Adversarial Training against Location-Optimized Adversarial Patches

Sukrut Rao

David Stutz

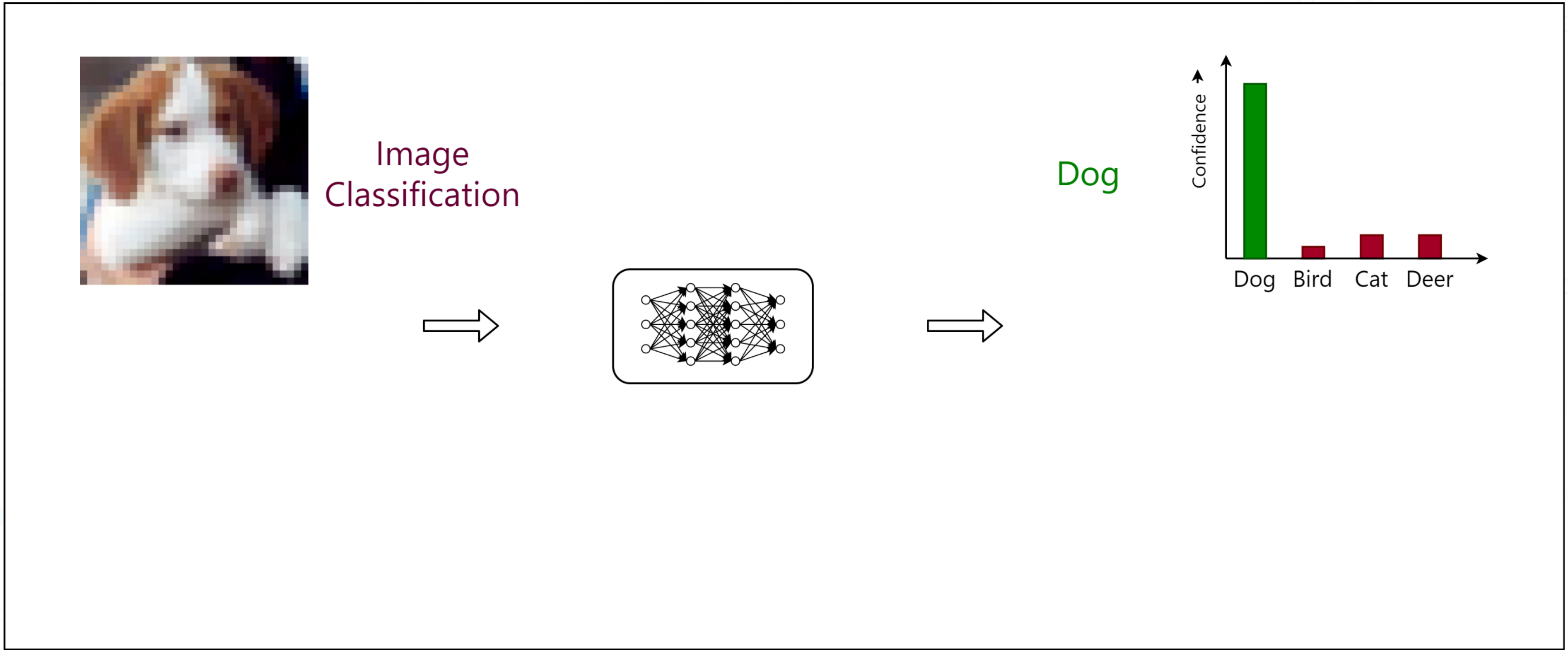
Bernt Schiele

Max Planck Institute for Informatics, Saarland Informatics Campus

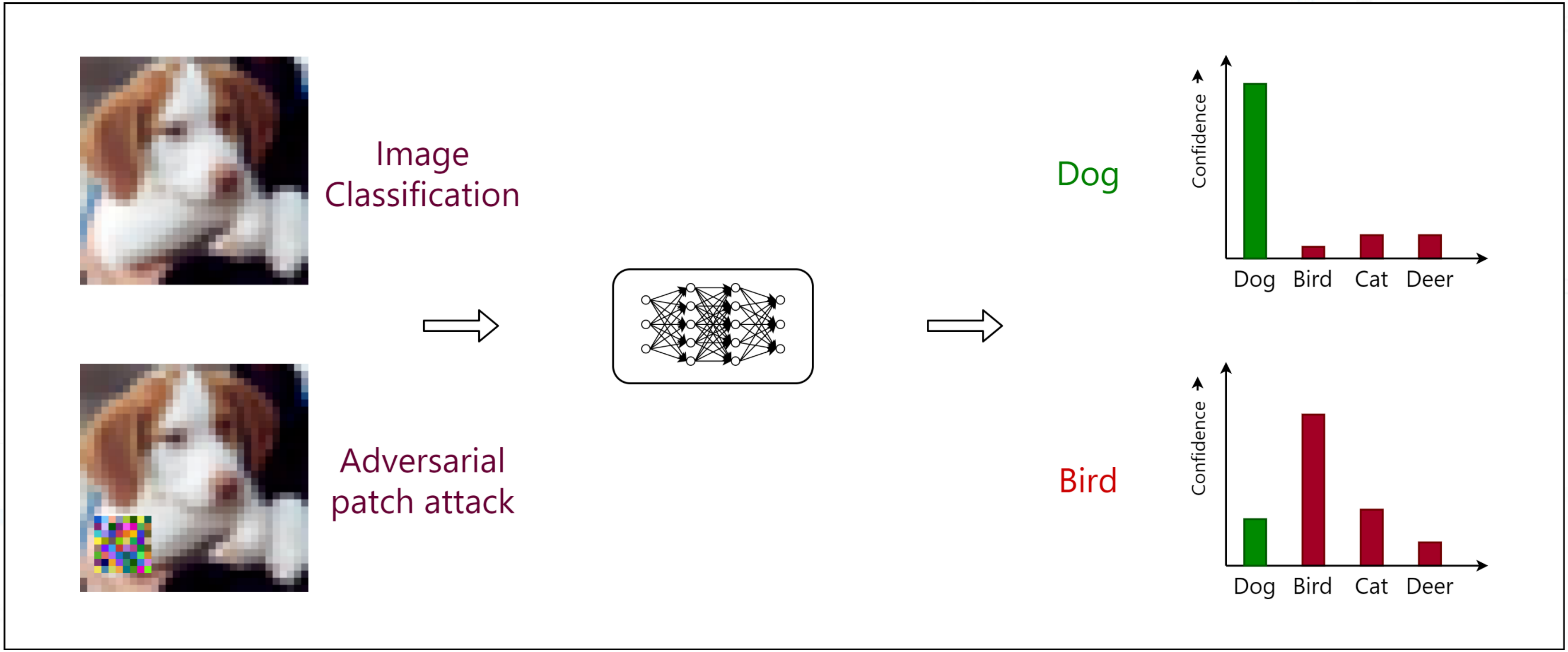
ECCV Workshop on The Bright and Dark Sides of Computer Vision: Challenges and Opportunities for Privacy and Security (CV-COPS) 2020

