Defending Against Adversarial Attacks

Logistics for final project

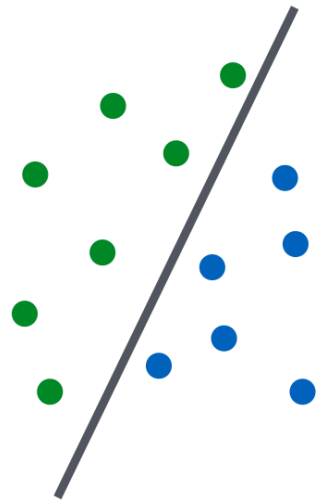
- Proposal presentation – 5 mins
  - Feedbacks from others and me
- Final presentation
  - You can choose to do a 20 mins presentation
  - Or you can choose to present a poster
- Review format
  - Paper summary
  - Advantages/disadvantages
  - Potential improvement, what you have learned

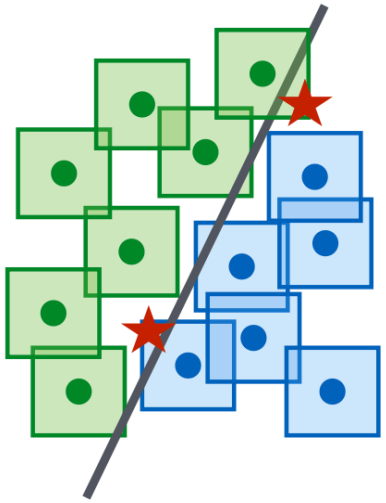
Towards Deep Learning Models Resistant to Adversarial Attacks

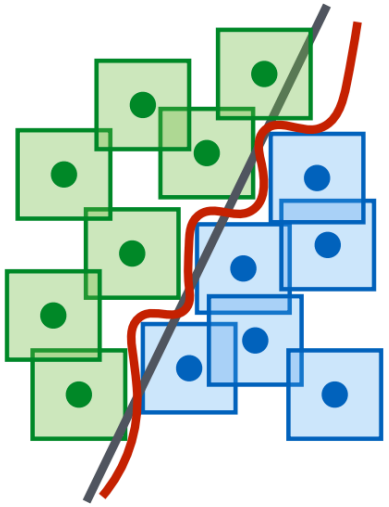
$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y) \right]$$

- Use a natural saddle point (min-max) formulation to capture the notion of security against adversarial attacks in a principled manner.
- The formulation casts both attacks and defenses into a common theoretical framework.
- Motivate projected gradient descent (PGD) as a universal “first-order adversary”.

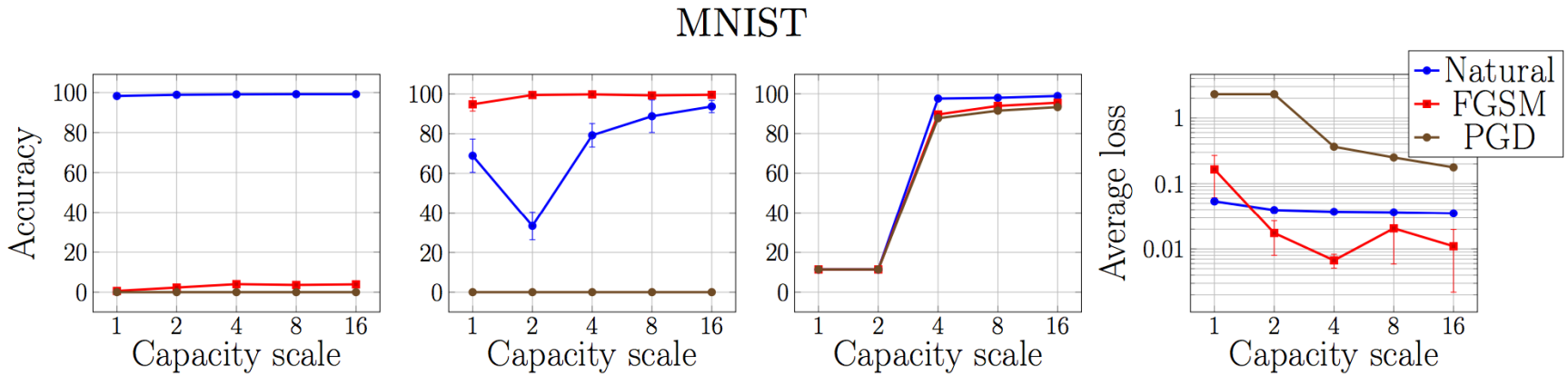
Model Capacity







Towards Deep Learning Models Resistant to Adversarial Attacks



PeerNets: Exploiting Peer wisdom against adversarial attacks

- Design robust neural networks that are robust to adversarial attacks
- Defense: recover the ground truth instead of just tell adversarial instance apart
- Necessary step: design novel and advanced architectures built on new computational paradigms
- PeerNets:
  - Euclidean convolutions -> graph convolutions
  - Non-local forward propagation: Capture global structure induced by the data graph

