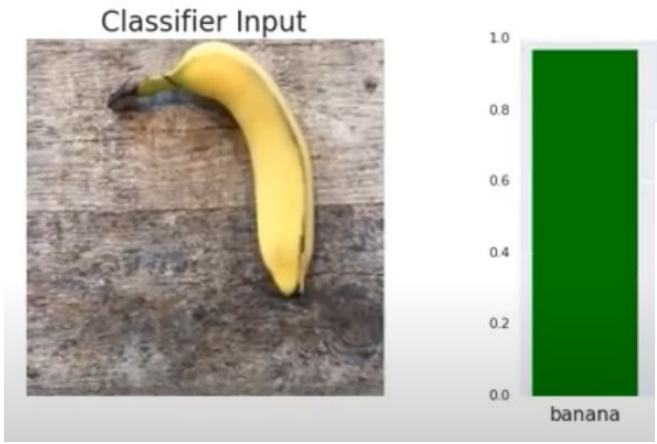
# Image Classification




https://huggingface.co/datasets/uoft-cs/cifar100

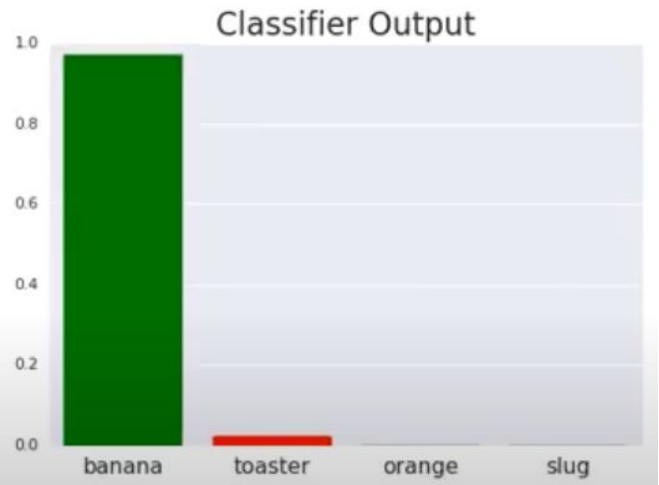
# Adversarial Patch

# Adversarial Patch







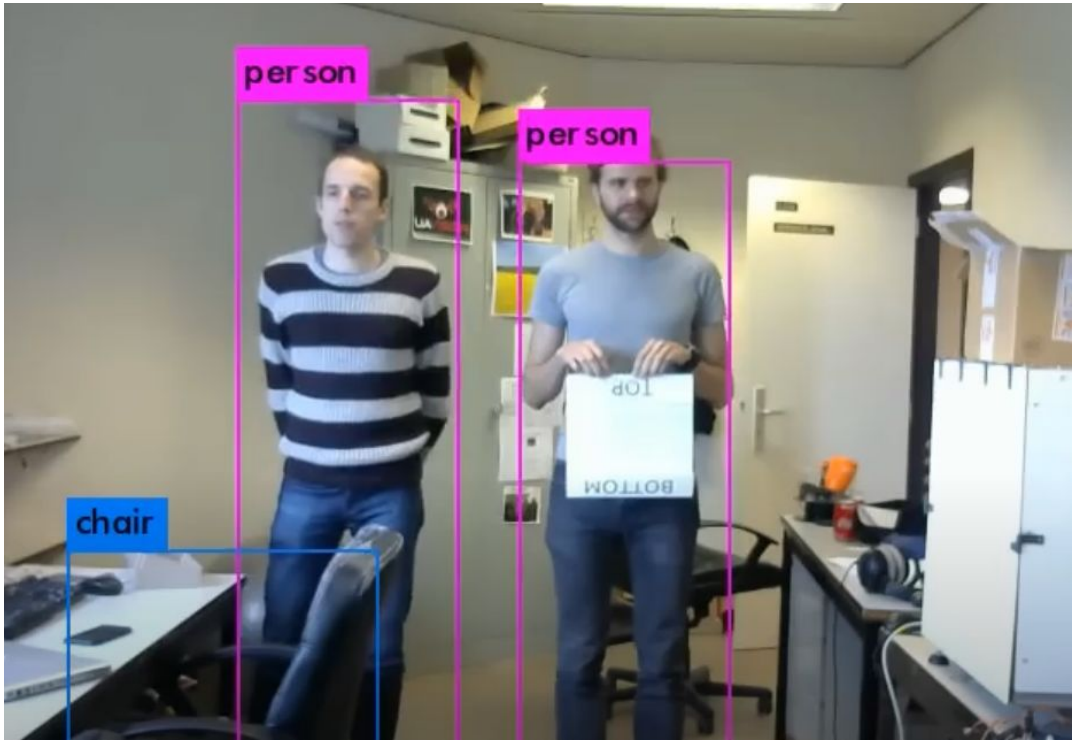



# Adversarial Patch


https://www.youtube.com/watch?v=i1sp4X57TL4

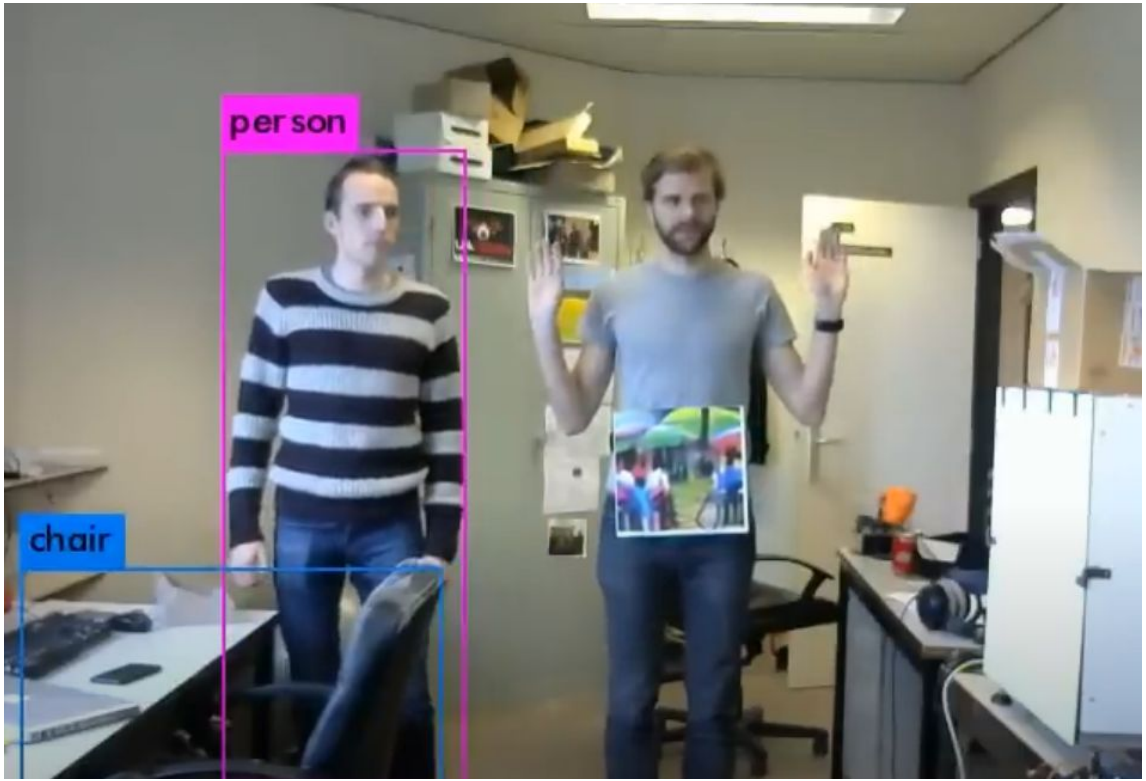
# SQL Injection

# Adversarial Patch




Generating Adversarial Patches Against YOLOv2

# Adversarial Patch

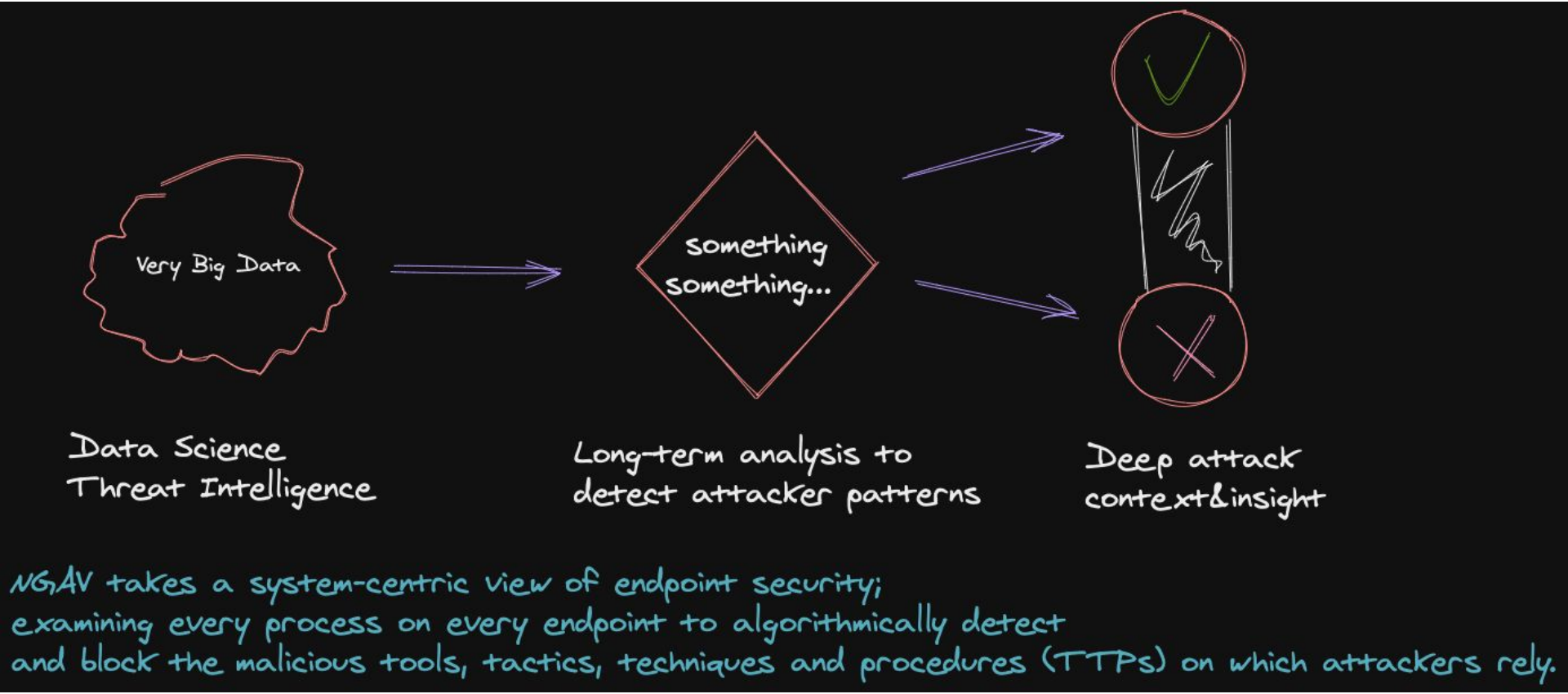



Generating Adversarial Patches Against YOLOv2

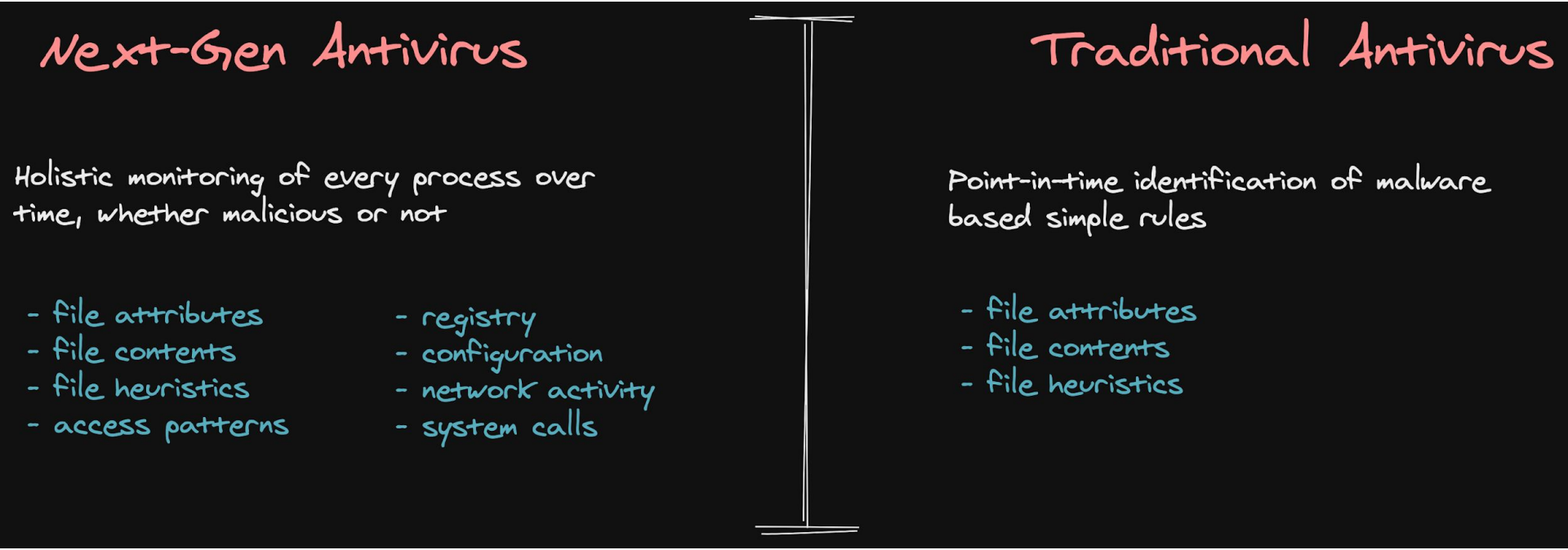