2-Minute Overview

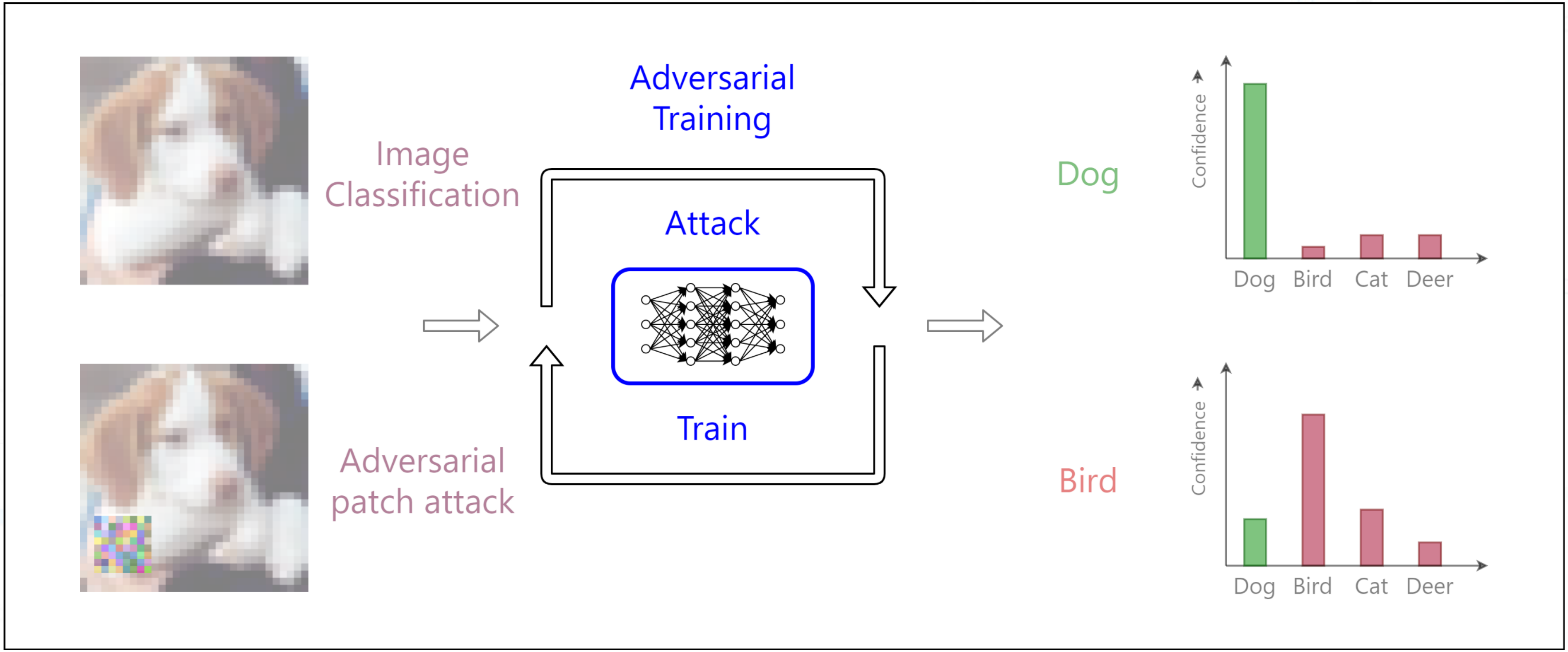
2-Minute Overview



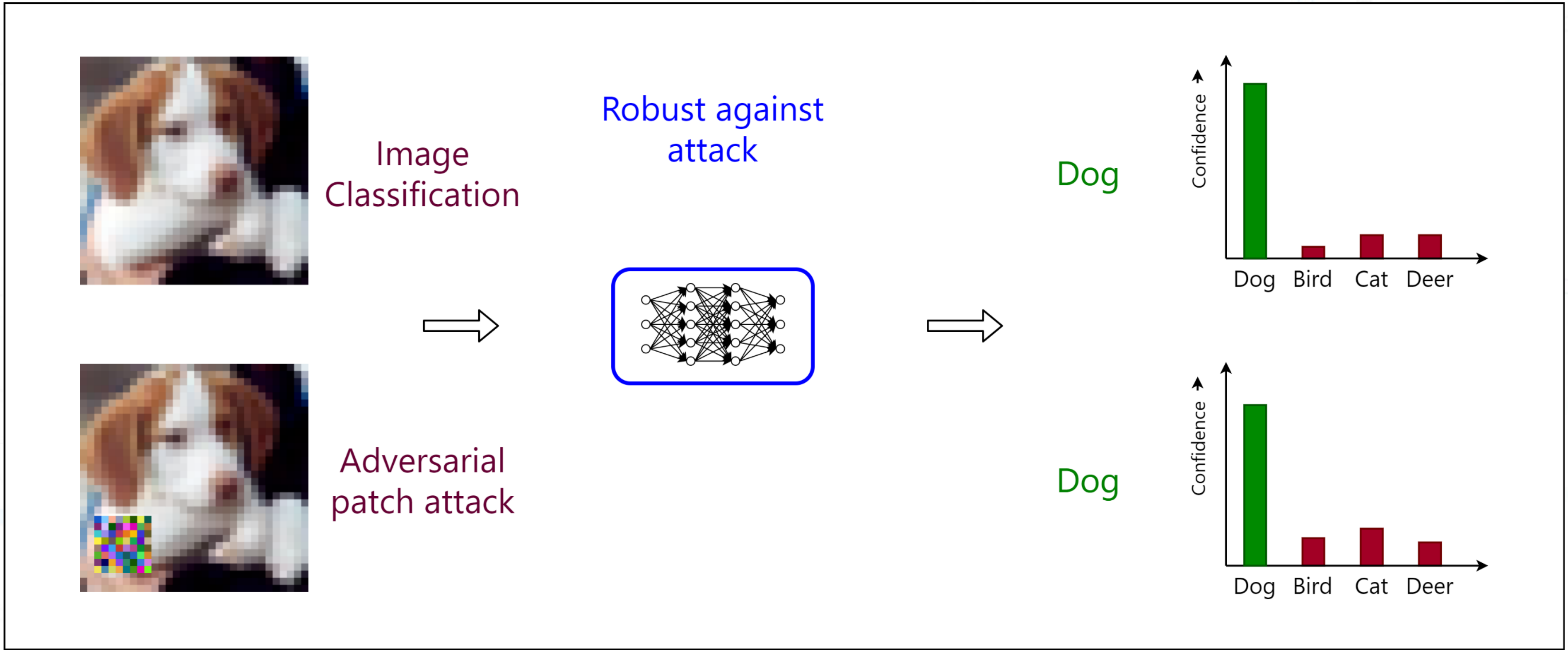
2-Minute Overview



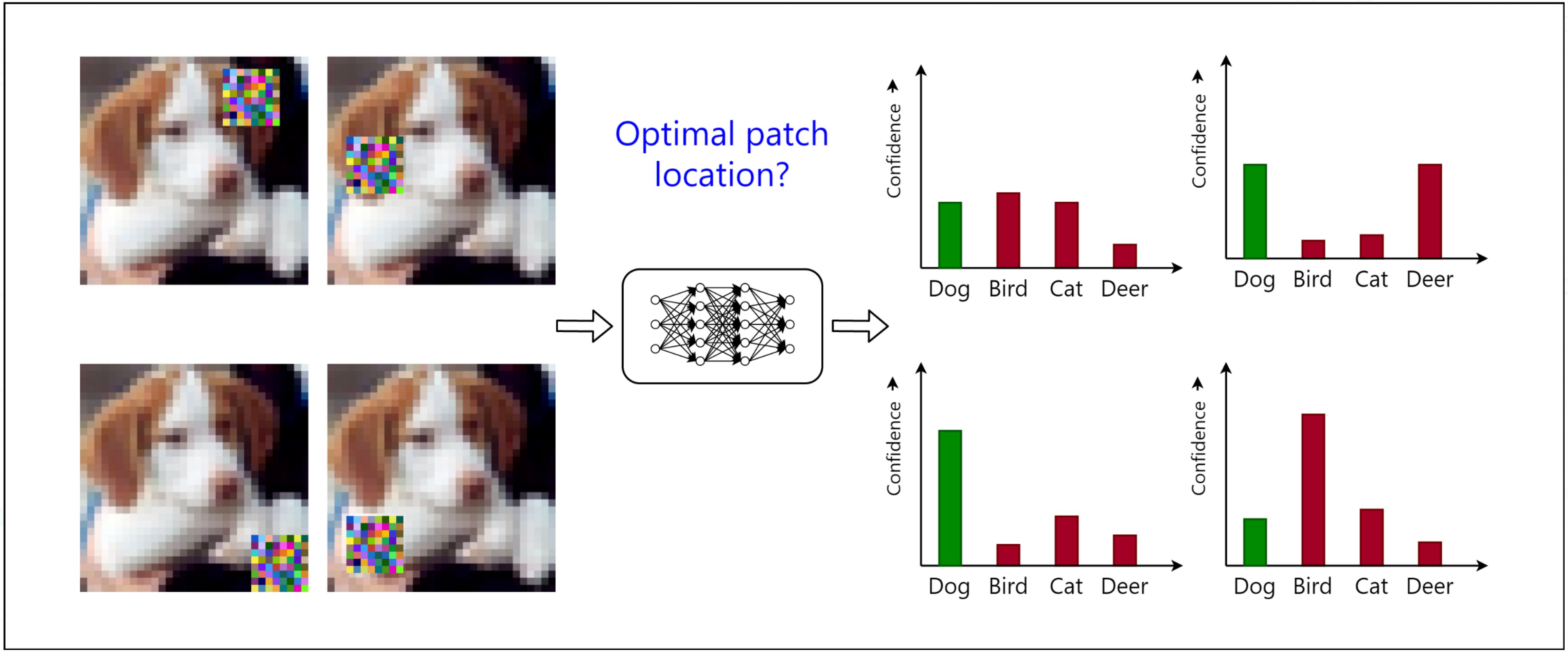
2-Minute Overview



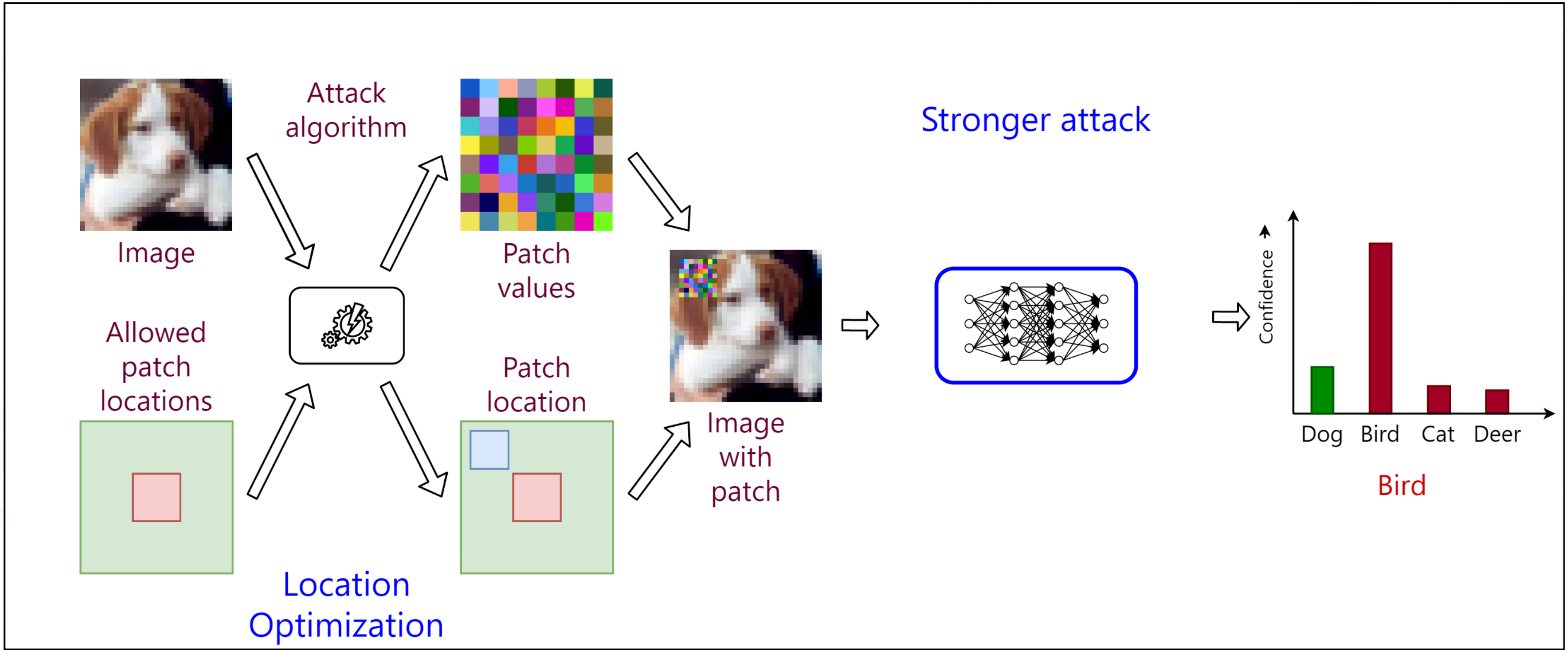
2-Minute Overview



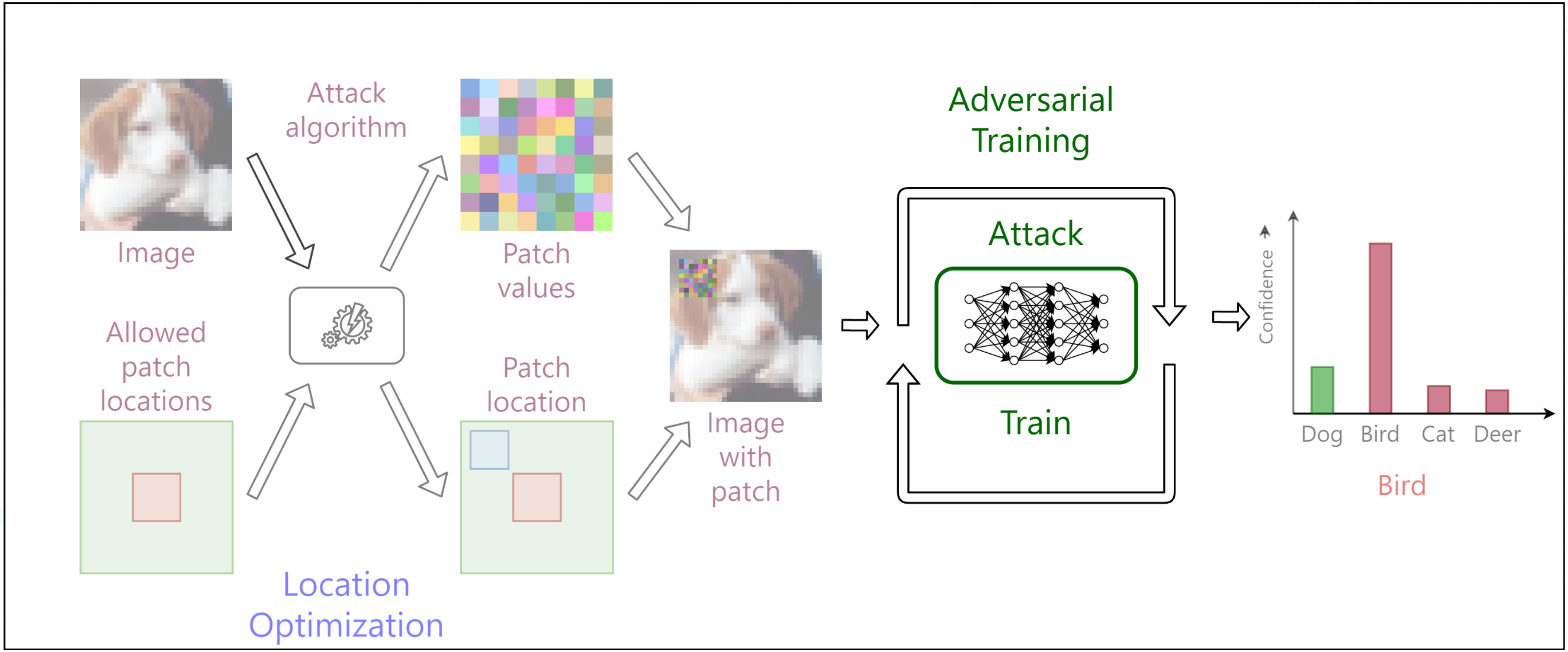
2-Minute Overview



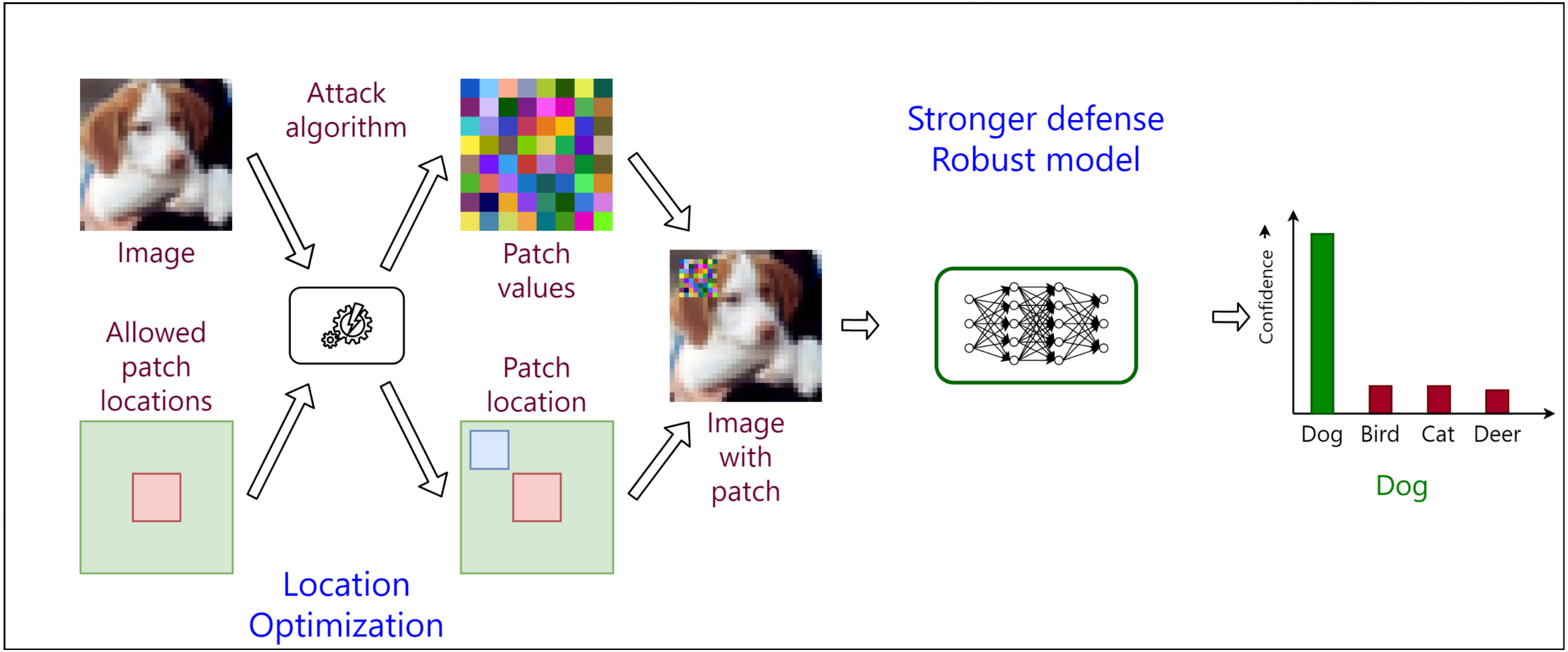
2-Minute Overview



2-Minute Overview



2-Minute Overview

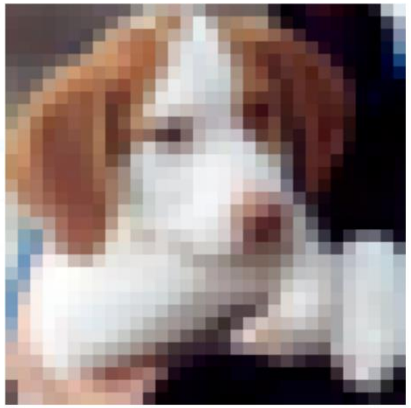


Outline

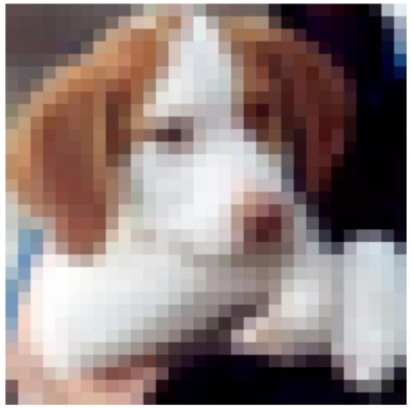
- Objective and Contributions
- Adversarial Patch Attack with Location Optimization
- Adversarial Patch Training
- Experimental Evaluation

Adversarial Patch

- A small contiguous patch of pixels to cause image misclassification
- Practical form of attack

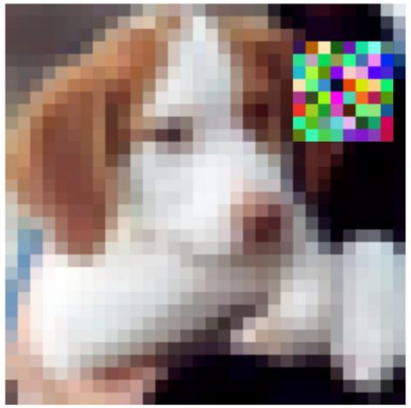


Dog



Bird

Imperceptible attack



Bird

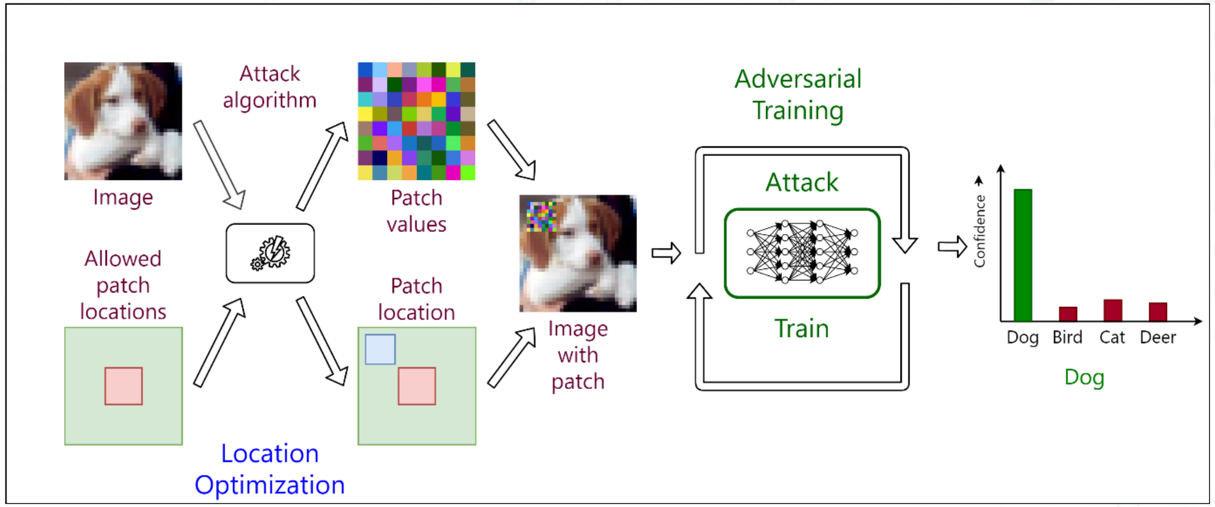
Adversarial patch

Objective and Contributions

Objective: Can adversarial training make a classifier robust against adversarial patches?

Contributions:

- Adversarial patch attack with location-optimization
- Adversarial training defense



Adversarial Patch Attack: Design Choices

Desired Property: Use strongest possible attack for each image

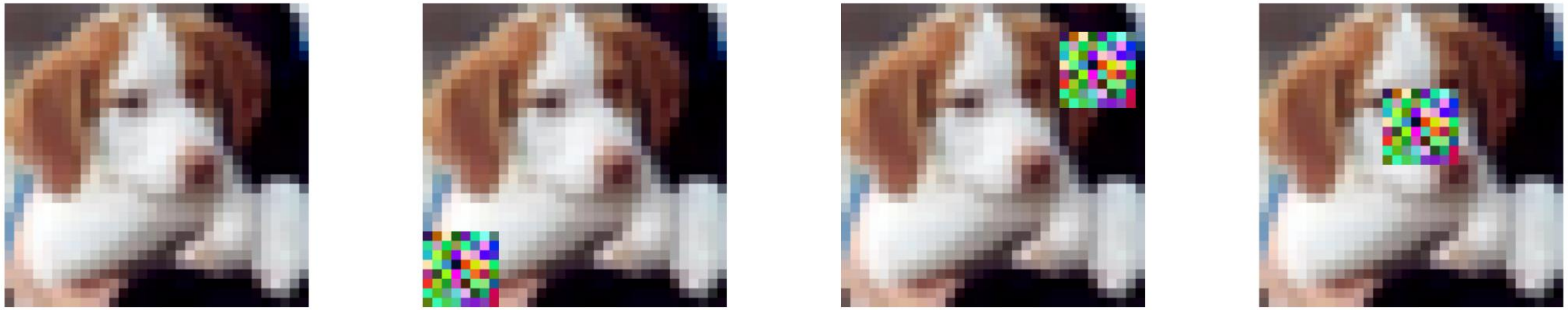
Motivation: Network robust against strong attacks is likely to be robust against weaker attacks

Design choices for adversarial patch attack:

- **Image-specific:** Separately generated patch for each image
- **Untargeted:** No target class for misclassification
- **Location-optimized:** Find optimal patch location

Adversarial Patch Attack: Location Optimization

- All patch locations not equally effective
- Find optimal location to place patch on the image
- Avoid locations likely to block vital features: image center

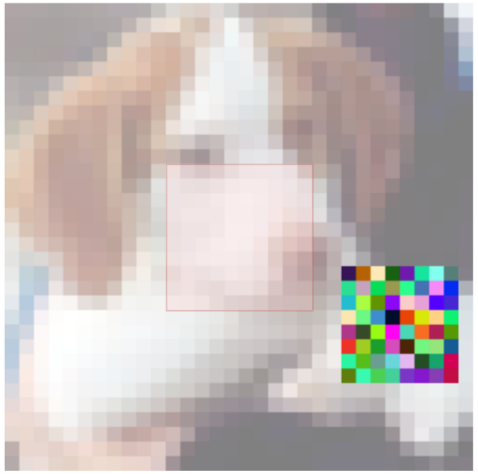


Dog | Dog | Bird |

Unsuccessful attack | Successful attack | Disallowed patch location

Adversarial Patch Attack: Initial Patch Locations

Fixed location
near image corner

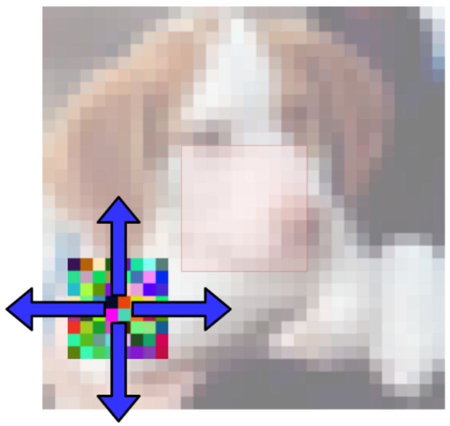


Random location
outside center region

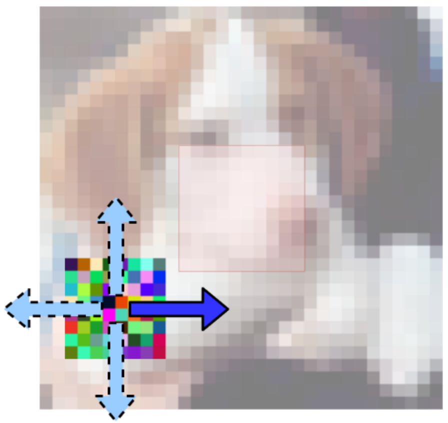
Adversarial Patch Attack: Location Optimization Strategies

Strategy:

- Check if a location in neighborhood of current location is better
- Move patch to each such location to check effectiveness



Full location optimization
All four directions



Random location optimization
One direction at random

Adversarial Patch Attack

Optimization function:

$$\max_{\delta, m} L(\underbrace{f}_{\text{Network}}(\underbrace{(1-m)\odot x + m\odot\delta}_{\text{Patched image}}; w), \underbrace{y}_{\text{Label}})$$

Perturbations → δ
Mask → m

Performing the attack:

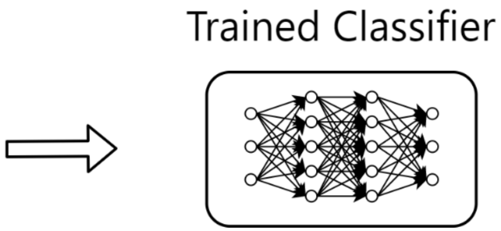
- Initialize patch with random values
- Alternating steps:
  - Update patch values using gradients
  - Update patch location

