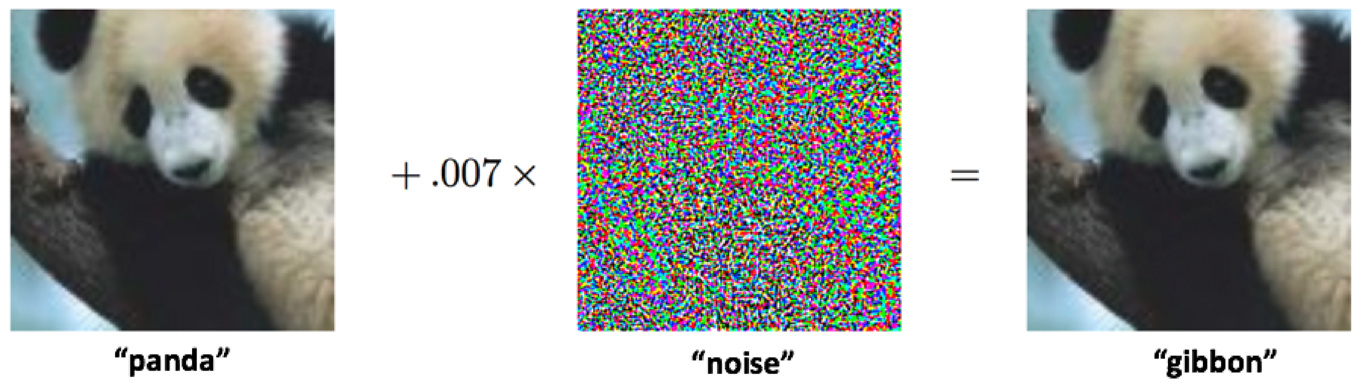
Cost-Sensitive Robustness against Adversarial Examples

University of Virginia

Xiao Zhang David Evans

Preliminaries

Adversarial examples: an input, generated by some adversary, which is visually indistinguishable from an example from the natural distribution, but is able to mislead the target classifier.



Famous "panda-gibbon" illustration of adversarial examples

More formally, the set of adversarial examples w.r.t. seed example $\{\boldsymbol{x}_0, y_0\}$, classifier $f_\theta(\cdot)$ and ℓ_∞ perturbations is defined as

$$\{\boldsymbol{x} \in \mathcal{X} : \|\boldsymbol{x} - \boldsymbol{x}_0\|_\infty \leq \epsilon \text{ and } \operatorname*{argmax}_j [f_\theta(\boldsymbol{x})]_j \neq y_0\}.$$

Defenses with certified robustness (Wong & Zico, 2018)