PeerNets: Exploiting Peer wisdom against adversarial attacks

- Peer Regularization layer
- Given images $X_1, X_2 \ldots, X_N$
- For N images, each image will look for its K nearest neighbors based on cosine similarity
  - $$\tilde{\mathbf{x}}_p^i = \sum_{k=1}^{K} \alpha_{ij_kpq_k} \mathbf{x}_{q_k}^{j_k}, \qquad \alpha_{ij_kpq_k} = \frac{\text{LeakyReLU}(\exp(a(\mathbf{x}_p^i, \mathbf{x}_{p_k}^{j_k})))}{\sum_{k'=1}^{K} \text{LeakyReLU}(\exp(a(\mathbf{x}_p^i, \mathbf{x}_{p_{k'}}^{j_{k'}})))}$$

PeerNets: Exploiting Peer wisdom against adversarial attacks

- Randomized approximation
- Monte Carlo approximation
  - Select smaller batch and sample the nearest neighbor from each batch

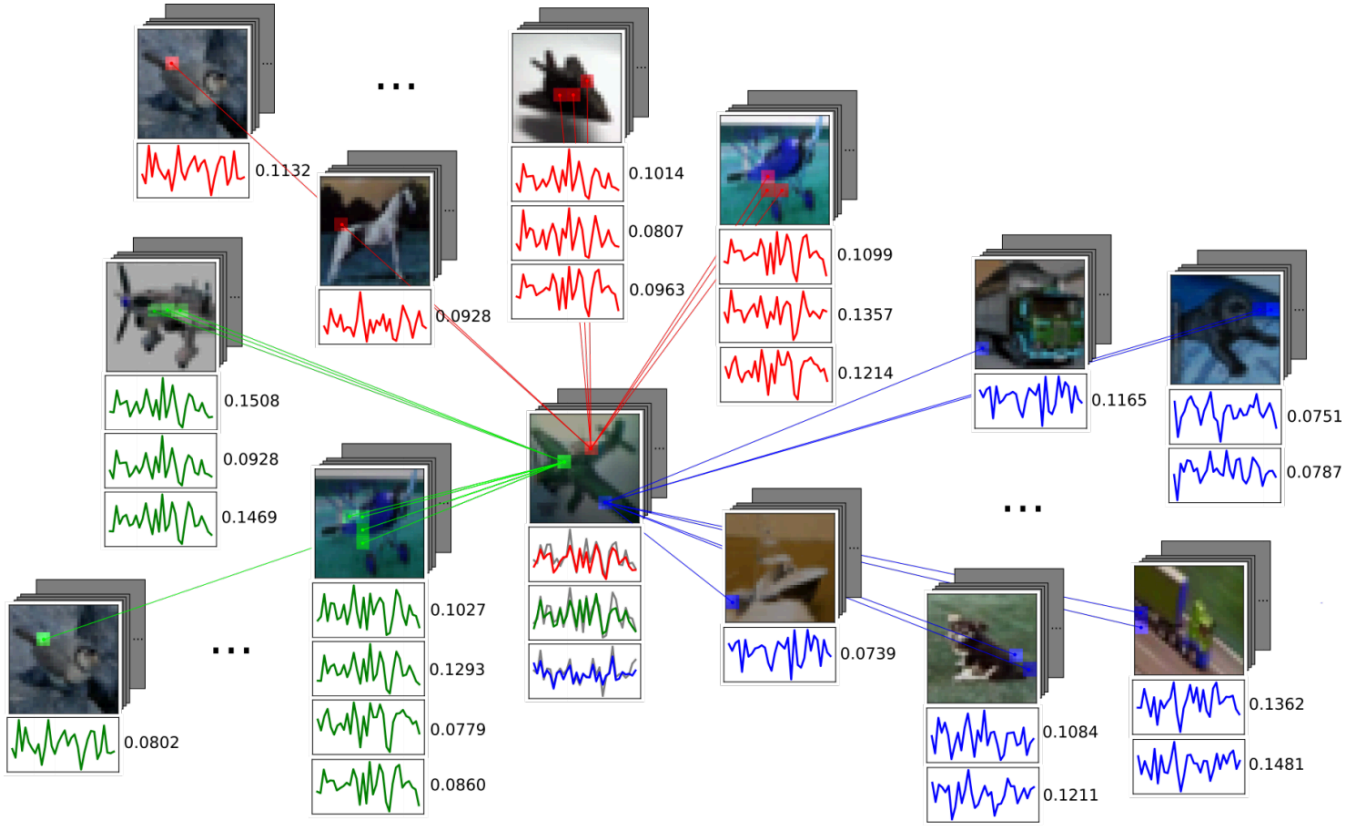
$$\{l_{m1}, \ldots, l_{mN}\} \subset \{1, \ldots, N'\}$$

