Reinforcement Learning in Adversarial Attack (RLAT)

Khanh Le Tran Quoc Khoa Tran Nhat Linh Ly Nguyen Thuy

University of Information Technology

7th June 2024

[Link slide with animation]

[Link github]

Motivation

- Machine learning models, especially DNNs, have power in classification tasks.
- However, they lack defenses:

  A small change in the input that is hard recognized by the naked eye, then successfully fool these models.

Motivation

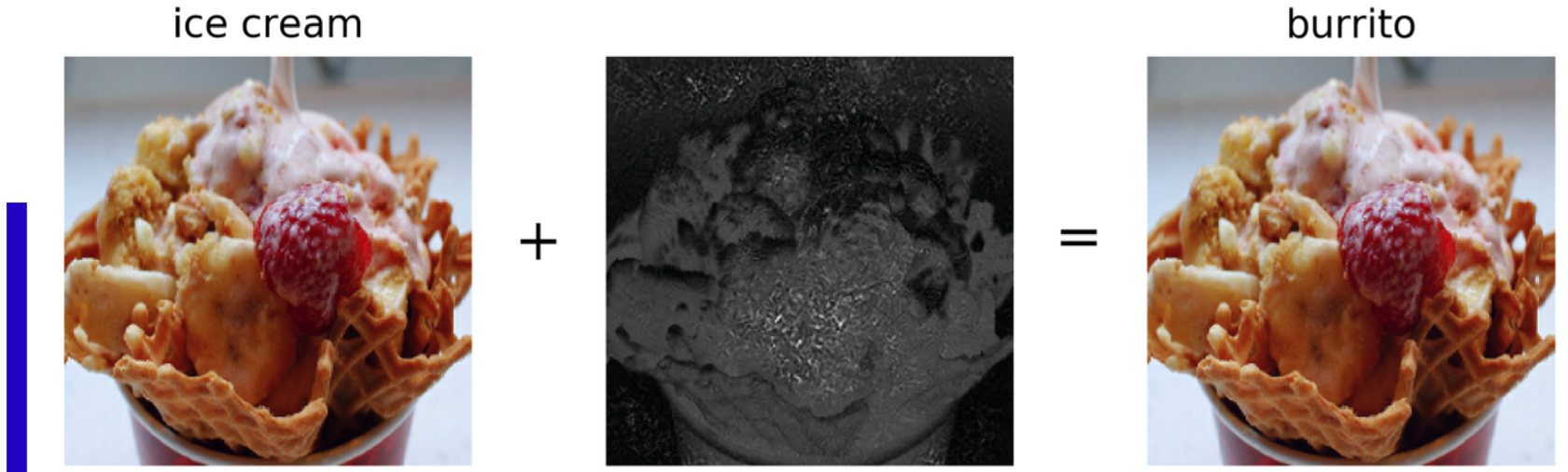
- Machine learning models, especially DNNs, have power in classification tasks.
- However, they lack defenses:

  A small change in the input that is hard recognized by the naked eye, then successfully fool these models.

→ Study in Adversarial Attack is important to the reliability of ML model.

Motivation

- Looking at this ice-cream image:
  - Only by changing a small amount of input value, a new image, which is no different from the old one, is created.
  - However, the new one successfully fools the classfication model.



Motivation

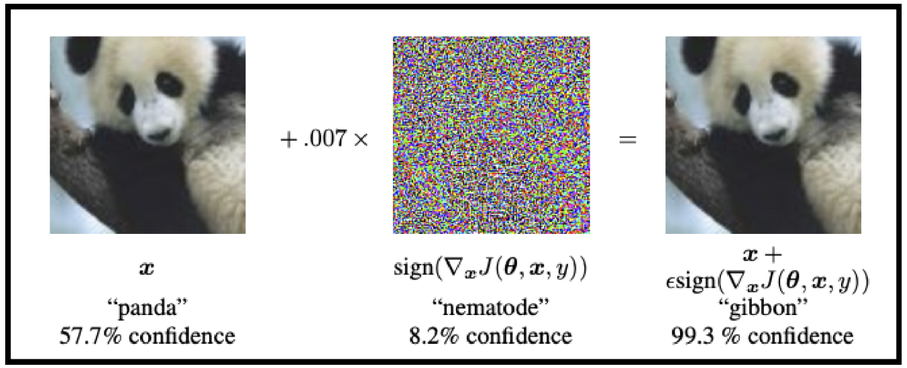
- Adversarial attack:
  Disruptions to machine learning models that successfully fools the models.
- 2 kinds of adversarial attack:
  - White-box attack
  - Black-box attack

Motivation

- White-box attack:

  The attack that allows to access the model details, such as: architecture, gradient, …

White-box Attack



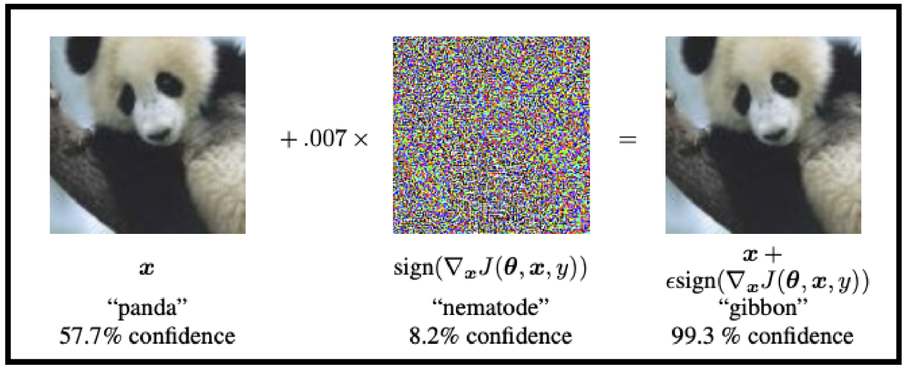
Impractical

Motivation

- White-box attack:

  The attack that allows to access the model details, such as: architecture, gradient, …

White-box Attack

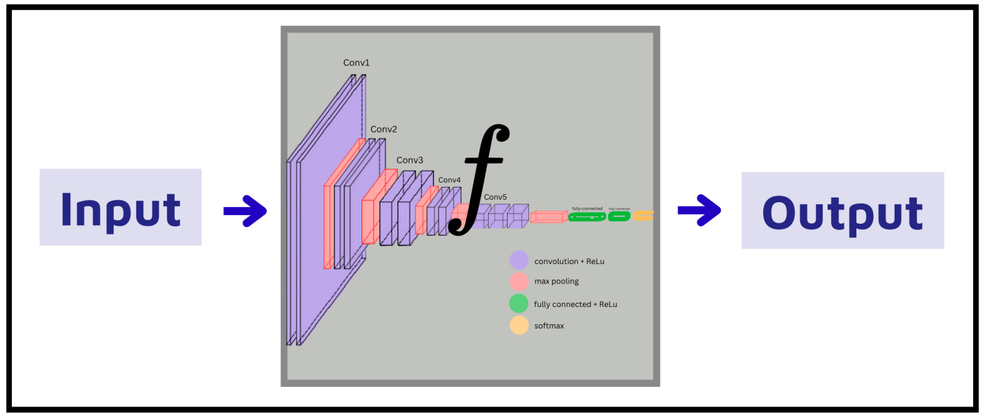


Impractical

- Black-box attack:

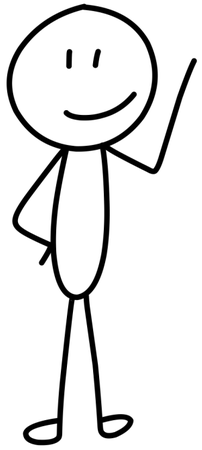
  The attack that only uses input & output