# Antivirus Concept

# Legacy Antivirus

# Next-Gen Antivirus



# Next-Gen / Legacy AV



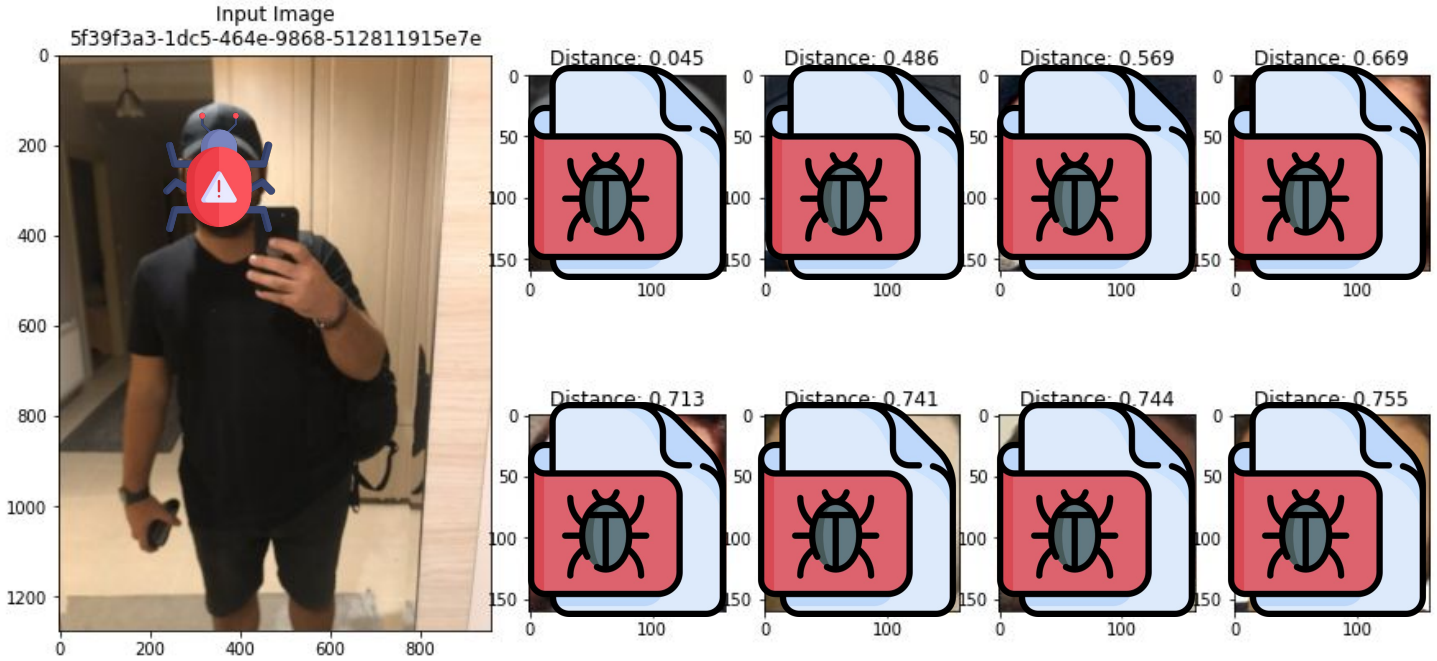
# Idea!

# Idea!

```
FOUND CLASS 5f39f3a3-1dc5-464e-9868-512811915e7e

Face is found  5f39f3a3-1dc5-464e-9868-512811915e7e Min distance: 0.04509807378053665
```

# Idea!

```
FOUND CLASS 5f39f3a3-1dc5-464e-9868-512811915e7e

Face is found  5f39f3a3-1dc5-464e-9868-512811915e7e Min distance: 0.04509807378053665
```



# Malware Detection via Binary Visualization

# AI-Based Antivirus – How??

AI

me

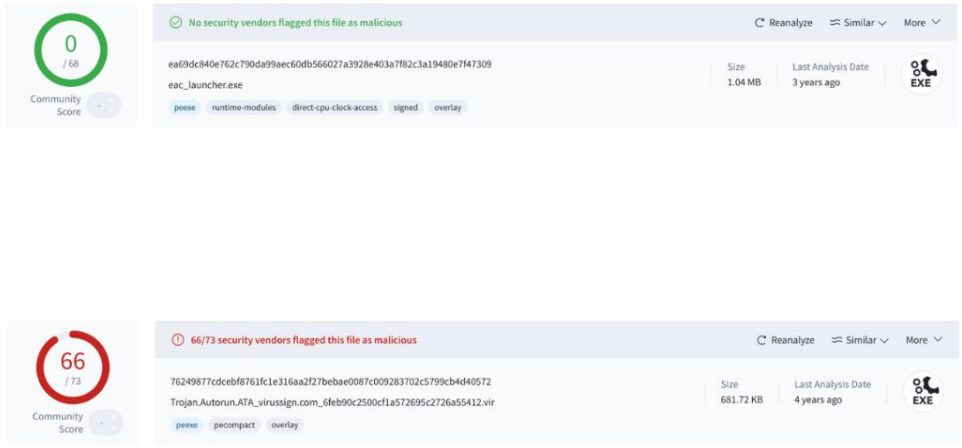
# Training Environment

- Deep Learning
  - Tensorflow
  - Keras
- NVIDIA RTX 2080Ti
- AMD Ryzen 5 - 16 Core
- Google Colab
- Jupyter Notebook




