Can robust ensembling schemes improve defenses against adversarial inputs?

Aditya Saligrama

Mentors: Guillaume Leclerc, Prof. Aleksander Mądry

Mądry Lab, MIT CSAIL Theory of Computation

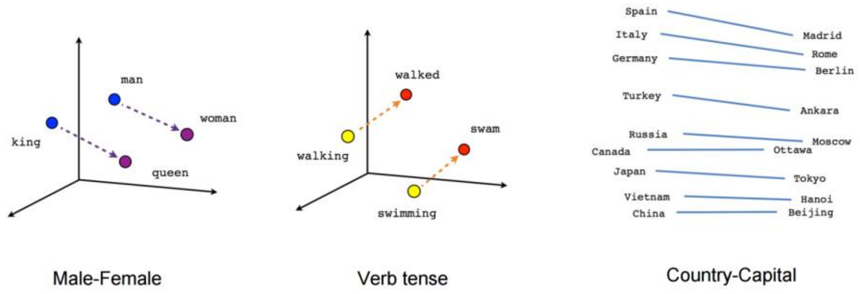
9.5th Annual MIT PRIMES Conference, October 20, 2019

Deep learning and adversarial examples

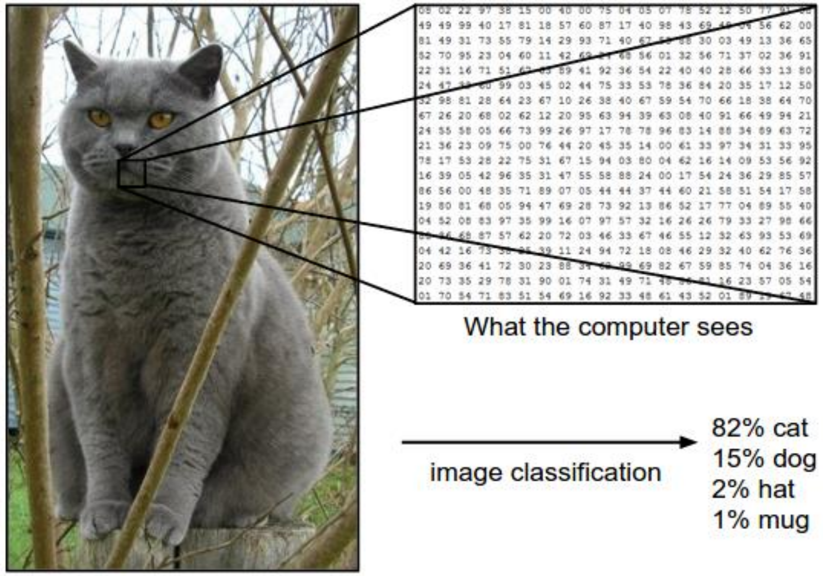
Deep learning

- Has become ubiquitous in the last few years and can outperform humans on some tasks

(DeepAI 2019)



(Ruizendaal 2017)

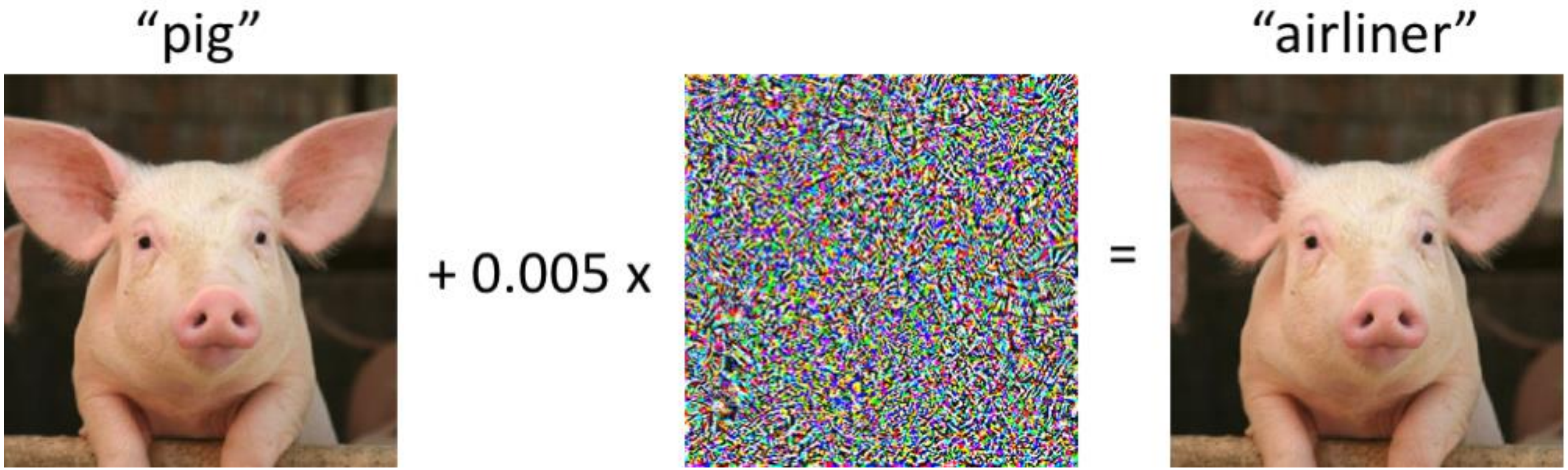


(Karpathy 2015)

Adversarial attacks

- Modify image in a set S, such as L2-ball of size ε, to maximize loss L
  - Imperceptible to human observer
  - Fools deep learning models

$$\hat{\delta} = \underset{||\delta||<\epsilon}{\operatorname{argmax}} L(\theta, x + \delta, y)$$