Black-box attack



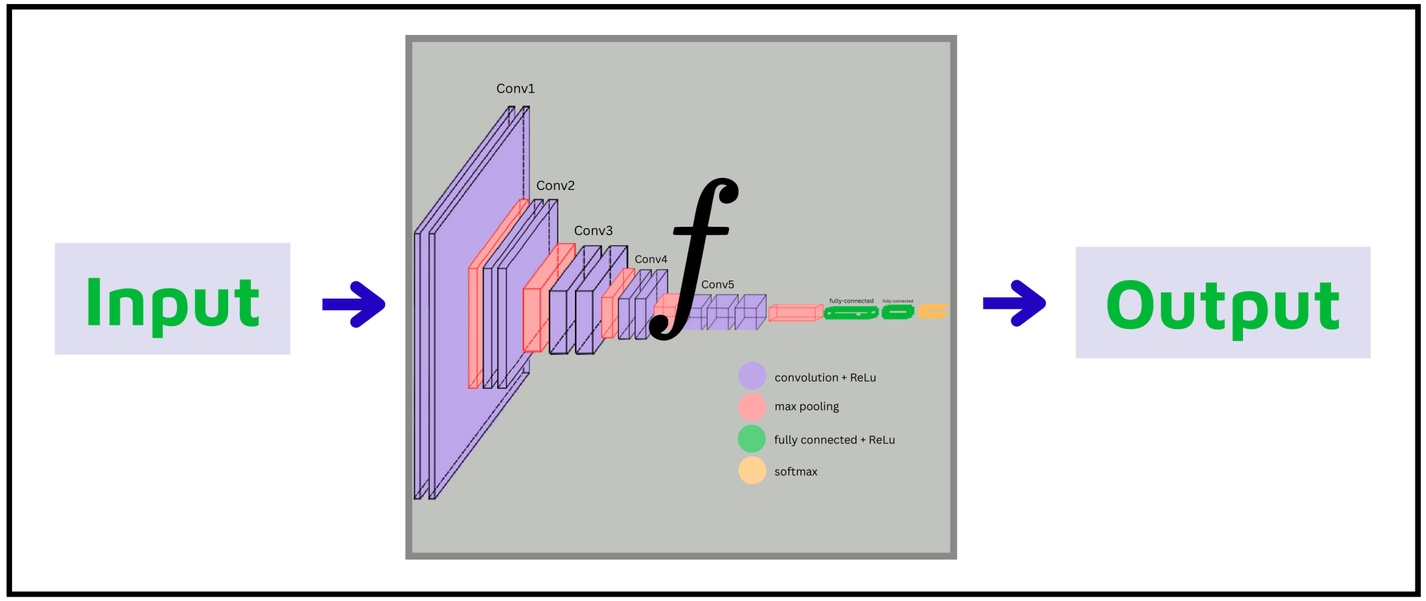
Practical

Purpose

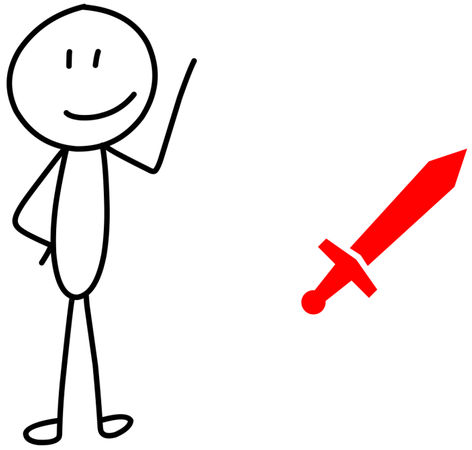


Reinforcement Learning Agent

BLACK-BOX ATTACK

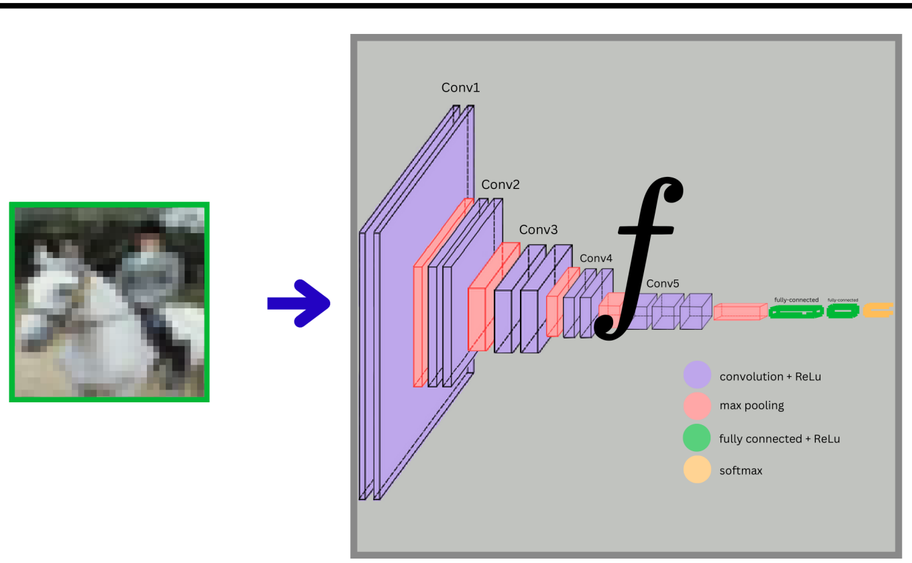


Purpose



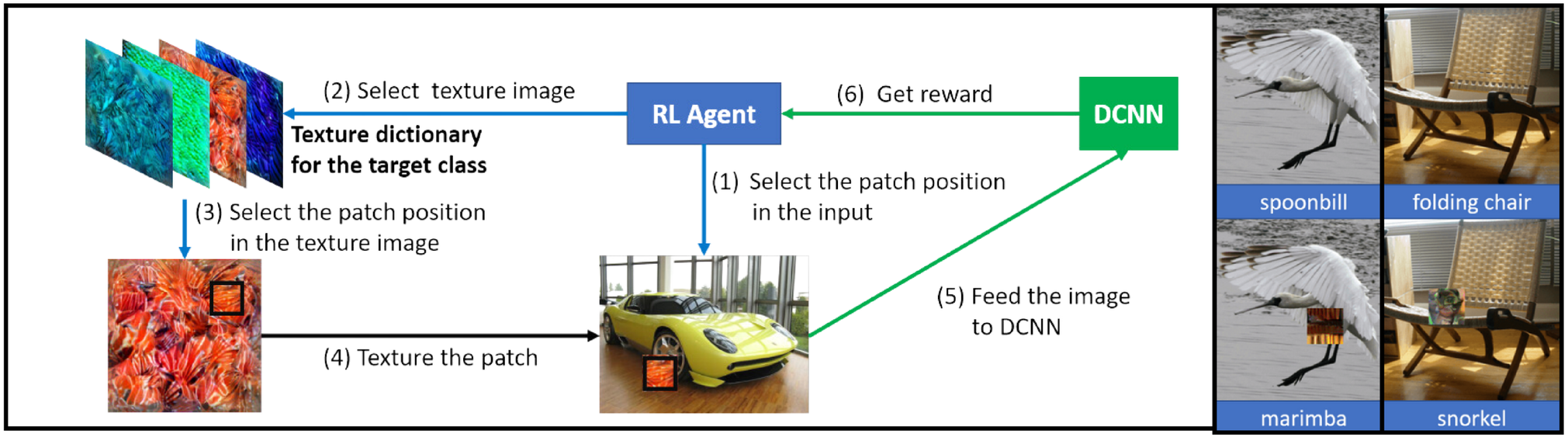
Reinforcement Learning Agent

BLACK-BOX ATTACK



Insight

- Image has some regions that is sensitive with the output of classification model.