$$\tilde{\mathbf{x}}_p^i = \frac{1}{M} \sum_{m=1}^{M} \sum_{k=1}^{K} \alpha_{ij_{mk}pq_{mk}} \mathbf{x}_{q_{mk}}^{j_{mk}}$$

- Other optimization method?

PeerNets: Exploiting Peer wisdom against adversarial attacks

- Select M = 1 during training and large M during inference
  - Limitations?

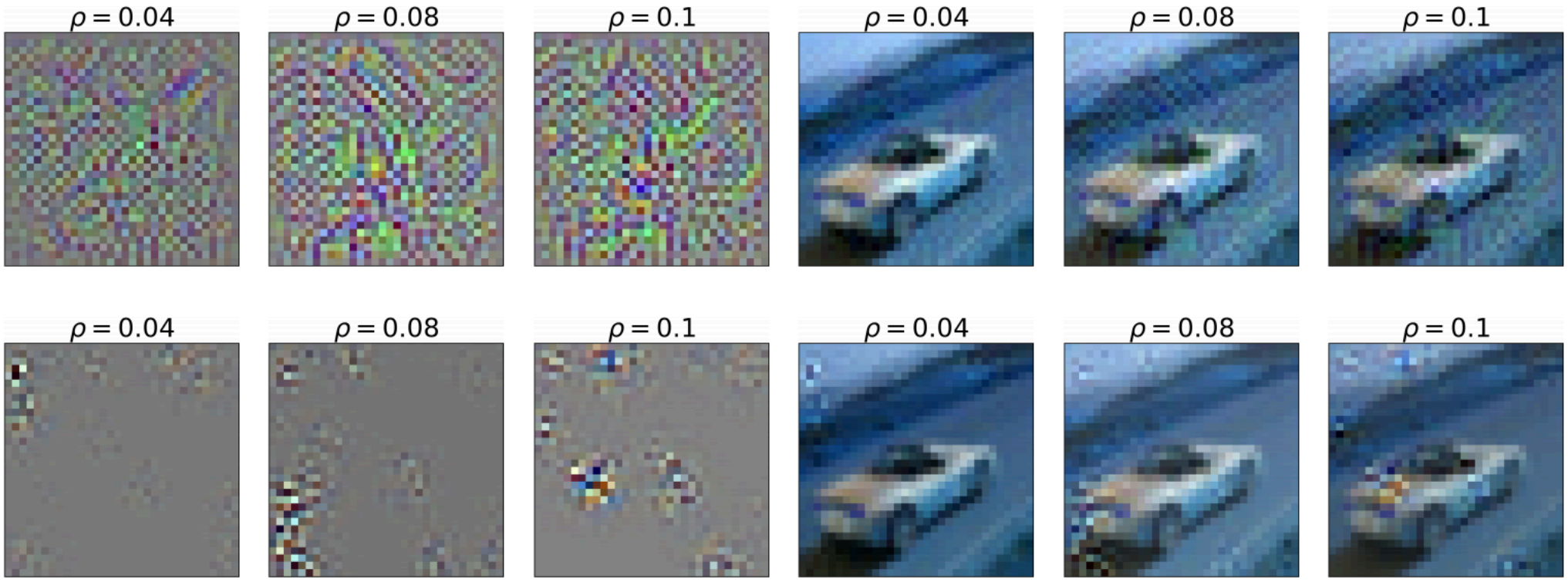


Results for PeerNets

Original

Reconstructed

Visualization of perturbation



Takeaways

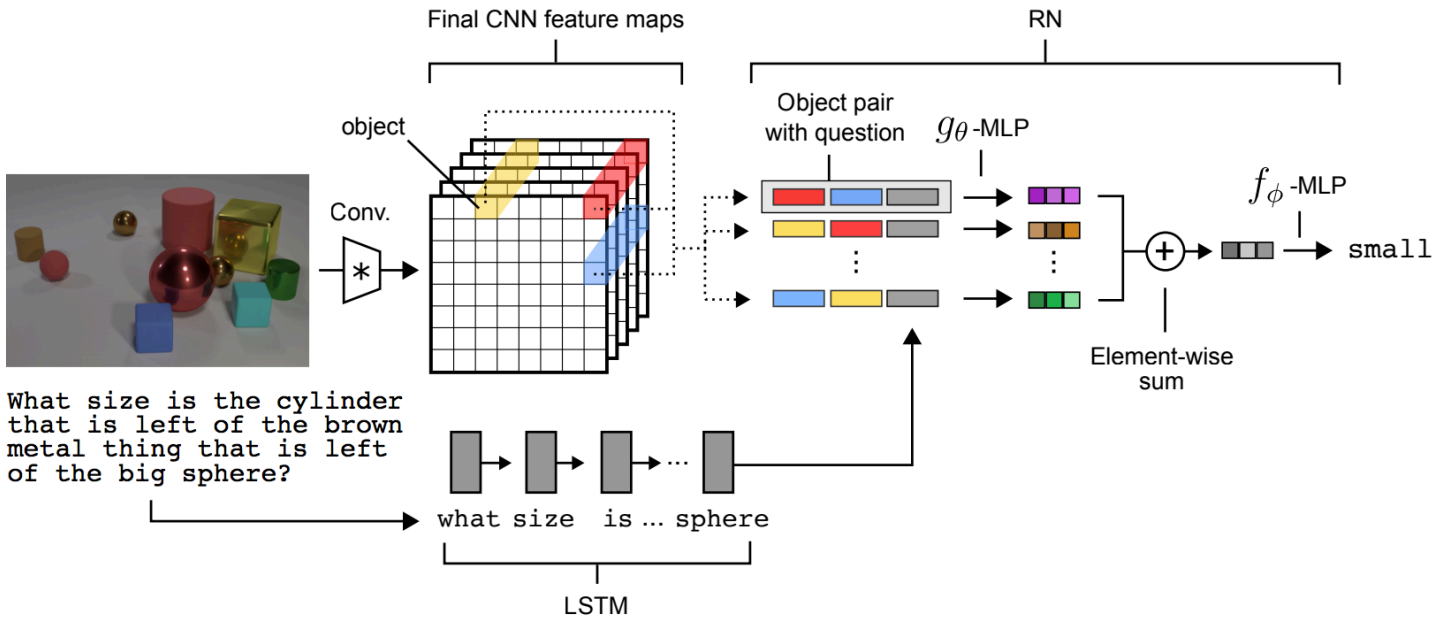
- Alternate Euclidean Graph convolution to harness information from peers can provide global information
- Can be added to any models as regularized layer –> good principle
- Not affect the benign accuracy -> important
- How to scale up?
- How to consider more peer images instead of pixels?
- Temporal information?

Similar reading

- Countering adversarial images using input transformations
  - Image quilting – nearest patches
  - Computationally expensive