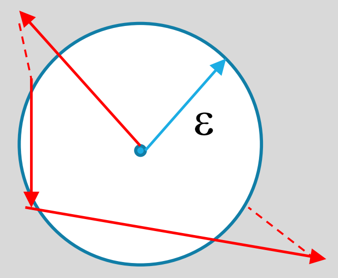


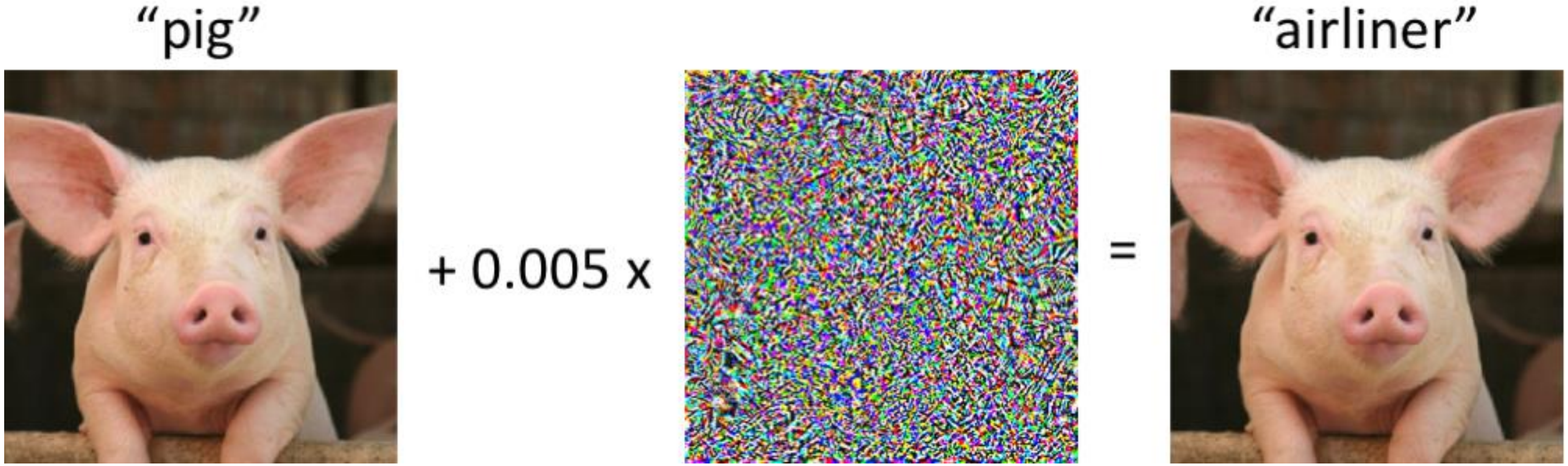
(Mądry and Schmidt 2018)

Adversarial attacks

- Modify image in a set S, such as L2-ball of size ε, to maximize loss L
  - Imperceptible to human observer
  - Fools deep learning models
- Many ways of synthesizing adversarial examples:
  - Such as PGD - projected gradient descent (Mądry et al. 2017)

$$\hat{\delta} = \underset{||\delta||<\epsilon}{\operatorname{argmax}} L(\theta, x + \delta, y)$$





(Mądry and Schmidt 2018)

Robust training

- Train robust model θ on dataset *D*:
  - Resistant to adversarial attacks
  - Robust training via PGD (Mądry et al. 2017)
    - Many other ways…

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, E_{(x,y)\sim D}\left[\max_{||\delta||\le\epsilon} L(\theta, x+\delta, y)\right]$$

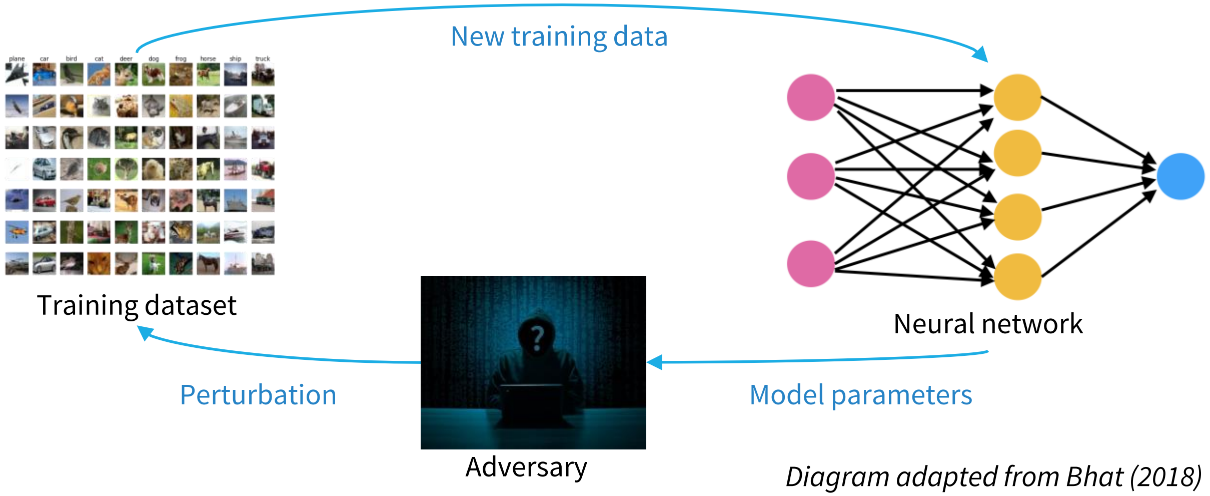


Diagram adapted from Bhat (2018)

Robust training

- Train robust model θ on dataset *D*:
  - Resistant to adversarial attacks
  - Robust training via PGD (Mądry et al. 2017)
    - Many other ways…

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, E_{(x,y)\sim D}\left[\max_{||\delta||\leq\epsilon} L(\theta, x+\delta, y)\right]$$

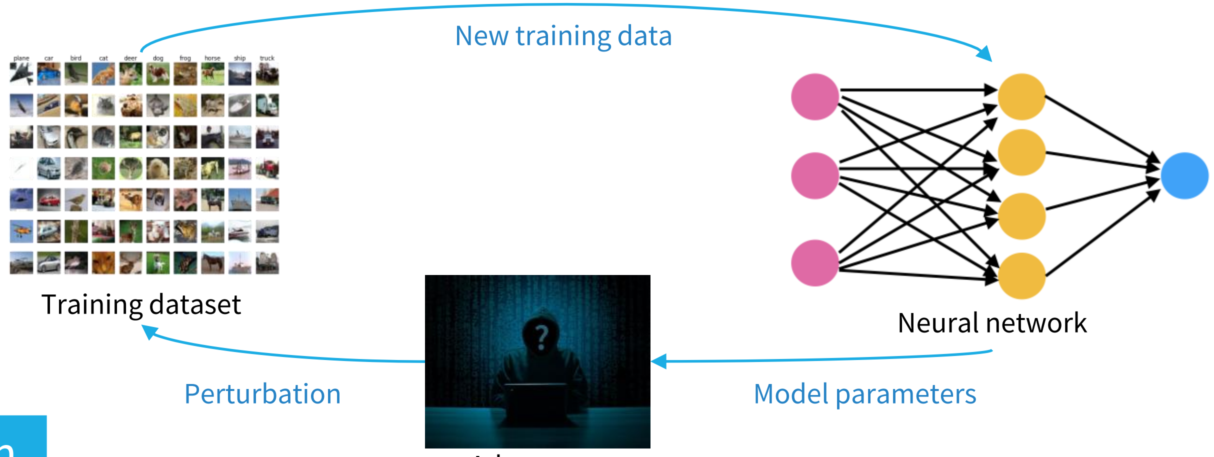


Diagram adapted from Bhat (2018)

ResNet18 models (He et al. 2015)
trained on CIFAR10

	Natural train	Robust train (ε=0.031)
Natural test		
Adv. test (ε=0.031)		