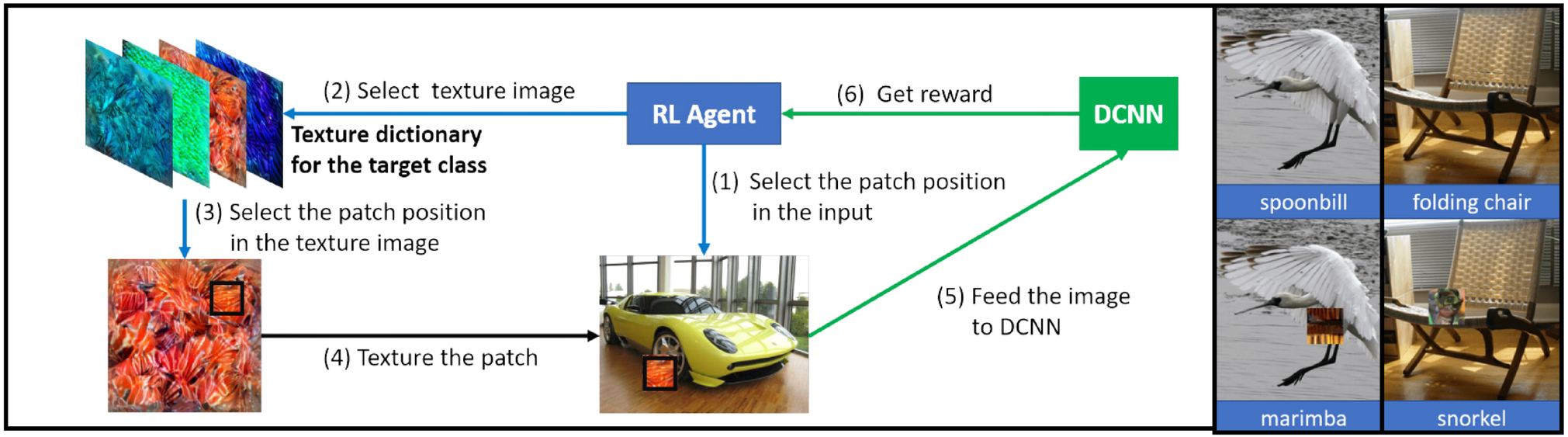


PatchAttack

Insight

- Image has some regions that is sensitive with the output of classification model.

  Ex: Patch-Attack using RL agent find the image that sensitive to Classification model and is pasted to the input image.



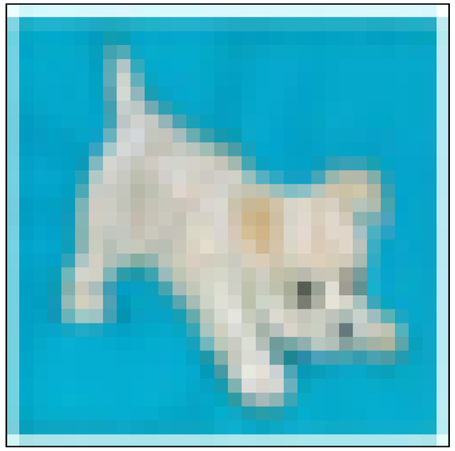
PatchAttack

Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.

Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.
- **Attack** process:

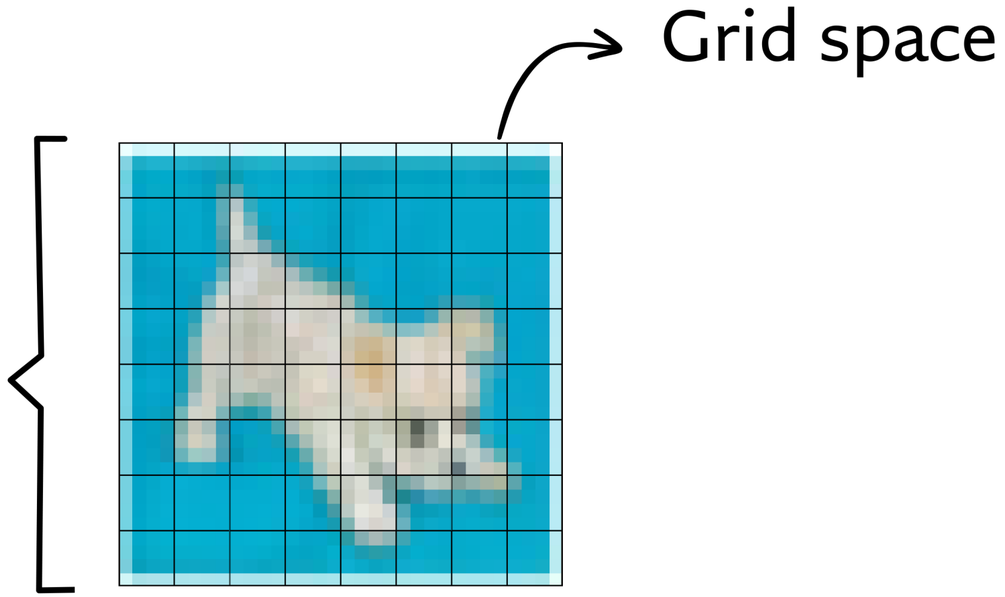


Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.
- **Attack** process:

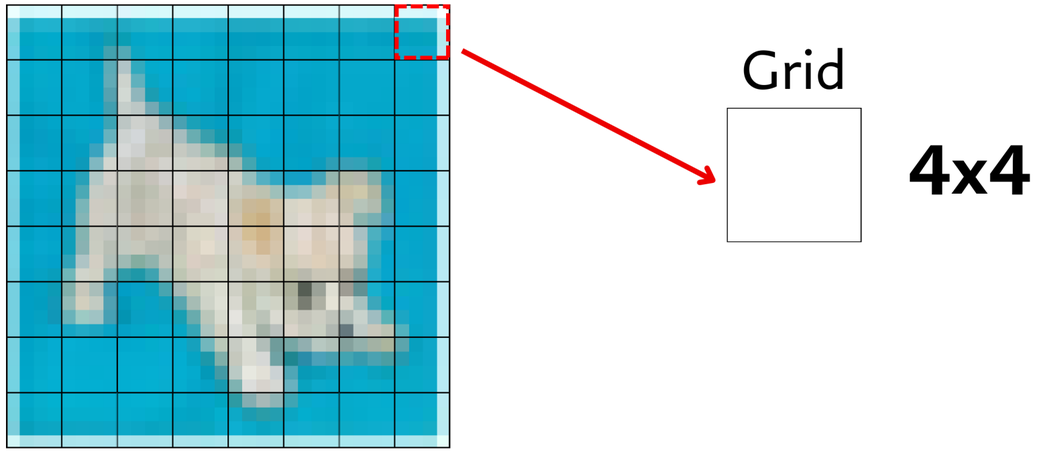
image_size = 32x32
grid_size = 4x4

grid_space_size = 8x8



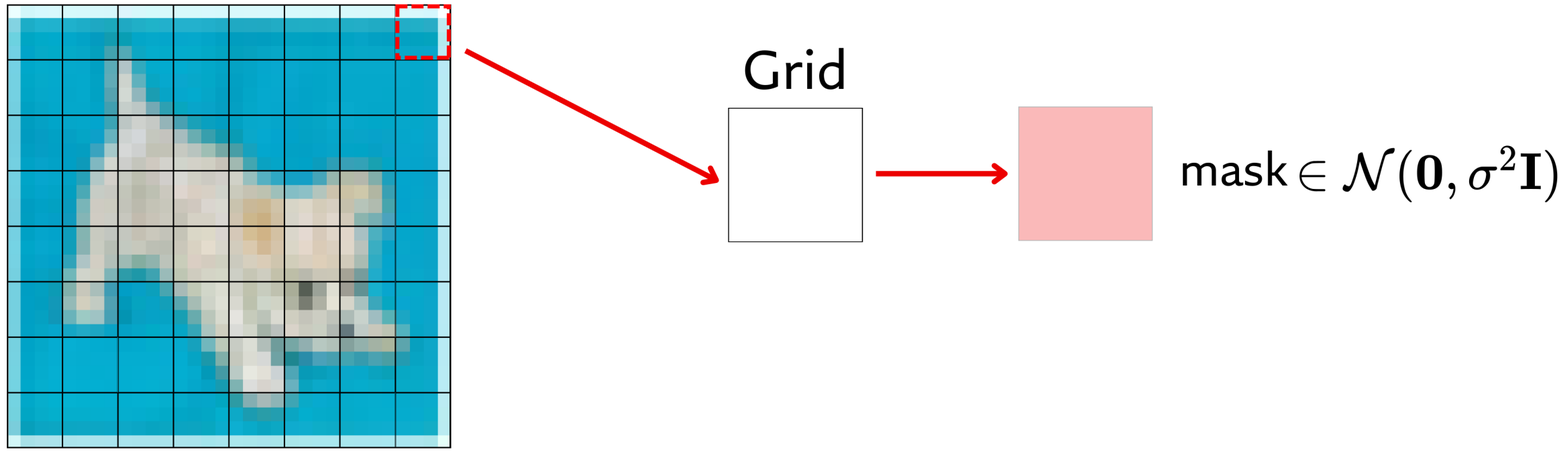
Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.
- **Attack** process:



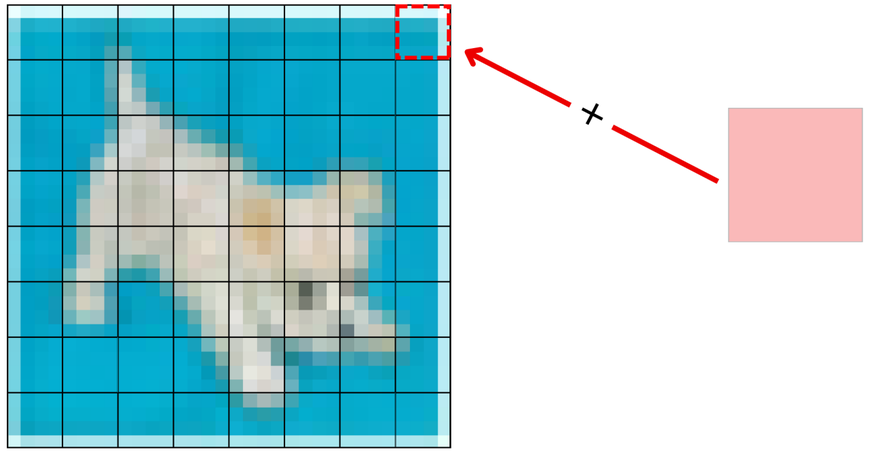
Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.
- **Attack** process:



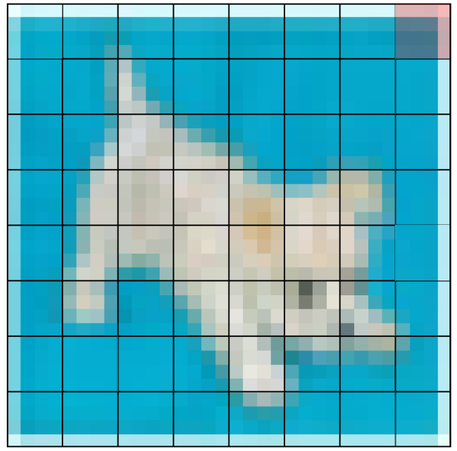
Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.
- **Attack** process:



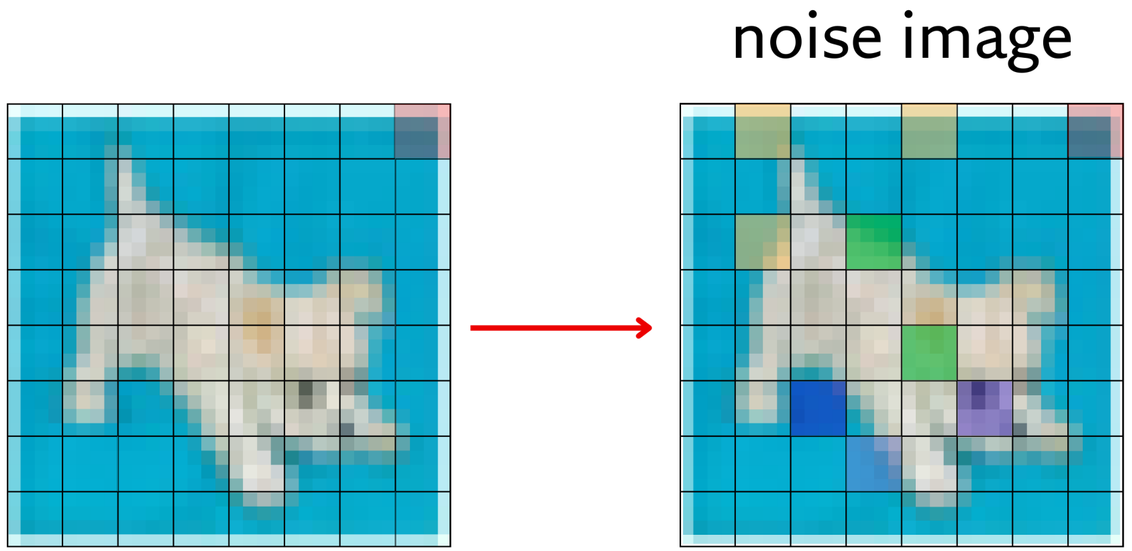
Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.
- **Attack** process:



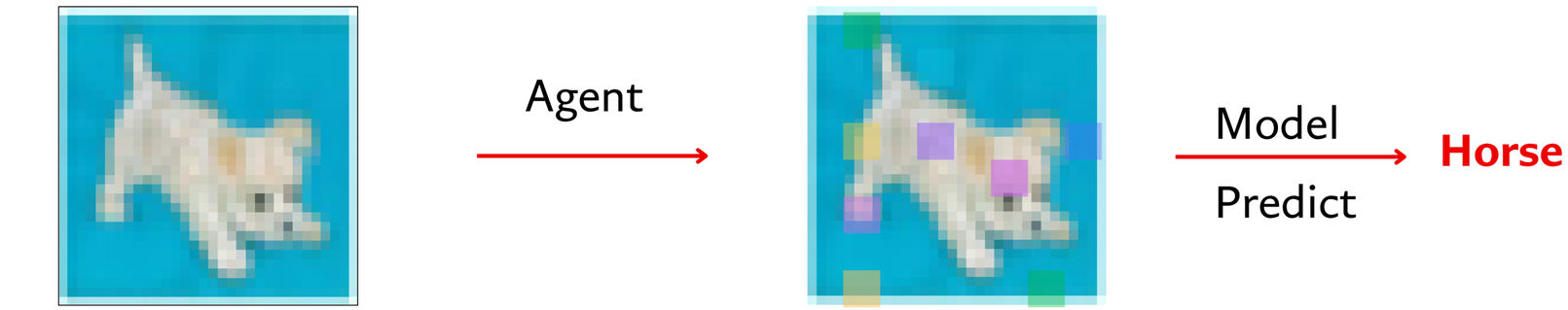
Insight

- Image has some region that is sensitive with the classification model.
- Improve a RL agent has policy to **find a region** and **apply adversarial noise** to that specific area.
- **Attack** process:



Problem Statement

- Input: Image, Ground truth (Model's prediction)
- Output: The agent's policy can identify the grid where noise should be added