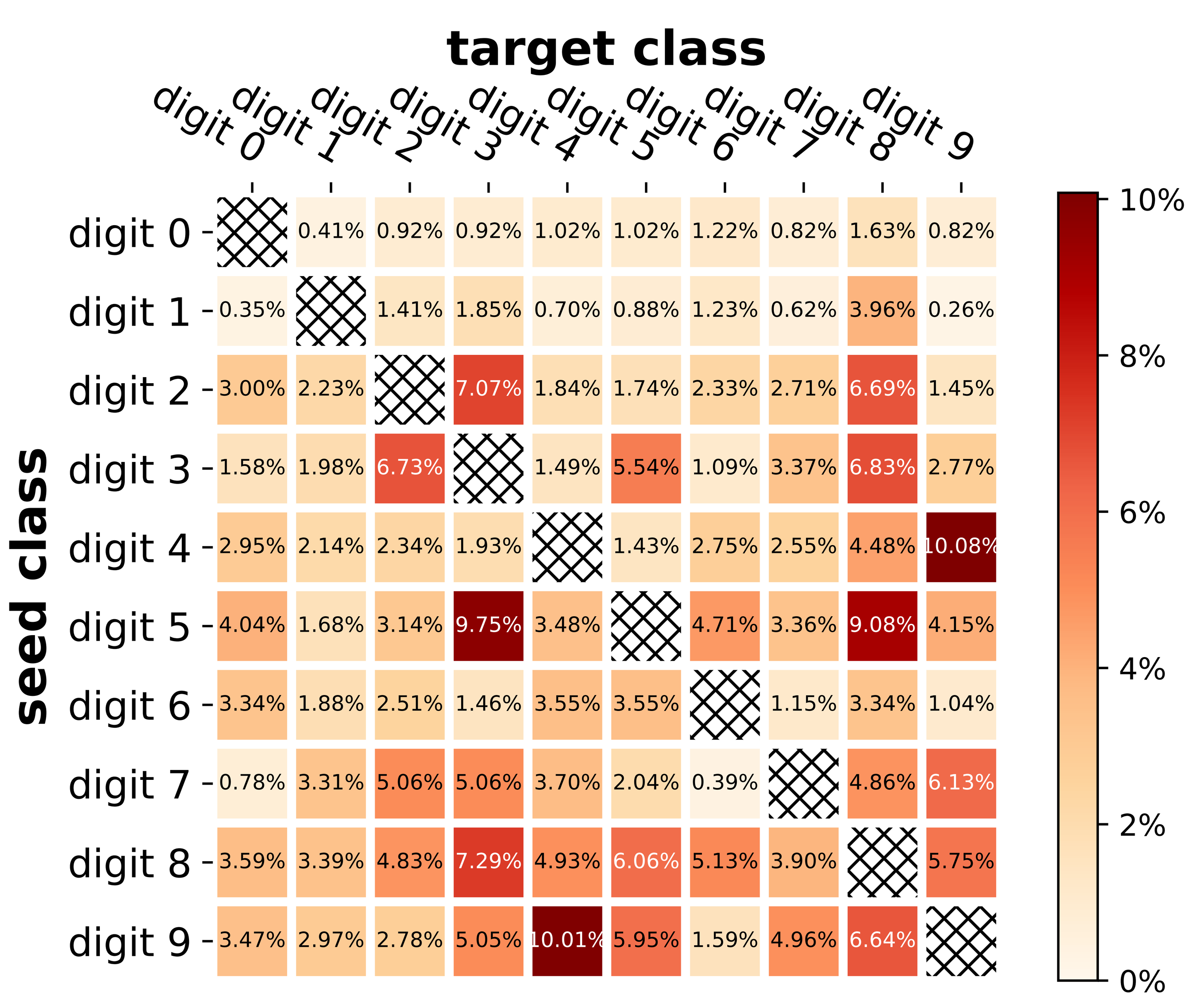
- Construct a convex outer bound on the "adversarial polytope"
- Develop robust certificate for testing given inputs
- Propose training methods to optimize for certifiable robustness

$$\operatorname*{minimize}_\theta \ \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}\Big(-J_\epsilon\big(\boldsymbol{x}_i, g_\theta(\boldsymbol{e}_{y_i} \cdot \boldsymbol{1}^\top - \boldsymbol{I})\big), y_i\Big),$$

where $-J_\epsilon\big(\boldsymbol{x}_i, g_\theta(\boldsymbol{e}_{y_i} \cdot \boldsymbol{1}^\top - \boldsymbol{I})\big)$ is a guaranteed lower bound.

Pairwise robust heatmap of certified robust classifier

- (i, j)-th entry is a robustness bound of that seed-target pair.
- The vulnerability to transformations differs among class pairs.



Heatmap of pairwise robust test error

Motivations

Overall robustness: designed for preventing seed examples in **any** class from being misclassified as **any** other class.

- Existing defensive methods focus on such robustness definition.
- May not be the appropriate criteria for security applications.
- Only certain kinds of adversarial misclassifications pose meaningful threats that provide value for potential adversaries.