Robust training

- Train robust model θ on dataset *D*:
  - Resistant to adversarial attacks
  - Robust training via PGD (Mądry et al. 2017)
    - Many other ways…

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, E_{(x,y)\sim D}\left[\max_{||\delta|| \le \epsilon} L(\theta, x+\delta, y)\right]$$

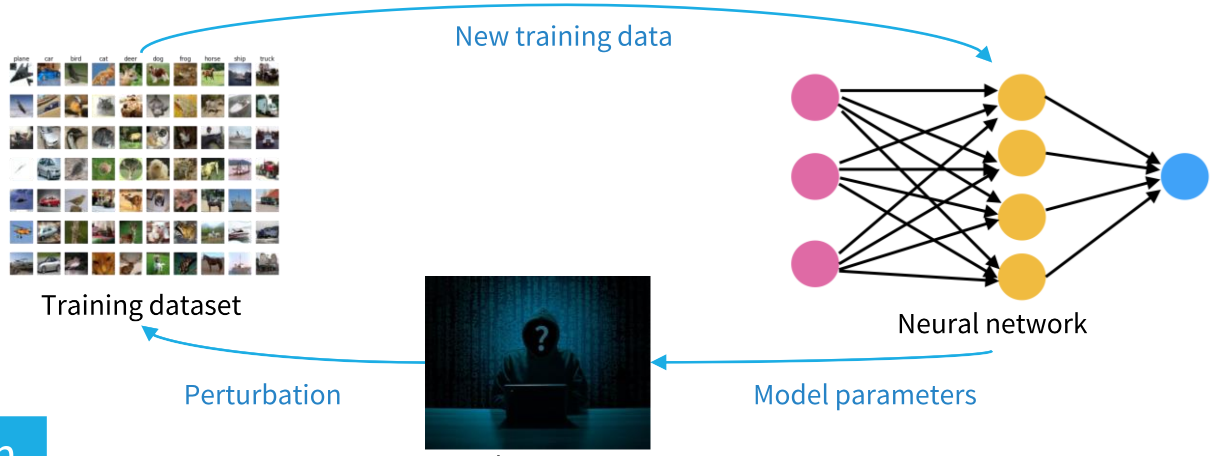


Diagram adapted from Bhat (2018)

ResNet18 models (He et al. 2015)
trained on CIFAR10

	Natural train	Robust train (ε=0.031)
Natural test	93%	83%
Adv. test (ε=0.031)	0%	51%

Our goal:
Robust train on natural test → natural train on natural test
Robust train on adv. test → natural test

Ensembling schemes

Adversarial ensembling

Using ensembling for training (lots of prior work, different from previous slide):

- Vanilla ensembling (baseline for this talk)
  - Random initializations, train M standard models
- Ensemble Adversarial Training (Tramèr et al. 2017)
  - Collect adversarial examples from multiple models
  - Transfer examples to train single model
- Ensemble diversity (Pang et al. 2019)
  - Coupled training of all *M* models to promote diversity

	Robust training (Mądry et al. 2017)	Vanilla ensembling	Ensemble diversity (Pang et al. 2019)
Natural test	83%	94%	93%
Adv. test	51% (ε=0.03)	0%	30% (ε=0.02)

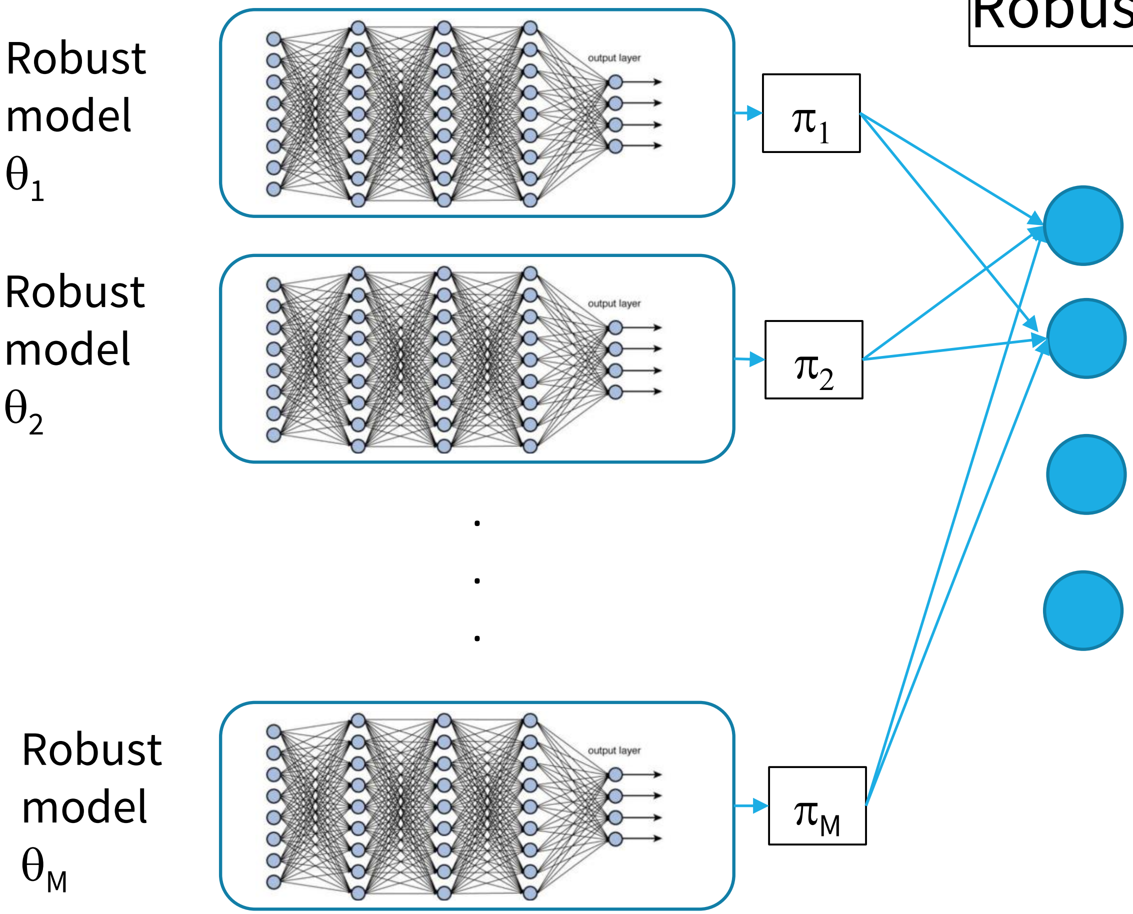
Our proposed methods

Robust ensembling

- Train M independent models robustly
  - *i*'th model with seed *i*

$$\widehat{\theta}_{\mathrm{i}} = \underset{\theta}{\operatorname{argmin}}\, E_{(x,y)\sim D}\left[\max_{||\delta||\leq\epsilon} L(\theta, x+\delta, y)\right]$$

Robust training with initialization seed i



$$c(x, \boldsymbol{\theta}, \boldsymbol{\pi}) = \max_{y} \sum_{i=1}^{M} \pi_i \theta_i(x, y)$$

$\theta_i(x, y)$: model *i*'s probability of class *y* on instance *x*

How to understand ensembles?