# Dataset Preparation

VirusTotal



# Malicious Files

```
/home/lambda1/Desktop/Neural-Engine/dataset/2020_haziran/malware/dce043360bfb5b46ca87456bbd9ba20cedfaa2d6c2fee2ebba8c13ddb832584f|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/ddfa8770a35499259345fc66b736660aac4cb74c1f544bb5e751b84b7d79ace6|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mart/malware/6d8b54952ca602ba1d5e3571b999a87e00fce8366a2cba64d8eda5d7857ccf50|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_haziran/malware/e1434d8d560f934abcffef6f75288958c0e19c9c32fba20cf08efee17f070938|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/6612960b24f0f8b8a050a880ec48cf589c67a7ece212e7426db648e79d5711c9|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_nisan/malware/ee99f32c194d801ee56722dfeec9fac5fb93fc17546a70e6cf96e86b736cd00b|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/bc746ea8946ad2747bfe98a3da66022475d1dfef079e36ca097ed36e735cb032|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/cd9d21a197b0676c9c42a7978dfd3087b1f8fc03f59f24fccdf98563e587a747|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_haziran/malware/297e1be16ed9da14f0ee89f02c5d5689eefca281835f8de76a2c52f16aa0c4f4|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_haziran/malware/bb7a4b9ea4e66f4921168bdc6cef6bc01733d3e96d50a6f256d4270ce343d983|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_haziran/malware/a8d92f05dd51571aaad80c4d402e064679b826ac422a6c3d01bac4369ac9c1f5|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_nisan/malware/f649683451ac4c3a53015b9d3a5094c2e98fd182d7fa623f8f5bdb22d03eadae|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/66e5b6a93845525643938c525380a4f82972ead6c57f83c178416201ebd8b906|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/11c2691d210030f62f0d111d33598172a7dab33730ed5290e5a7df1b17debad4|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mart/malware/82e5acfcc66397008e0182c1bf30e8964b9e7d36cac424afbf03b23861ea66a4|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/7f2a6e179d727cef2075928b73cbae4372cf339d843ae736bc383bdaa580c032|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_şubat/malware/c6ce5a6ebd6da8acb326b5eef59d2a086ad82d3ff51a319f8f6ed96a881e32e2|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_ocak/malware/63a7ec2fe4fe4405bb25d8ee8c1c12ecb515ea5cc6a77e26cde424626966327c|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/68622260da9047f9a5be9eb34626857ca56a3979383ede01ea3d42c9571b2603|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/f04f48b6006bce222d8a17065459a71d8e19d8a36a821b698b9c1e66aff26e6c|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/d9b4215f118a22d7ce610c93d49796c66e6d7a2177149b667d2fa0049f1d09dd|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_mayis/malware/fba80bab650f9a5945bb51c503c906ce963e023840488beaff2e33c3a7791465|1
/home/lambda1/Desktop/Neural-Engine/dataset/2020_nisan/malware/cfd1bf47e7ee631839816cccfe0942753761b2da83f7dd87b2f414603bbf4ba2|1
➜  mw_dataset
➜  mw_dataset cat train_files.txt | wc -l
   51832
➜  mw_dataset
```

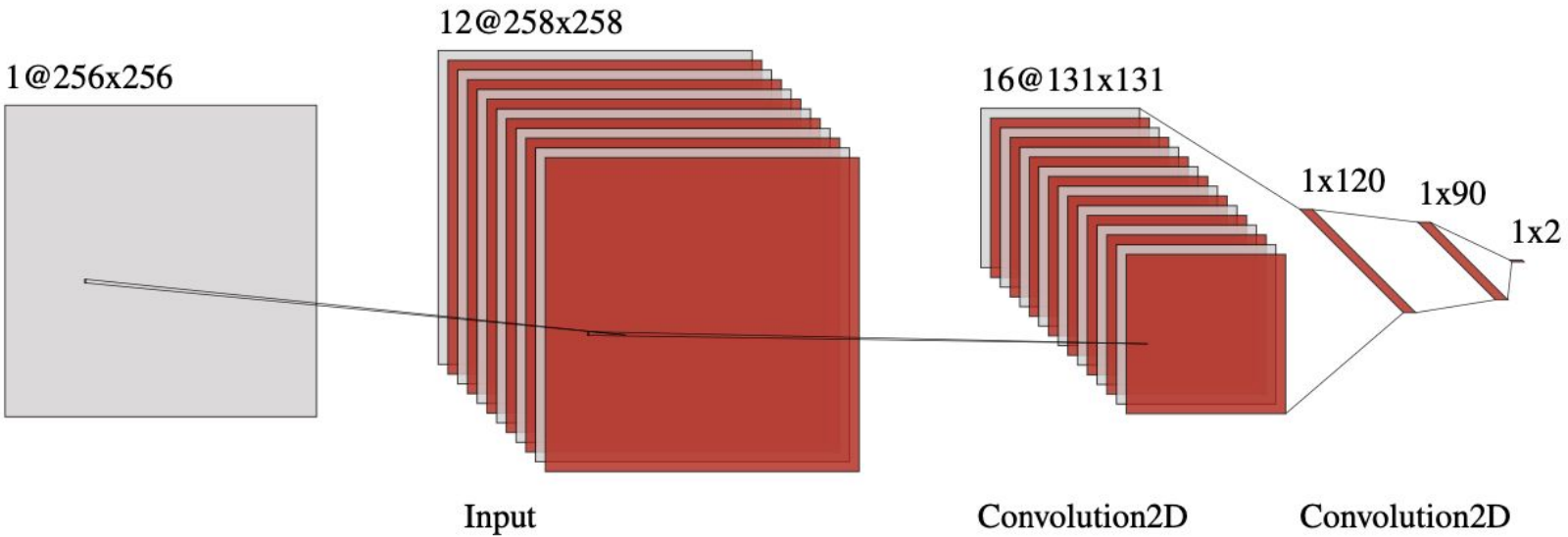
# Benign Files

```
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/64bd7c8cced1d5b9abebfe0c714bdf78fcde8c86feb2708600c61e9fa157eb34|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/a05b081b3cb44619d61d6063969fcc2fcfad0d59a3e9389ac44997b77149a517|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/97fc0e218eca2b437dea442b94f079119a7edc6515717b7941d8656b488dcc7d|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/d66de7b4077485dc215cea2cc35123da36ebe58ff2bc9dbdd5cc7bfbf9bccc7a|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/72fb8bbc8c0e5fa1e4eeb676d819641f2bb988d841a14320ae423111755aa63a|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/6e099a7d4841a4c2730462a331df1c3cbb108874954f8fa43fc1775df7afeb5e|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/6b94c93e00546c332618321138e46412546a942c9763850dee7998bd7c23c041|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/e18474ac9ecdf400829862b0d0e94c5782c76a8adec648b5bf00c315289893a9|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/9907bd2cc2e6933ab877fd693005344632f5f07c45b8ad33477585e754cd809b|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/255962d6c3423215dd00d93545cc6202555c9ee176e2d1a5c3dd2860176fd459|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/fdb665f883b928cc6a86804cca39033960b0318050f4ec987d3079f92e469ec8|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/ea69dc840e762c790da99aec60db566027a3928e403a7f82c3a19480e7f47309|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/9e34501109d23d17ff3836bbfe6e50f7314848dc44fc13ac2b0869077b36d402|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/e2dc15bfab264fc92d35b423606dfb762e43ea7459f252737d3cc862c99a0b66|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/e3cfbd82117066f2ae2aca0f1e88698b3dcb50fb4c1b00487e8911309d9300a0|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/42e1a138e8f928a8bf1d2edff2259b564b311da69e3716737c12199b351ddac8|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/4b1a847399d02b3fdf09f3cb39f8a5c820f1f81f02c1661de26e8d8ffa17b531|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/9637ca2316b97a66a345133bfdcbe5b600db3fd0c227fde7d24260cd20e1a567|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/974d756972a907b597b94ee6574d8e960d2340d97907e7f5e412692bfe7eaf6a|0
/home/lambda1/Desktop/Neural-Engine/dataset/2020_temmuz/clean/bfc66c4857eca5bac52d7f73773a42f48c95f263013b15949fe10681932d021d|0
➜  mw_dataset cat test_files.txt | grep "|0" | wc -l
    7420
➜  mw_dataset
➜  mw_dataset
➜  mw_dataset cat train_files.txt | grep "|0" | wc -l
   20733
➜  mw_dataset
```