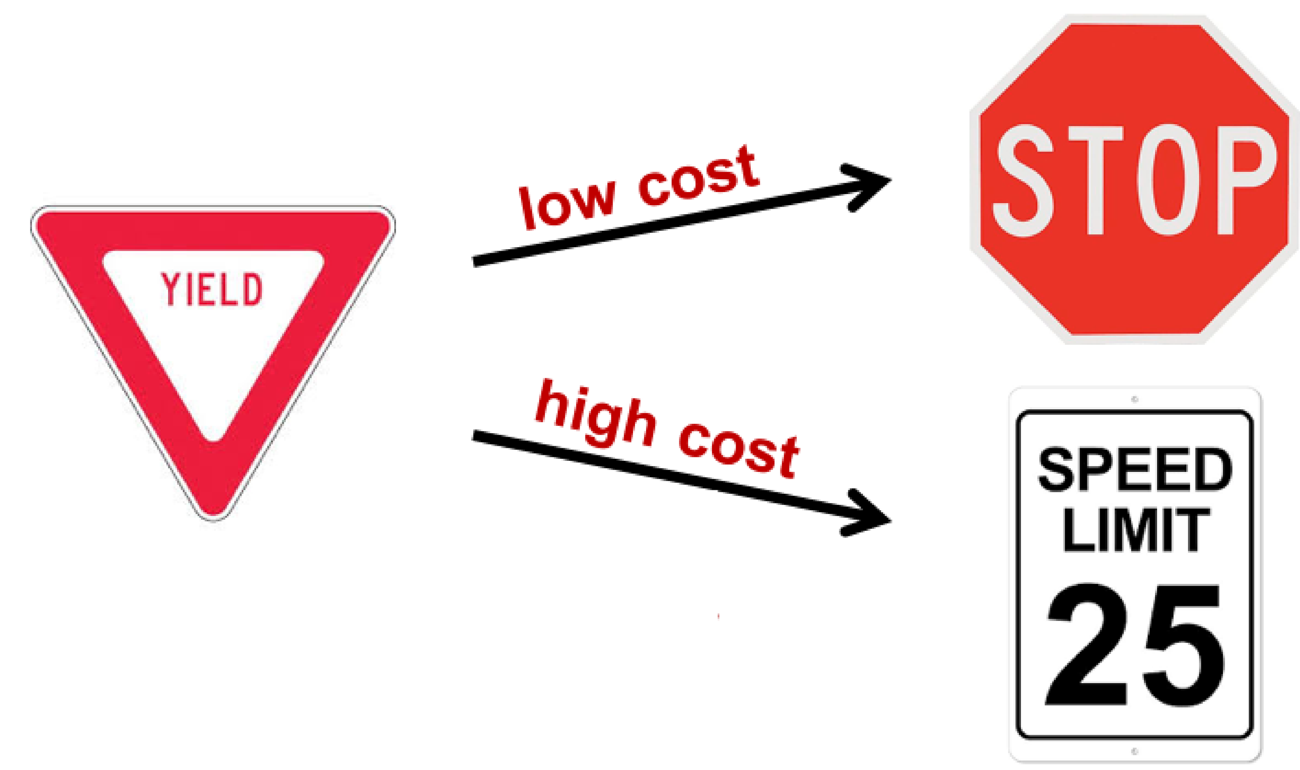


Illustration of our motivation in the application of autonomous vehicles

Cost-Sensitive Robustness

- Use a **cost matrix** C to encode the cost (i.e., potential harm to model deployer) of different adversarial transformations.
- **Binary cost matrix**
  - An example $\boldsymbol{x}$ in class j is said to be certified cost-sensitive robust, if $J_\epsilon(\boldsymbol{x}, g_\theta(\boldsymbol{e}_j - \boldsymbol{e}_{j'})) \geq 0$ for all $j' \in \Omega_j$.
  - Define **cost-sensitive robust error** as

$$\frac{\#\{\text{examples not guaranteed to be cost-sensitive robust}\}}{\#\{\text{candidate seed examples with non-zero cost}\}}.$$

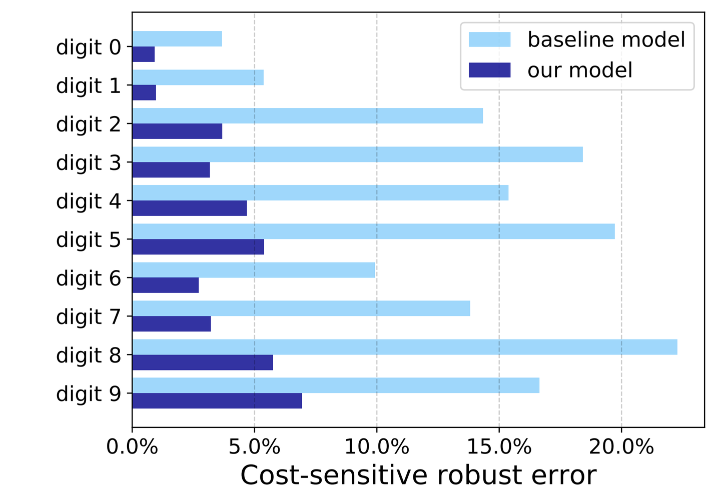
- **Real-valued cost matrix**
  - The cost of an adversarial example $\boldsymbol{x}$ in class j is defined as the sum of all $C_{jj'}$ such that $J_\epsilon(\boldsymbol{x}, g_\theta(\boldsymbol{e}_j - \boldsymbol{e}_{j'})) < 0$.
  - Define **robust cost** as averaged cost of adversarial examples.
- **General cost-sensitive training method**

$$\operatorname*{minimize}_\theta \ \frac{1}{N}\sum_{i\in[N]} \mathcal{L}(f_\theta(\boldsymbol{x}_i), y_i) + \alpha \sum_{j\in[m]} \frac{\delta_j}{N_j} \sum_{i|_{y_i=j}} \log\Big(1 + \sum_{j'\in\Omega_j} C_{jj'} \cdot \exp\big(-J_\epsilon(\boldsymbol{x}_i, g_\theta(\boldsymbol{e}_j - \boldsymbol{e}_{j'}))\big)\Big)$$