Value of the game (discrete):

- Player: random strategy over M models
  - Probability $\pi_1 \dots \pi_M$
- Adversary: perturbation $\delta_1 \dots \delta_S$ ($S \to \infty$) with probability $q_1 \dots q_S$

$$\ell(\mathbf{q}, \pi, L) = E_{\delta \sim \mathbf{q}} E_{\theta_j \sim \pi} L(\theta_j, x + \delta, y)$$

Player strategy

Adversary strategy

	$\boldsymbol{\theta_1}$	$\boldsymbol{\theta_2}$	$\boldsymbol{\theta_3}$
δ_1	Loss		
δ_2			
δ_3			

Key point: Adversary plays against ensemble rather than single model for each instance

$$\min_\pi \max_{\mathbf{q}} \ell(\mathbf{q}, \pi, L) \leq \max_\delta \frac{1}{M} \sum_j L(\theta_j, x + \delta, y)$$

vs.

$$\max_{\delta \in S} L(\theta, x + \delta, y)$$

How to understand ensembles?

Value of the game (discrete):

- Player: random strategy over M models
  - Probability $\pi_1 \ldots \pi_M$
- Adversary: perturbation $\delta_1 \ldots \delta_S$ ($S \to \infty$) with probability $q_1 \ldots q_S$

$$\ell(\mathbf{q}, \pi, L) = E_{\delta \sim \mathbf{q}} E_{\theta_j \sim \pi} L(\theta_j, x + \delta, y)$$

Adversary strategy ↓ Player strategy →

	$\boldsymbol{\theta_1}$	$\boldsymbol{\theta_2}$	$\boldsymbol{\theta_3}$
δ_1	Loss		
δ_2			
δ_3			

Key point: Adversary plays against ensemble rather than single model for each instance

$$\min_\pi \max_{\mathbf{q}} \ell(\mathbf{q}, \pi, L) \leq \max_\delta \frac{1}{M} \sum_j L(\theta_j, x + \delta, y)$$

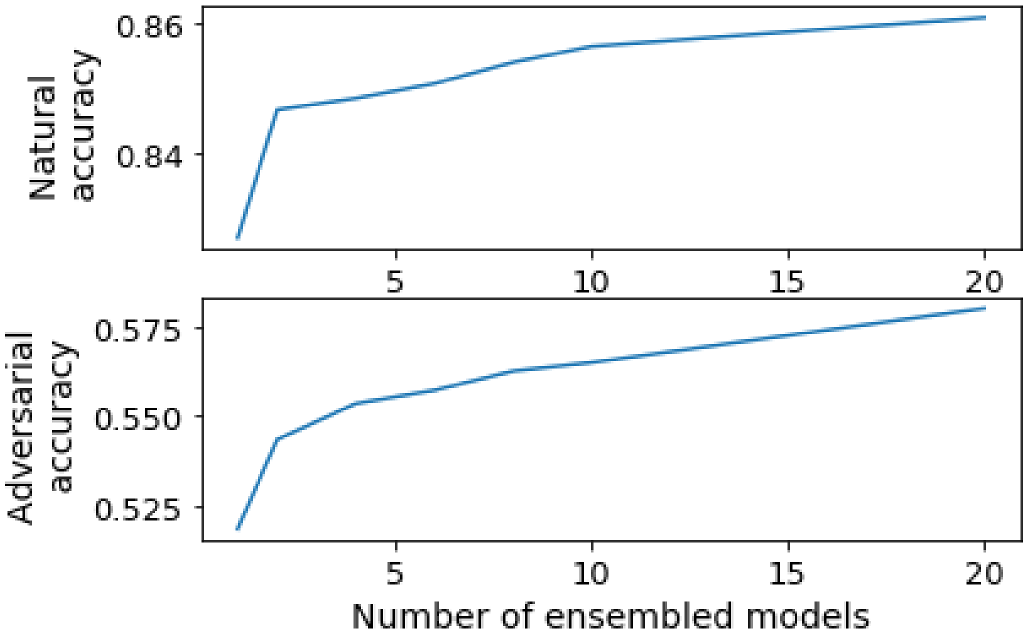
vs.

$$\max_{\delta \in S} L(\theta, x + \delta, y)$$

robust ensemble loss $\leq$ single robust model loss
Why? Choose **q** to focus on single model

Robust ensembling: Results

	Single non-robust model	Single robust model	Robust ensemble (20 models)
Natural test	93.2%	82.7%	86.1%
Adv. Test ($\varepsilon = 0.031$, $k = 14$)	0.0%	51.8%	58.0%



Different models may mispredict on same instances, but require different perturbations

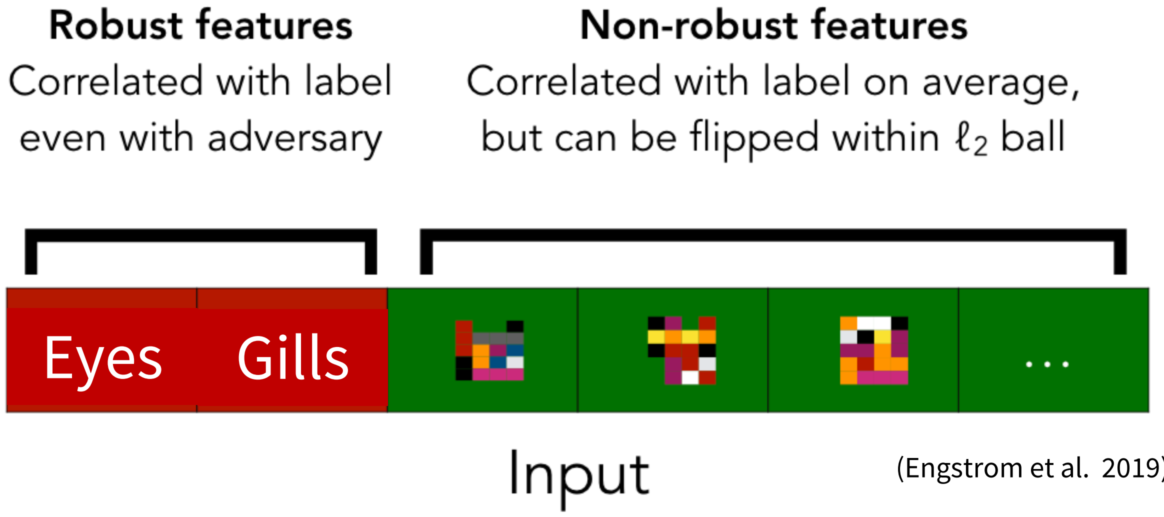
Still, large gap between natural performance of non-robust model and robust ensembles!