Adversarial Patch Attack

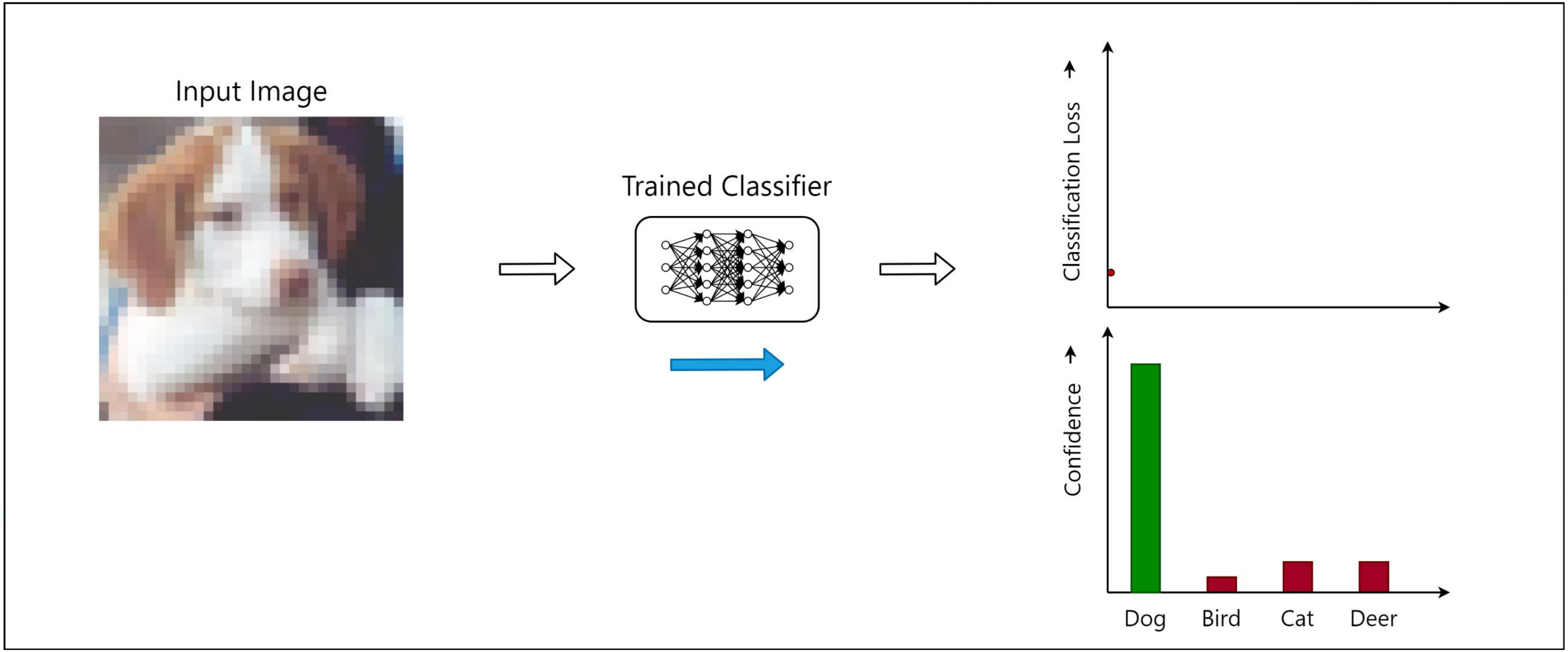
Input Image

Adversarial Patch Attack

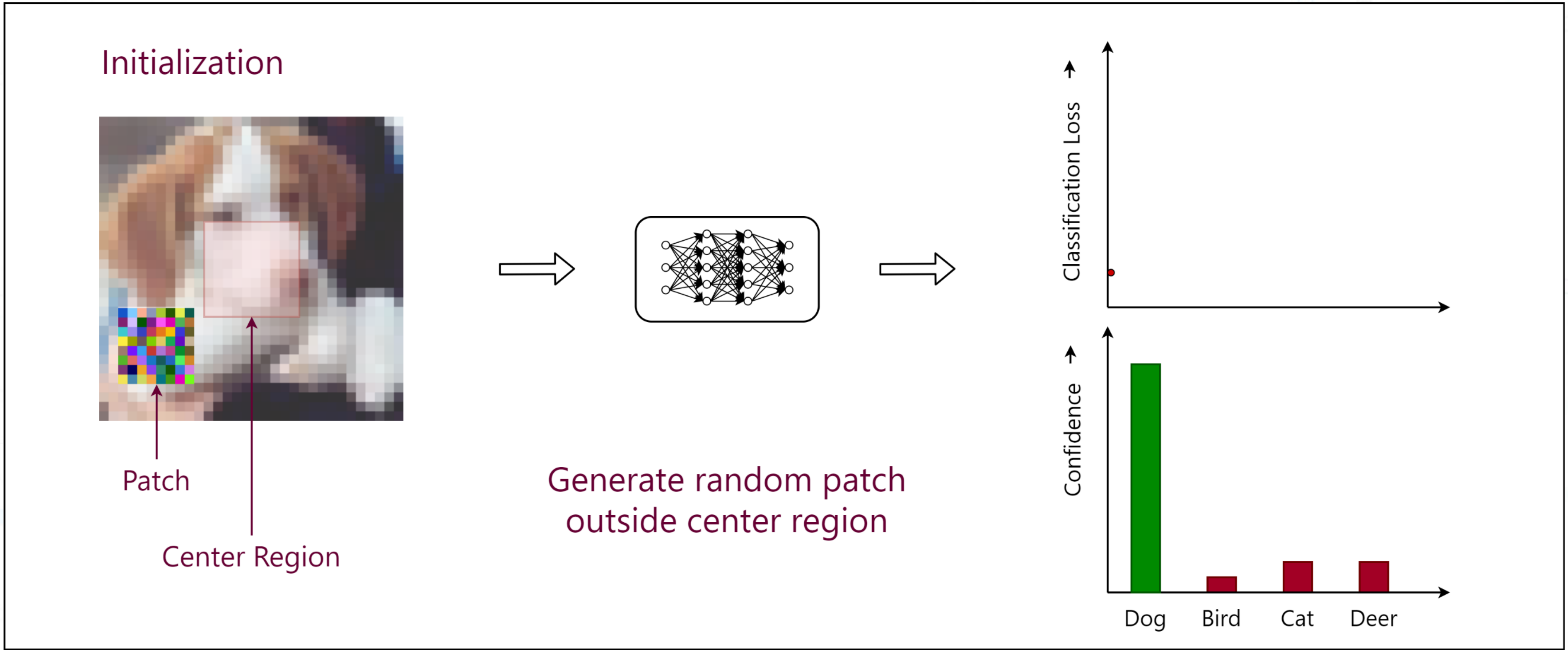
Input Image



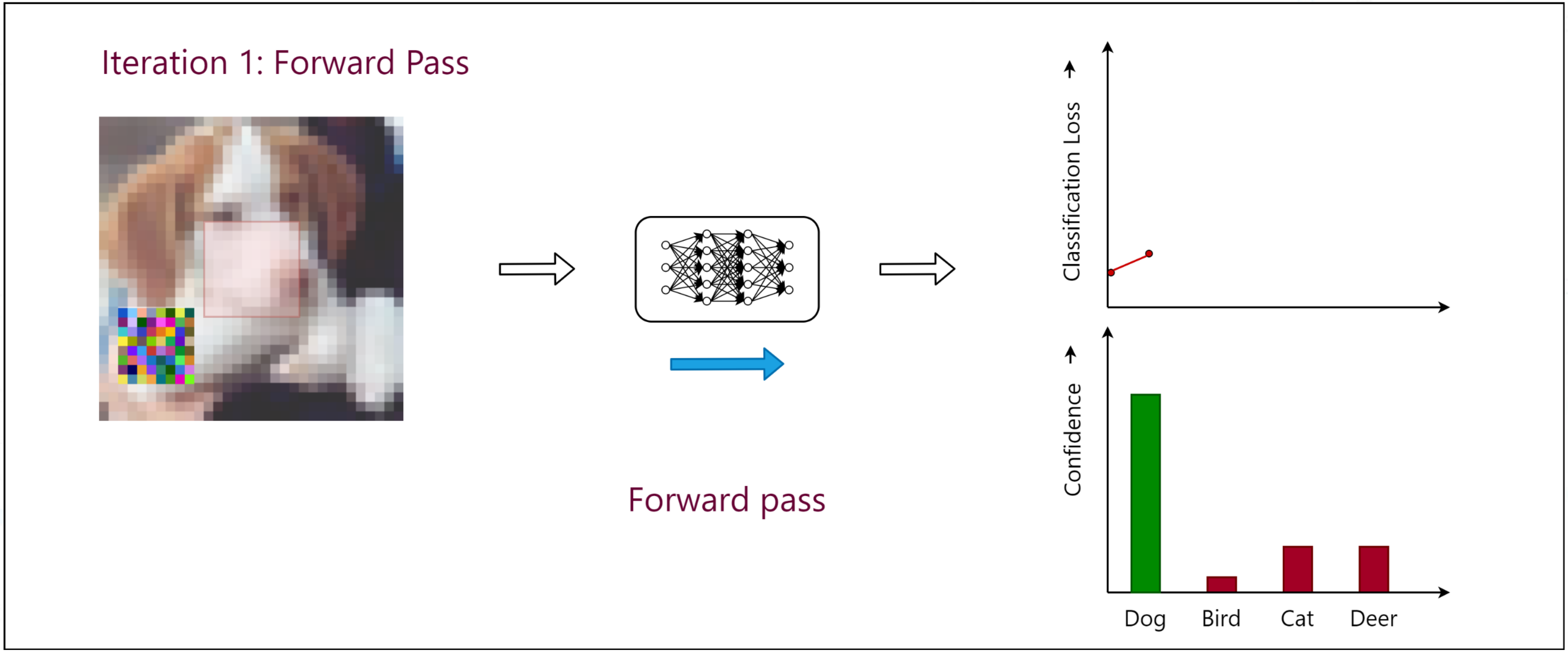
Adversarial Patch Attack



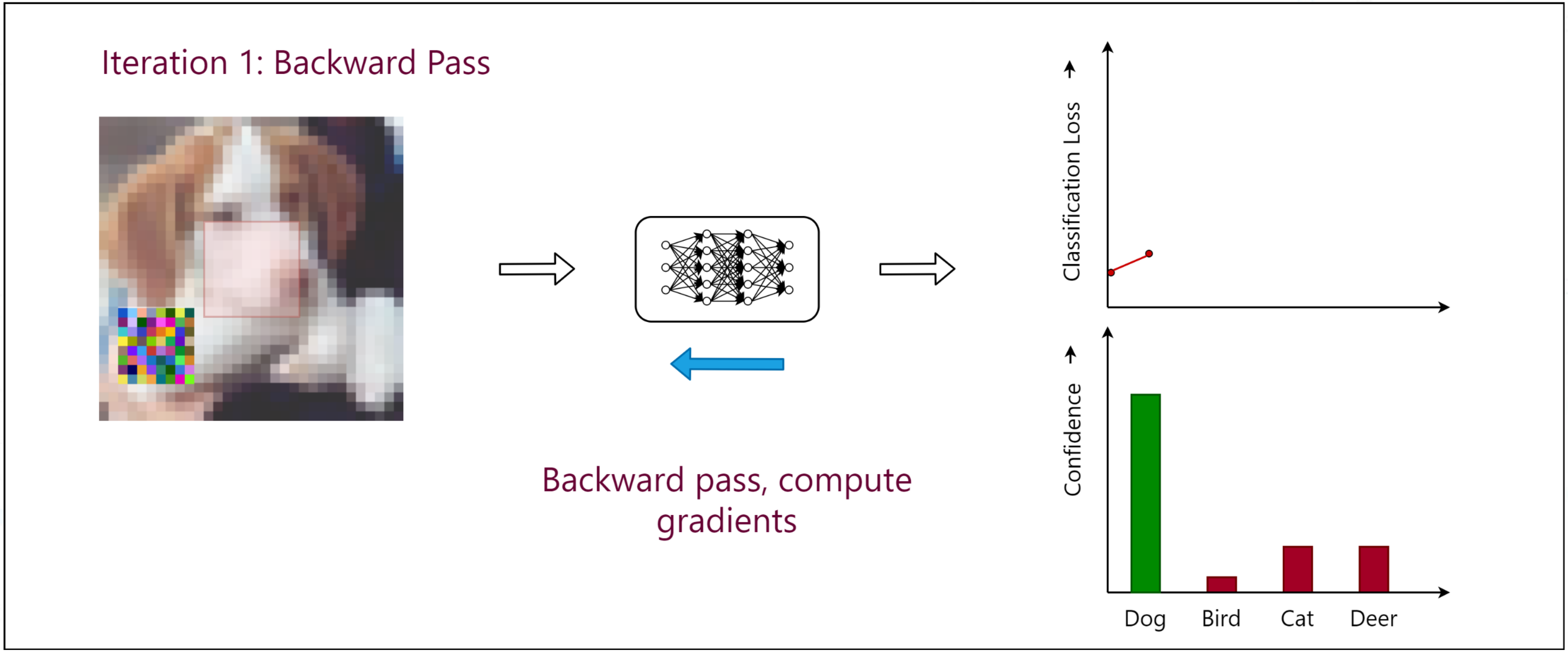
Adversarial Patch Attack: Initialization



Adversarial Patch Attack: Forward Pass

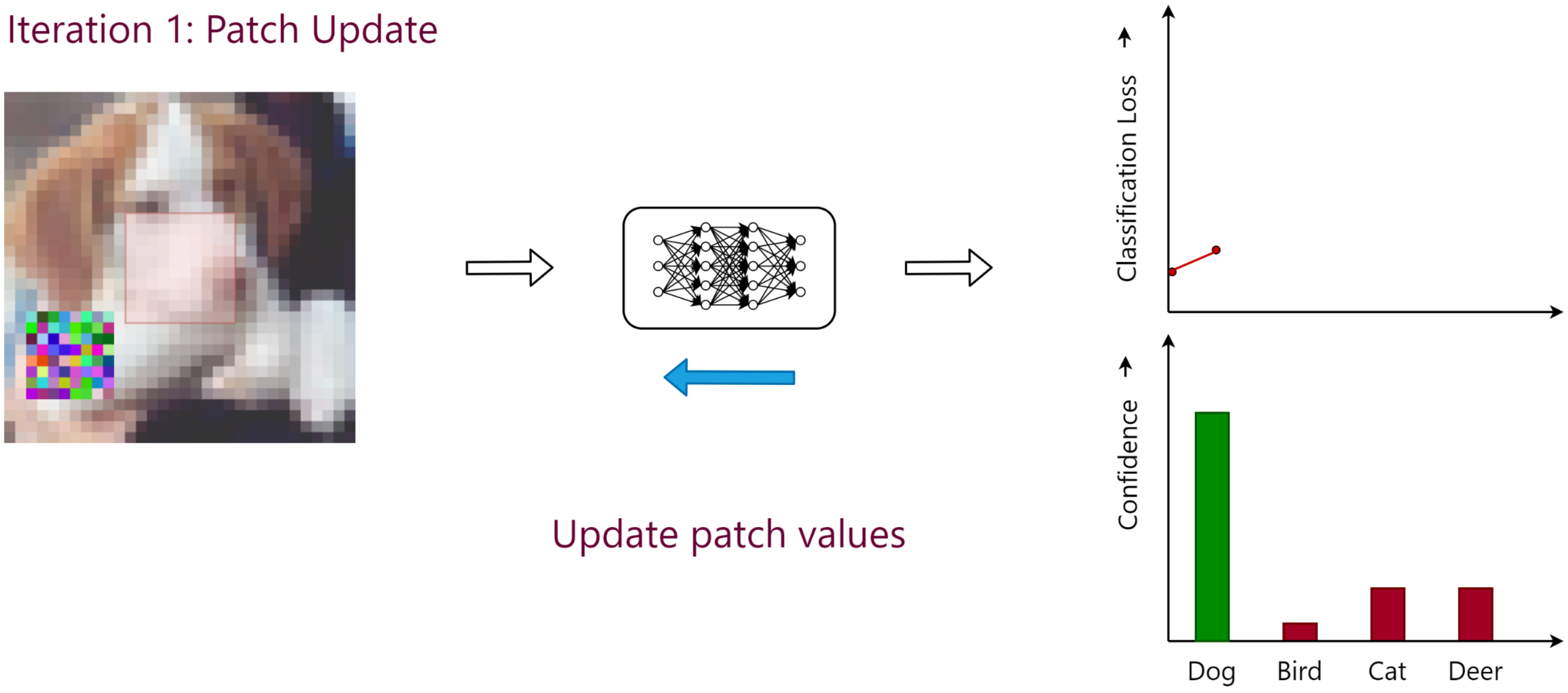


Adversarial Patch Attack: Backward Pass



Adversarial Patch Attack: Patch Update

Iteration 1: Patch Update

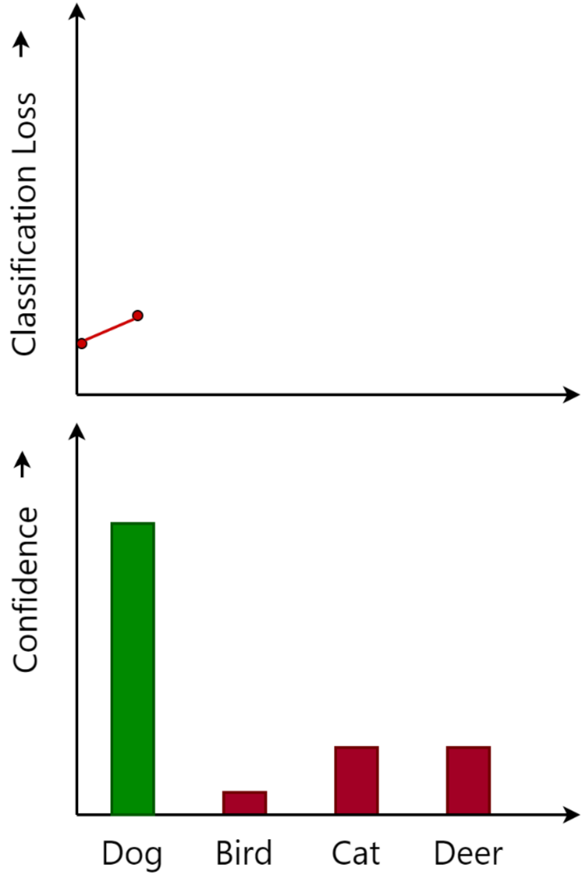


Update patch values

Adversarial Patch Attack: Location Optimization

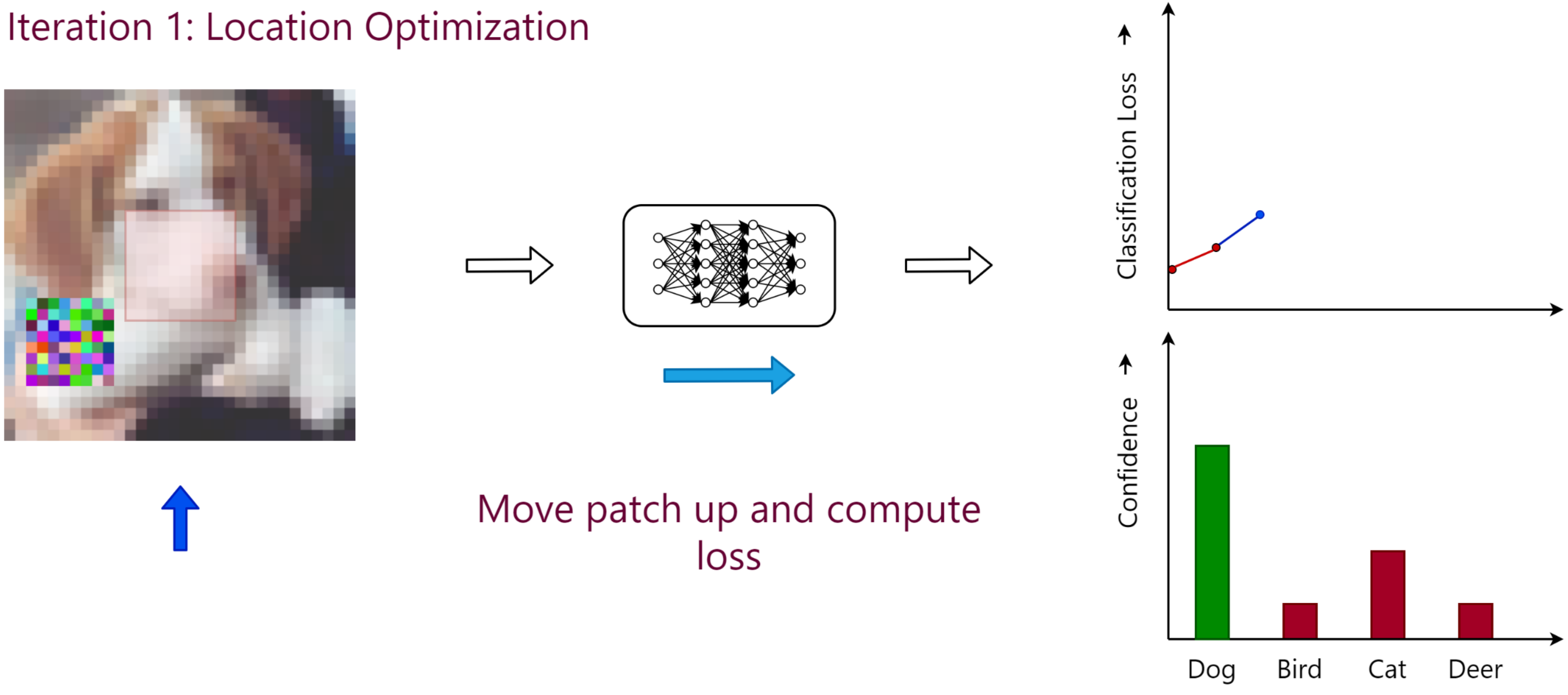
Iteration 1: Location Optimization

Perform location optimization



Adversarial Patch Attack: Location Optimization

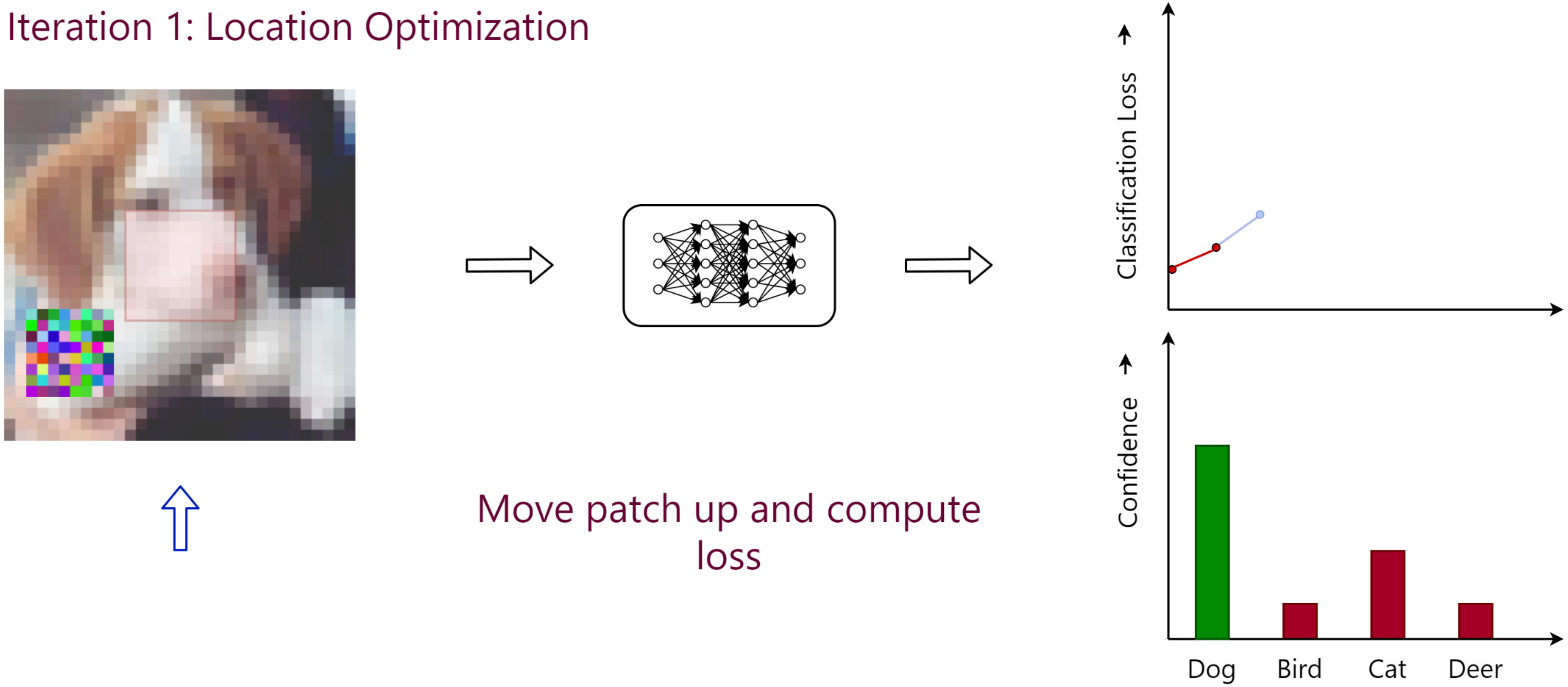
Iteration 1: Location Optimization



Move patch up and compute loss

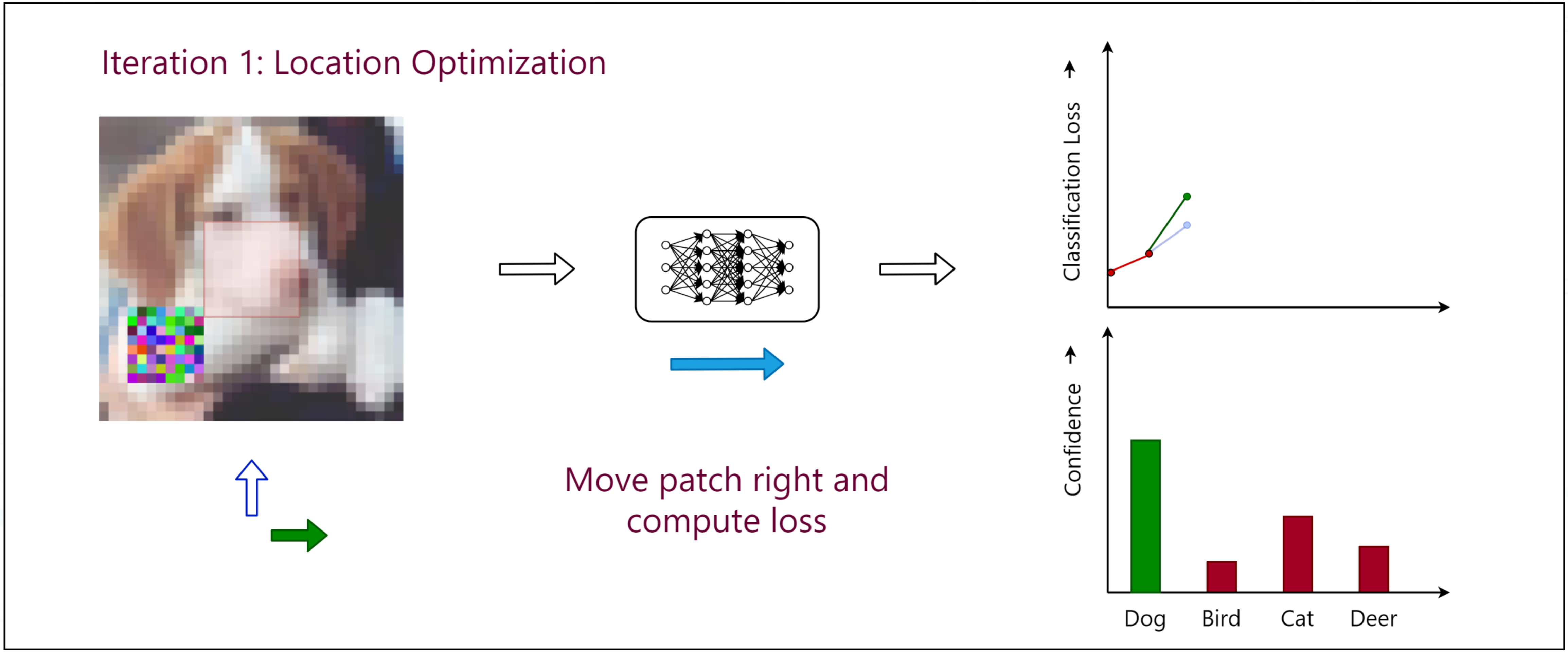
Adversarial Patch Attack: Location Optimization