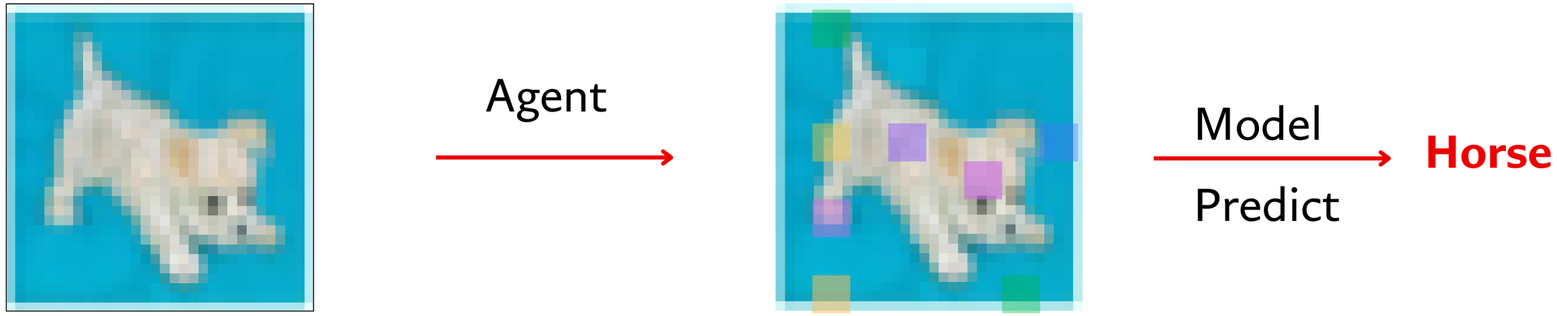


Ground truth: **Dog**

Adversarial sample

Problem Statement

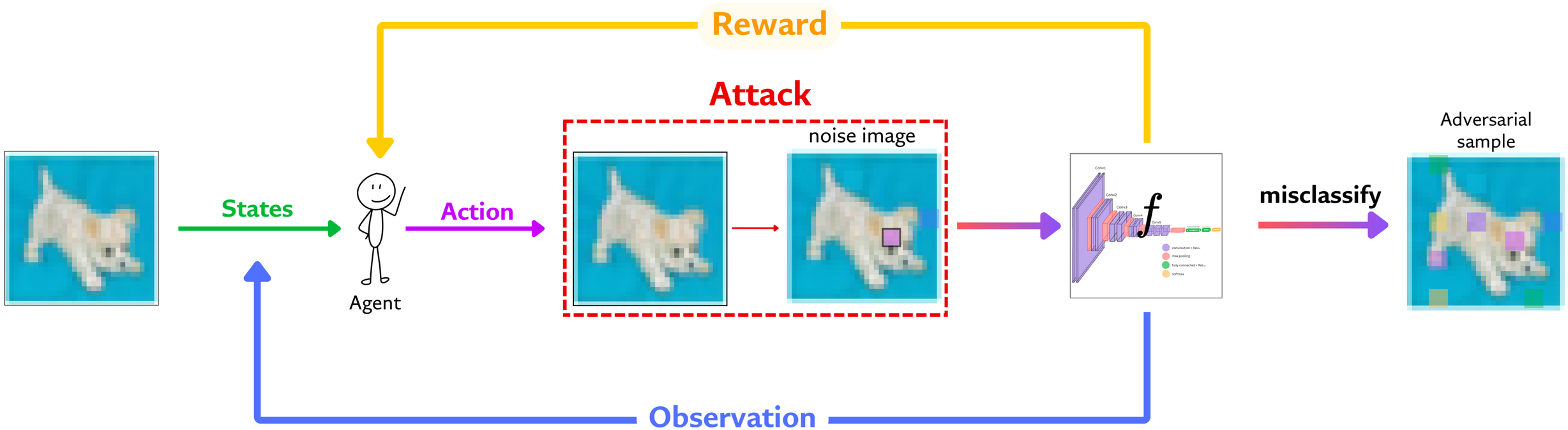
- Input: Image, Ground truth (Model's prediction)
- Output: The agent's policy can identify the grid where noise should be added
- Aims:
  1. Make a huge change in output of the model in Grouth-Truth class.
  2. The noise image has less differences from the original image.



Ground truth: **Dog**

Adversarial sample

Agent Training Pipeline



Modeling

Denote:

$\mathbf{x}$: original image

$\mathbf{x}_i$: noise image at $i\text{th}$ step

f : classifier

$\mathbf{E}$: feature extraction

P_{GT} : GT Classfication Probability of $\mathbf{x}$

P_{GT_i} : GT Classification Probability of $\mathbf{x}_i$

$\boldsymbol{s}_i$: sensitive analysis vector of $\mathbf{x}_i$

$m \in \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$: random vector mask with the same shape with grid

Modeling

Denote:

$\mathbf{x}$: original image

$\mathbf{x}_i$: noise image at $i\text{th}$ step

f : classifier

$\mathbf{E}$: feature extraction

P_{GT} : GT Classfication Probability of $\mathbf{x}$

P_{GT_i} : GT Classification Probability of $\mathbf{x}_i$

s_i : sensitive analysis vector of $\mathbf{x}_i$

$m \in \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$: random vector mask with the same shape with grid

- Agent definition: finding region from original image to add the noise mask
- State: $S_i = \{\mathbf{E}(\mathbf{x}_i), \mathbf{s}_i\}$ is state of $i\text{th}$ step
- Action: a_i, adding to specific grid of $\mathbf{x}_i$
- Reward: $R_i = -||\mathbf{x} - \mathbf{x}_i||_2 + |P_{GT} - P_{GT_i}|$

Methodology

Q - learning

Q-learning cores

Bellman equation:

$$Q^*(S,a) = \mathbb{E}_{S'}[r + \gamma max_{a'} Q(S',a')^* | S,a]$$

Target:

$$r + \gamma max_{a'} Q(S',a',\mathbf{w})$$

Minimise MSE:

$$I = (r + \gamma \max_{a'} Q(S',a',\mathbf{w}) - Q(S,a,\mathbf{w}))^2$$

Two problems:

- Correlations
- Non-stationary targets

Q - learning

Q - learning problems

Correlations

Solve

Replay Buffer

S_1, a_1, R_2, S_2
S_2, a_2, R_3, S_3
S_3, a_3, R_4, S_4
$\cdots$
$S_t, a_t, R_{t+1}, S_{t+1}$