How to bridge this gap?

Robust and non-robust features

- Images comprised of robust and non-robust features (Ilyas et al. 2019)
- **Key insight: Robust features do not have enough info about particular instances**
  - **Non-robust features contain remaining info**

Robust features



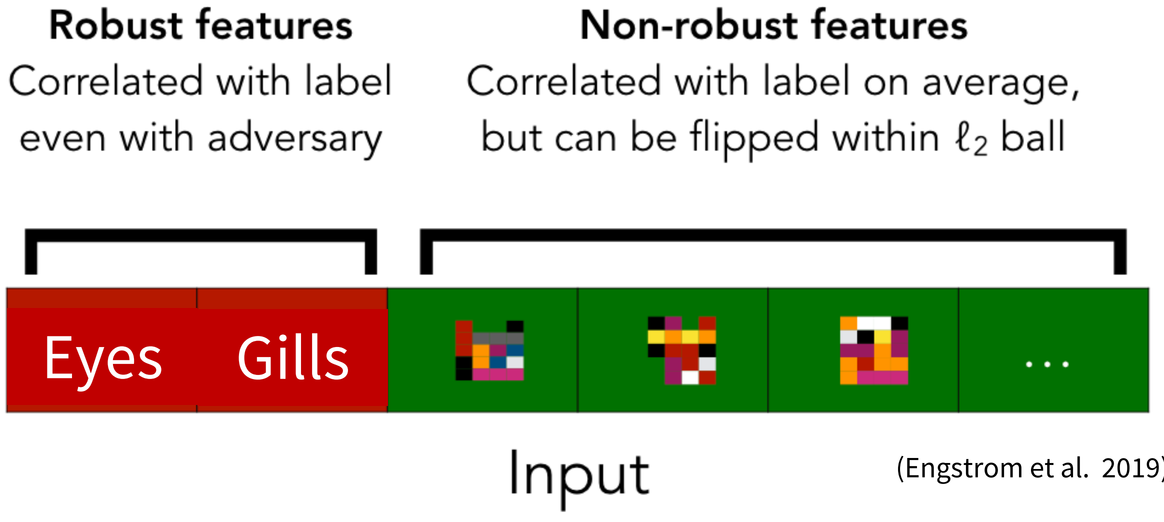
(Engstrom et al. 2019)

Robust + non-robust features

Robust and non-robust features

- Images comprised of robust and non-robust features (Ilyas et al. 2019)
- **Key insight: Robust features do not have enough info about particular instances**
  - **Non-robust features contain remaining info**
  - **Objective: Augment non-robust features with robust features without losing robustness**

Robust features

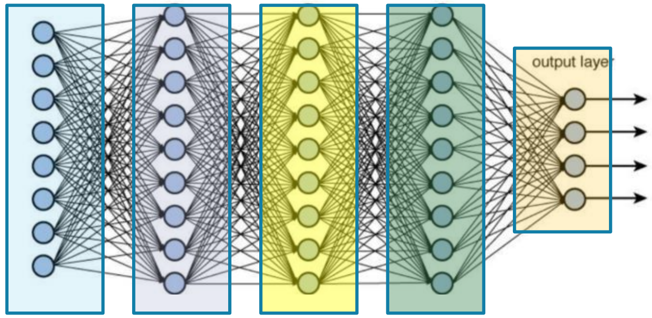


(Engstrom et al. 2019)