Iteration 1: Location Optimization

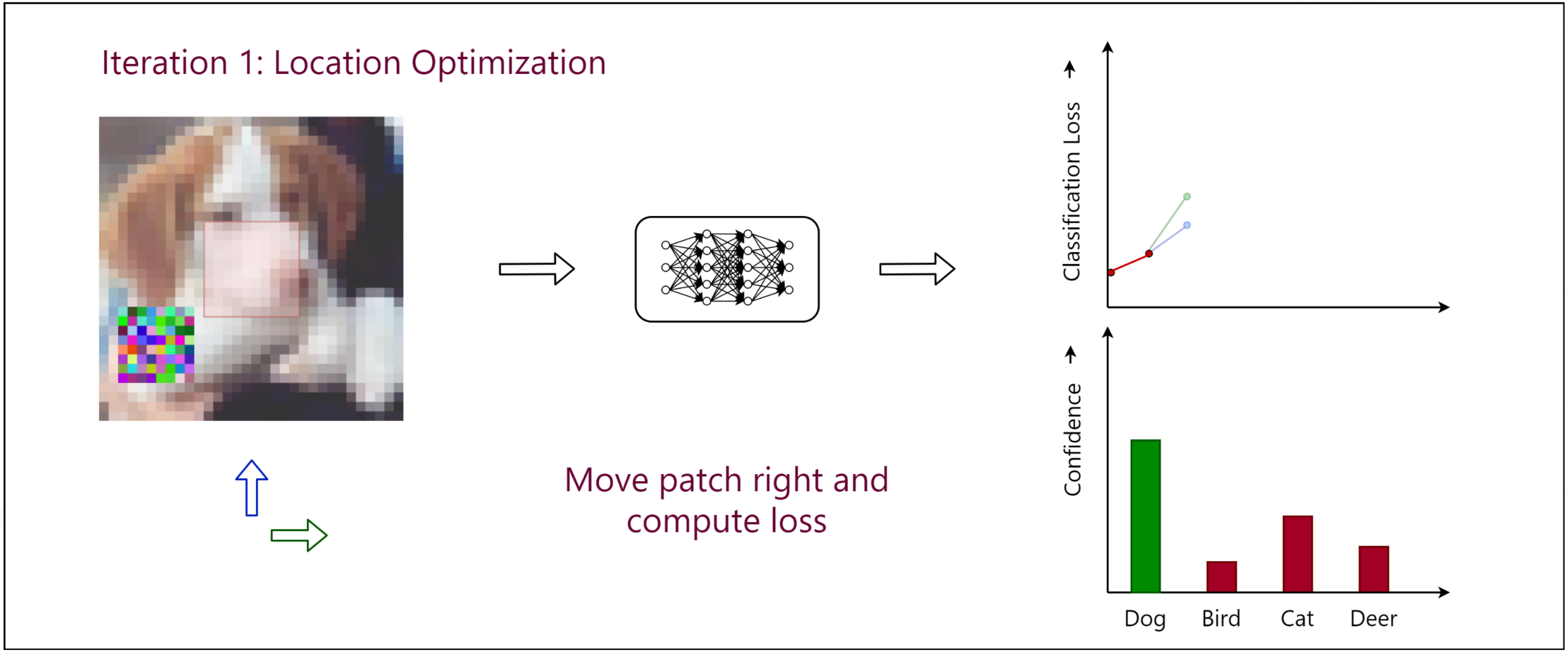


Move patch up and compute loss

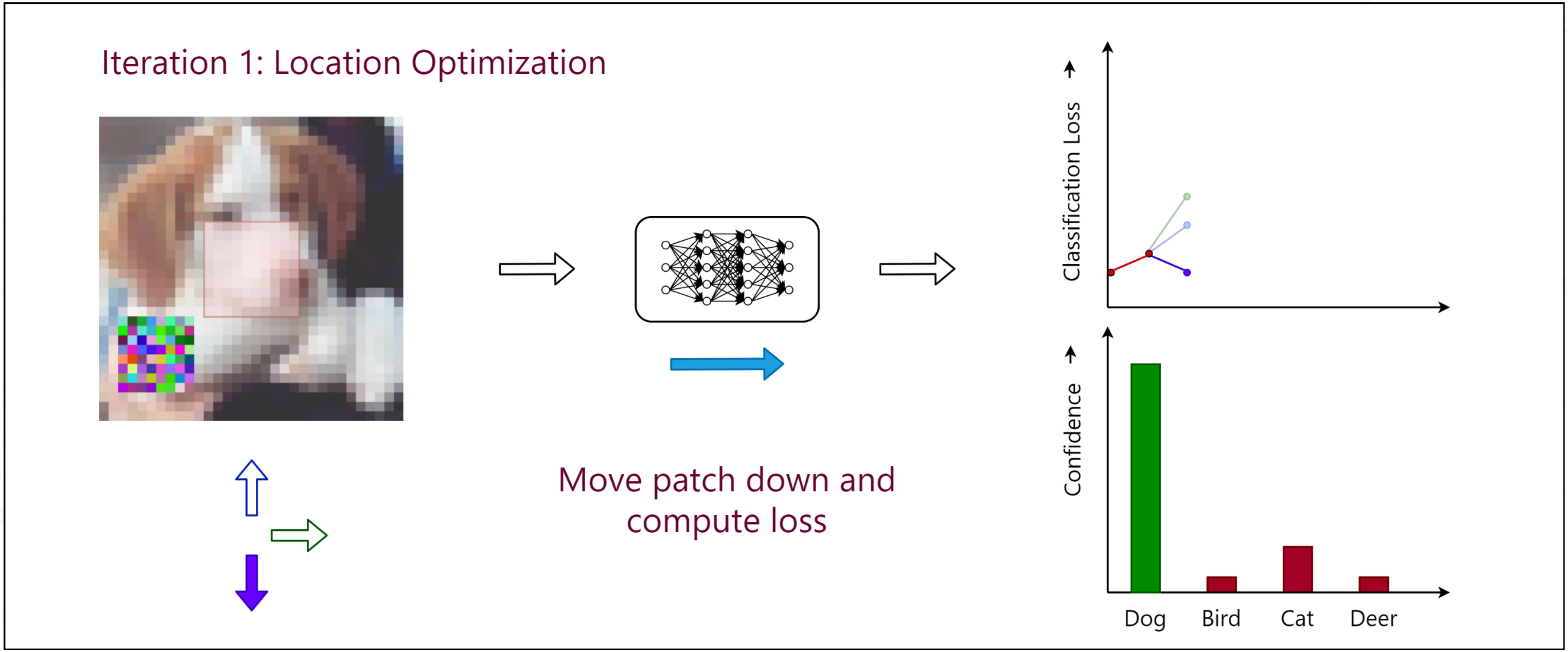
Adversarial Patch Attack: Location Optimization



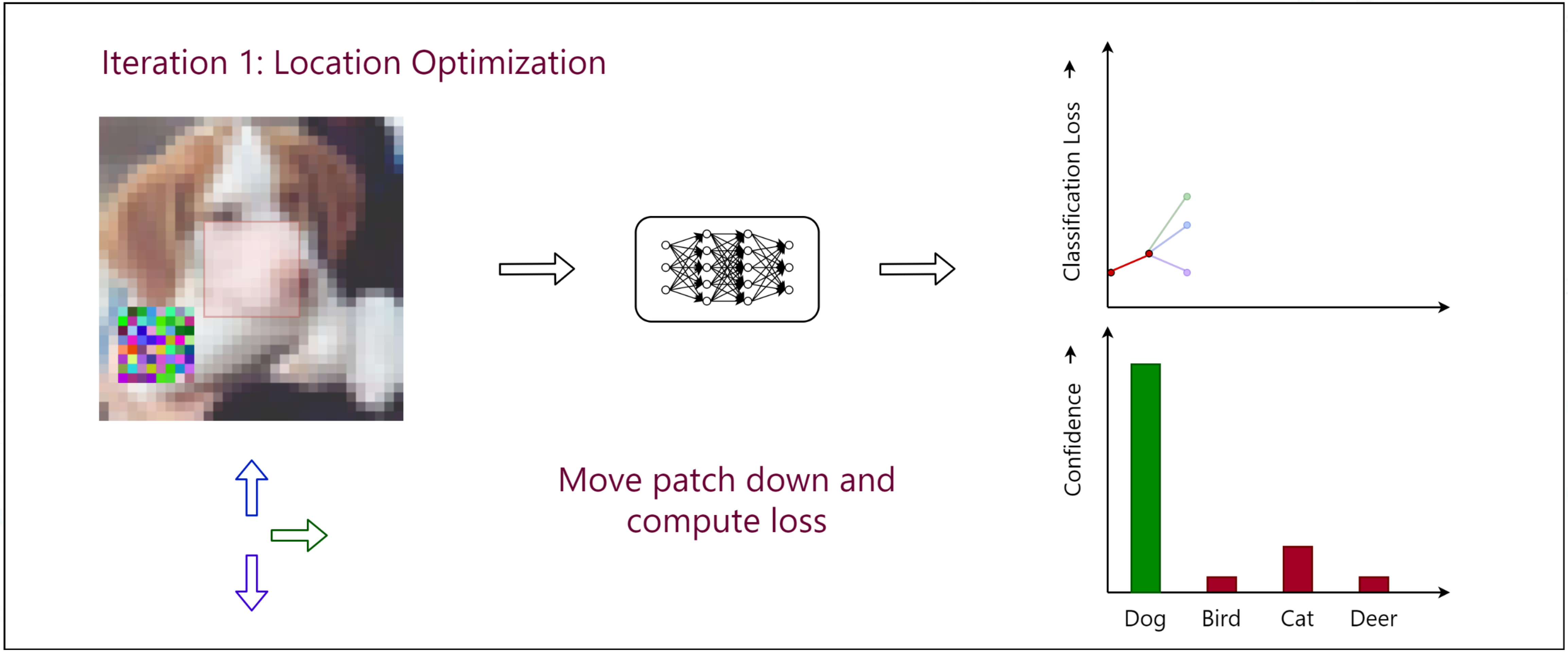
Adversarial Patch Attack: Location Optimization



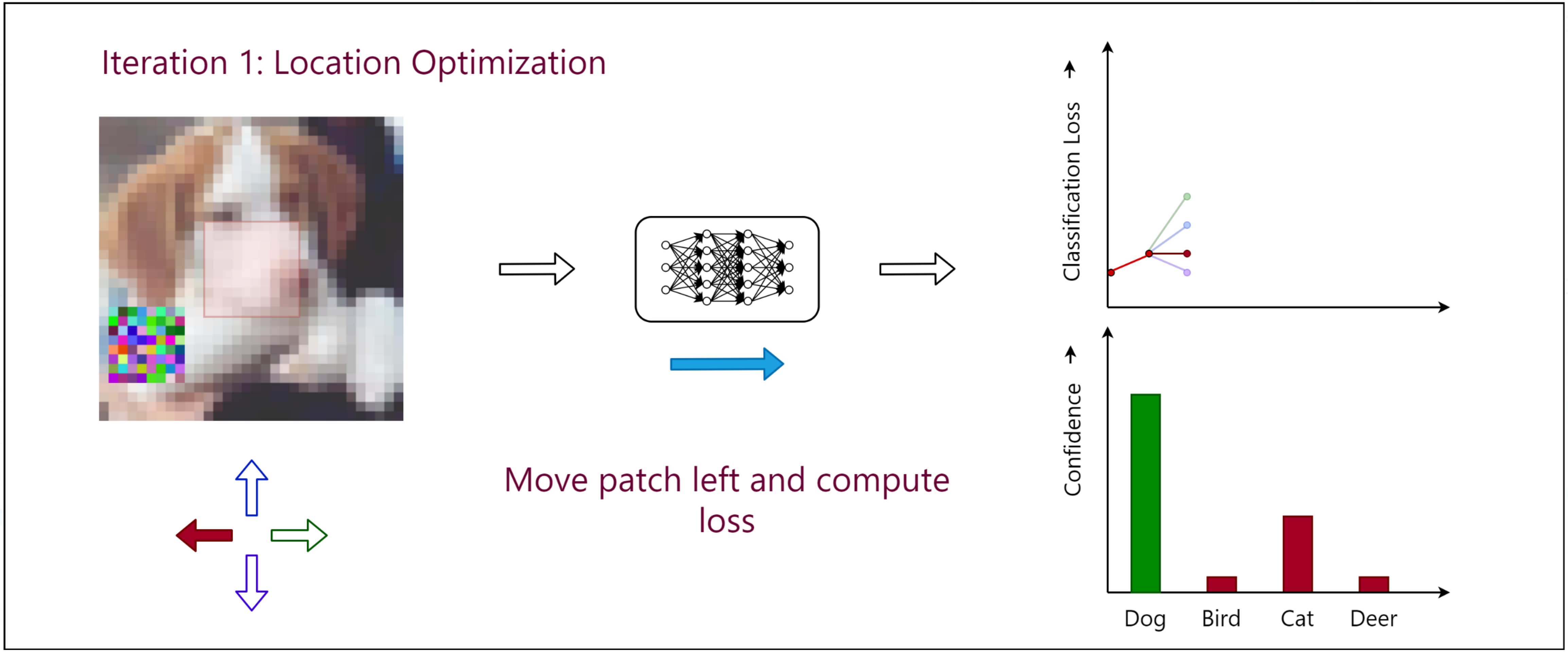
Adversarial Patch Attack: Location Optimization



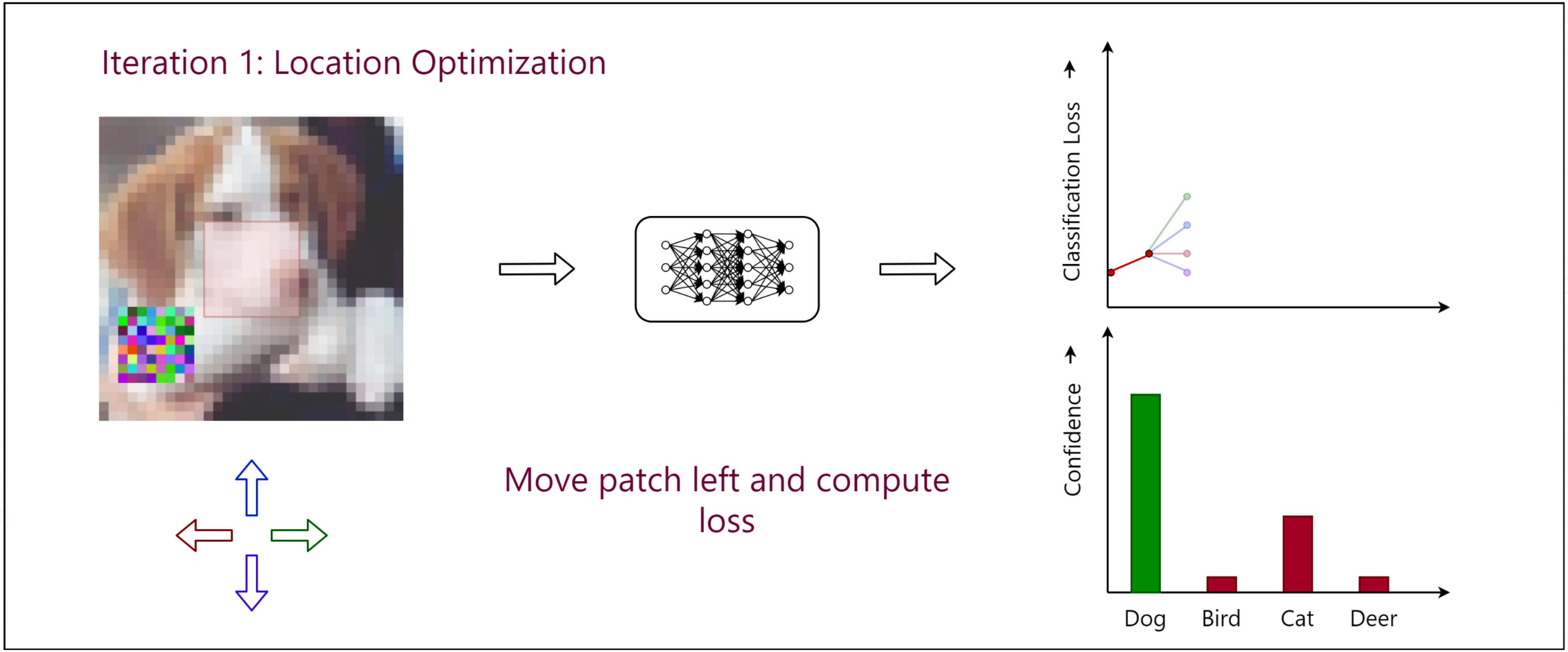
Adversarial Patch Attack: Location Optimization



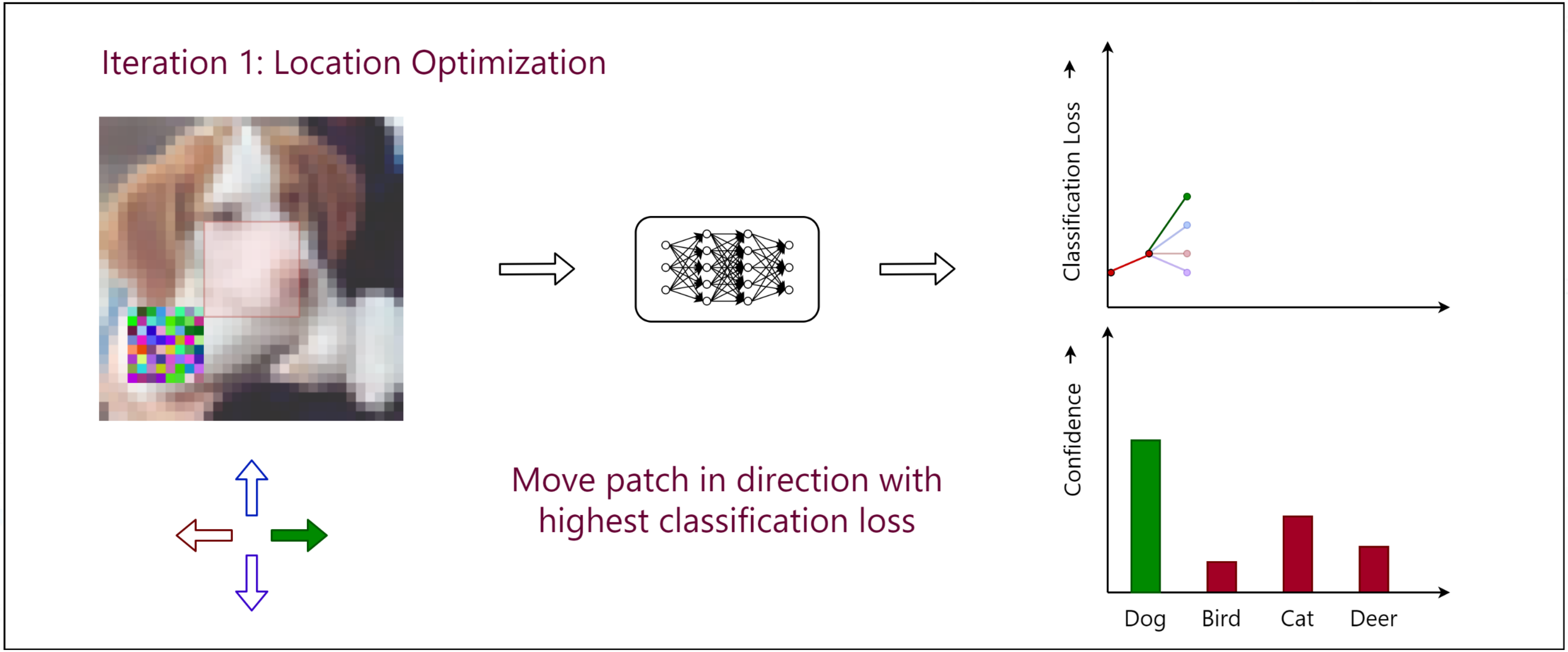
Adversarial Patch Attack: Location Optimization



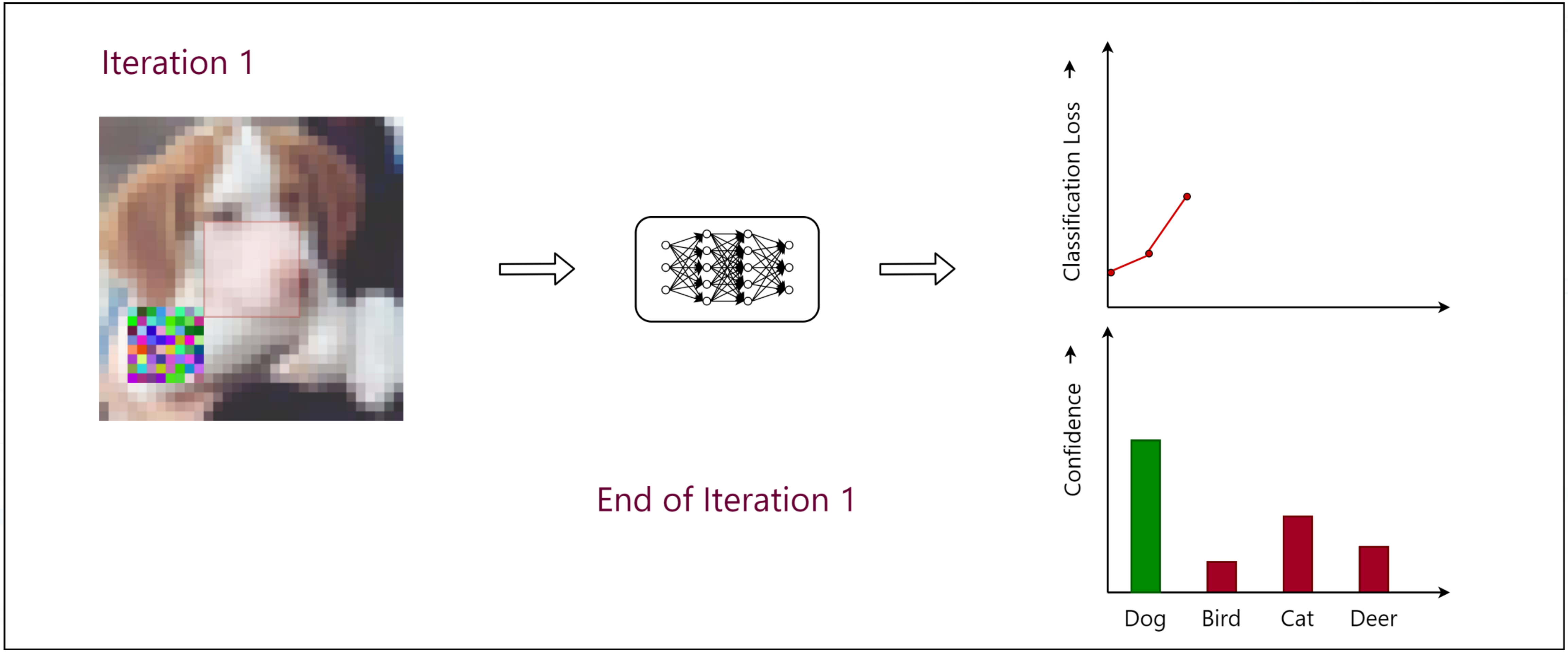
Adversarial Patch Attack: Location Optimization