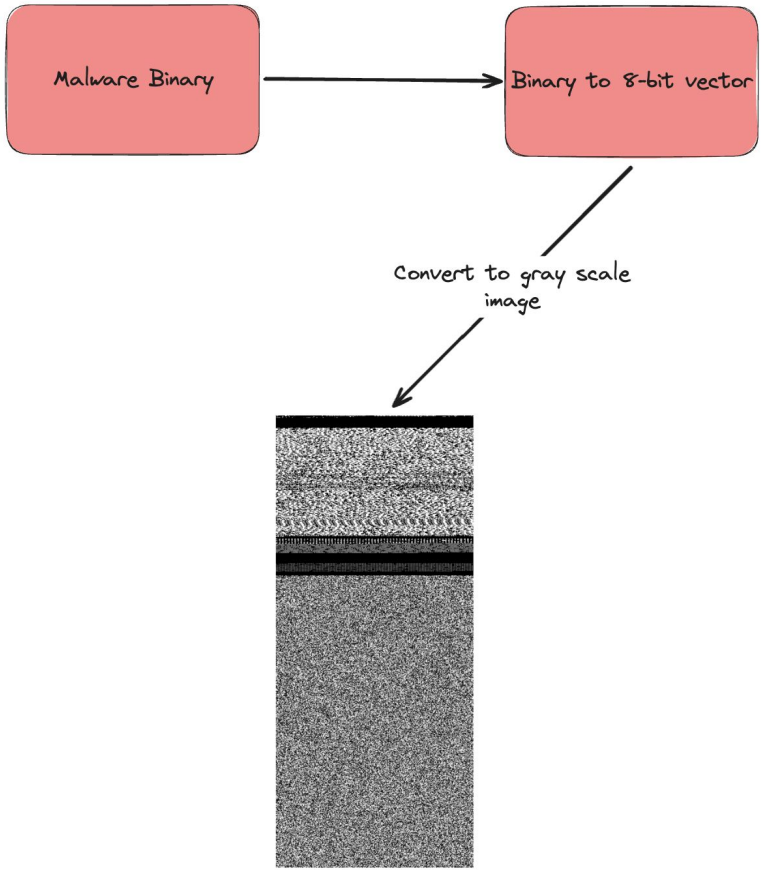
# CNN Model

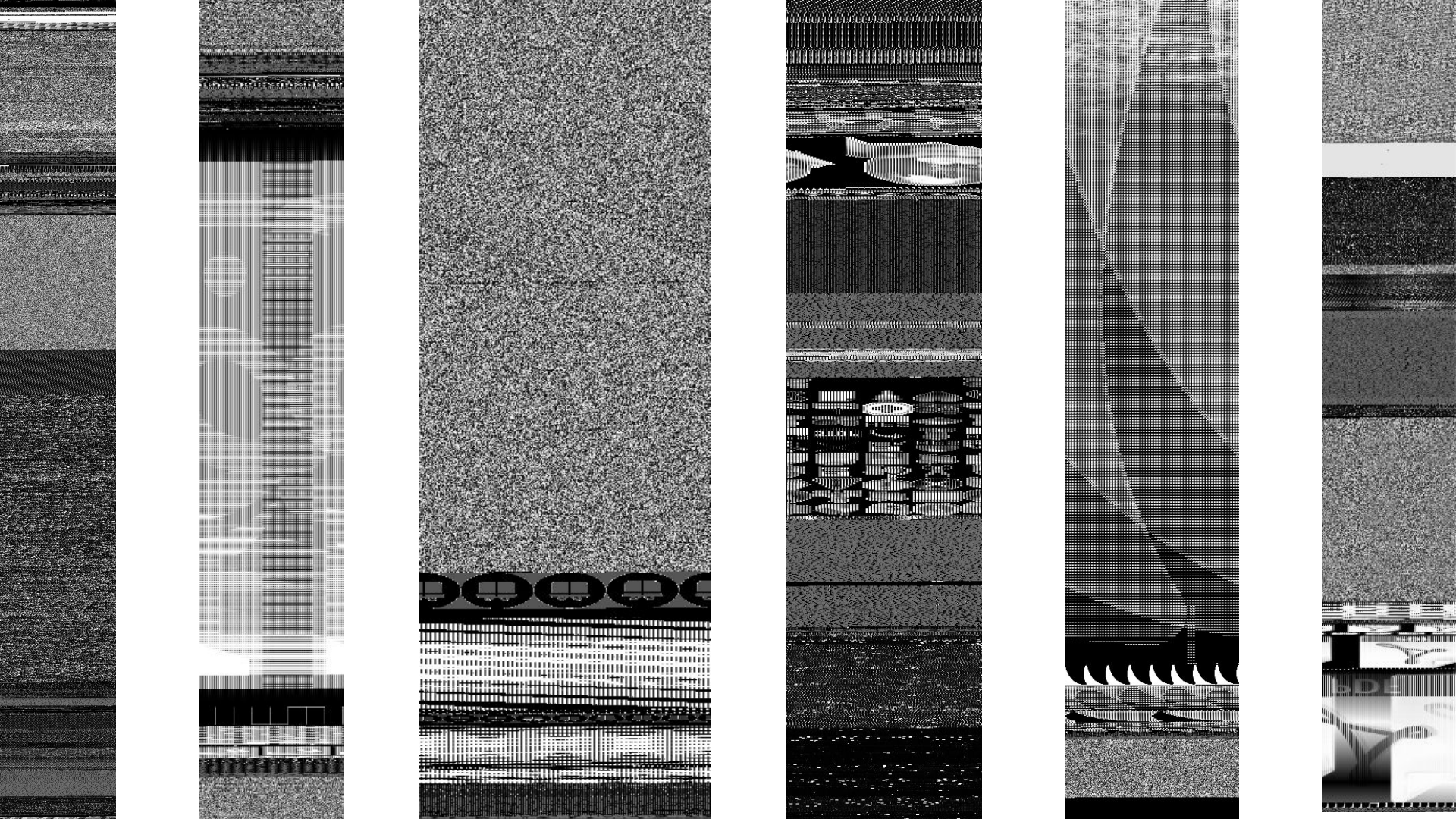
```
class CNNMalware_Model1(nn.Module):
    def __init__(self, image_dim=32, num_of_classes=20):
        super().__init__()

        self.image_dim = image_dim
        self.num_of_classes = num_of_classes

        self.conv1_out_channel = 12
        self.conv1_kernel_size = 3

        self.conv2_out_channel = 16
        self.conv2_kernel_size = 3

        self.linear1_out_features = 120
        self.linear2_out_features = 90

        self.conv1 = nn.Conv2d(1, self.conv1_out_channel, self.conv1_kernel_size, stride=1,
                               padding=(2, 2))

        self.conv2 = nn.Conv2d(self.conv1_out_channel, self.conv2_out_channel, self.conv2_kernel_size,
stride=1,                      padding=(2, 2))

        self.temp = int((((self.image_dim + 2) / 2) + 2) / 2)

        self.fc1 = nn.Linear(self.temp * self.temp * self.conv2_out_channel, self.linear1_out_features)
        self.fc2 = nn.Linear(self.linear1_out_features, self.linear2_out_features)
        self.fc3 = nn.Linear(self.linear2_out_features, self.num_of_classes)

    def forward(self, X):
        X = F.relu(self.conv1(X))
        print(x.shape)
        X = F.max_pool2d(X, 2, 2)
        print(x.shape)
        X = F.relu(self.conv2(X))
        print(x.shape)
        X = F.max_pool2d(X, 2, 2)
        print(x.shape)
        X = X.view(-1, self.temp * self.temp * self.conv2_out_channel)
        print(x.shape)
        X = F.relu(self.fc1(X))
        print(x.shape)
        X = F.relu(self.fc2(X))
        print(x.shape)
        X = self.fc3(X)
        print(x.shape)
        return F.log_softmax(X, dim=1)
```