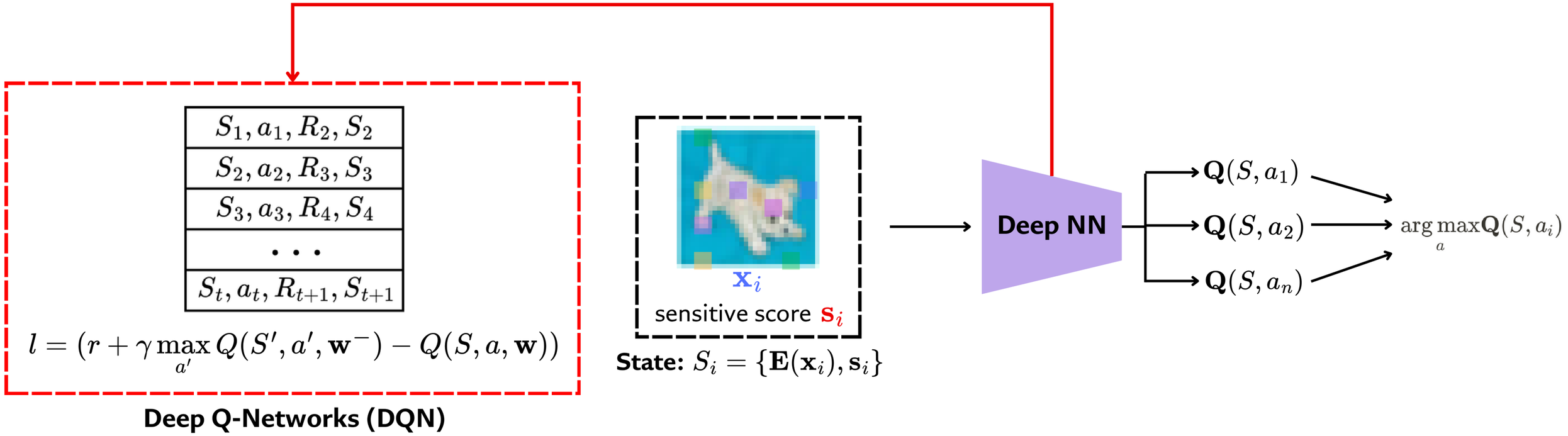
Non-stationary

Solve

Target Net

$$l = (r + \gamma \max_{a'} Q(S', a', \mathbf{w}^-) - Q(S, a, \mathbf{w}))$$

Deep Q - Network



Algorithm

Algorithm Sensitive Analysis

Require: : $\mathbf{x}_i, \mathbf{m}$, f

Ensure: $\mathbf{s}_i$

Initialize $\mathbf{s}_i \leftarrow$ empty set
$P \leftarrow f(\mathbf{x})$
$GT \leftarrow$ argmax P, $P_{GT} \leftarrow P[GT]$
for each grid in grid_space **do**
    $\mathbf{x}_{\text{new}} \leftarrow \mathbf{x}_i$ with mask $\mathbf{m}$ added at the grid location
    $P_{GT_i} \leftarrow f(\mathbf{x}_{new})[GT]$
    $\mathbf{s} \leftarrow |P_{GT} - P_{GT_i}|$
    append $\mathbf{s}$ to $\mathbf{s}_i$
end for

Algorithm

Algorithm RL Agent for Training

Require: Dataset, Max_iter = 300
Ensure: Optimized DQN agent

Initialize: Q Network parameters
for image $\mathbf{x}$ in Dataset **do**
    $m \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$
    $\mathbf{e}_0 \leftarrow \mathbf{E}(\mathbf{x})$
    $\mathbf{s}_0 \leftarrow$ sensitive_analysis($\mathbf{x}$)
    $S_0 \leftarrow \{\mathbf{e}_0, \mathbf{s}_0\}$, $\mathbf{x}_0 \leftarrow \mathbf{x}$
    $i \leftarrow 1$
    while not misclassify **and** $i <$ Max_iter **do**
        $a_i \leftarrow$ select_action(S_{i-1}, m)
        $\mathbf{x}_i \leftarrow$ take_action($\mathbf{x}_{i-1}, a_i$)
        $\mathbf{R}_i \leftarrow$ sensitive_score($\mathbf{x}_i$) $- \|\mathbf{x} - \mathbf{x}_i\|_2$
        Update DQN agent
        $\mathbf{s}_i \leftarrow$ sensitive_analysis($\mathbf{x}_i$)
        $\mathbf{e}_i \leftarrow \mathbf{E}(\mathbf{x}_i)$
        $S_i \leftarrow \{\mathbf{e}_i, \mathbf{s}_i\}$
        $i \leftarrow i + 1$
    end while
end for

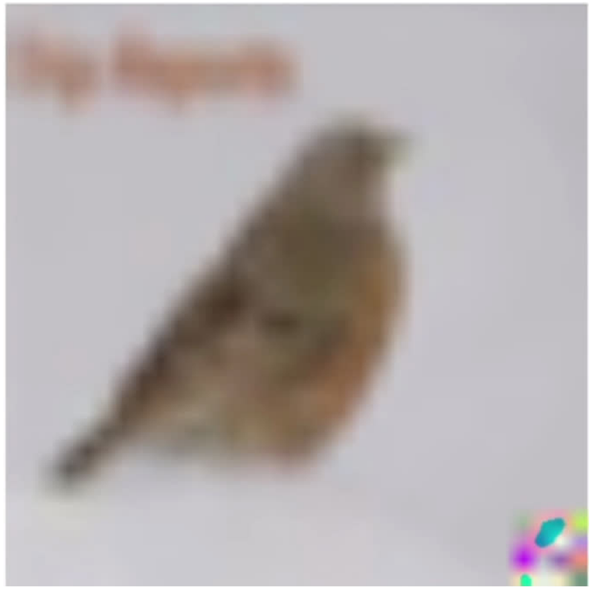
Problem

The previous attacked areas became **more sensitive.**
=> agent keep adding noise to those areas.

Even though the L2 norm is still guaranteed small, the noises still can be observed by human eyes clearly.

Problem

The previous attacked areas became **more sensitive.**
=> agent keep adding noise to those areas.



Even though the L2 norm is still guaranteed small, the noises still can be observed by human eyes clearly.

Problem

The previous attacked areas became **more sensitive.**
=> agent keep adding noise to those areas.

Even though the L2 norm is still guaranteed small, the noises still can be observed by human eyes clearly.

$F[a]$ frequency value of action a

Solution

- Agent definition: finding region from original image to add the noise mask
- State: $S_i = \{\mathbf{E}(\mathbf{x}_i), \mathbf{s}_i\}$ is state of $i\text{th}$ step
- Action: a_i, adding to specific grid of $\mathbf{x}_i$
- Reward: $R_i = -||\mathbf{x} - \mathbf{x}_i||_2 + |P_{GT} - P_{GT_i}|$

Solution

- Agent definition: finding region from original image to add the noise mask
- State: $S_i = \{\mathbf{E}(\mathbf{x}_i), \mathbf{s}_i, \mathbf{h}\}$ is state of $i\text{th}$ step
- Action: a_i, adding to specific grid of $\mathbf{x}_i$
- Reward: $R_i = -||\mathbf{x} - \mathbf{x}_i||_2 + |P_{GT} - P_{GT_i}| - F[a_i]$

Solution

Algorithm RL Agent for Training

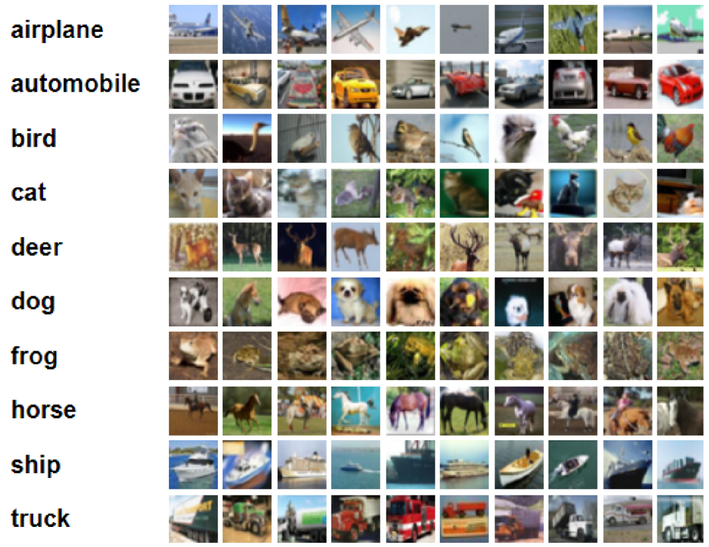
Require: Dataset, Max_iter = 300
Ensure: Optimized DQN agent