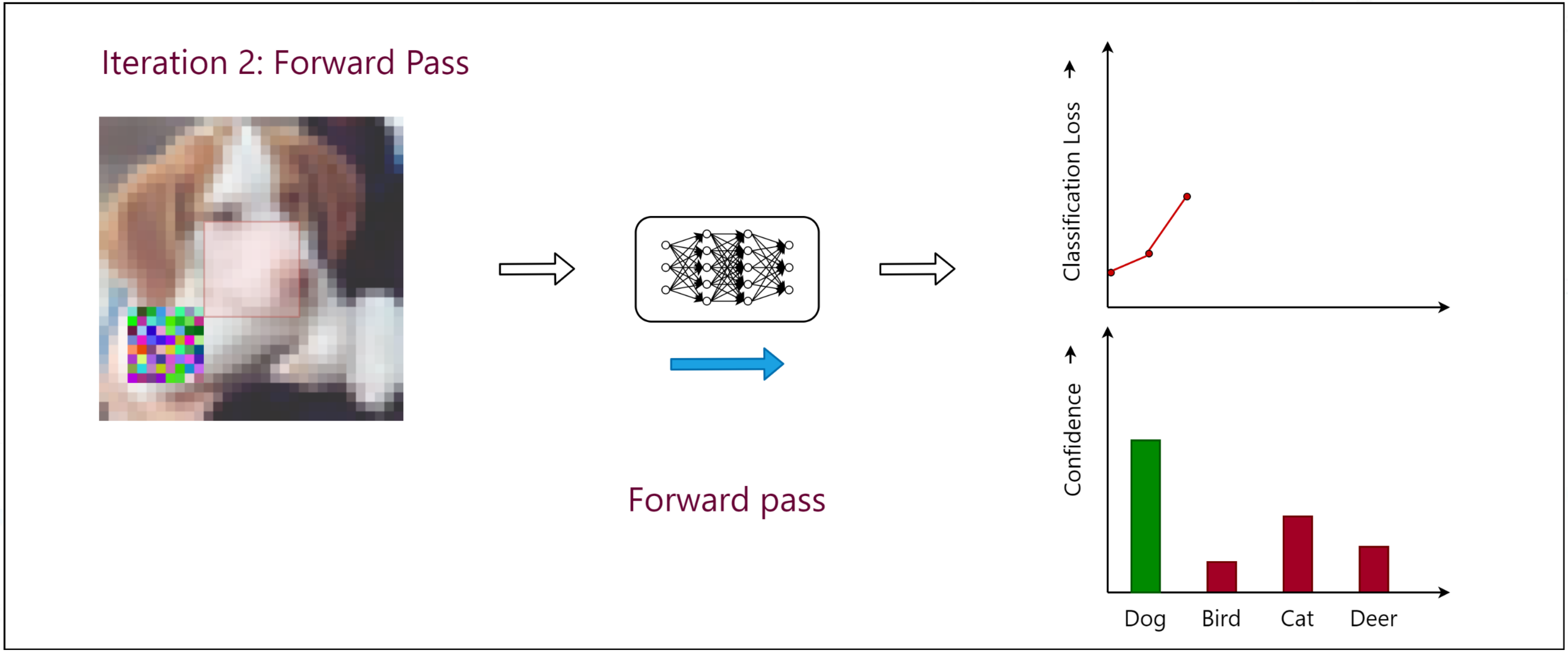
Adversarial Patch Attack: Location Optimization



