AI Security

Adversarial machine learning

- A classifier can possibly be tricked by malicious input
  - Either during training or live
- Carefully crafted <u>live</u> input can fool the classifier
- Malicious <u>training</u> input can cause the classifier to learn incorrectly
- Deep learning classifiers are surprisingly vulnerable to these attacks
  - "Discontinuity" in input space

Adversarial machine learning

- Asset: Classifier (costs a lot to train!)
- Threat:
  - Loss of classification accuracy
- Attacker:
  - Someone who does not want their instances to be classified correctly
  - Someone trying to sabotage the classifier

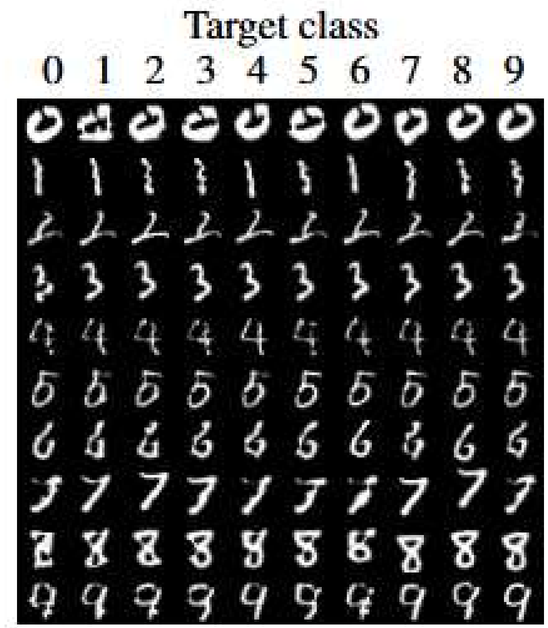
Live input: Adversarial perturbations

Scenario:

1. A classifier (usually image classifier) is trained
2. Attacker has access to the classifier's outputs, cannot affect the classifier
3. Attacker has instances that are correctly classified as class A, but she wants them to be classified as class B (or any class)
4. By repeatedly querying the classifier, the attacker perturbs the instances minimally to achieve her goal

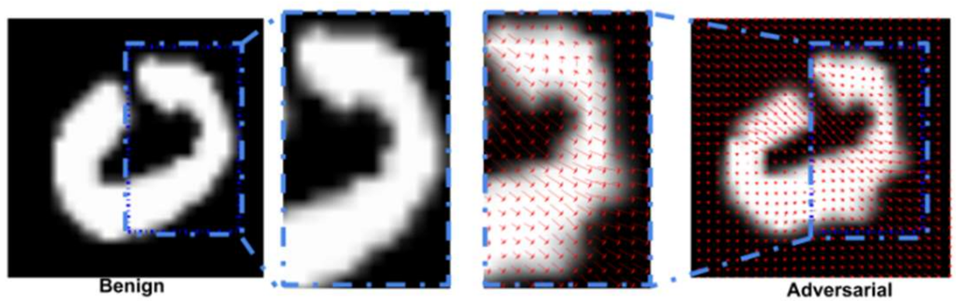
Spatially transformed adversarial examples

- Xiao et al. (2018) used minimal spatial transformations to trick classifiers
  - Black box attack: no access to gradients required
  - Only requires access to classifier output and confidence
- Spatial transformations minimize visual changes
  - Lateral movement of pixels along a flow
- Nearly 100% success rate on MNIST



Spatially transformed adversarial examples

- Adversarial perturbations are found by optimizing over an objective function that minimizes flow distance
- Optimization algorithm is L-BFGS solver
- Also confirmed effective against human perception



One pixel attack

- Su et al. (2019) found that convolutional image classifiers can be fooled by changing only **one pixel** on an image
  - Also a black box attack
- 68.7% chance in non-targeted scenario
- 19.8% chance in targeted scenario