# CNN Based Malware Detection

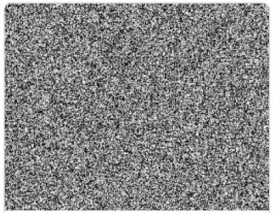


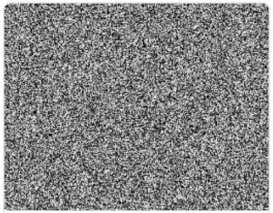
# Preprocessing: Binary to Image

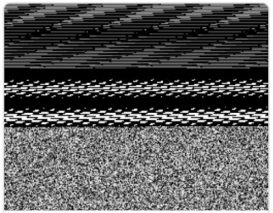


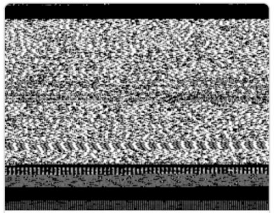


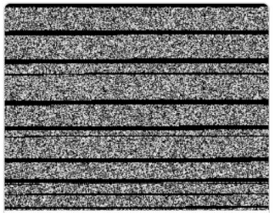
# Preprocessing: Binary to Image

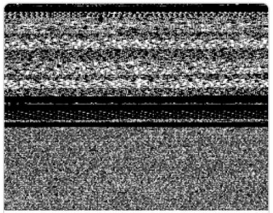

agenttesla_7.png

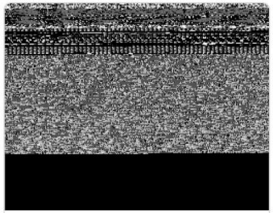

agenttesla_8.png

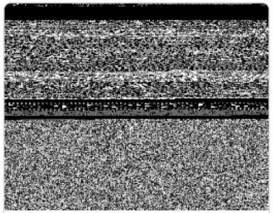

autoit_malware.png

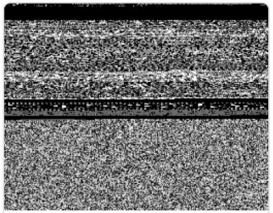

calypso_dropper.png

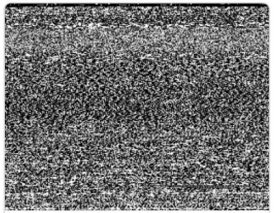

dridex_1.png

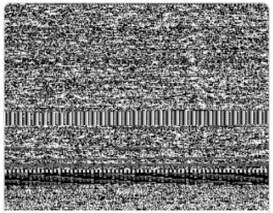

dridex.png

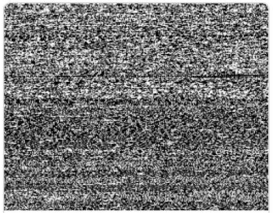

formbook.png

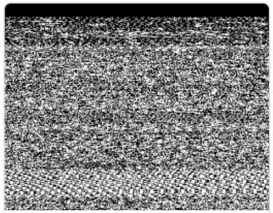

ghost_rat_1.png

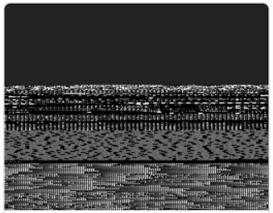

ghost_rat.png

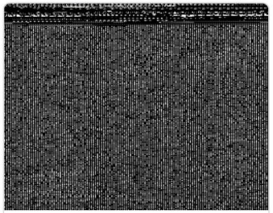

grandcab.png


hfs.png

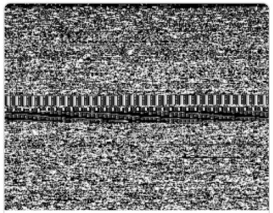

lokibot_1.png

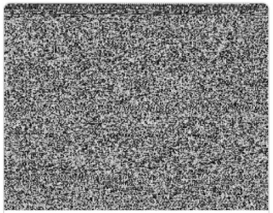

lokibot_2.png


lokibot_3.png


lokibot_5.png


lokibot_9.png


lokibot_13.png


lokibot_15.png

# Detection Scores

```
1 folder_scan(model, '/content/drive/My Drive/AV_RESEARCH/samples/malware_image')
```

```
/usr/local/lib/python3.6/dist-packages/torch/tensor.py:358: UserWarning: non-inplace resize is deprecated
  warnings.warn("non-inplace resize is deprecated")
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/mimidrv.png : CLEAN 0.011830936186015606
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/beRoot.png : CLEAN 0.09846494346857071
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/mimikatz.png : CLEAN 0.21892967820167542
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/mimilove.png : MALICIOUS 0.5499293208122253
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/mimilib.png : CLEAN 0.023451529443264008
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/rev-shell-32-msf.png : MALICIOUS 0.7929524779319763
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/rev-meter-32-msf.png : MALICIOUS 0.7725298404693604
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/rev-vnc-32-msf.png : MALICIOUS 0.6748470664024353
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/trickbot_1.png : MALICIOUS 0.9998303651809692
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/trickbot_2.png : CLEAN 0.31961530447006226
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/malware.png : CLEAN 0.08822183310985565
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/malware_nullpadding.png : CLEAN 0.10340498387813568
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/turkojan.png : MALICIOUS 0.9966475367546082
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/hfs.png : MALICIOUS 0.803017795085907
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/grandcab.png : MALICIOUS 0.9737184643745422
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/gandcrab.png : CLEAN 0.17672014236450195
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/trickbot.png : MALICIOUS 0.9998303651809692
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/azorult.png : CLEAN 0.10591787844896317
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/nanocore.png : MALICIOUS 0.6681360602378845
/content/drive/My Drive/AV_RESEARCH/samples/malware_image/ta505.png : MALICIOUS 0.782324492931366
```

# True-Negative

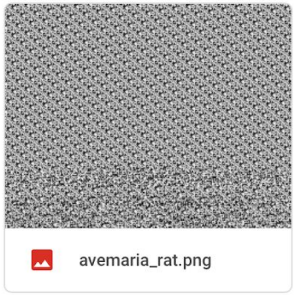
My Drive > AV_RESEARCH > samples > true_negatives_images

Files

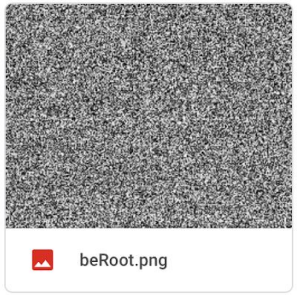
Name


artra_downloader_1.png

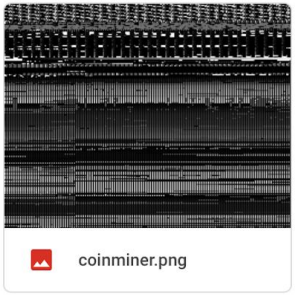

artra_downloader.png


avemaria_rat.png

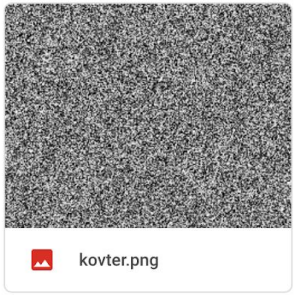

azorult.png


beRoot.png


bitter_rat.png


coinminer.png


gandcrab.png


kovter.png


kpotstealer.png


lokibot_4.png


lokibot_6.png

# Model Accuracy: %60

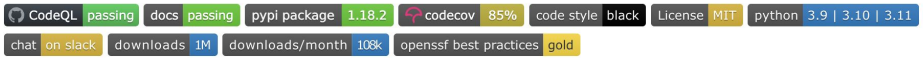
```
[ ]  1 len(glob.glob('/content/drive/My Drive/AV_RESEARCH/samples/malware_image/*.png'))

60
```

```
[ ]  1 len(glob.glob('/content/drive/My Drive/AV_RESEARCH/samples/true_negatives_images/*.png'))

40
```

# Designing to Adversarial Attacks

# ART: Adversarial Robustness Toolbox

## Adversarial Robustness Toolbox (ART) v1.18


中文README请按此处

LF AI & DATA
GRADUATE PROJECT

Adversarial Robustness Toolbox (ART) is a Python library for Machine Learning Security. ART is hosted by the Linux Foundation AI & Data Foundation (LF AI & Data). ART provides tools that enable developers and researchers to defend and evaluate Machine Learning models and applications against the adversarial threats of Evasion, Poisoning, Extraction, and Inference. ART supports all popular machine learning frameworks (TensorFlow, Keras, PyTorch, MXNet, scikit-learn, XGBoost, LightGBM, CatBoost, GPy, etc.), all data types (images, tables, audio, video, etc.) and machine learning tasks (classification, object detection, speech recognition, generation, certification, etc.).

https://github.com/Trusted-AI/adversarial-robustness-toolbox

# ART's Attack Types

## 1.2 Black-box

- **Square Attack** (Andriushchenko et al., 2020)
- **HopSkipJump Attack** (Chen et al., 2019)
- **Threshold Attack** (Vargas et al., 2019)
- **Pixel Attack** (Vargas et al., 2019, Su et al., 2019)
- **Simple Black-box Adversarial (SimBA)** (Guo et al., 2019)
- **Spatial Transformation** (Engstrom et al., 2017)
- **Query-efficient Black-box** (Ilyas et al., 2017)
- **Zeroth Order Optimisation (ZOO)** (Chen et al., 2017)
- **Decision-based/Boundary Attack** (Brendel et al., 2018)
- **Geometric Decision-based Attack (GeoDA)** (Rahmati et al., 2020)

# ART's Attack Types

## 1.2 Black-box

- **Square Attack** (Andriushchenko et al., 2(
- **HopSkipJump Attack** (Chen et al., 2019)
- **Threshold Attack** (Vargas et al., 2019)
- **Pixel Attack** (Vargas et al., 2019, Su et al
- **Simple Black-box Adversarial (SimBA)** (
- **Spatial Transformation** (Engstrom et al.,
- **Query-efficient Black-box** (Ilyas et al., 2
- **Zeroth Order Optimisation (ZOO)** (Cher
- **Decision-based/Boundary Attack** (Bren
- **Geometric Decision-based Attack** (Geo

## 1.1 White-box

- **Auto Projected Gradient Descent (Auto-PGD)** (Croce and Hein, 2020) `all/Numpy`

  Auto Projected Gradient Descent attacks classification and optimizes its attack strength by adapting the step size across iterations depending on the overall attack budget and progress of the optimisations. After adapting its steps size Auto-Attack restarts from the best example found so far.

- **Shadow Attack** (Ghiasi et al., 2020) `TensorFlow` , `PyTorch`

  Shadow Attack causes certifiably robust networks to misclassify an image and produce "spoofed" certificates of robustness by applying large but naturally looking perturbations.

- **Wasserstein Attack** (Wong et al., 2020) `all/Numpy`

  Wasserstein Attack generates adversarial examples with minimised Wasserstein distances and perturbations according to the content of the original images.

- **PE Malware Attacks** (Suciu et al., 2018, Demetrio et al., 2020, Demetrio et al., 2019) `TensorFlow`

  White-box attacks related to PE malware.

- **Imperceptible, Robust, and Targeted Adversarial Examples for Automatic Speech Recognition** (Qin et al., 2019) `TensorFlow` , `PyTorch`

  The attack extends the previous work of Carlini and Wagner (2018) to construct effective imperceptible audio adversarial examples.

- **Brendel & Bethge Attack** (Brendel et al., 2019) `all/Numpy`

  Brendel & Bethge attack is a powerful gradient-based adversarial attack that follows the adversarial boundary (the boundary between the space of adversarial and non-adversarial images as defined by the adversarial criterion) to find the minimum distance to the clean image.

- **Targeted Universal Adversarial Perturbations** (Hirano and Takemoto, 2019) `all/Numpy`

  This attack creates targeted universal adversarial perturbations combining iterative methods to generate untargeted examples and fast gradient sign method to create a targeted perturbation.