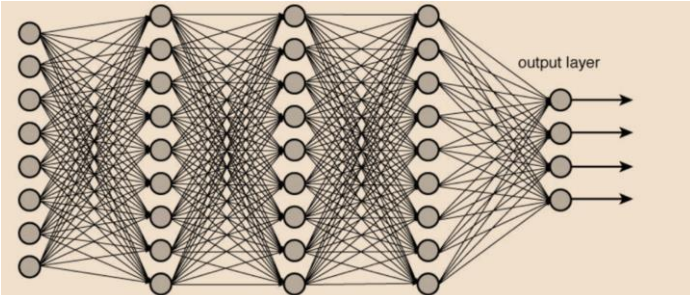
Robust + non-robust features

Composite ensembling

Robust



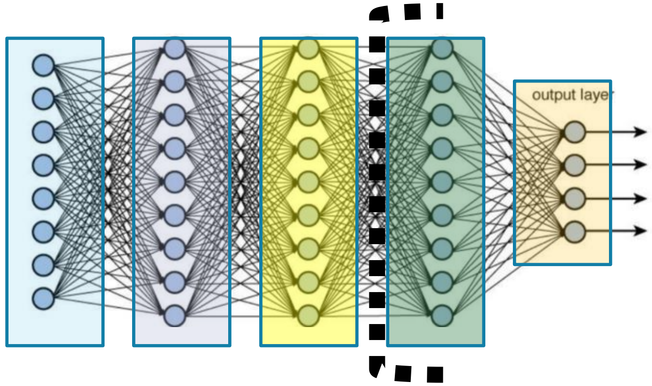
Natural



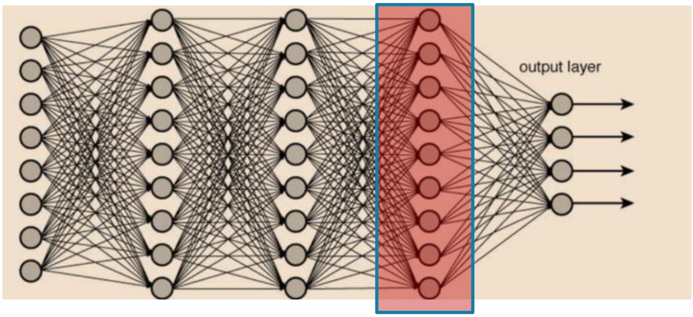
Composite ensembling

Extract Last Layers

Robust



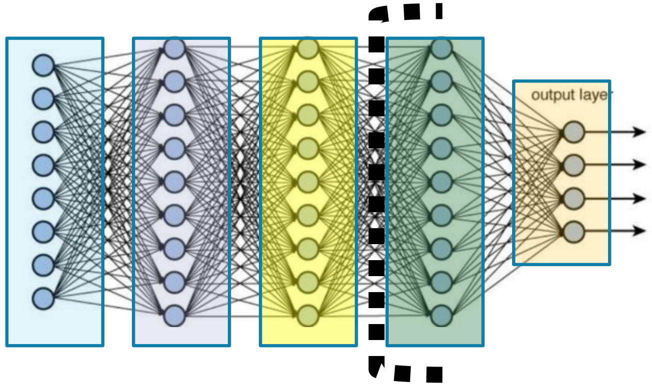
Natural

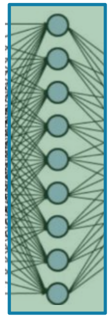


Composite ensembling

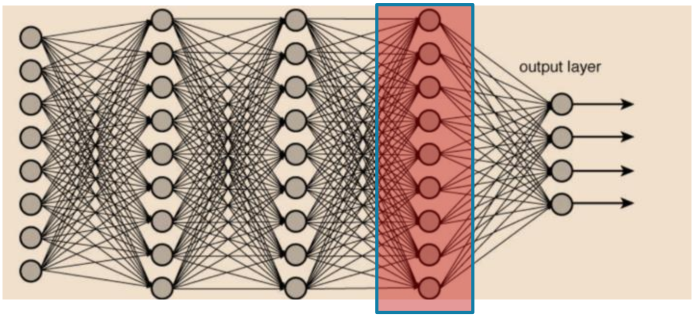
Replicate Last Robust Layer

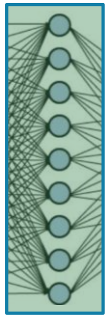
Robust





Natural

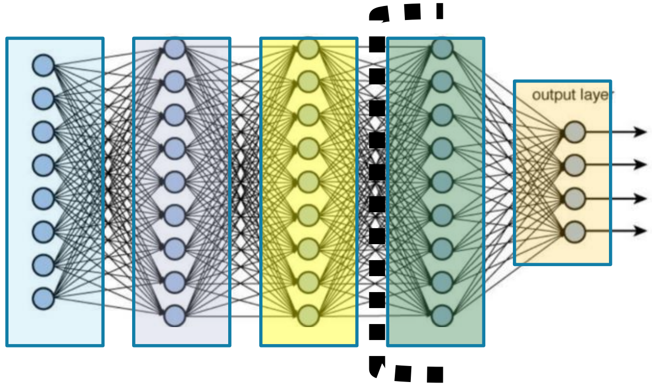




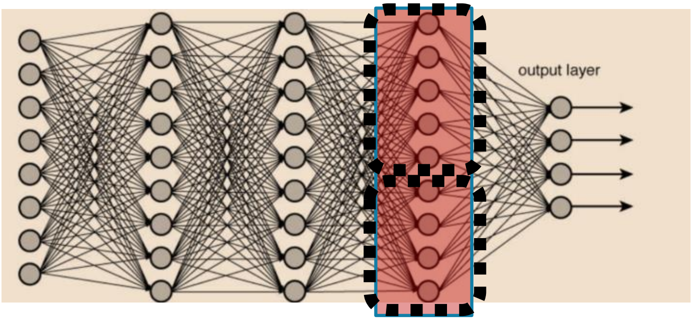
Composite ensembling

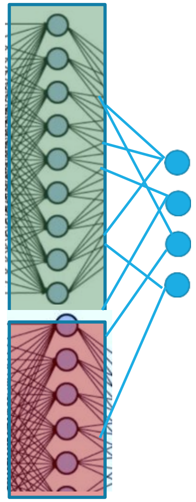
Replicate Last Robust Layer + Attach Natural Last Layer + Train Last Composite Layer Independently

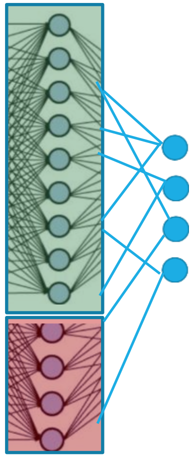
Robust



Natural





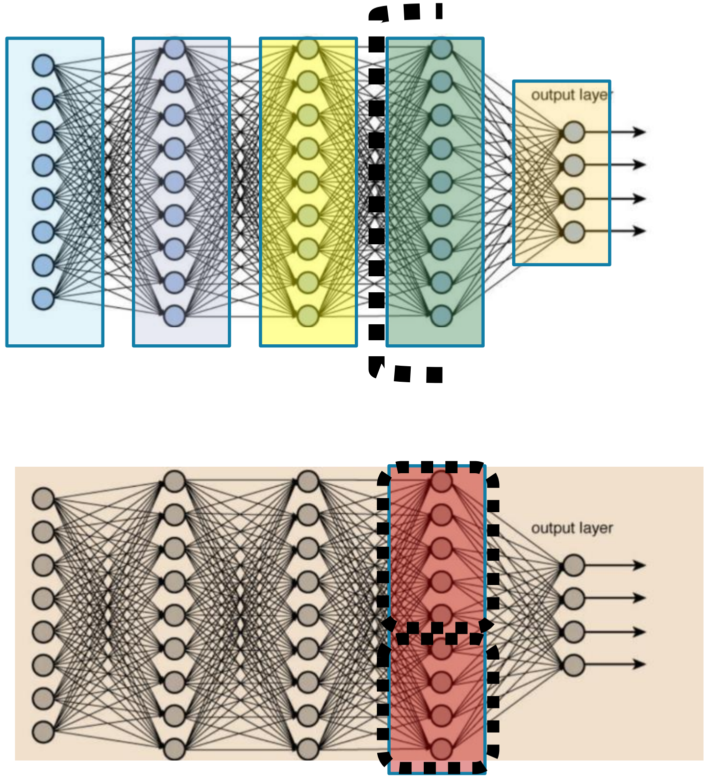


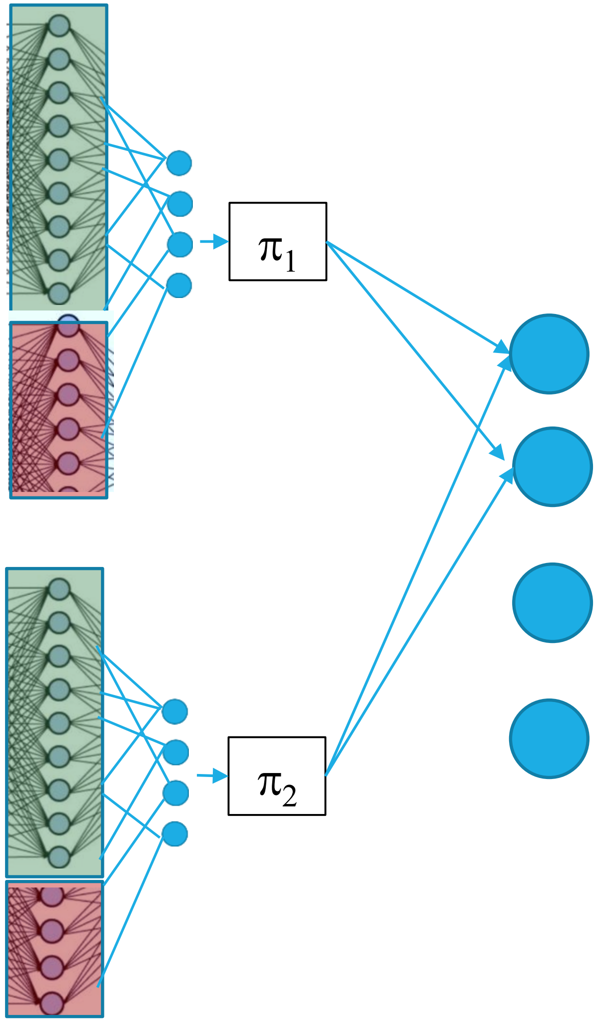
Composite ensembling?