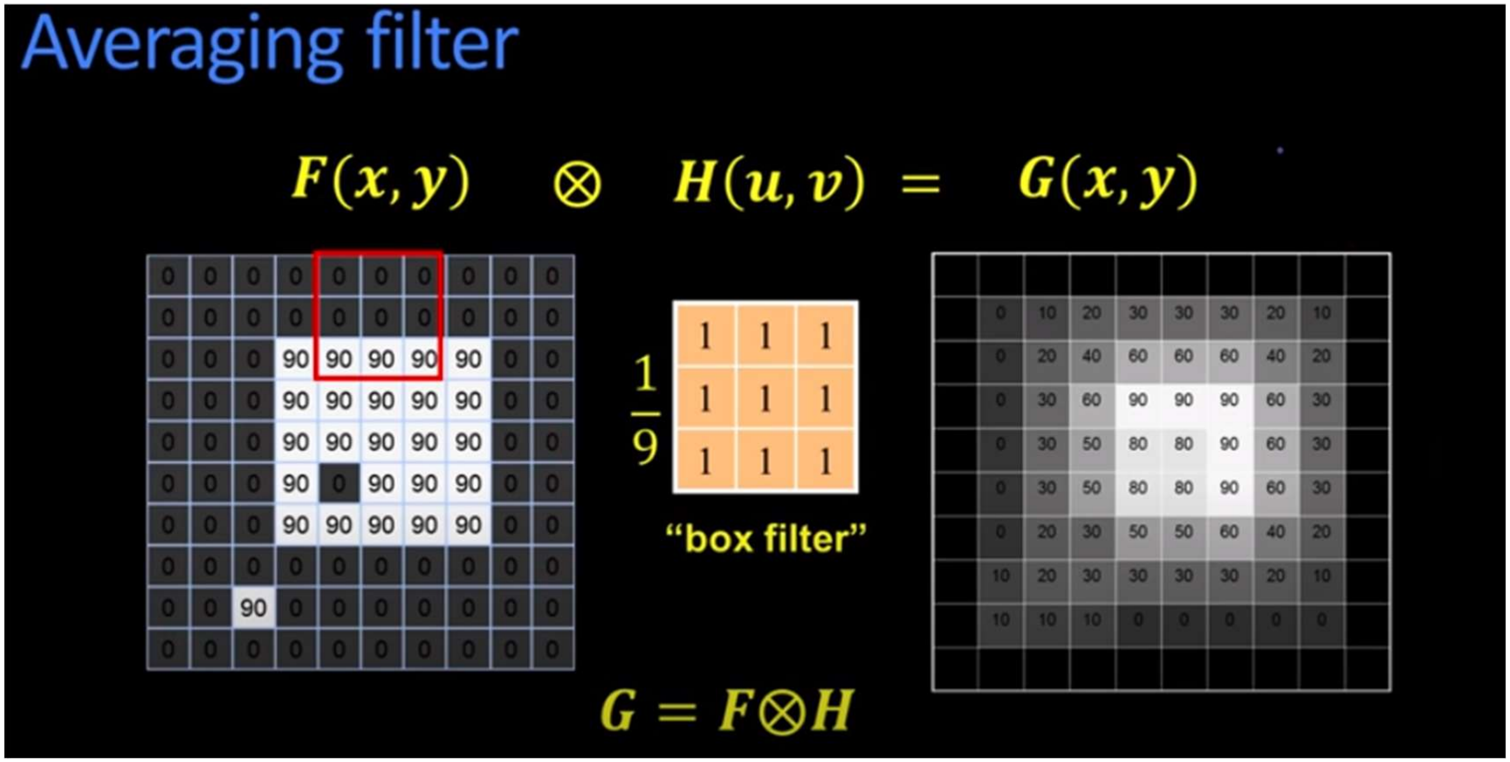
Transferability of adversarial perturbation

- For adversarial perturbations, transferability refers to the ability of a perturbation fooling classifier A to also fool classifier B
- Liu et al. (2017) developed an approach using ensemble learning
  - Intuition: if several known classifiers can be fooled with the same perturbation, then unseen classifiers should also be fooled as well
- Studied perturbations were transferable if non-targeted, but were not transferable if targeted

Defense: Adversarial training

- Training a classifier to defeat adversarial perturbations
- Recall that the classifier cannot know what perturbations will occur
  - Can be formulated as empirical risk minimization
- Ensemble adversarial training: Use transferable samples to the defense’s advantage
  - Transferable samples can be thought of the most powerful samples that we need to defend against
- Mean blur defense: Add a averaging filter to convolutional neural networks – sufficient to defeat older attacks

Defense: Adversarial training



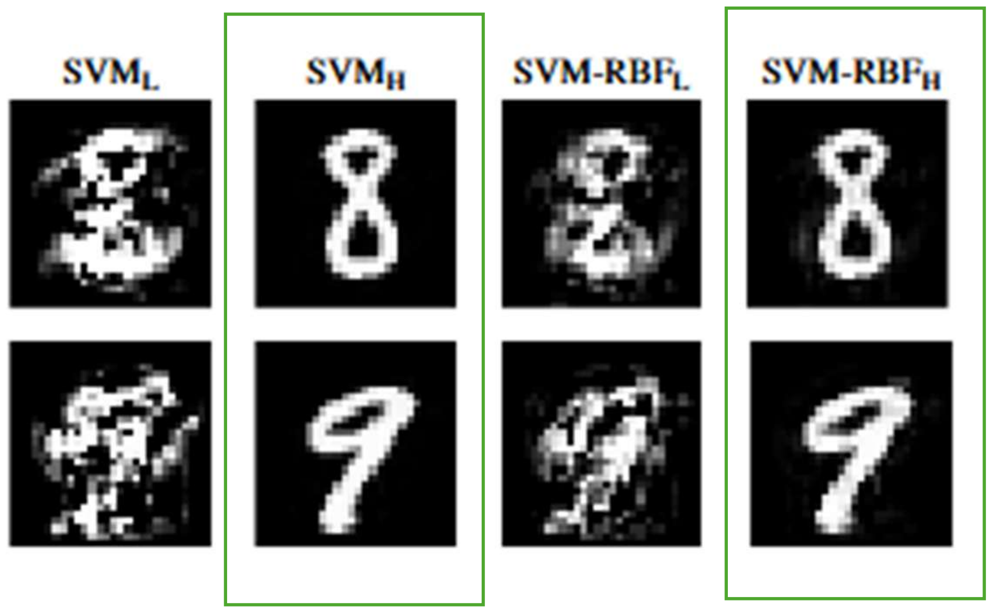
Defense: Adversarial training

- Some techniques have been adopted that are beneficial to classifier training even without an attack
- Data augmentation: Add augmented versions of training data to the training set (e.g. spatial transformation, color variance)
- Generative adversarial networks: Repeatedly train the classifier against a generator that applies perturbation to the data to fool the classifier
    - GANs are usually used as a generative model