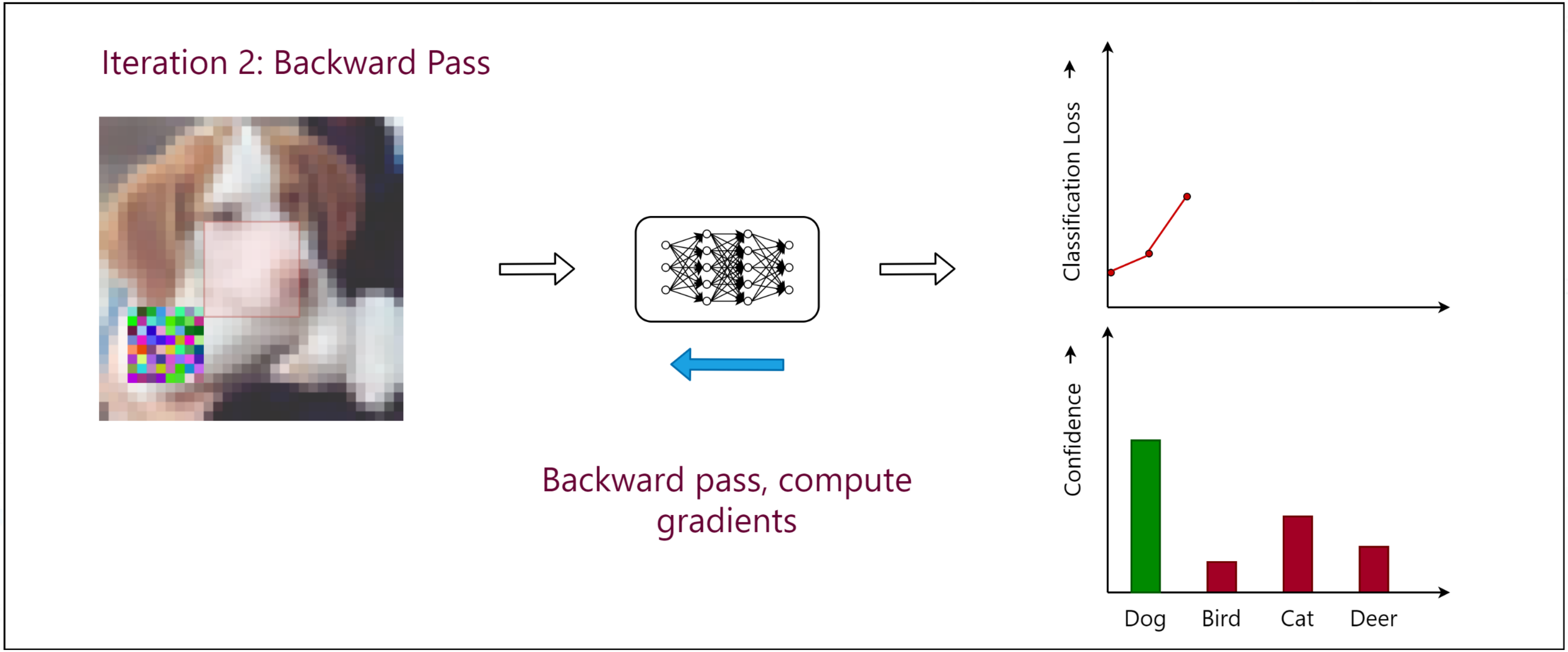
Adversarial Patch Attack



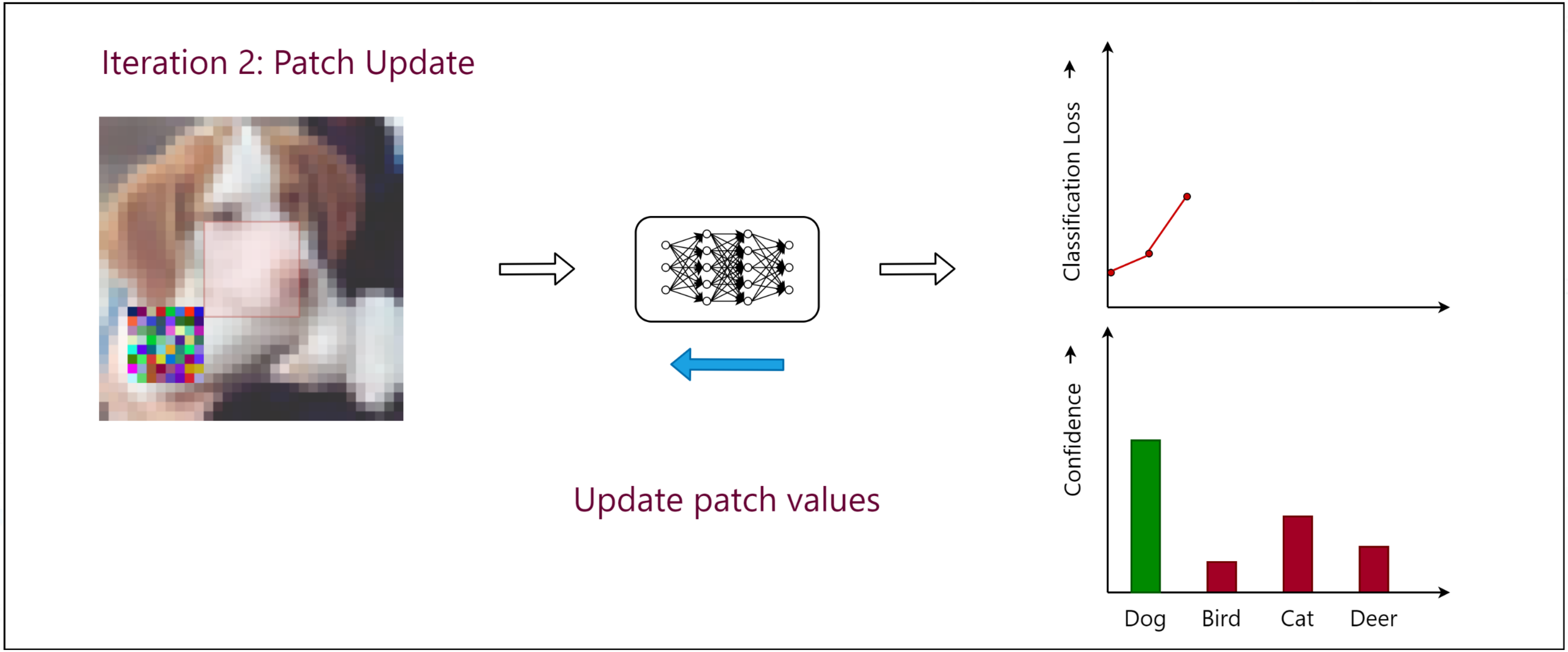
Adversarial Patch Attack: Forward Pass



Adversarial Patch Attack: Backward Pass

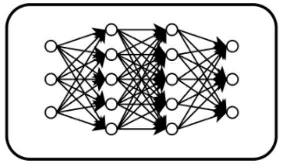


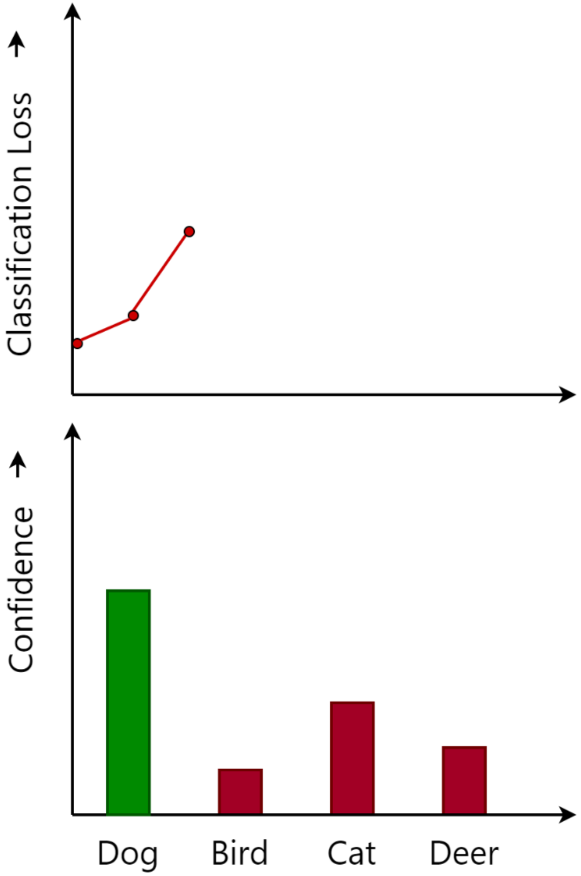
Adversarial Patch Attack: Patch Update



Adversarial Patch Attack: Location Optimization

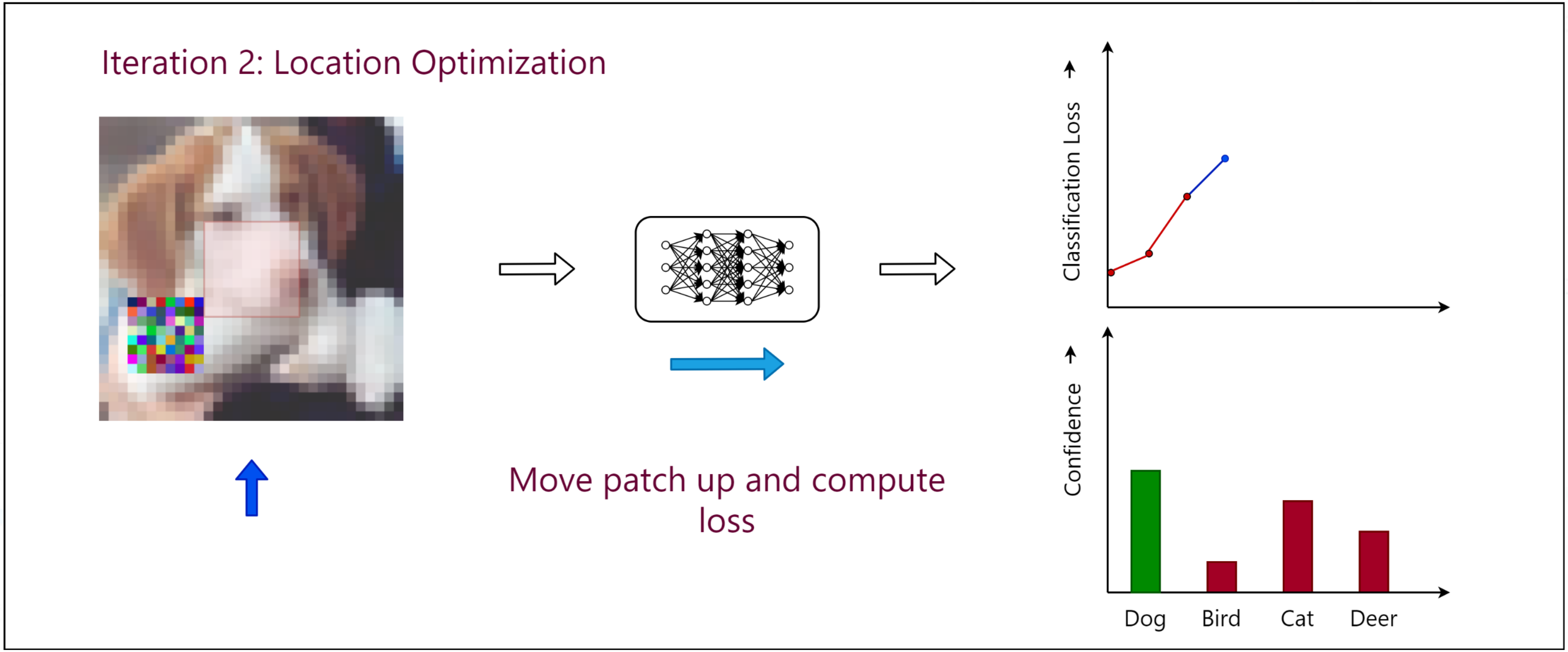
Iteration 2: Location Optimization



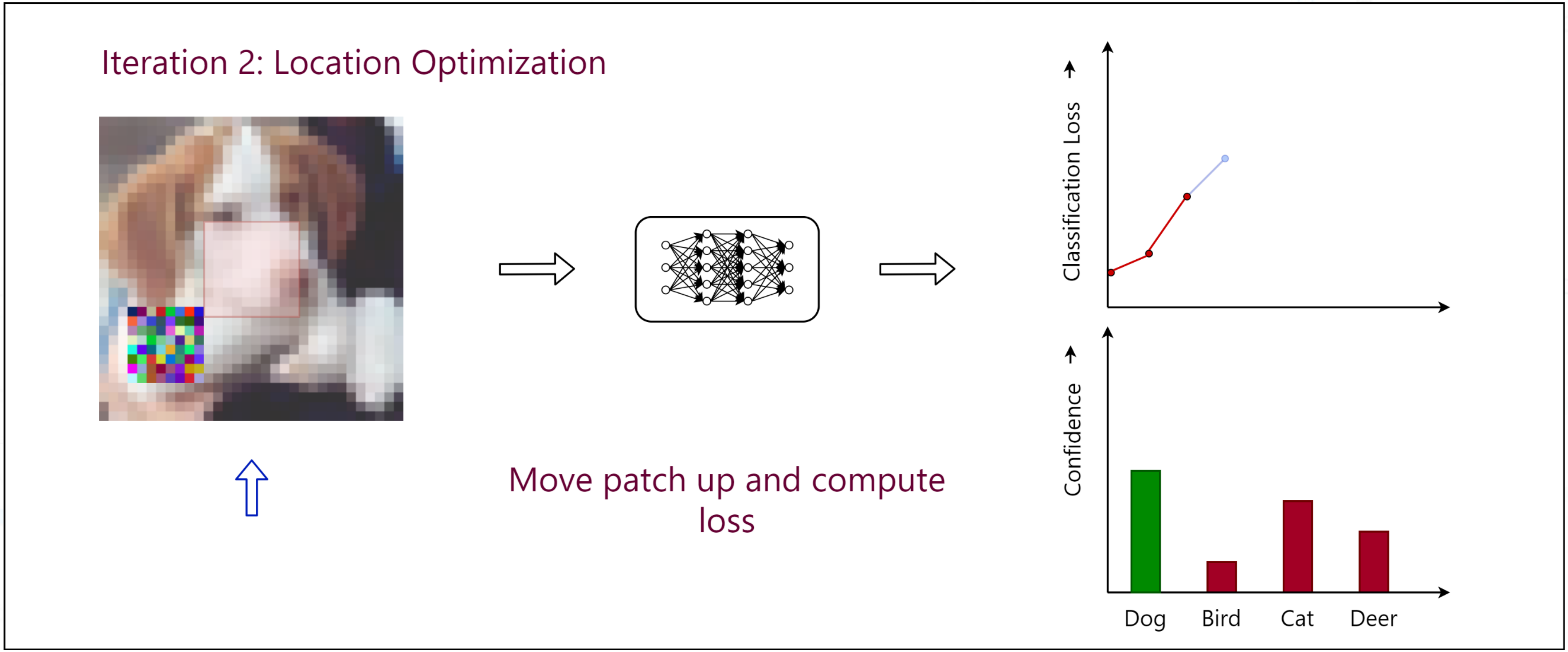


Perform location optimization

Adversarial Patch Attack: Location Optimization

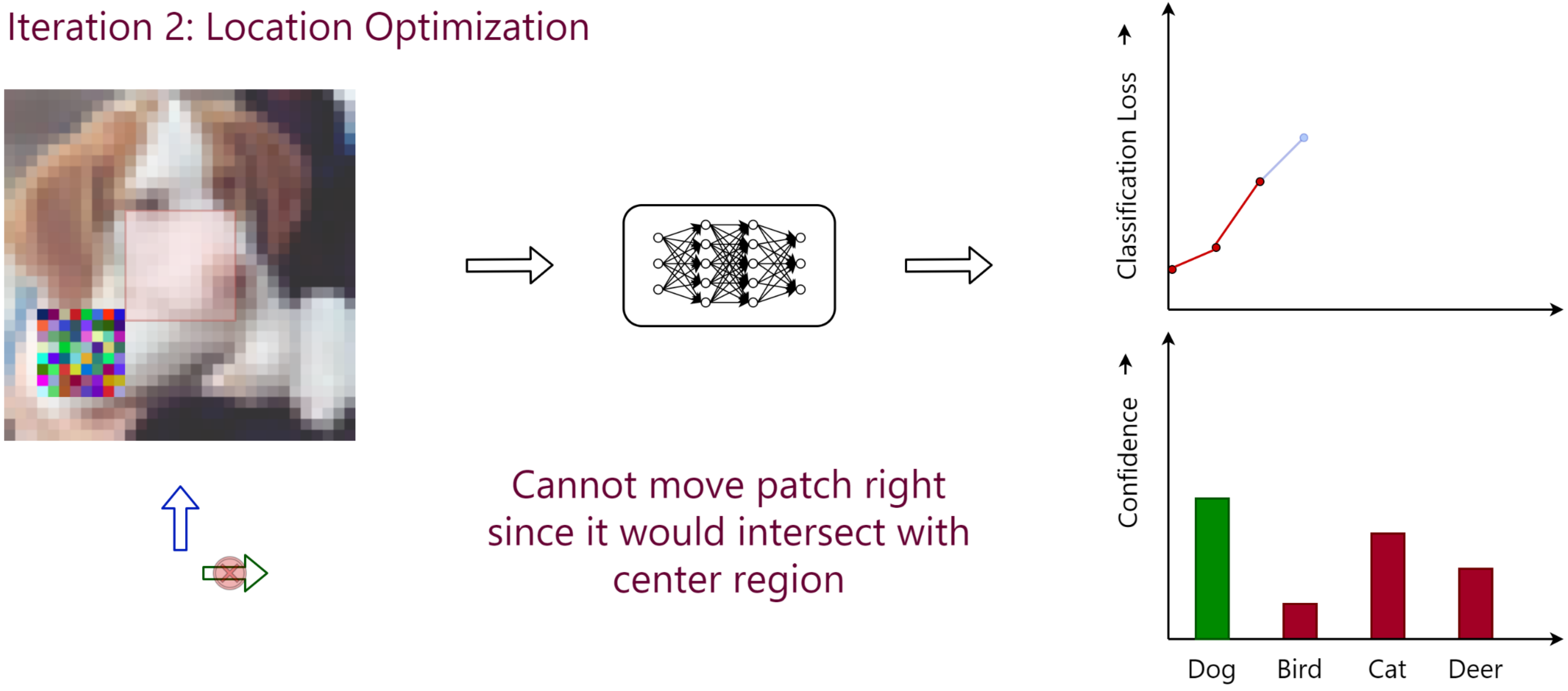


Adversarial Patch Attack: Location Optimization



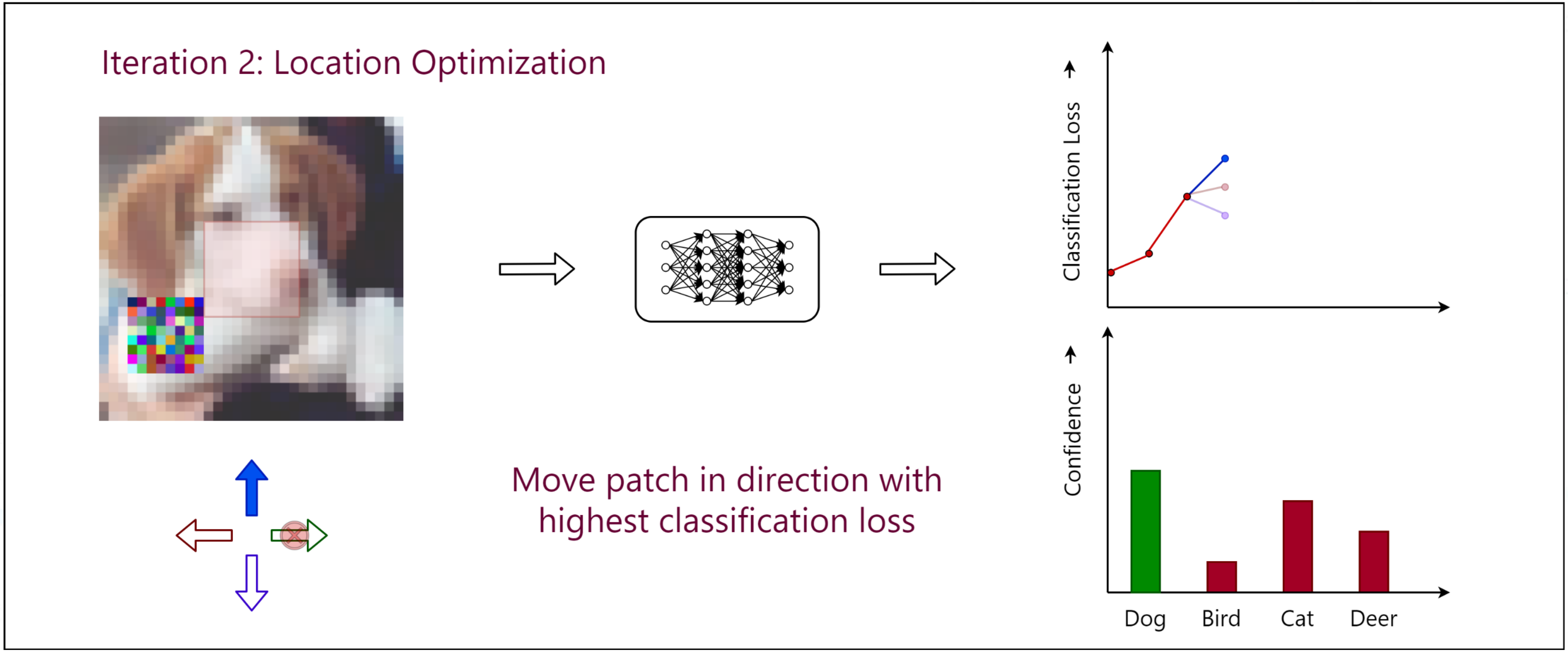
Adversarial Patch Attack: Location Optimization

Iteration 2: Location Optimization

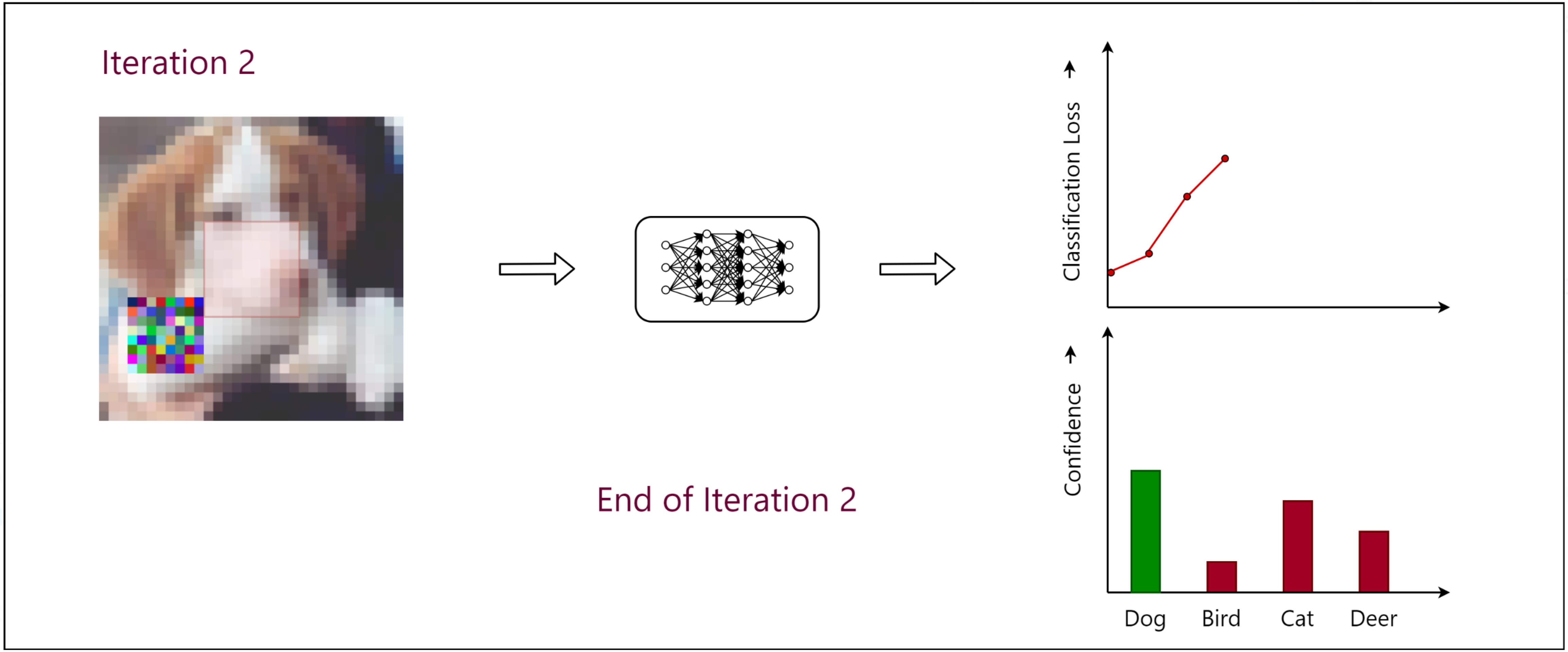


Cannot move patch right since it would intersect with center region

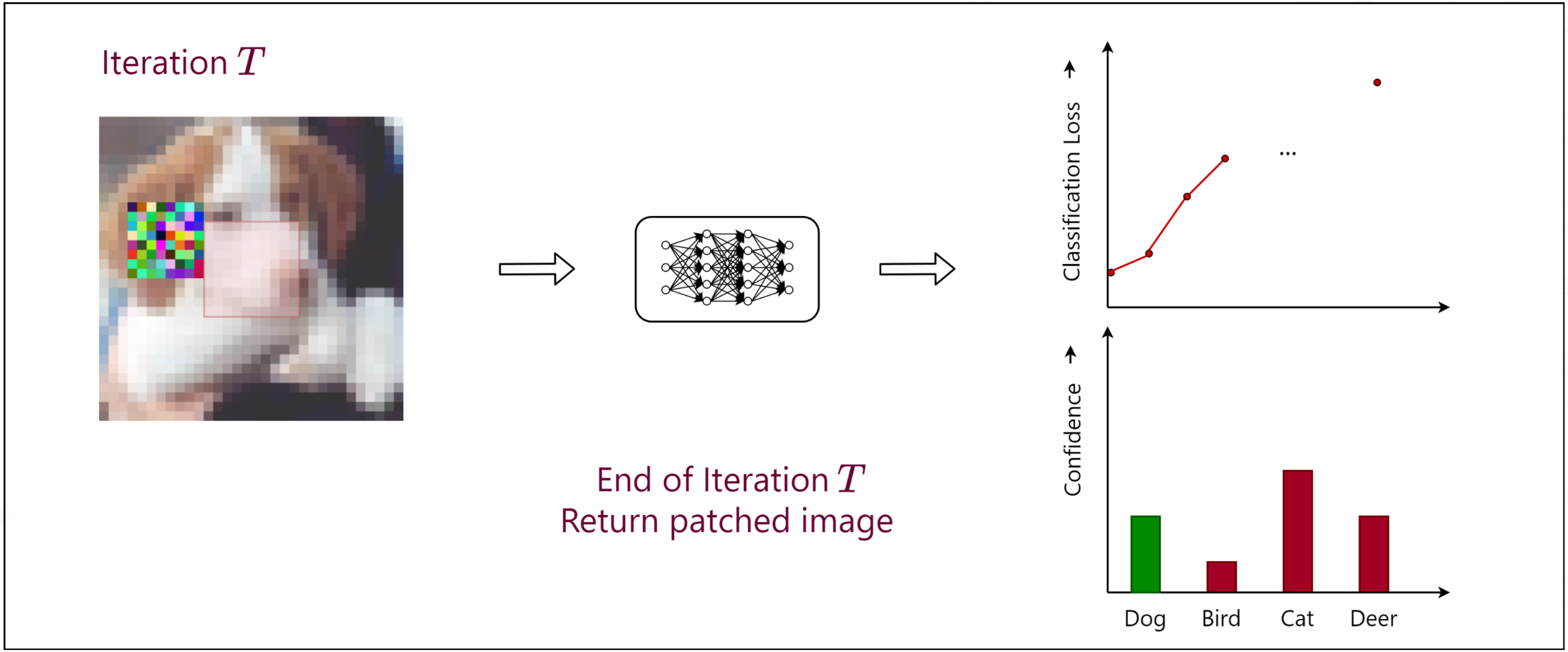
Adversarial Patch Attack: Location Optimization



Adversarial Patch Attack



Adversarial Patch Attack

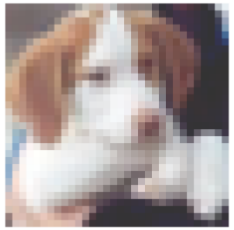


Adversarial Patch Attack: Multiple Attempts

Adversarial Patch Attack: Multiple Attempts

Run attack algorithm multiple times

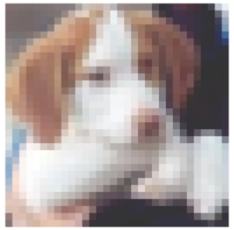
Input Image

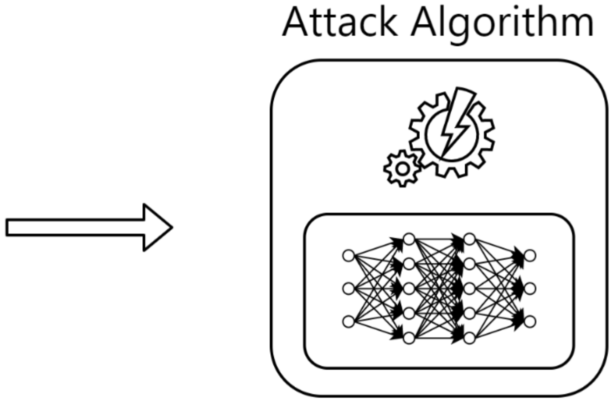


Adversarial Patch Attack: Multiple Attempts

Run attack algorithm multiple times

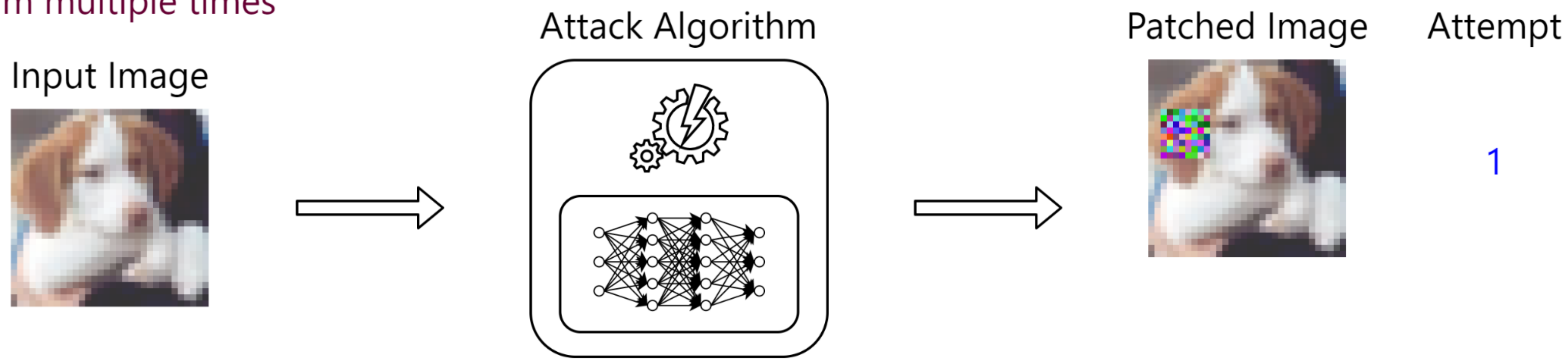
Input Image





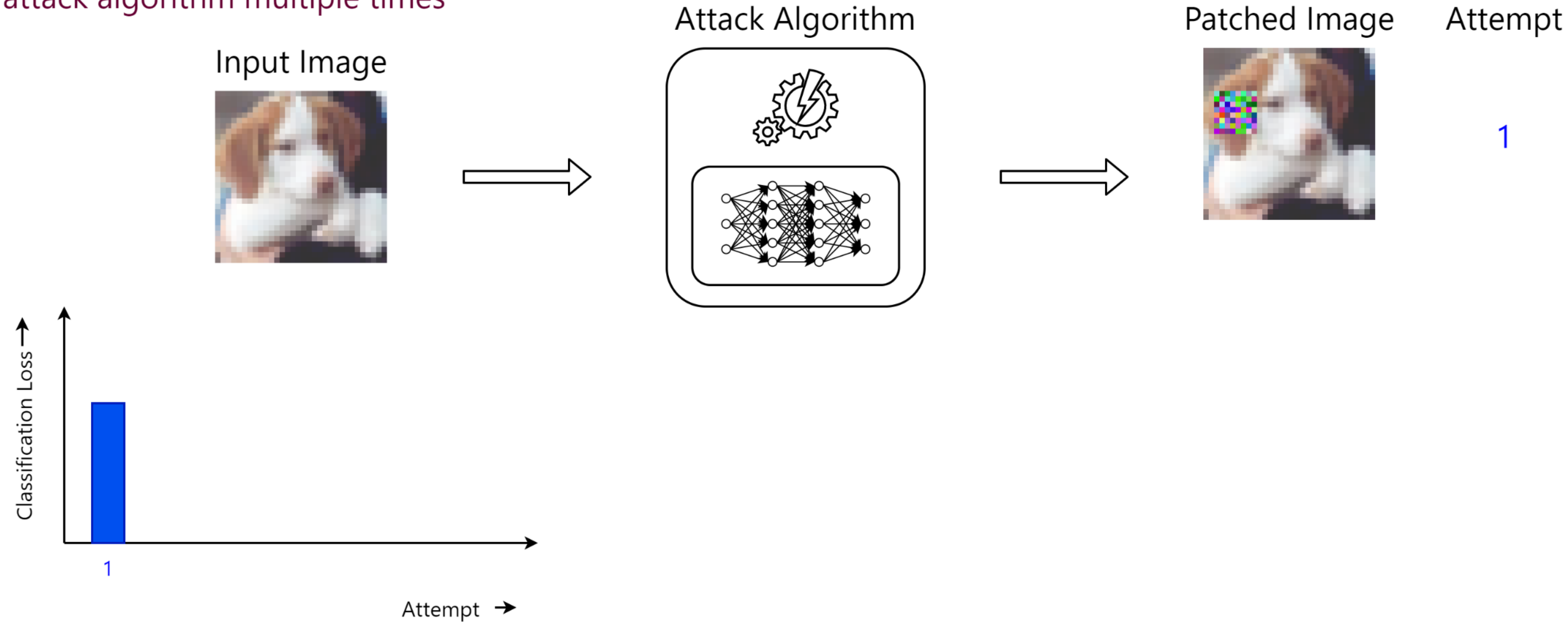
Adversarial Patch Attack: Multiple Attempts