# Selected Attack Types

- ❏ FGSM Attack
- ❏ Adversarial Patch
- ❏ Boundary Attack
- ❏ Carlini and Wagner L_2 Attack
- ❏ Carlini and Wagner L_inf Attack
- ❏ DeepFool
- ❏ Elastic Net Attack
- ❏ Frame Saliency Attack
- ❏ HopSkipJump Attack
- ❏ Basic Iterative Method
- ❏ Projected Gradient Descent
- ❏ NewtonFool
- ❏ Jacobian Saliency Map Attack
- ❏ Shadow Attack
- ❏ Spatial Transformations Attack
- ❏ Square Attack
- ❏ Universal Perturbation Attack

# Adversarial Attack Dataset

| **Zararlı yazılım çeşitleri** | ***Sayı*** |
|---|---|
| Nanocore | 25 |
| Lokibot | 23 |
| AgentTesla | 8 |
| Trickbot | 4 |
| Mimikatz | 4 |
| Ryuk Ransomware | 3 |
| Metasploit | 3 |
| Gh0st Rat | 2 |
| Artra Downloader | 2 |
| Gandcrab Ransomware | 2 |
| Dridex | 2 |
| TA-505 | 1 |
| Qakbot | 1 |
| Pushdo | 1 |
| Pony Rat | 1 |
| Shade Ransomware | 1 |
| Turkojan | 1 |
| Zeus | 1 |
| AutoIt Malware | 1 |
| AveMaria Rat | 1 |
| Bitter Rat | 1 |
| Nanobot | 1 |
| Kovter | 1 |
| Kpot Stealer | 1 |
| Formbook | 1 |
| Azorult Stealer | 1 |
| Calypso Dropper | 1 |
| CoinMiner | 1 |
| beRoot | 1 |
| hfs | 1 |

# Implementation

```python
# Adversarial Patch
def attack_adverserial_patch(classifier, x, y):
    attack = art.attacks.evasion.AdversarialPatch(classifier)
    adv_patch = attack.generate(x=x, y=y)

    applied_patch = attack.apply_patch(x, 0.25, adv_patch[0])
    score = np.exp(classifier.predict(applied_patch))

    return score, applied_patch
```

```python
def attack_carliniL2method(classifier, x, y):
    attack = art.attacks.evasion.CarliniL2Method(classifier)
    adv_attack = attack.generate(x=x, y=y)

    score = np.exp(classifier.predict(adv_attack))

    return score, adv_attack
```

```python
def attack_boundary_attack(classifier, x, y):
    attack = art.attacks.evasion.BoundaryAttack(classifier)
    adv_attack = attack.generate(x=x, y=y)

    score = np.exp(classifier.predict(adv_attack))

    re, adv_attack
```

```python
def attack_deep_fool(classifier, x, y):
    attack = art.attacks.evasion.DeepFool(classifier)
    adv_attack = attack.generate(x=x, y=y)

    score = np.exp(classifier.predict(adv_attack))

    return score, adv_attack
```

```python
_fast_gradient_method(classifier, x, y):
 art.attacks.evasion.FastGradientMethod(classifier, eps=0.2)
ck = attack.generate(x=x)

np.exp(classifier.predict(adv_attack))

    return score, adv_attack
```

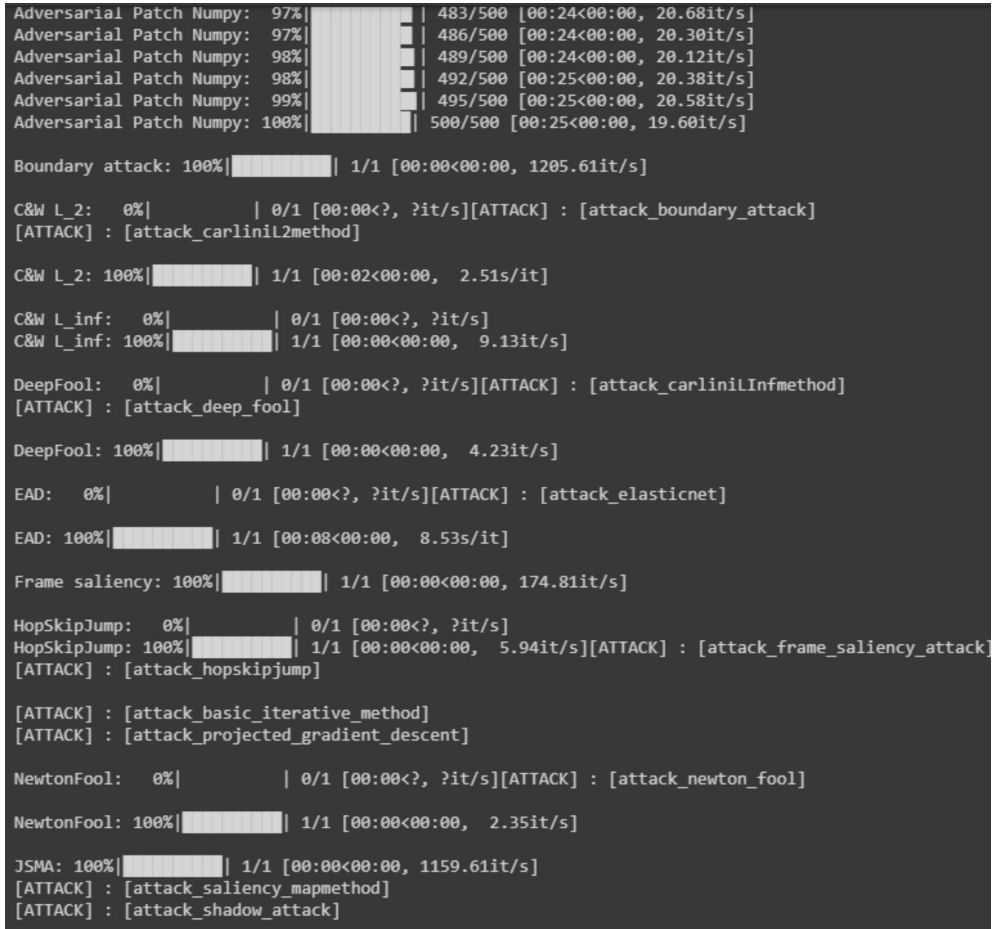
```python
def attack_frame_saliency_attack(classifier, x, y):
    attack = art.attacks.evasion.FrameSaliencyAttack(classifier=classifier,
                                                     attacker=art.attacks.evasion.FastGradientMethod(classifier),
                                                     method='iterative_saliency',
                                                     frame_index=1,
                                                     batch_size=1)
    adv_attack = attack.generate(x=x, y=y)

    score = np.exp(classifier.predict(adv_attack))

    return score, adv_attack
```

# Algorithm

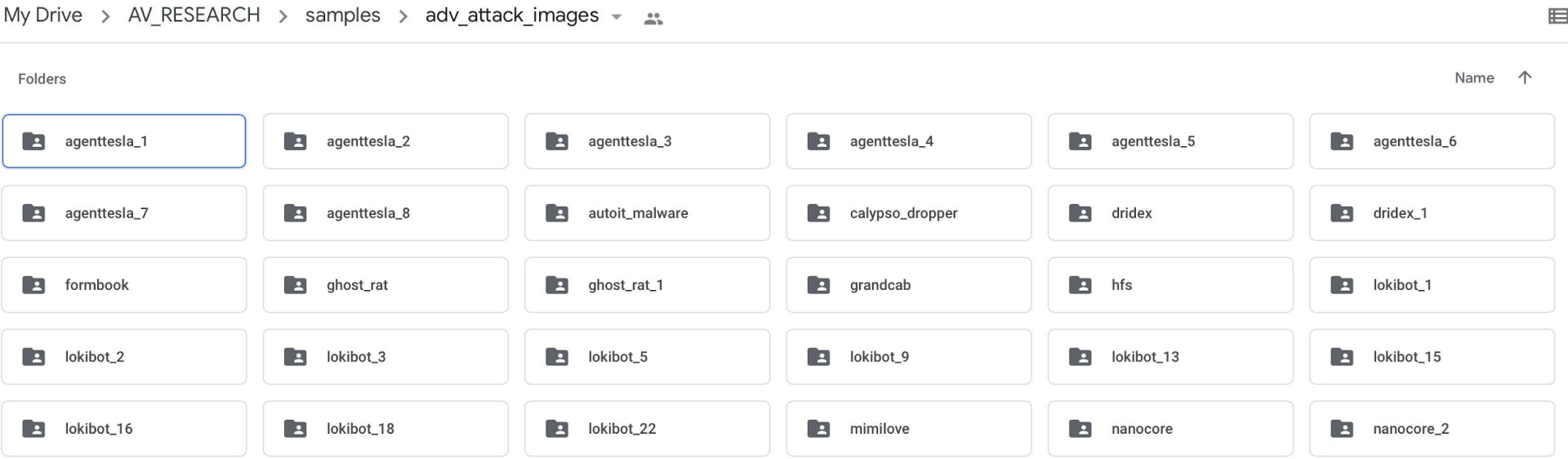
**Algorithm 1:** Çekişmeli atakların uygulanması

**Result:** Çekişmeli atak uygulanmış örnek
resim içeriğini dosyadan oku;
resim kanalını 1 yap;
resmi 256x256 boyutlandır;
modelden orijinal tespit skoru üret;
**for** *atak methodları* **do**
    atak oluştur;
    örnek üzerinde atağı uygula;
    modelden atak skorunu elde et;
    **if** *atak skoru* > $\alpha$ **then**
        başarılı atak;
    **else**
        başarısız atak;
    **end**
    atak skoru orijinal skoru karşılaştır;
**end**