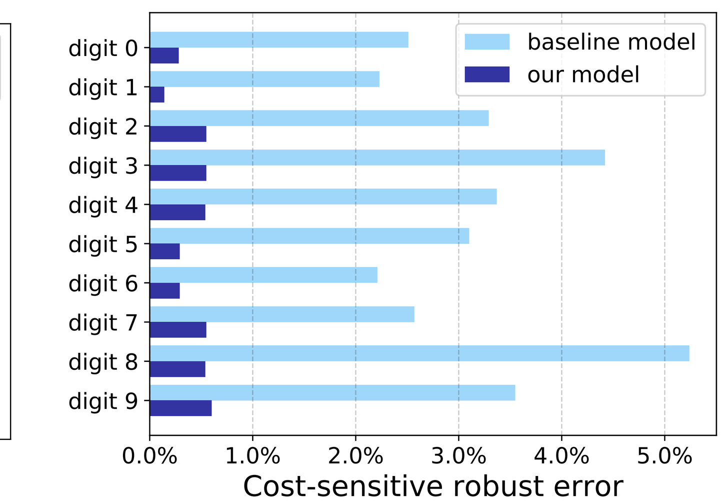
  - Optimize for both standard classification accuracy and certified cost-sensitive robustness, and use α to balance them.
  - Can be solved efficiently using gradient-based algorithms.

