

Revisiting Ensembles in an Adversarial Context: Improving Natural Accuracy

Aditya Saligrama

Mentors: Guillaume Leclerc, Prof. Alexander Madry (MIT CSAIL Theory of Computing)

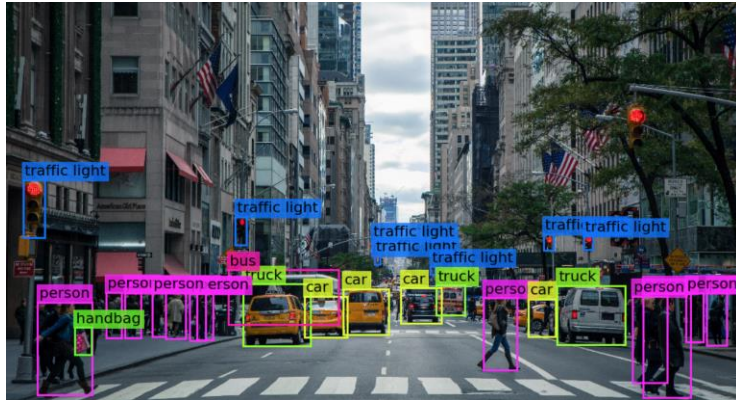
10th Annual MIT PRIMES Conference

June 7, 2020

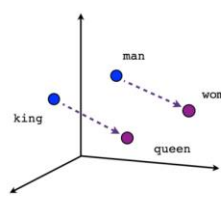
Deep learning and adversarial examples

Deep learning

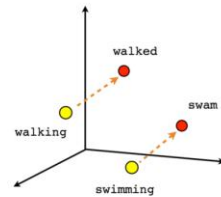
- Has become ubiquitous in the last few years and can outperform humans on some tasks



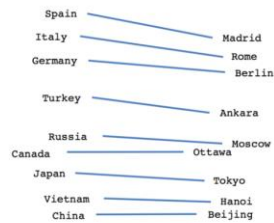
(DeepAI 2019)



Male-Female

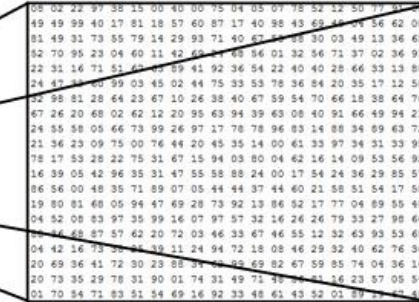
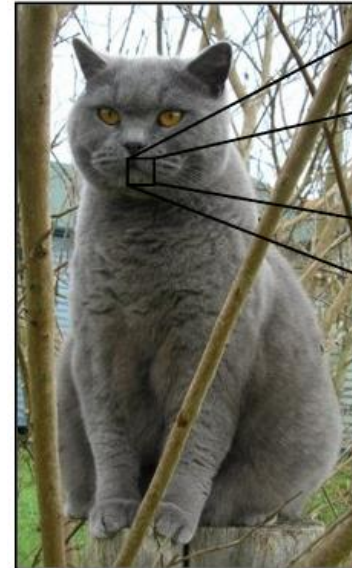


Verb tense



Country-Capital

(Ruizendaal 2017)