Replicate Last Robust Layer + Attach Natural Last Layer + Train Last Composite Layer Independently

Robust

Natural

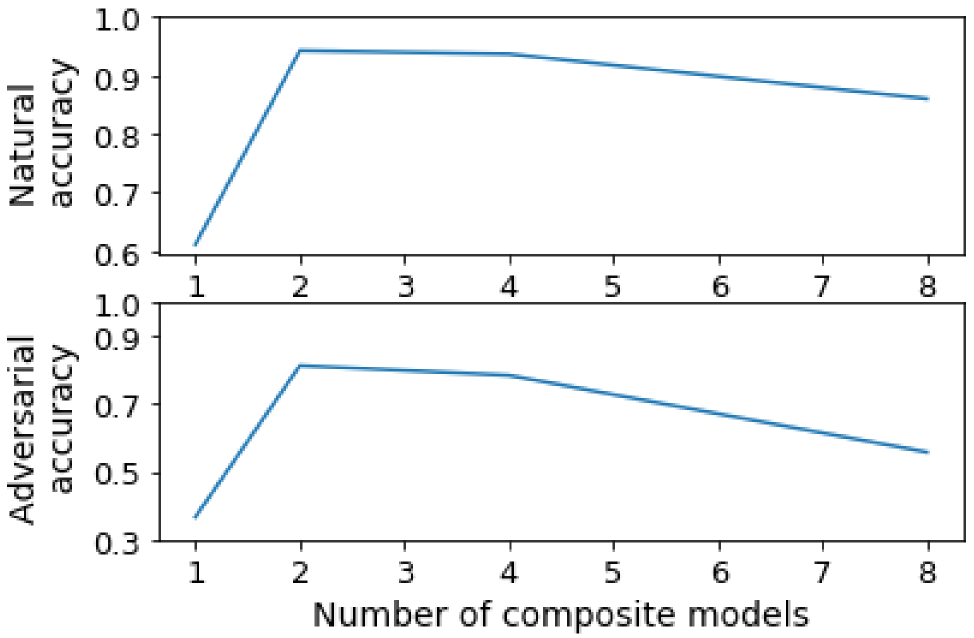




Composite prediction = ensemble average

Composite acc. $\geq$ single robust model acc.

Composite ensembling: Results

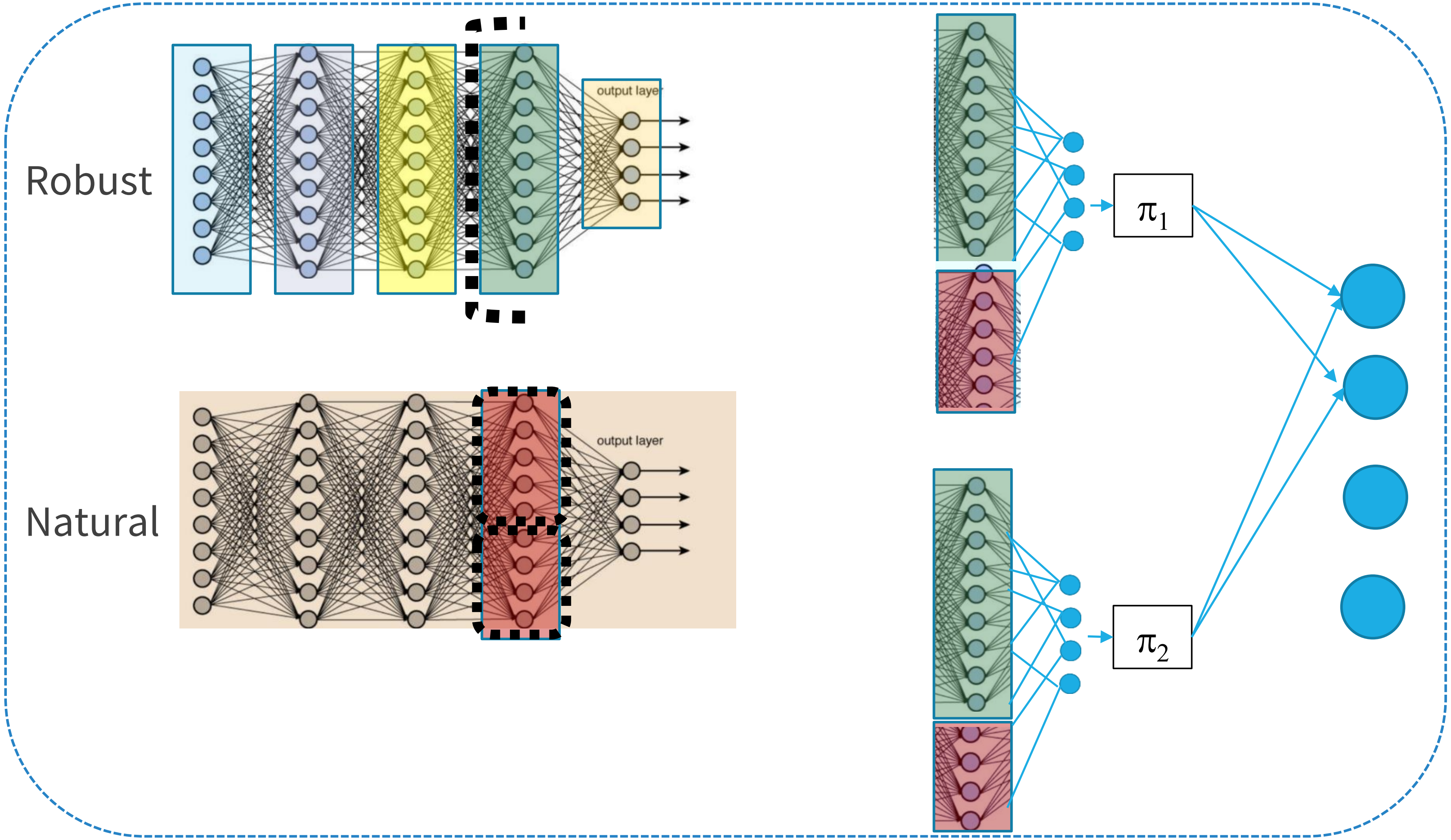


1-composite (**naïve stacking**) is a disaster!