Experimental Results

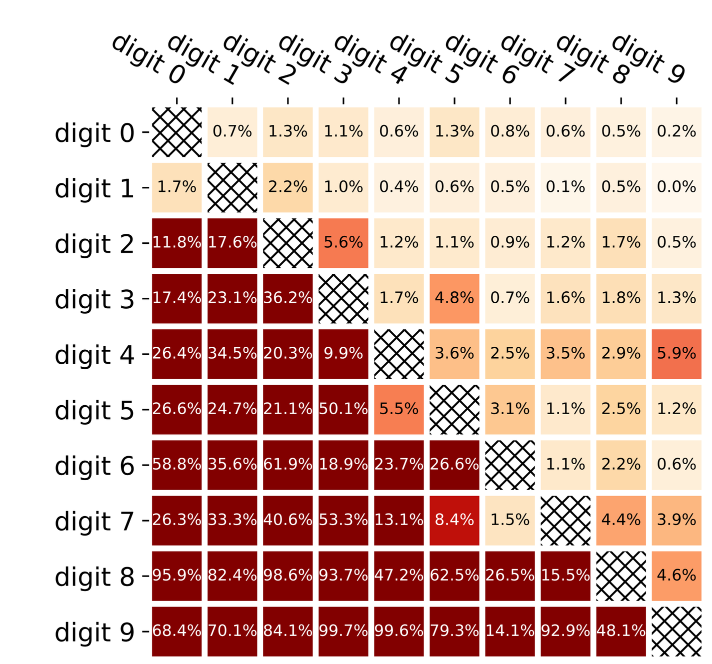
- **MNIST**



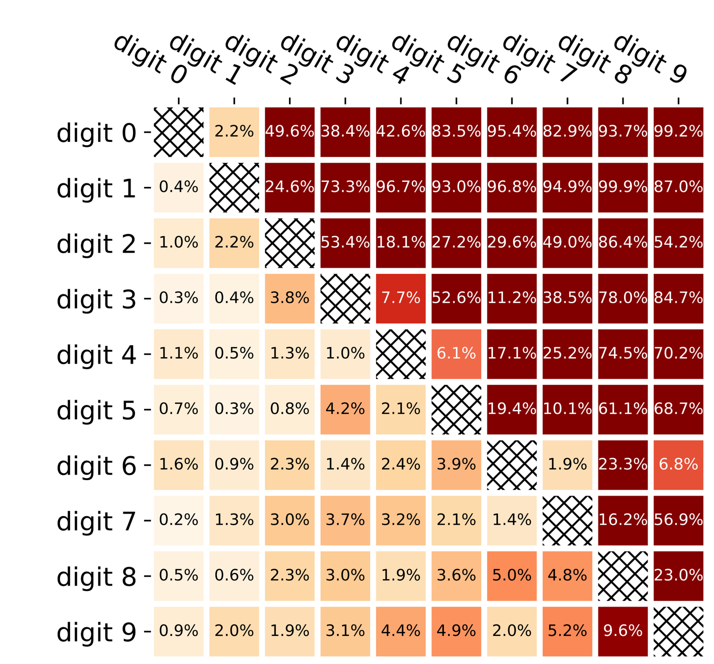
(a) single seed class



(b) single target class



(c) small-large

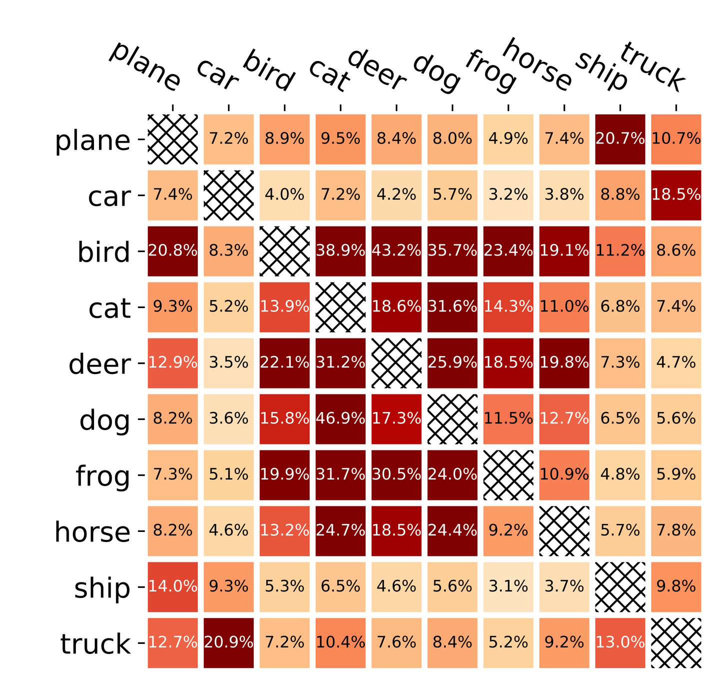


(d) large-small

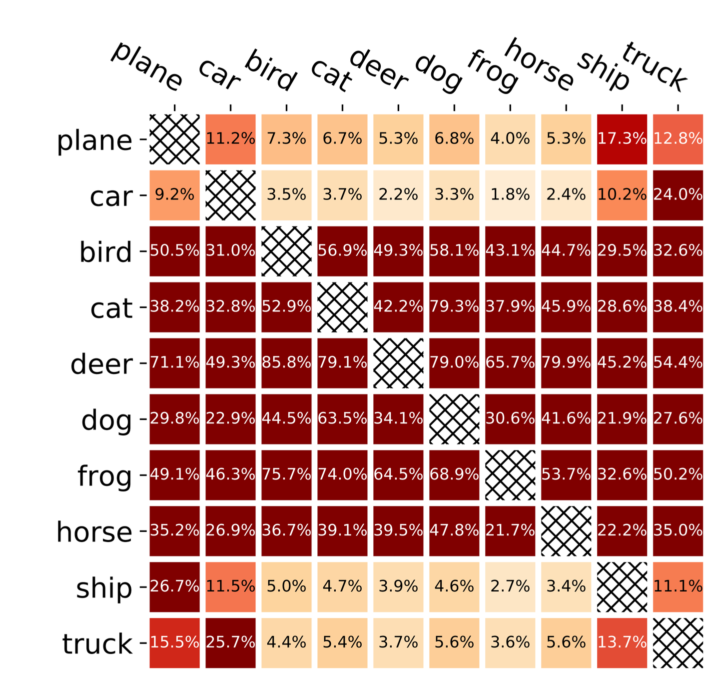
- **CIFAR-10**

Comparison results against ℓ_∞ perturbations with $\epsilon = 2/255$

Task Description		Classification Error		Robust Error	
		baseline	ours	baseline	ours
single pair	(frog, bird)	31.80%	27.88%	19.90%	1.20%
	(cat, plane)	31.80%	28.63%	9.30%	2.60%
single seed	dog	31.80%	30.69%	57.20%	28.90%
	truck	31.80%	31.55%	35.60%	15.40%
single target	deer	31.80%	26.69%	16.99%	3.77%
	ship	31.80%	24.80%	9.42%	3.06%
multiple	A-V	31.80%	26.65%	16.67%	7.42%
	V-A	31.80%	27.60%	12.07%	8.00%



(e) baseline model



(f) our model