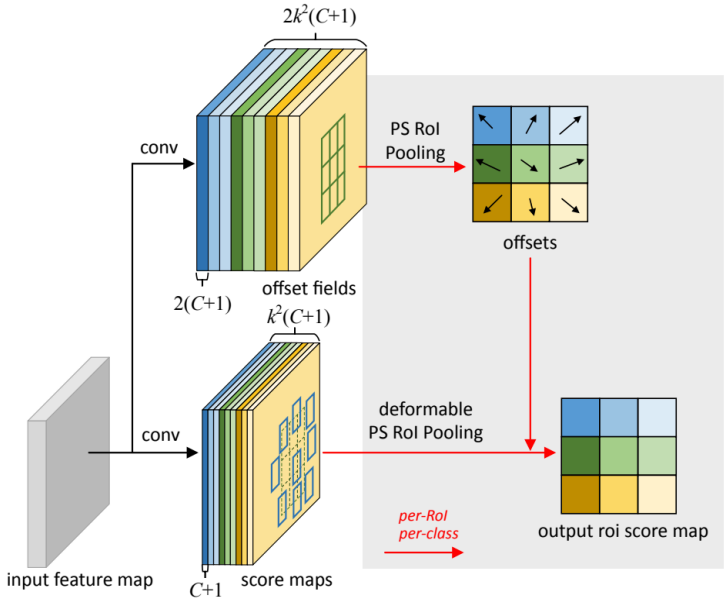
Interesting reading

- A simple neural network module for relational reasoning

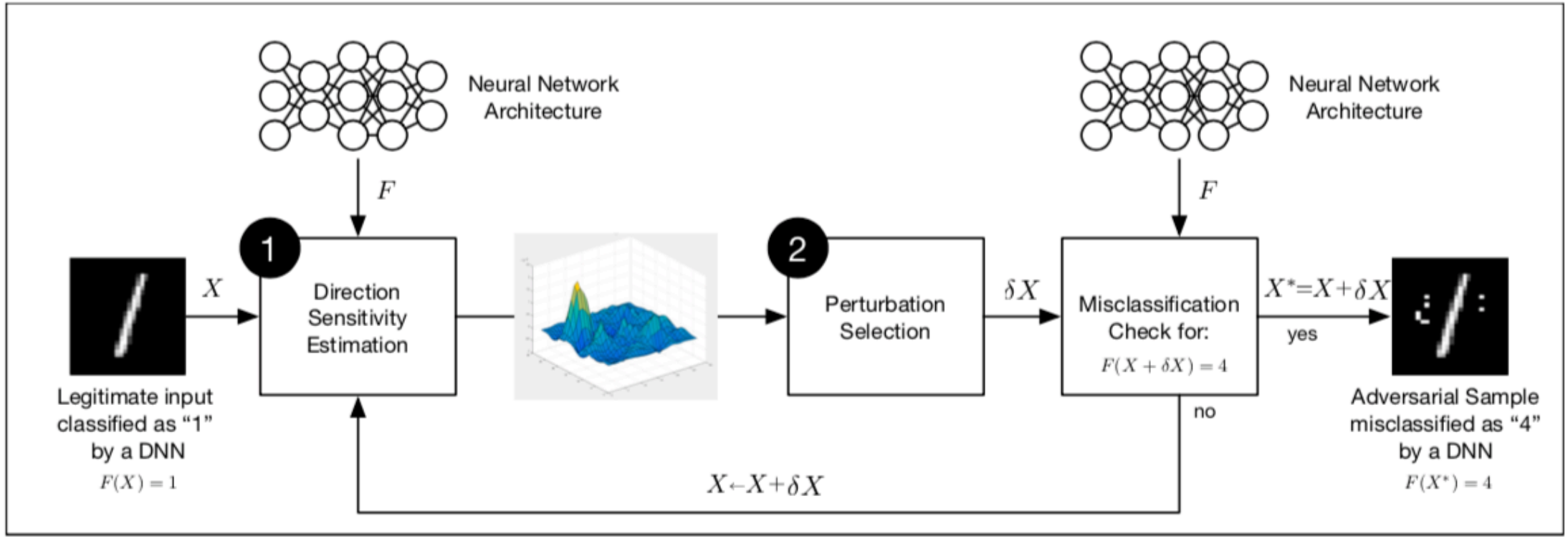


Interesting reading

- Deformable Convolutional Networks
  - Deformable convolution and deformable RoI pooling
  - Augment the spatial sapling locations with additional offset which can be learned



Distillation as a defense to adversarial perturbations against DNNs



Defending DNNs using distillation

- Previous methods:
  - Regularizations
  - Data augmentation
- Distillation as defense for DNNs