2-composite (**random splitting and stacking independently**) is optimal size for both natural and adversarial

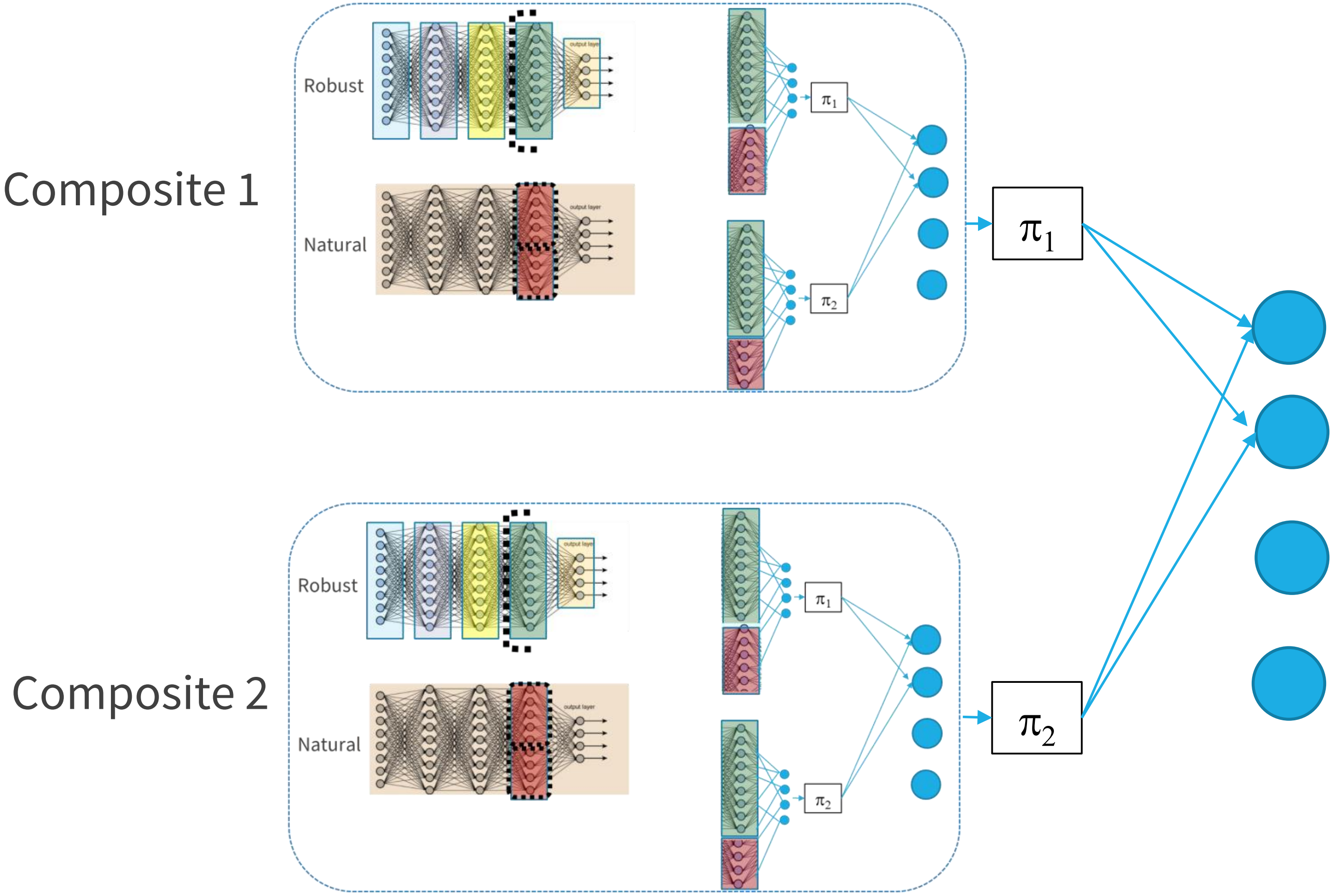
	Single non-robust model	Single robust model	Robust ensemble (20 models)	2-Composite of robust and non-robust features
Natural test	93.2%	82.7%	86.1%	94.2%
Adv. Test ($\varepsilon = 0.031$, $k = 14$)	0.0%	51.8%	58.0%	81.2%

Meta-composite ensembling



Take many composite ensembles

Meta-composite ensembling



Meta Composite prediction = ensemble of composites

Meta-composite ensembling

- Combine *M* independently trained composite models

	Single non-robust model	Single robust model	Robust ensemble (20 models)	2-Composite of robust and non-robust features	5 meta 2-composites
Natural test	93.2%	82.7%	86.1%	94.2%	94.9%
Adv. Test ($\varepsilon = 0.031$, $k = 14$)	0.0%	51.8%	58.0%	81.2%	83.5%

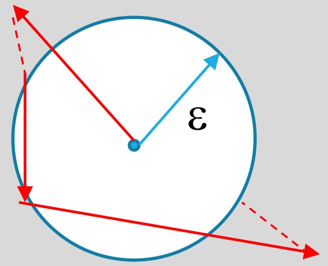
Key insights and Conclusions

- Robust ensembling outperforms single models
  - Choosing models randomly forces adversary to use average strategy
  - Different models may mispredict the same way, but require different perturbations
- Proposed composite and meta-composite models
  - Re-incorporate non-robust features
  - Significantly improve natural and adversarial accuracy
  - Adversary may be hamstrung trying to attack non-robust component only
    - (Robust natural approximately equal to meta-composite adversarial accuracy)
- Bridged natural and adversarial accuracy gap
  - Appears to resolve tension between robustness and accuracy suggested by Tsipras et al. (2018)
    - Non-robust features are an important component of achieving natural accuracy
  - Meta-composites achieve SOTA natural accuracy compared to ResNet18-based architectures

Future work

- Tune ensemble weights (π) and composite parameters
- PGD: Gradient ascent with projection onto ball
  - Tuning parameters: learning rate (η), attack steps (k), random restarts
  - Random restarts did not decrease performance
  - Attack steps and learning rate changed performance but not significantly
  - Tested along a 2D grid of attack steps and learning rate
- Validation with other adversarial attacks such as Carlini-Wagner (Carlini and Wagner 2017)
- Use meta-composite framework to improve natural accuracy outside adversarial context

$$\hat{\delta} = \underset{||\delta||<\epsilon}{\operatorname{argmax}}\, L(\theta, x + \delta, y)$$



Acknowledgements

- Guillaume Leclerc and Prof. Aleksander Mądry
- Logan Engstrom, Andrew Ilyas, and the rest of Mądry Lab
- Dr. Slava Gerovitch and Prof. Srini Devadas

Questions?