Initialize: Q Network parameters
for image $\mathbf{x}$ in Dataset **do**
  Initialize: $\mathbf{F}, \mathbf{h} \leftarrow$ zeros vector
  $m \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$
  $\mathbf{e}_0 \leftarrow \mathbf{E}(\mathbf{x})$
  $\mathbf{s}_0 \leftarrow$ sensitive_analysis($\mathbf{x}$)
  $S_0 \leftarrow \{\mathbf{e}_0, \mathbf{s}_0, \mathbf{h}\}$, $\mathbf{x}_0 \leftarrow \mathbf{x}$
  $i \leftarrow 1$
  while not misclassify **and** $i <$ Max_iter **do**
    $a_i \leftarrow$ select_action(S_{i-1}, m)
    $\mathbf{x}_i \leftarrow$ take_action($\mathbf{x}_{i-1}, a_i$)
    $\mathbf{R}_i \leftarrow$ sensitive_score($\mathbf{x}_i$) $- \|\mathbf{x} - \mathbf{x}_i\|_2 - \mathbf{F}[a_i]$
    Update DQN agent
    $\mathbf{s}_i \leftarrow$ sensitive_analysis($\mathbf{x}_i$)
    $\mathbf{F}[a_i] \leftarrow \mathbf{F}[a_i] + 1$
    update history $\mathbf{h}$
    $\mathbf{e}_i \leftarrow \mathbf{E}(\mathbf{x}_i)$
    $S_i \leftarrow \{\mathbf{e}_i, \mathbf{s}_i, \mathbf{h}\}$
    $i \leftarrow i + 1$
  end while
end for

Experiment

Dataset and Black-box Classification Model

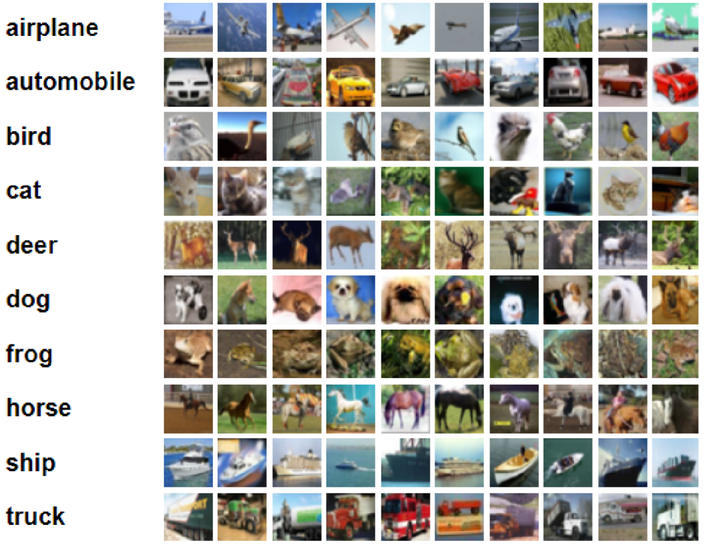
- **Cifar10:** Consists of 60000 32x32 color images in 10 classes. 50000 for training, 10000 for testing.
- **VGG19:** Training for 200 epoch with **0.9026** acc.



RESOURSE LIMITATION

Dataset and Black-box Classification Model

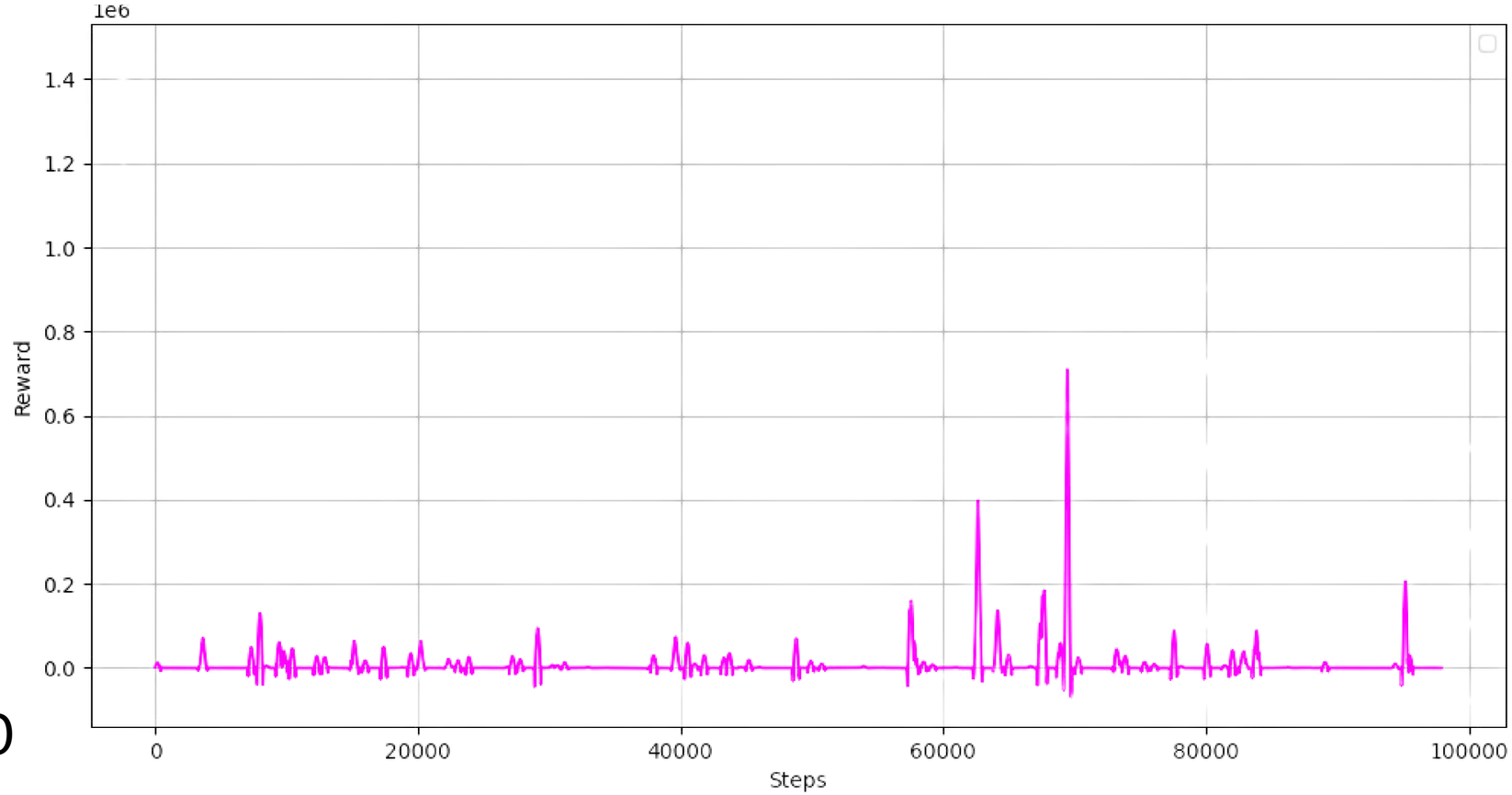
- **Cifar10:** Consists of 60000 32x32 color images in 10 classes. 50000 for training, 10000 for testing.
- **VGG19:** Training for 200 epoch with **0.9026** acc.