Poisoning attacks

- Maliciously crafted inputs that compromise the classifier
  - Especially concerning for federated learning
- Two types:
  - **Backdoor** attacks: after poisoning training set, specifically crafted test cases will fail
  - **Availability** attacks: after poisoning training set, overall accuracy of classifier drops
- Demontic et al. (2019):
  - Low complexity machines require heavy perturbation, while high complexity machines require minimal changes, need to be defended with regularization
  - Attacks on high complexity machines are, however, less transferable

Poisoning attacks



White box techniques

- More powerful techniques are available in white box scenarios
- Gradient optimization: Given the gradients used in each layer, an attacker could optimize an attack by finding the largest gradients
- Huang et al. 2020:

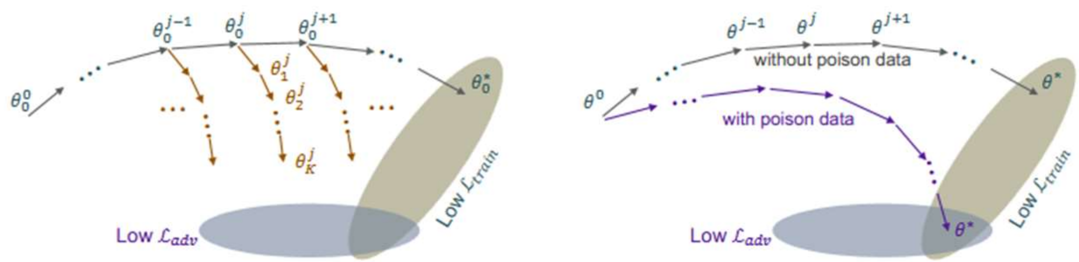


Figure 2: MetaPoison in weight space. Gray arrows denote normal training trajectory with weights θ_0^j at the j-th step. (Left) During the poison crafting stage, the computation graph consisting of the training pipeline is unrolled by K SGD steps forward in order to compute the perturbation to the poisons $\nabla_{X_p}\mathcal{L}_{\text{adv}}$, starting from various points along the trajectory. Optimally, those poisons will steer weights (brown arrows) toward regions of low $\mathcal{L}_{\text{adv}}$ regardless of which training step θ_0^j the poisons are inserted into. (Right) When the victim trains on the poisoned data (purple arrows), the weight trajectory is collectively and implicitly steered to regions of low $\mathcal{L}_{\text{adv}}$ whilst the learner explicitly drives the weights to regions of low $\mathcal{L}_{\text{train}}$.

A different problem: Data **privacy**

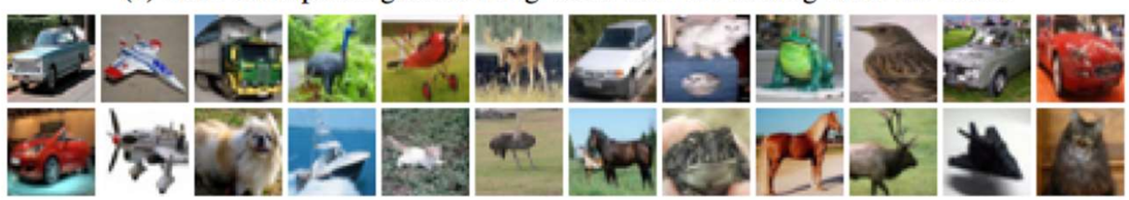
- A trained classifier can reveal elements of its training set
  - Asset is now the training data, not the classifier
- Example query: “Please fill out this social security number for me: 140…”
- Membership inference attacks: Was a given query part of the training set or not?
  - Shokri 2017: 90% accuracy against Google-trained models
  - Queries that were part of the training set have very high confidence values
- Implicit bias of gradient flow towards training points

A different problem: Data **privacy**

- Reconstruction results from Haim et al. (2022):

(a) Top 24 images reconstructed from a binary classifier trained on 50 CIFAR10 images

(b) Their corresponding nearest neighbours from the training-set of the model



Differentially private stochastic gradient descent

- Differential privacy guarantees that two neighboring datasets will have very similar outcomes under a query
- Stochastic gradient descent: To converge an optimization function, calculate and apply gradient on random batches repeatedly
- DP-SGD: We also clip and add noise to a gradient according to a privacy budget
- Composition theorem proves that the resulting classifier is differentially private