Run attack algorithm multiple times

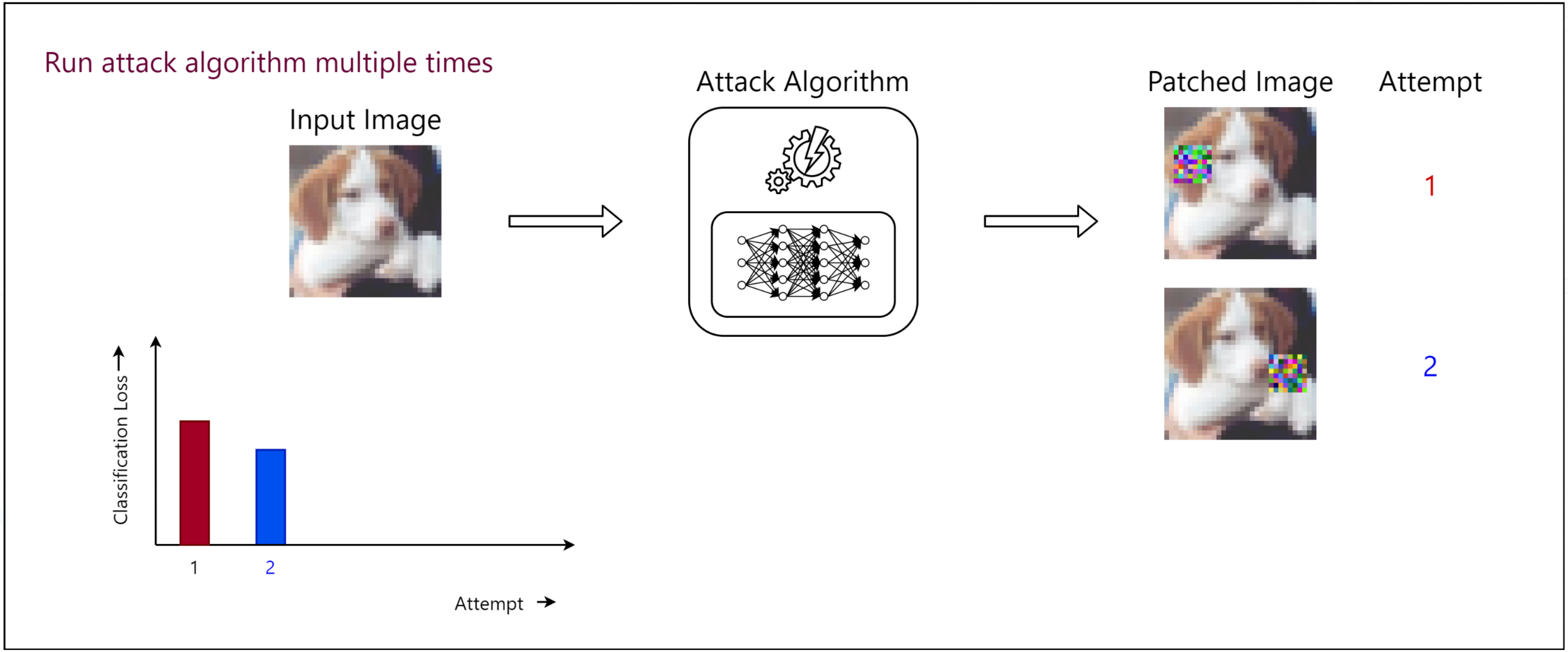


Adversarial Patch Attack: Multiple Attempts

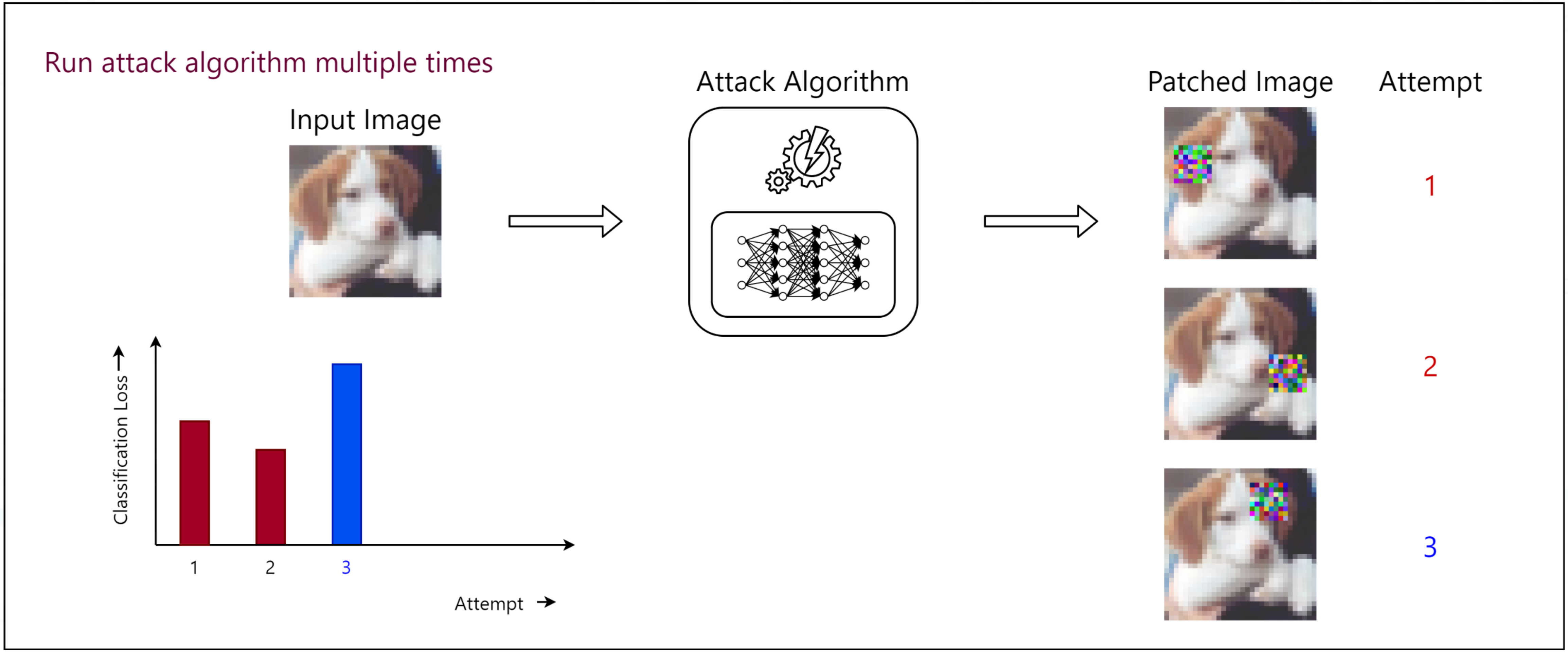
Run attack algorithm multiple times



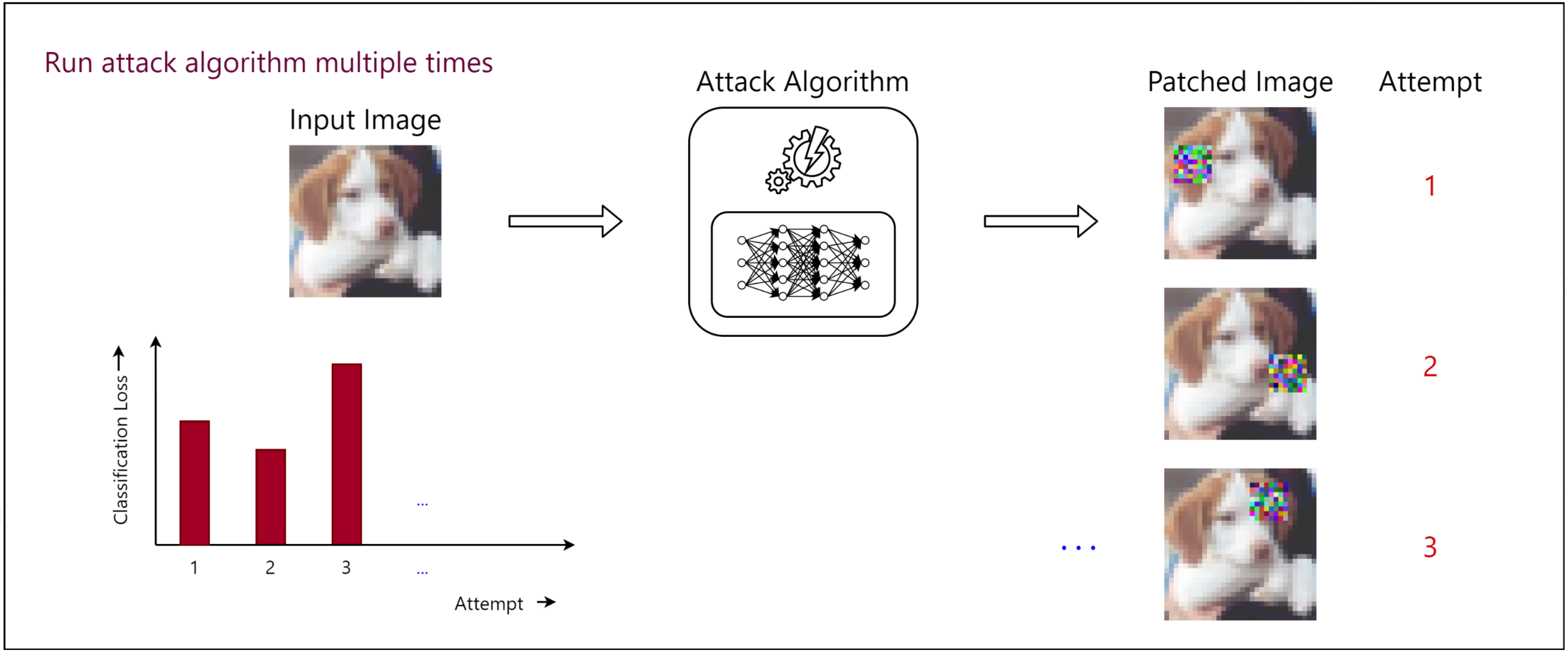
Adversarial Patch Attack: Multiple Attempts



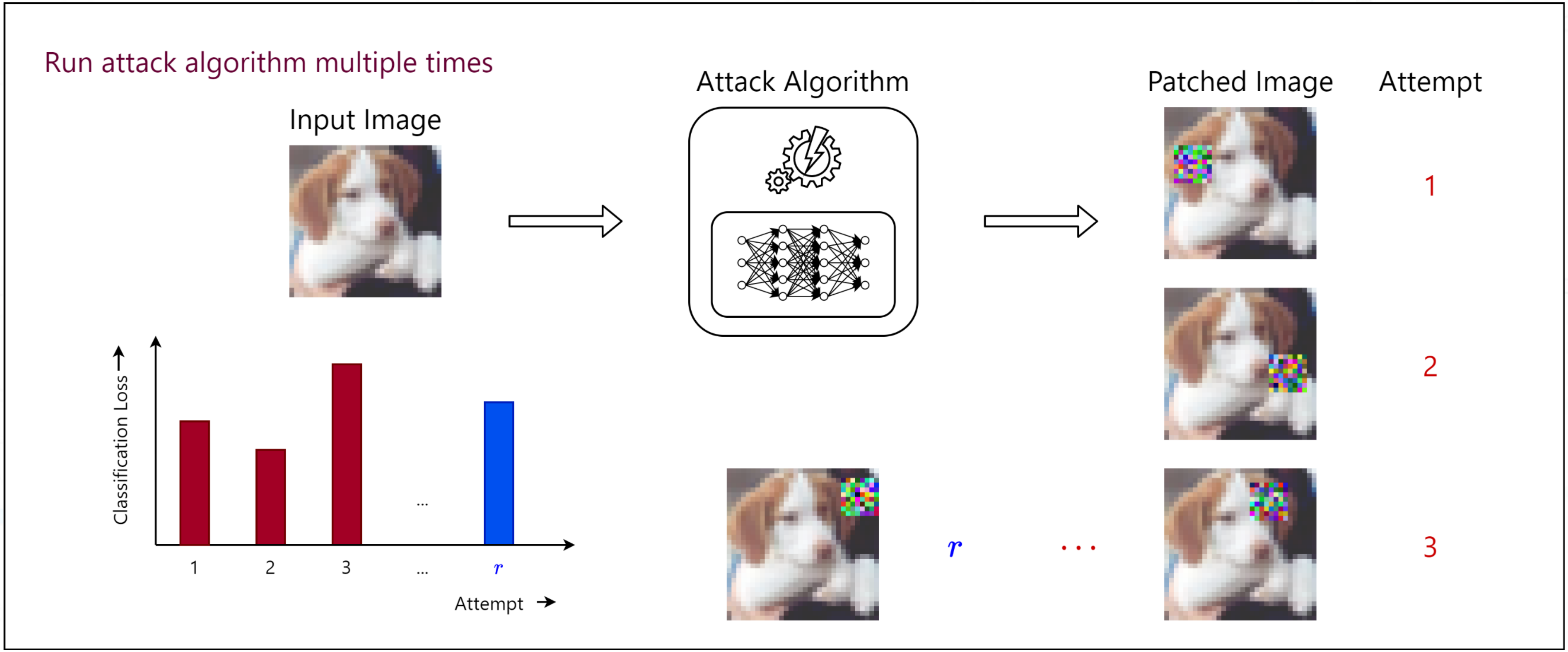
Adversarial Patch Attack: Multiple Attempts



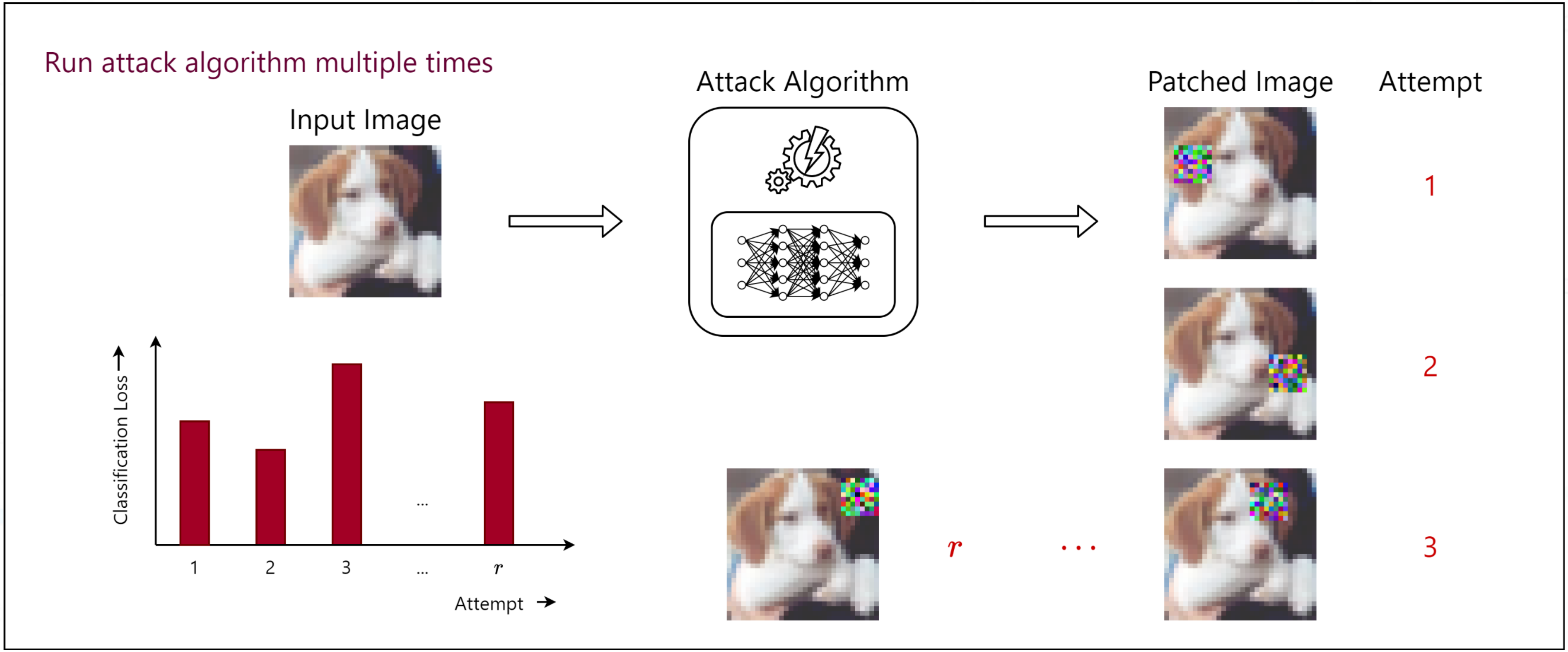
Adversarial Patch Attack: Multiple Attempts



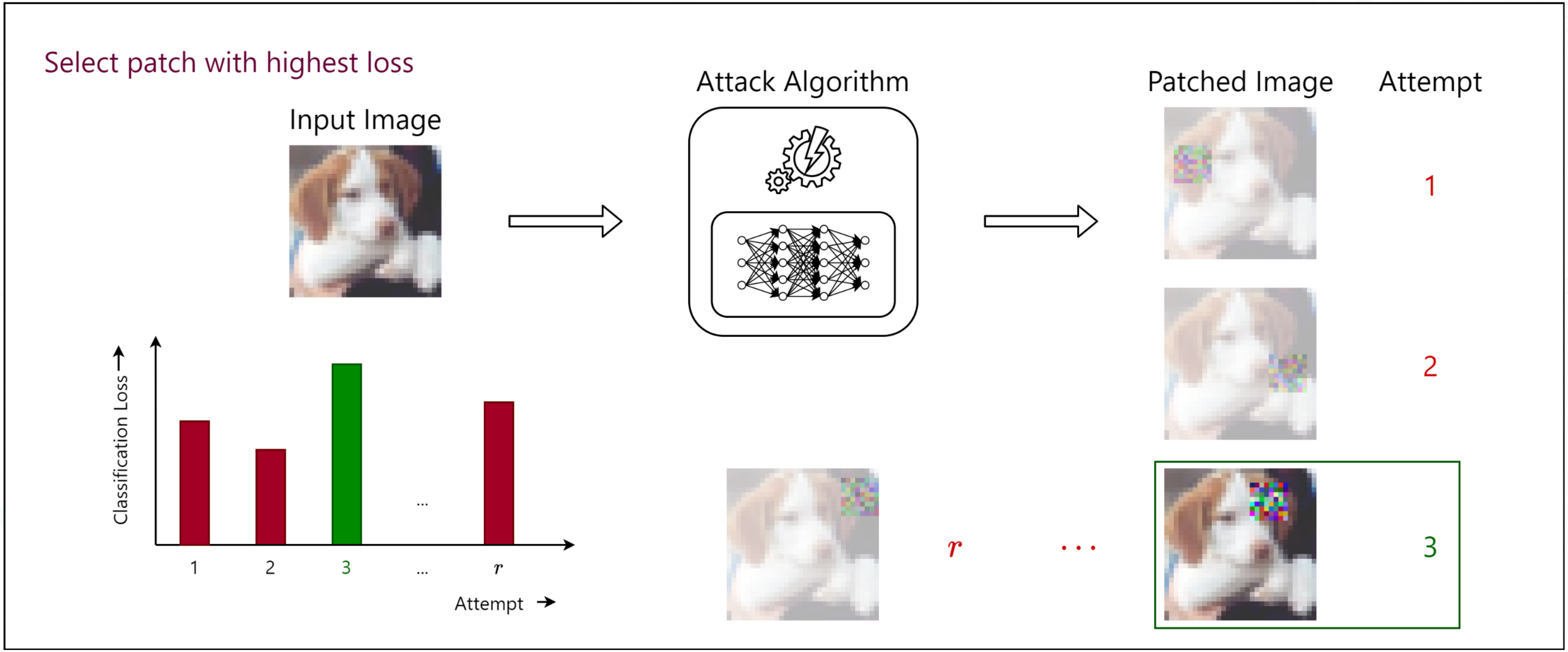
Adversarial Patch Attack: Multiple Attempts



Adversarial Patch Attack: Multiple Attempts



Adversarial Patch Attack: Multiple Attempts



Adversarial Patch Training

Objective: Correctly classify both clean and adversarially patched images

Optimization function:

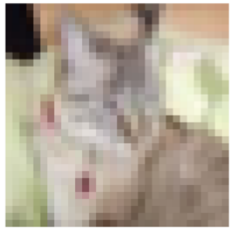
$$\min_{w} \left\{ \underbrace{\mathbb{E}\left[\max_{m,\delta} L(f((1-m)\odot x + m\odot\delta; w), y)\right]}_{\text{Optimize for adversarially patched images (50\% of batch)}} + \underbrace{\mathbb{E}\left[L(f(x;w),y)\right]}_{\text{Optimize for clean images (50\% of batch)}} \right\}$$