Traning	Testing
500 (50 image per class)	100 (10 image per class)

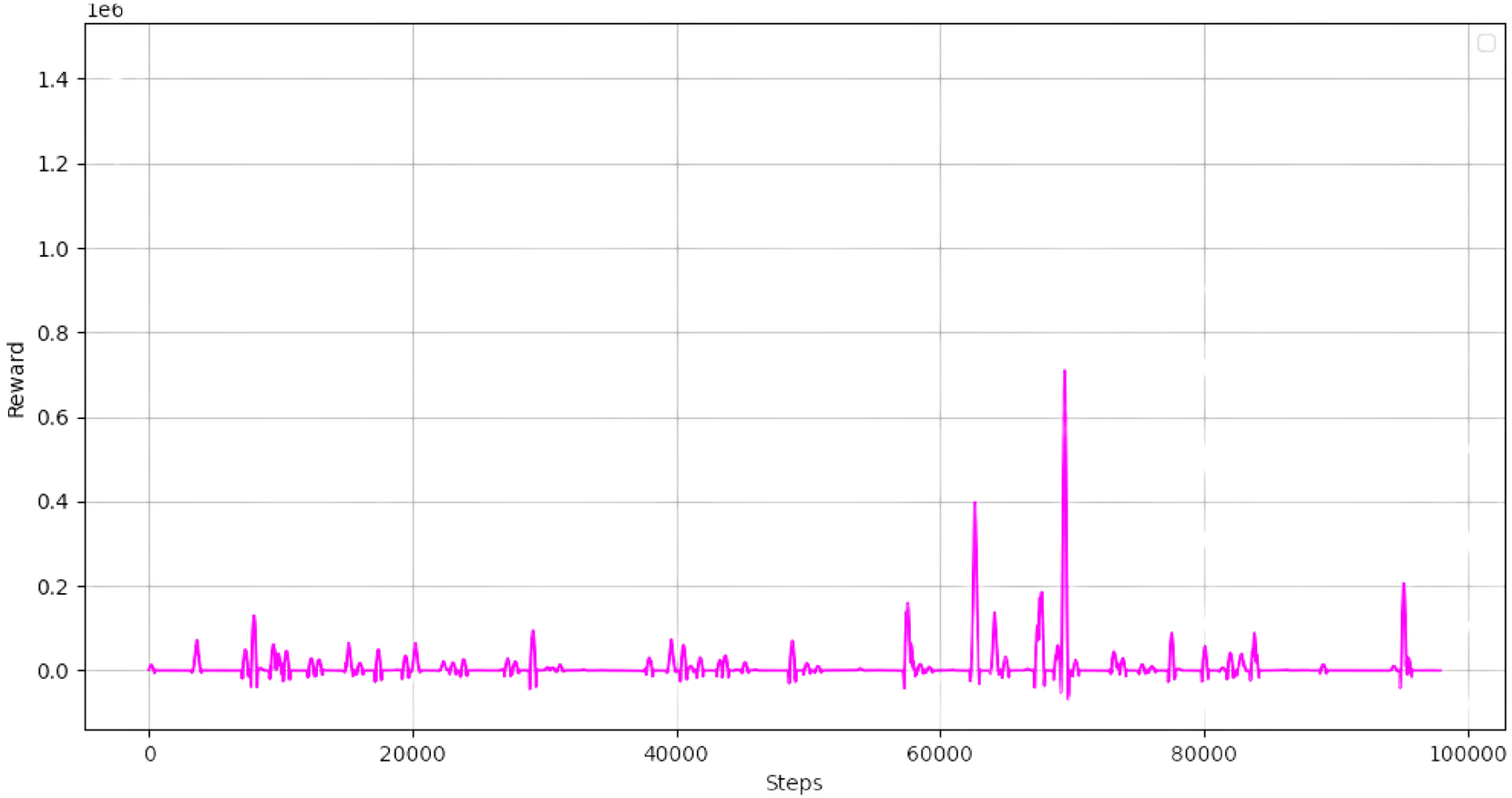
DQN Agent Training

- Total steps ~ **98000**
  Max iter / image: **300**
- Batch size: **32**
- Target update: **1000**
- Noise sd: **0.005**
- Gamma: **0.9**
- Epsilon: **0.1**
- Kaggle T4 GPU
  NVIDIA GetForce 3060



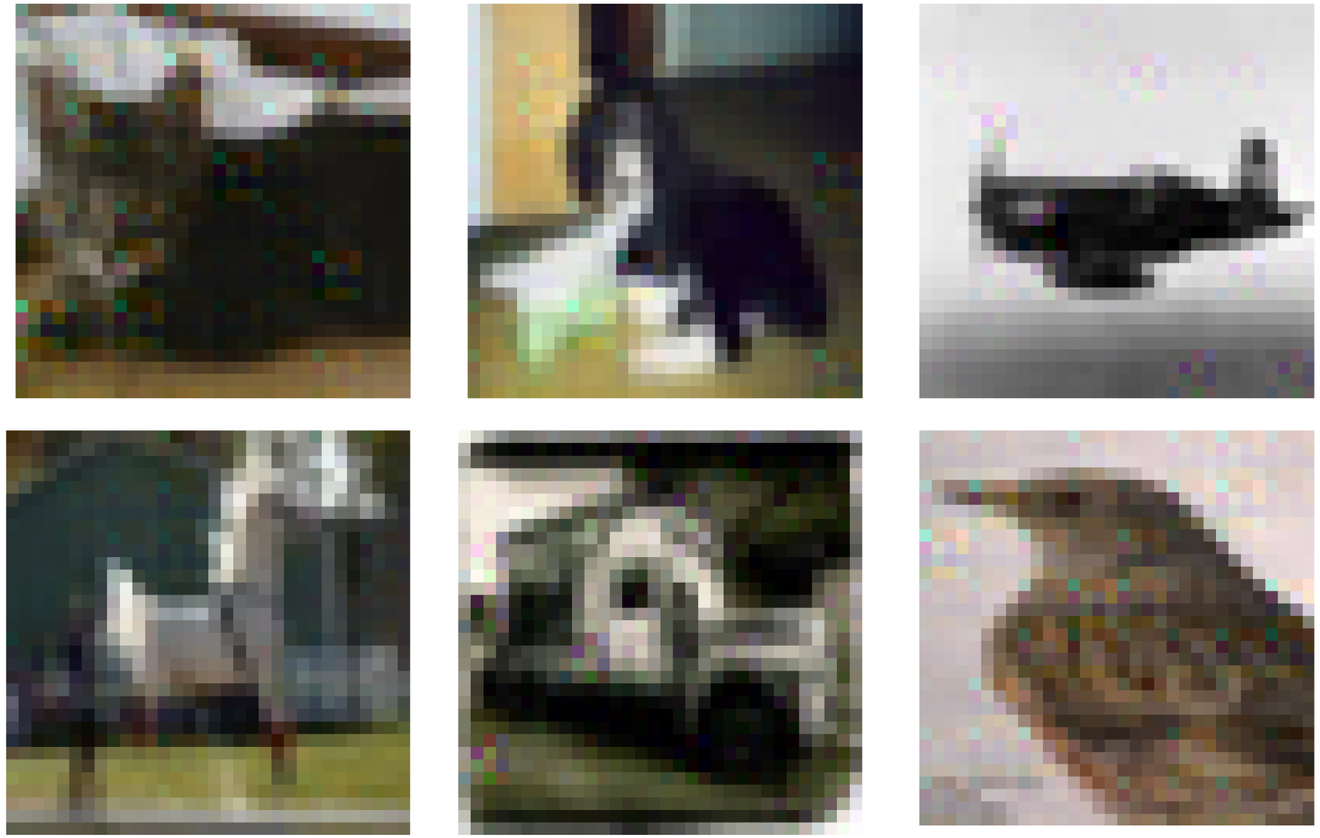
Limitations

- The process of training is not stable.
- Reward will reduce when **L2 norm and frequent of actions increases.**



Attack Result

- Success rate: 21 / 100
- Avg L2 norm: 7.48809
- AVG step: 280.1



Limitations

- The process of training is not stable.
- Reward function does not have the score for terminated state.
- Adding random noise at grid level is not as well as pixel level.
- Time for **sensitive analysis** is really long.

  100 images ~ 20 hours → 50000 image ~ 1000 days

References

- [1] Soumyendu Sarkar. “Robustness with Query-efficient Adversarial Attack using Reinforcement Learning”. CVPR. 2023
- [2] Chenglin Yang. “PatchAttack: A Black-Box Texture-Based Attack with Reinforcement Learning”. ECCV. 2020
- [3] Jonathan Peck. “An Introduction to Adversarially Robust Deep Learning”. IEEE Trans. Pattern Anal. Mach. Intell. 2024

THANKS FOR WATCHING!