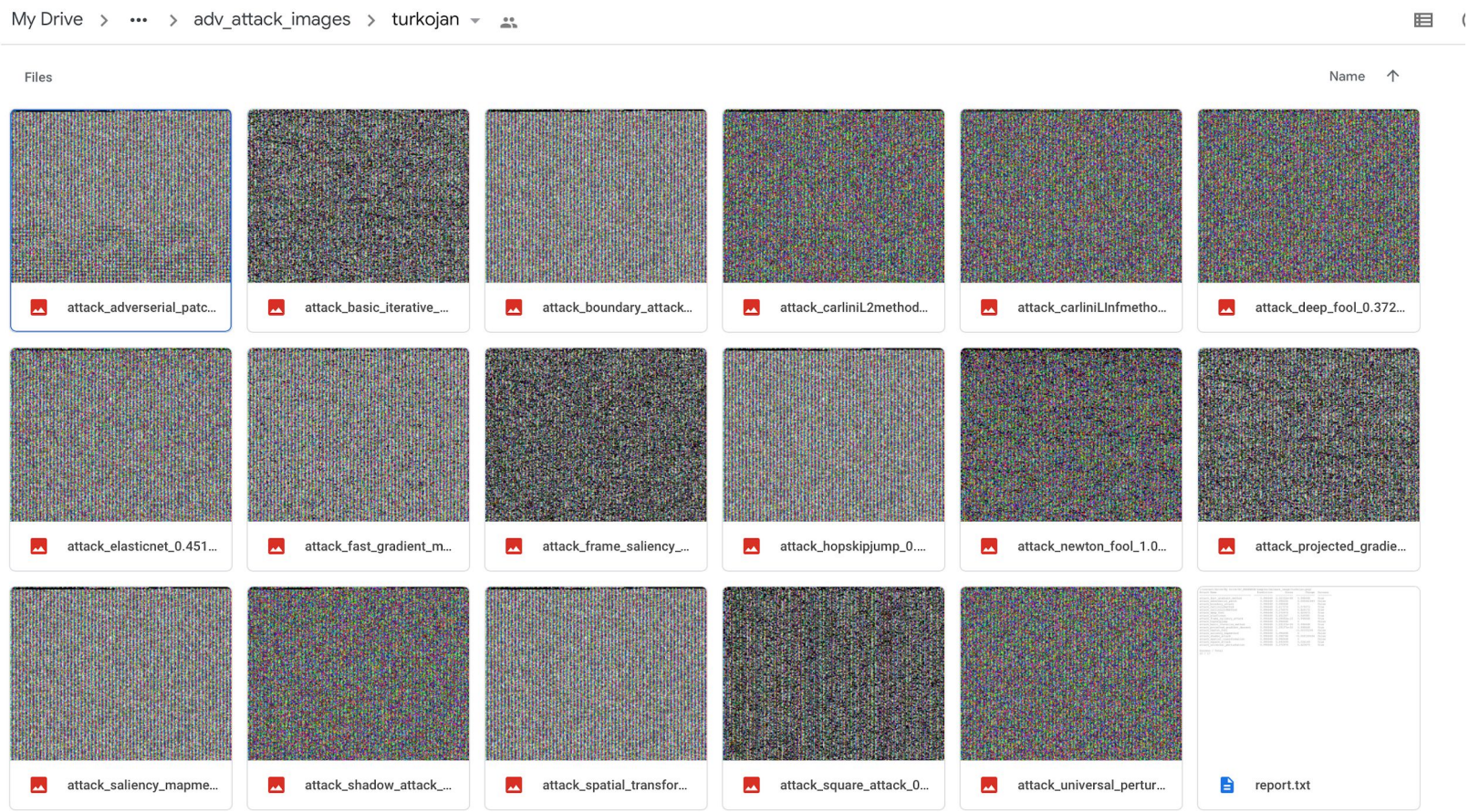
# Attack Simulation



# Adversarial-ed Samples



# Adversarial-ed Samples



# Reporting

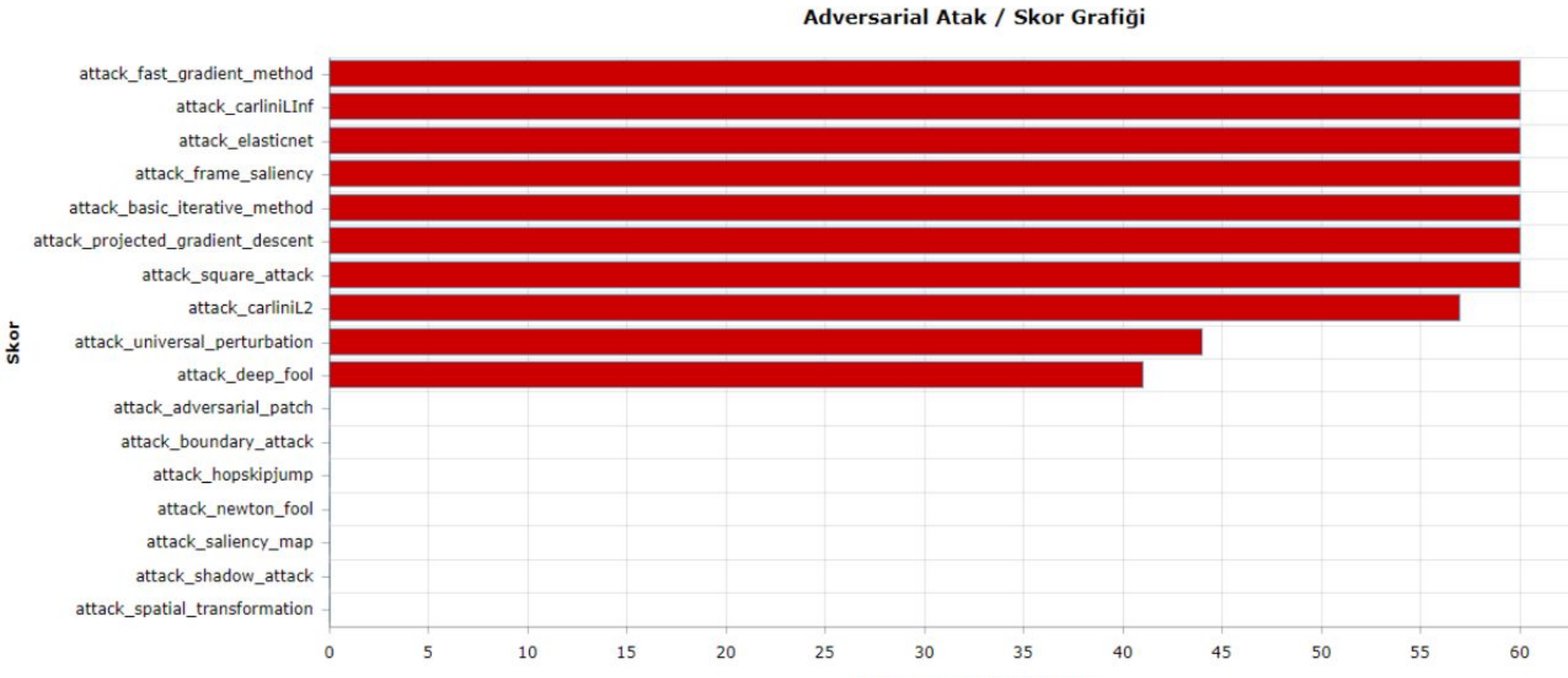
```
[/content/drive/My Drive/AV_RESEARCH/samples/malware_image/turkojan.png]
Attack Name                          Prediction        Score         Change  Success
-----------------------------------  ------------  -----------  ------------  ---------
attack_fast_gradient_method              0.996648  2.32332e-08    0.996648    True
attack_adverserial_patch                 0.996648  0.996206       0.000441849 False
attack_boundary_attack                   0.996648  0.996648       0           False
attack_carliniL2method                   0.996648  0.417274       0.579373    True
attack_carliniLInfmethod                 0.996648  0.176475       0.820172    True
attack_deep_fool                         0.996648  0.372974       0.623673    True
attack_elasticnet                        0.996648  0.451067       0.545581    True
attack_frame_saliency_attack             0.996648  6.28682e-10    0.996648    True
attack_hopskipjump                       0.996648  0.996648       0           False
attack_basic_iterative_method            0.996648  1.29157e-24    0.996648    True
attack_projected_gradient_descent        0.996648  1.29157e-24    0.996648    True
attack_newton_fool                       0.996648  1             -0.00335246  False
attack_saliency_mapmethod                0.996648  0.996648       0           False
attack_shadow_attack                     0.996648  0.996748      -0.000100434 False
attack_spatial_transformation            0.996648  0.996648       0           False
attack_square_attack                     0.996648  0.440499       0.556149    True
attack_universal_perturbation            0.996648  0.372974       0.623673    True

Success / Total
10 / 17
```