Implementation: Attack half the images in each batch when training

Adversarial Patch Training

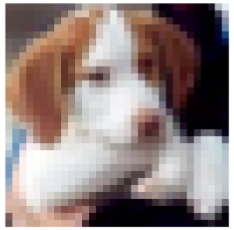
Adversarial Patch Training

Truck



Cat

Frog



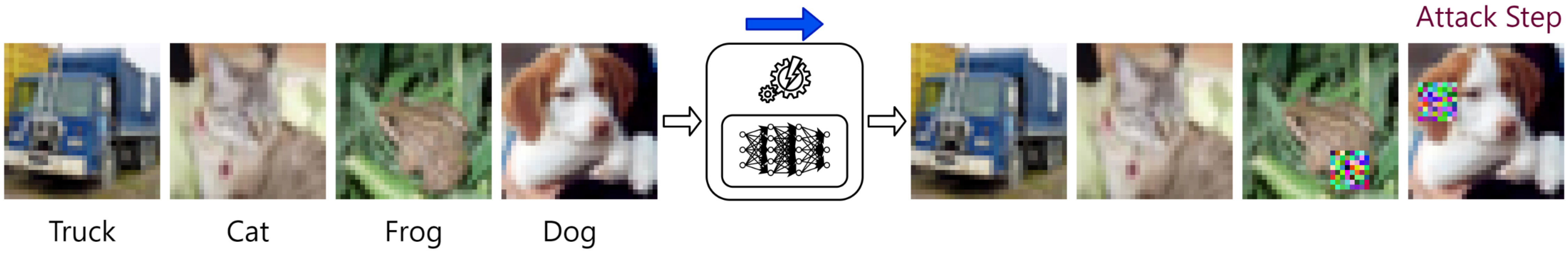
Dog

Adversarial Patch Training

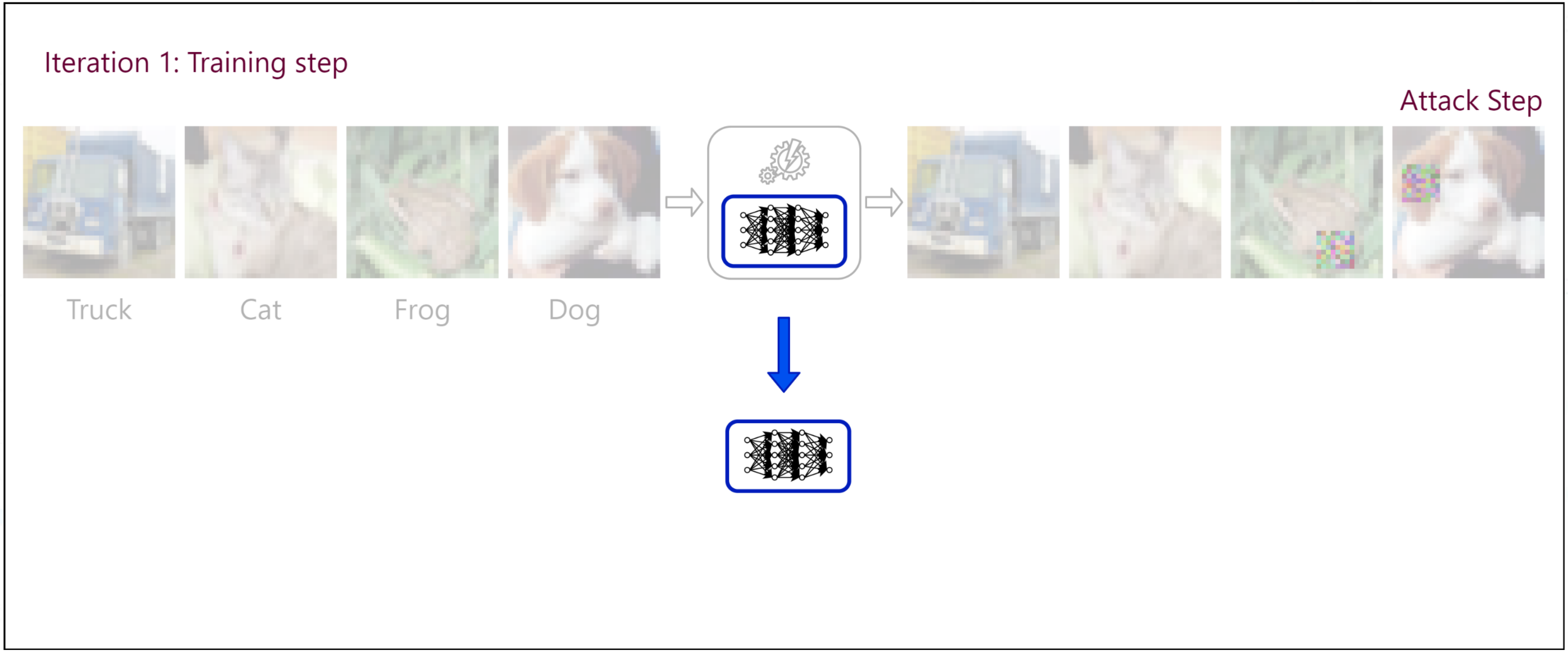
Adversarial Patch Training

Adversarial Patch Training

Iteration 1: Attack half the images in the batch



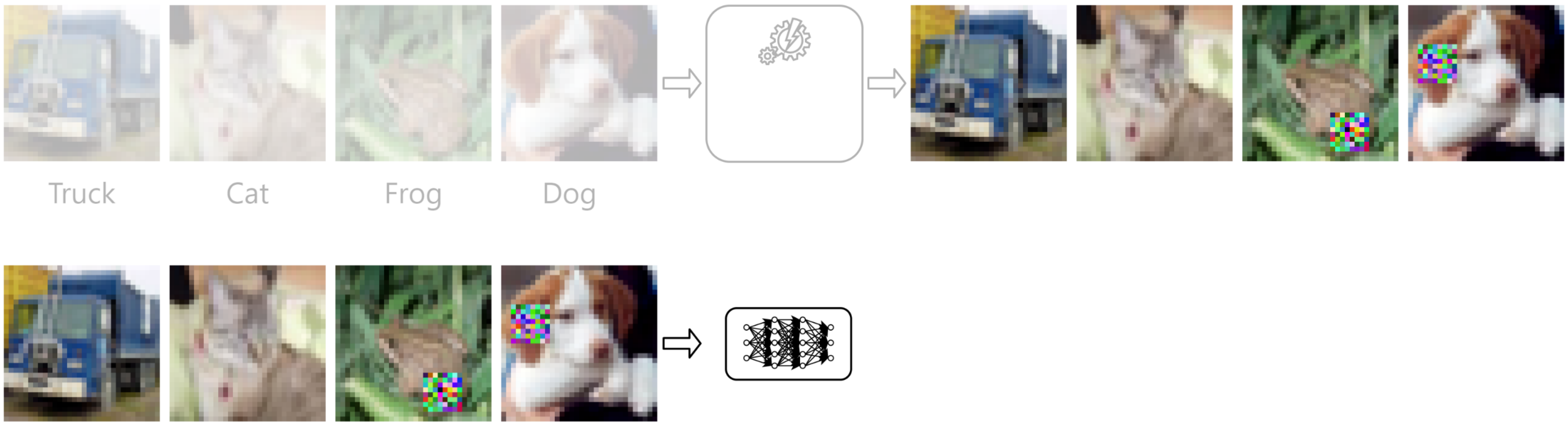
Adversarial Patch Training



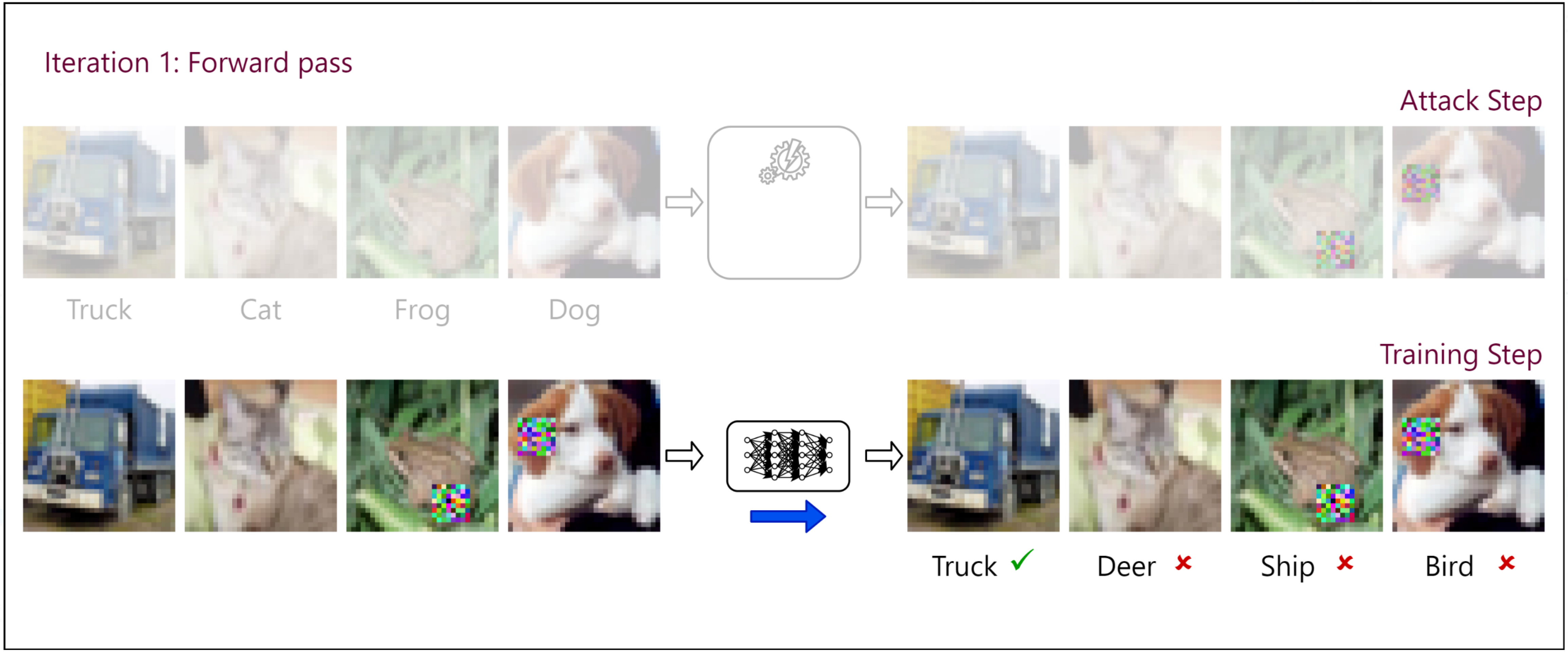
Adversarial Patch Training

Iteration 1: Training step

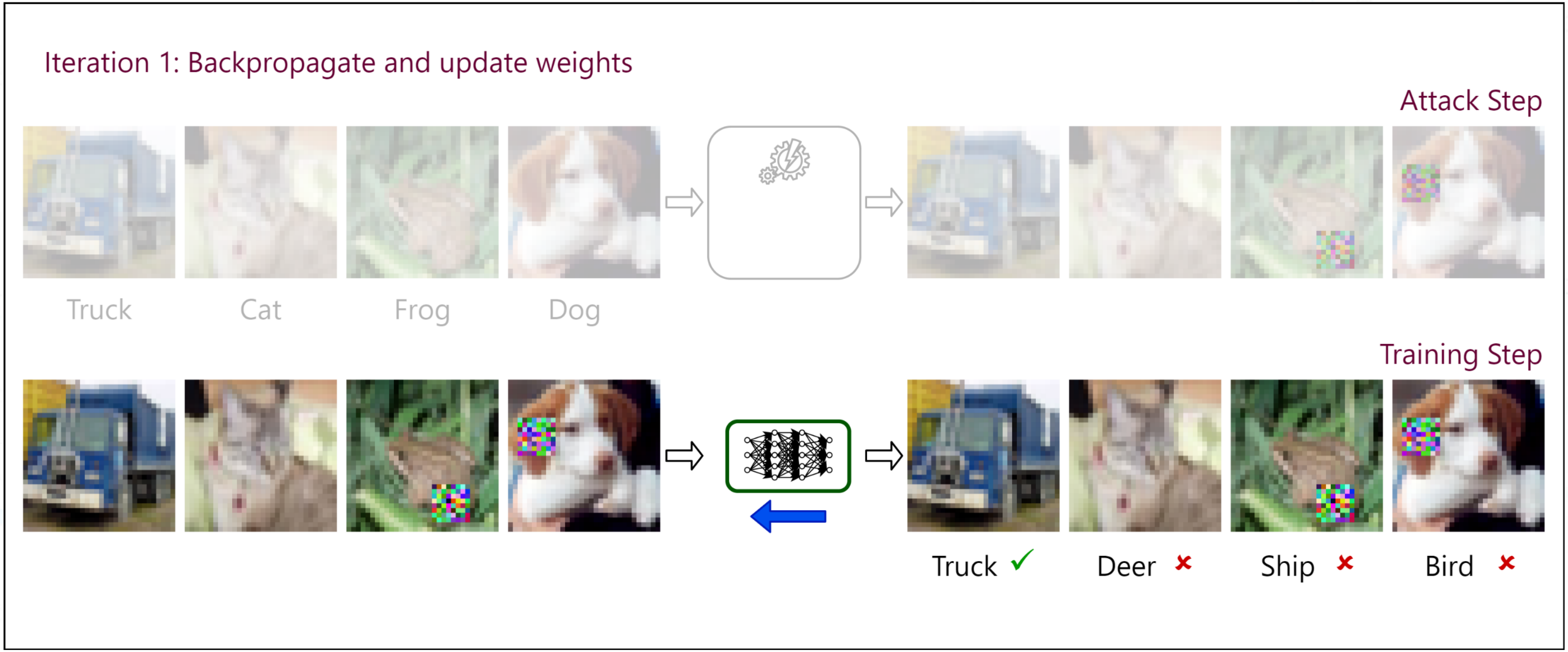
Attack Step



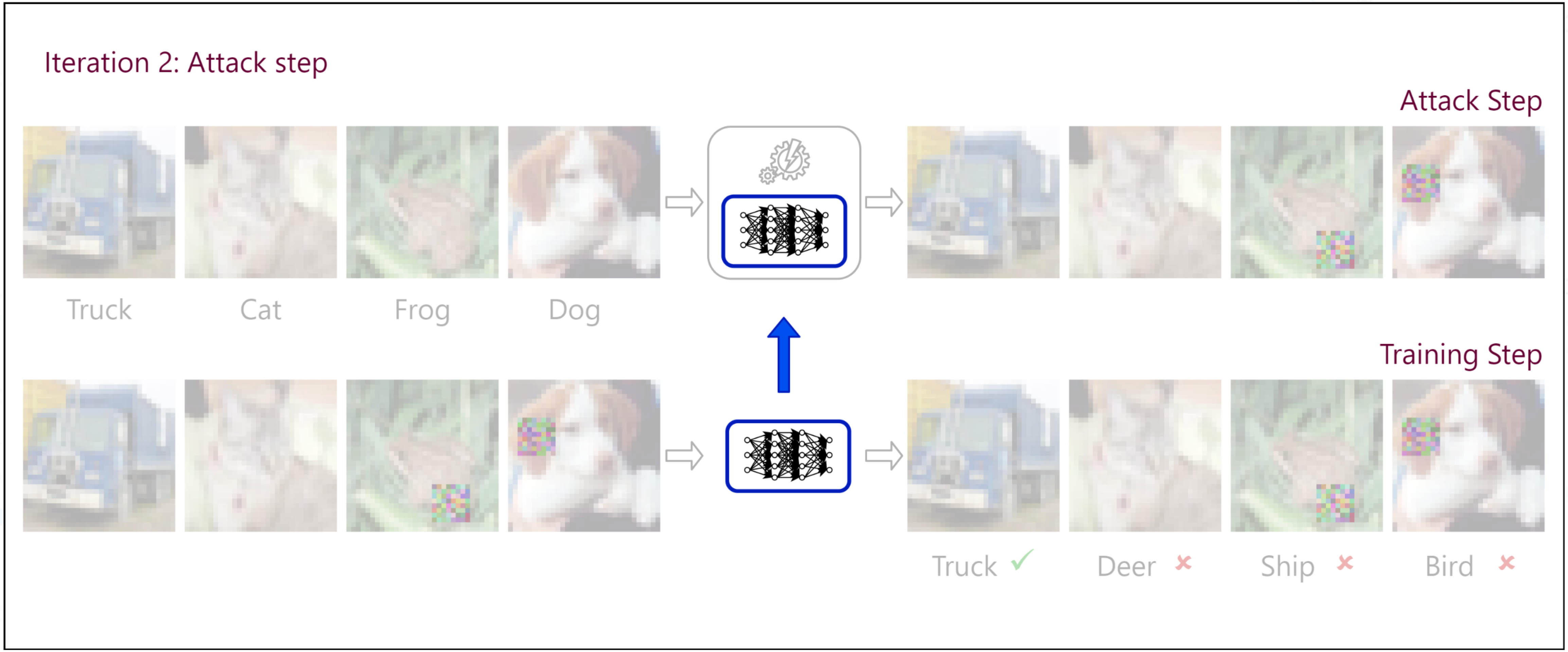
Adversarial Patch Training



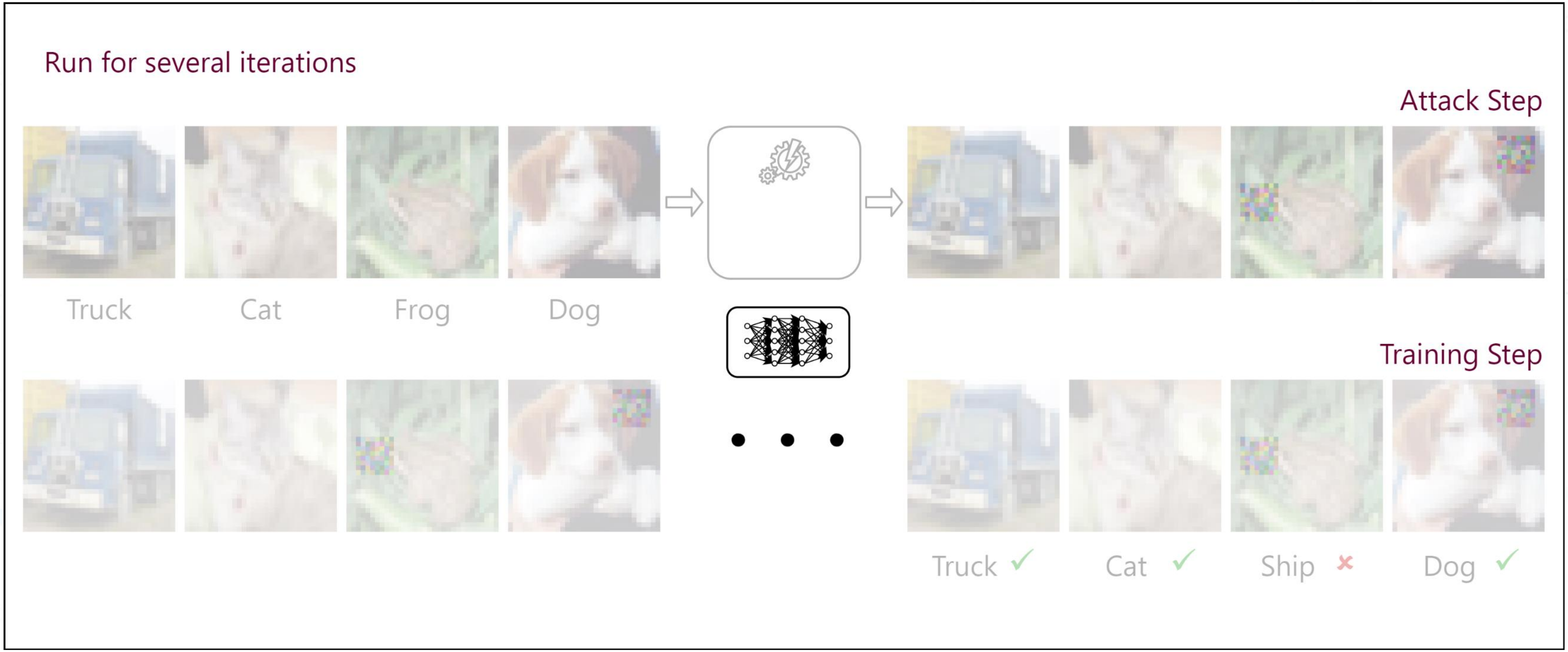
Adversarial Patch Training



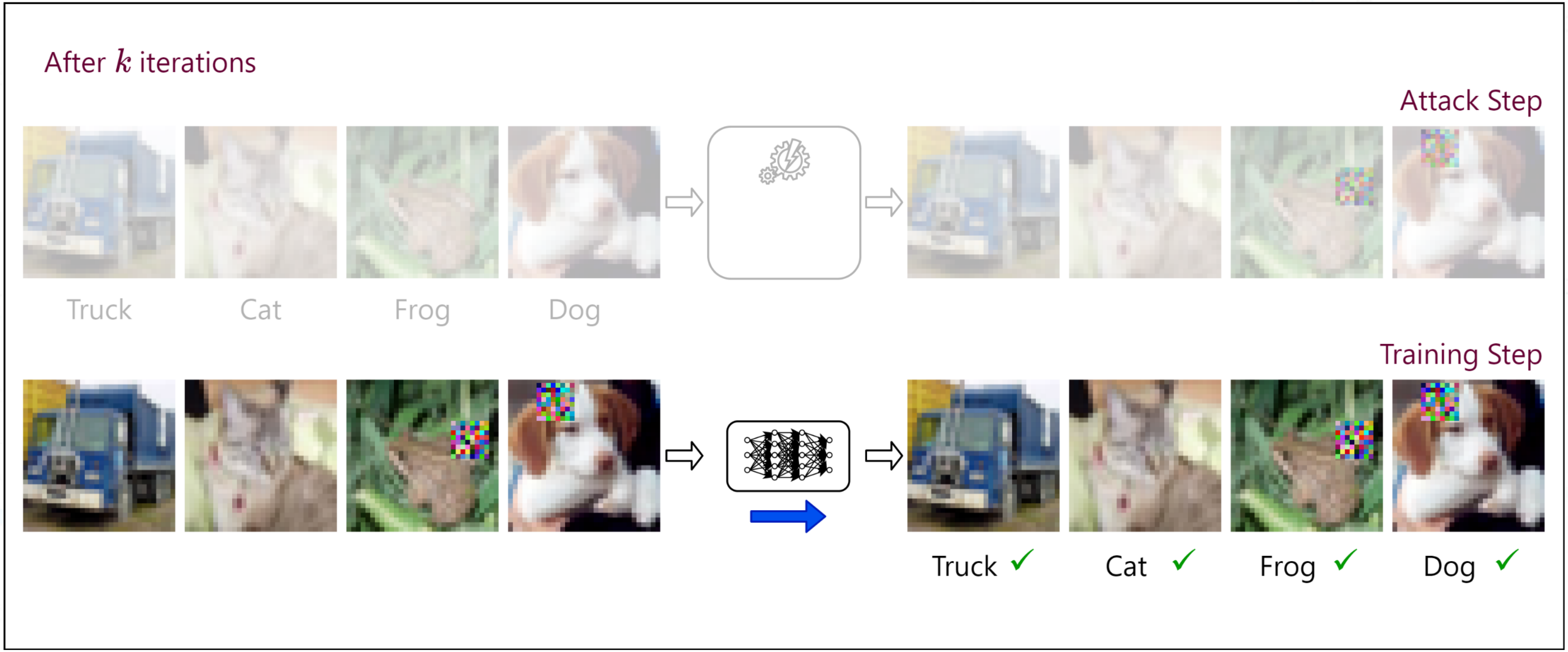
Adversarial Patch Training



Adversarial Patch Training

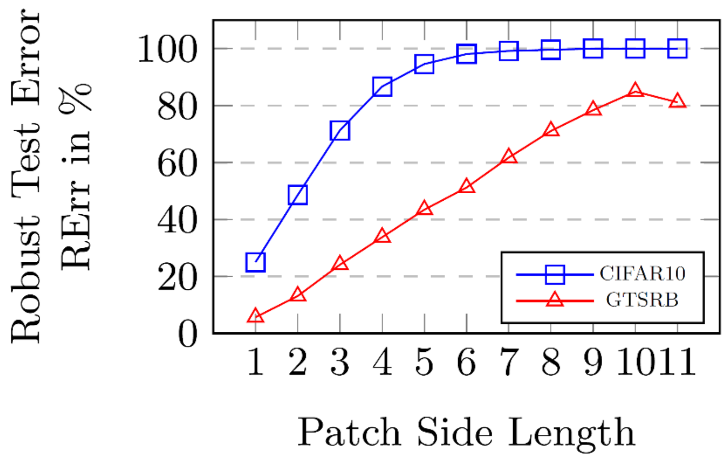


Adversarial Patch Training



Experimental Evaluation

- **Datasets:** CIFAR10, GTSRB
- **Network:** ResNet-20
- **Patch size:** 8 x 8



Attacks:

- Fixed location (AP-Fixed)
- Random location (AP-Rand)
- Random location initialization + random location optimization (AP-RandLO)
- Random location initialization + full location optimization (AP-FullLO)

Experimental Evaluation

Models: one trained per attack type

- Fixed location (AT-Fixed)
- Random location (AT-Rand)
- Random location initialization + random location optimization (AT-RandLO)
- Random location initialization + full location optimization (AT-FullLO)

Attack Effort (#attempts × #iterations):

- Adversarial patch training: 25
- Evaluation of trained models: 3000

Experimental Evaluation: Results

Model \ Attack	AP-Fixed	AP-Rand	AP-RandLO	AP-FullLO
Normal	99.9	100.0	100.0	100.0
AT-Fixed	63.4	82.1	85.5	85.1
AT-Rand	51.0	60.9	61.5	63.3
AT-RandLO	40.4	54.2	60.6	62.8
AT-FullLO	**27.9**	**39.6**	**44.2**	**45.1**

Robust Test Error (%) on CIFAR10

Experimental Evaluation: Results

Model \ Attack	AP-Fixed	AP-Rand	AP-RandLO	AP-FullLO
Normal	99.9	100.0	100.0	100.0
AT-Fixed	63.4	82.1	85.5	85.1
AT-Rand	51.0	60.9	61.5	63.3
AT-RandLO	40.4	54.2	60.6	62.8
AT-FullLO	**27.9**	**39.6**	**44.2**	**45.1**

Robust Test Error (%) on CIFAR10

Experimental Evaluation: Results

Model \ Attack	AP-Fixed	AP-Rand	AP-RandLO	AP-FullLO
Normal	99.9	100.0	100.0	100.0
AT-Fixed	63.4	82.1	85.5	85.1
AT-Rand	51.0	60.9	61.5	63.3
AT-RandLO	40.4	54.2	60.6	62.8
AT-FullLO	**27.9**	**39.6**	**44.2**	**45.1**

Robust Test Error (%) on CIFAR10

Experimental Evaluation: Results

Model \ Attack	AP-Fixed	AP-Rand	AP-RandLO	AP-FullLO
Normal	99.9	100.0	100.0	100.0
AT-Fixed	63.4	82.1	85.5	85.1
AT-Rand	51.0	60.9	61.5	63.3
AT-RandLO	40.4	54.2	60.6	62.8
AT-FullLO	**27.9**	**39.6**	**44.2**	**45.1**

Robust Test Error (%) on CIFAR10

Experimental Evaluation: Results

Model	Clean Test Error
Normal	9.7
AT-Fixed	10.1
AT-Rand	9.1
AT-RandLO	8.7
AT-FullLO	8.8

Clean Test Error (%) on CIFAR10

Experimental Evaluation: Heatmaps

Adversarial patch training reduces the region where attack is successful

Normal AT-Fixed AT-Rand AT-RandLO AT-FullLO

Summary

- Proposed adversarial patch attack with location optimization
- Location optimization strengthens attack
- Adversarial patch training with location-optimized patches improves model robustness

Resources:

- Paper: https://arxiv.org/abs/2005.02313
- Code: https://github.com/sukrutrao/adversarial-patch-training
- Contact: sukrut.rao@mpi-inf.mpg.de