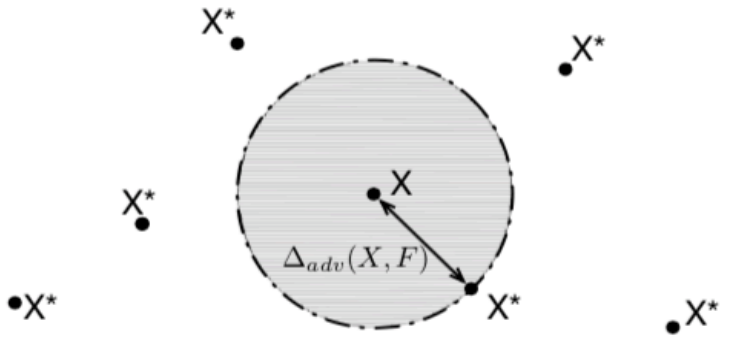
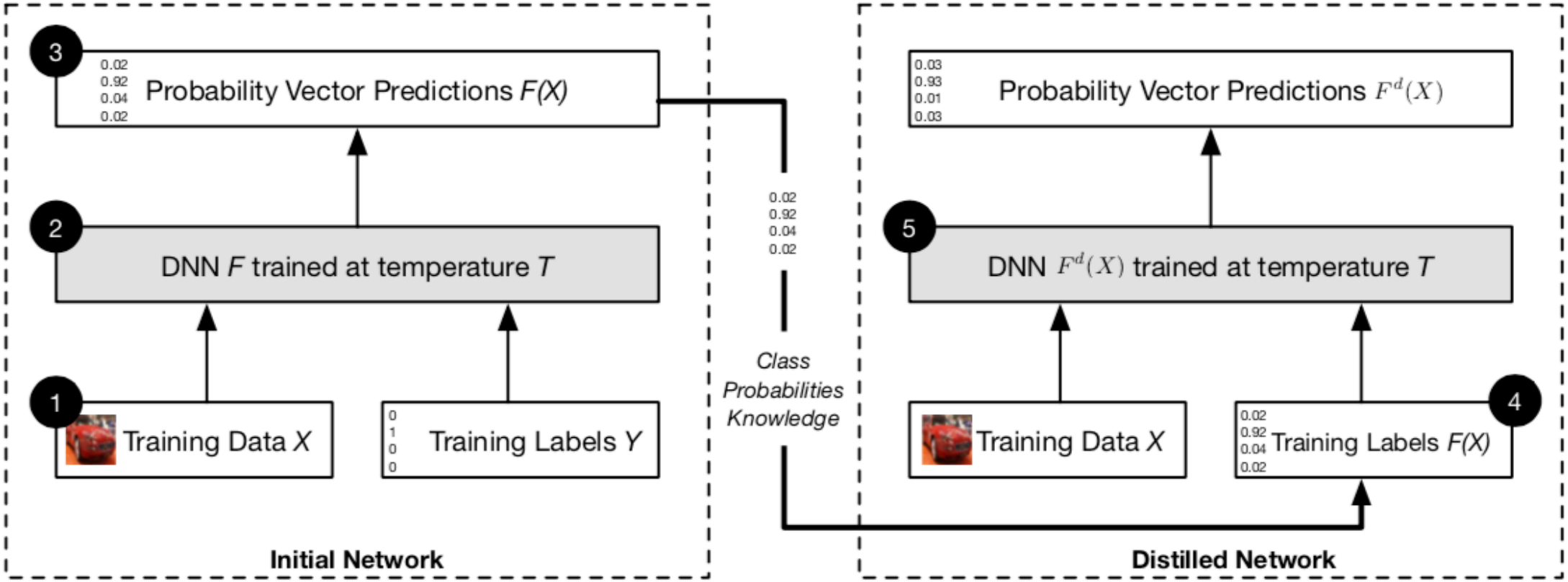


Fig. 4: **Visualizing the hardness metric:** This 2D representation illustrates the hardness metric as the radius of the disc centered at the original sample X and going through the closest adversarial sample X^* among all the possible adversarial samples crafted from sample X. Inside the disc, the class output by the classifier is constant. However, outside the disc, all samples X^* are classified differently than X.

Defending DNNs using distillation

- Adversarial examples are primarily exploiting gradients computed to estimate the sensitivity of DNNs
- We need to smooth the model learned during training to encourage the model to generalize better
- Robustness: ensure the classification output by a DNN remains constant in a closed neighborhood around anv given sample extracted from the input distribution $\rho_{adv}(F) = E_{\mu}[\Delta_{adv}(X, F)]$

Defending DNNs with distillation



First train an initial network F on data X with a softmax temperature of T . We then use the probability vector F(X), which includes additional knowledge about classes compared to a class label, predicted by network F to train a distilled network F_d at temperature T on the same data X

$$F(X) = \left[\frac{e^{z_i(X)/T}}{\sum_{l=0}^{N-1} e^{z_l(X)/T}} \right]_{i \in 0..N-1}$$