# Result

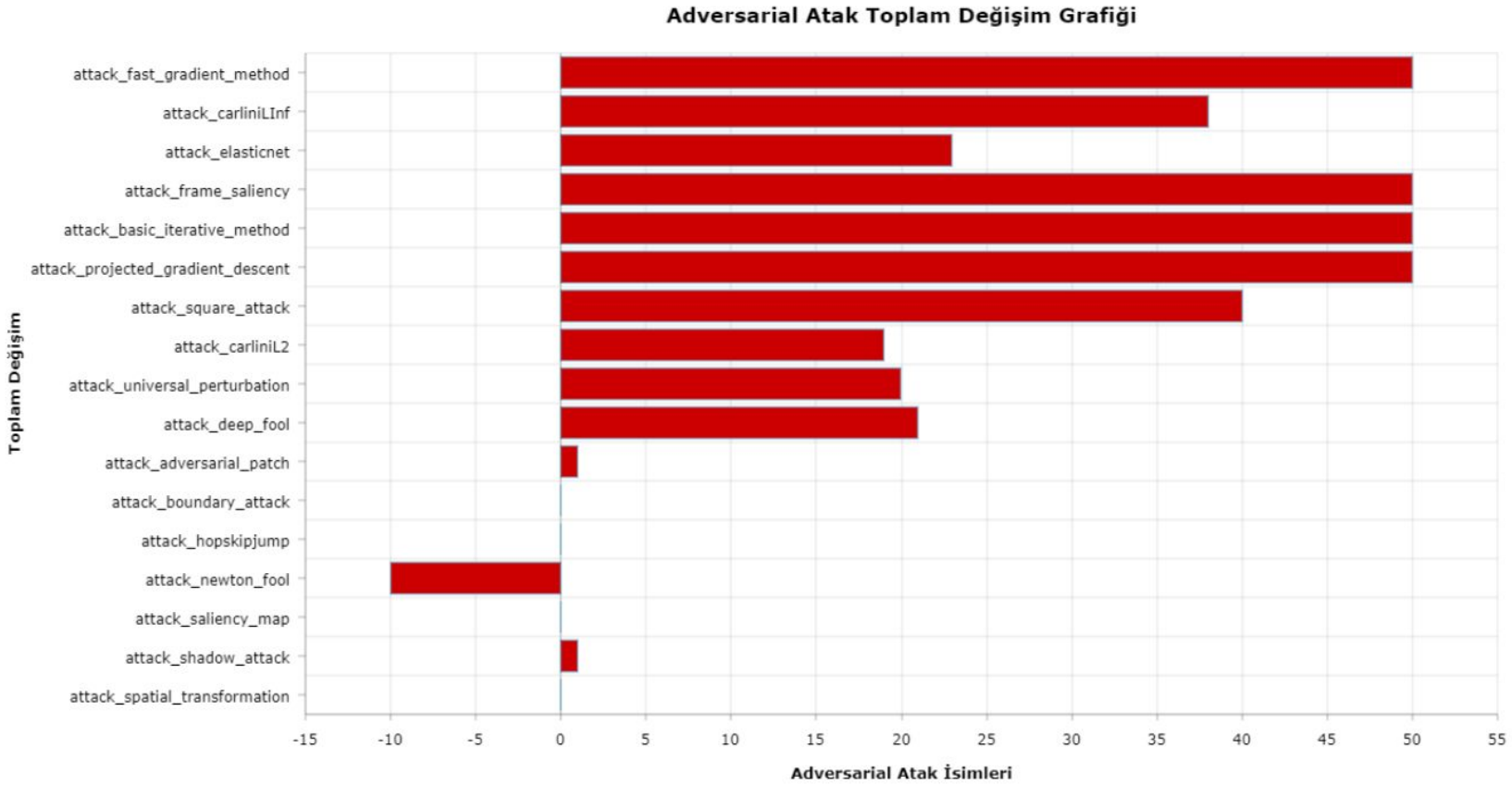
# Success

| **Başarılı olan çekişmeli ataklar** | ***Sayı*** |
|---|---|
| Fast Gradient (FGSM) Attack | 60 |
| Carlini and Wagner Linf Attack | 60 |
| Elastic Net Attack | 60 |
| Frame Saliency Attack | 60 |
| Basic Iterative Method | 60 |
| Projected Gradient Descent | 60 |
| Square Attack | 60 |
| Carlini and Wagner L2 Attack | 57 |
| Universal Perturbation Attack | 43 |
| DeepFool | 41 |

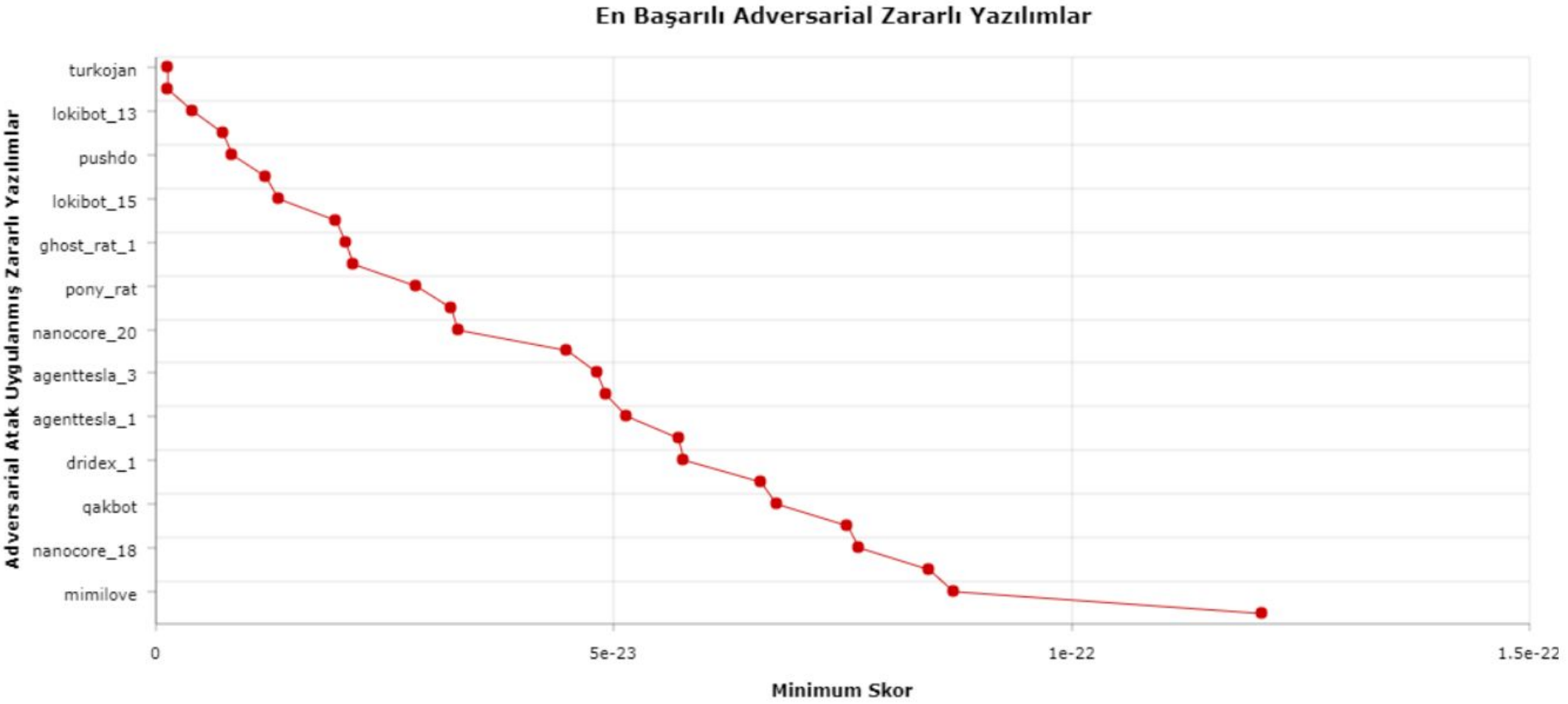
# Success Graph



# Success/Unsuccess Attacks



# Top Adversarial-ed Sample




AI MODELS
38
ADVERSARIAL ATTACKS
imgflip.com

# Research Paper

## Derin Öğrenme Tabanlı Antivirüs Modellerinin Açık Kaynak Kodlu Çekişmeli Atak Kütüphaneleri Kullanılarak Atlatılması

Yıl 2021, Cilt 3, Sayı 1, 66 - 71, 30.04.2021

Fatih ERDOĞAN Mert Can ALICI

https://doi.org/10.47769/izufbed.879611


### Öz

Gelişen teknoloji ve internetin gelişmesi ile birlikte; insanların bu gelişen teknolojiyi kullanım oranının artması, kullanıcıları ve sistemlerini siber saldırganlar tarafından hedef haline getirmektedir. Siber saldırganlar tarafından kullanılan en etkili atak yöntemlerinden biri zararlı yazılımlardır. Zararlı yazılımlar aracılığı ile kişi ve kurumlara ait sistemler ele geçirilebilir, farklı enfeksiyonlara sebep olunarak daha büyük çaplı ataklar gerçekleştirilebilir. Bu saldırılar karşısında siber güvenlik firmaları tarafından geliştirilen yapay zeka tabanlı son jenerasyon antivirüs yazılımlarının yüzde yüz başarılı olamadığı görülmektedir. Gerçekleştirilen ilgili çalışmada; zararlı yazılım ve zararlı ofansif araçlara uygulanacak çekişmeli(adversarial) ataklar sonucunda üretilecek çekişmeli örnekler sayesinde, geliştirilen yapay zeka tabanlı son jenerasyon güvenlik ürünlerinin başarılı bir şekilde atlatılabildiği gözlemlenmiştir.



### Anahtar Kelimeler

Derin Öğrenme, Çekişmeli Atak, Antivirüs Atlatma, Zararlı Yazılım Tespiti


{UAKK 2020}
Ulusal Açık Kaynak ve Özgür Yazılım Konferansı
18-19 ARALIK 2020

#dfist24

# THANK YOU!

devfest
Istanbul 2024


**Fatih ERDOĞAN**

Sr. Blue Team Engineer, Picus Security