What the computer sees

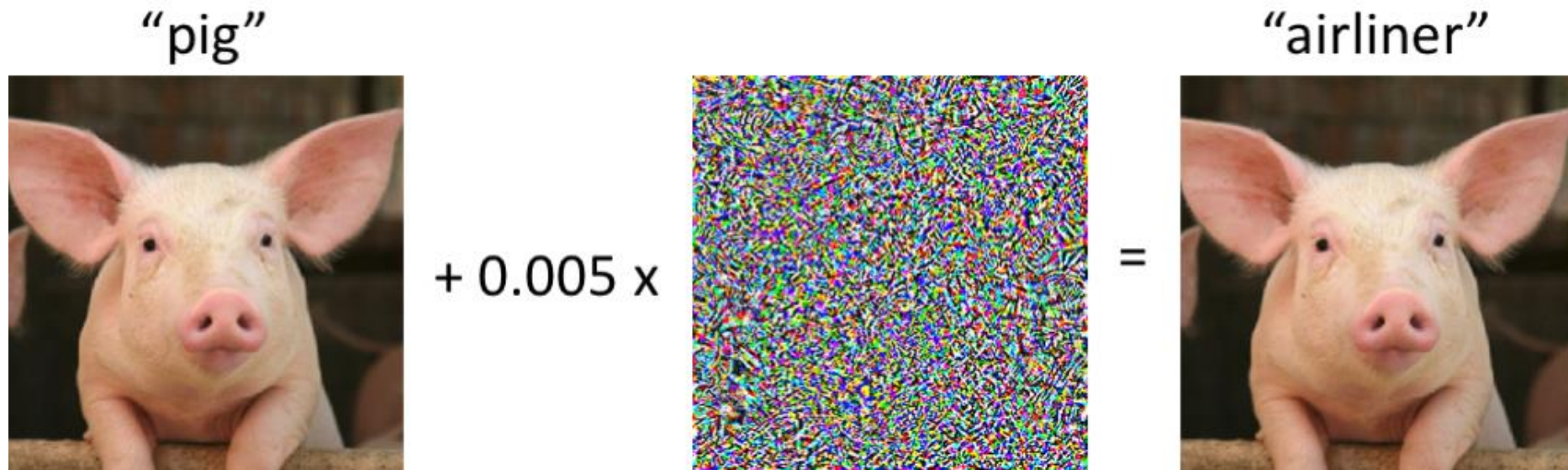
image classification → 82% cat
15% dog
2% hat
1% mug

(Karpathy 2015)

Adversarial attacks

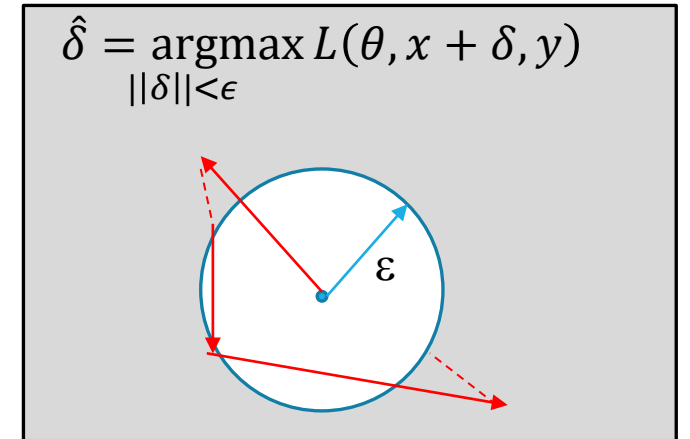
- Modify image in a set S , such as L2-ball of size ϵ , to maximize loss L
 - Imperceptible to human observer
 - Fools deep learning models

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



Adversarial attacks

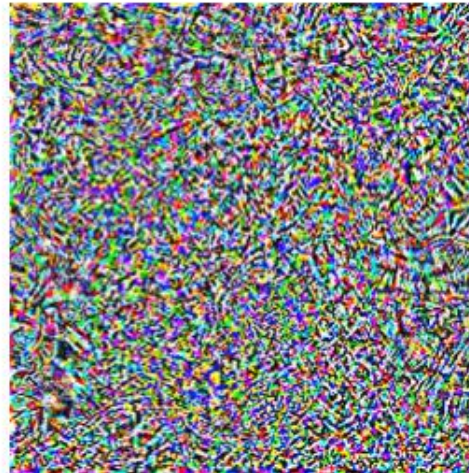
- Modify image in a set S , such as L2-ball of size ϵ , to maximize loss L
 - Imperceptible to human observer
 - Fools deep learning models
- Many ways of synthesizing adversarial examples:
 - Such as PGD - projected gradient descent (Mądry et al. 2017)



“pig”



+ 0.005 x



=

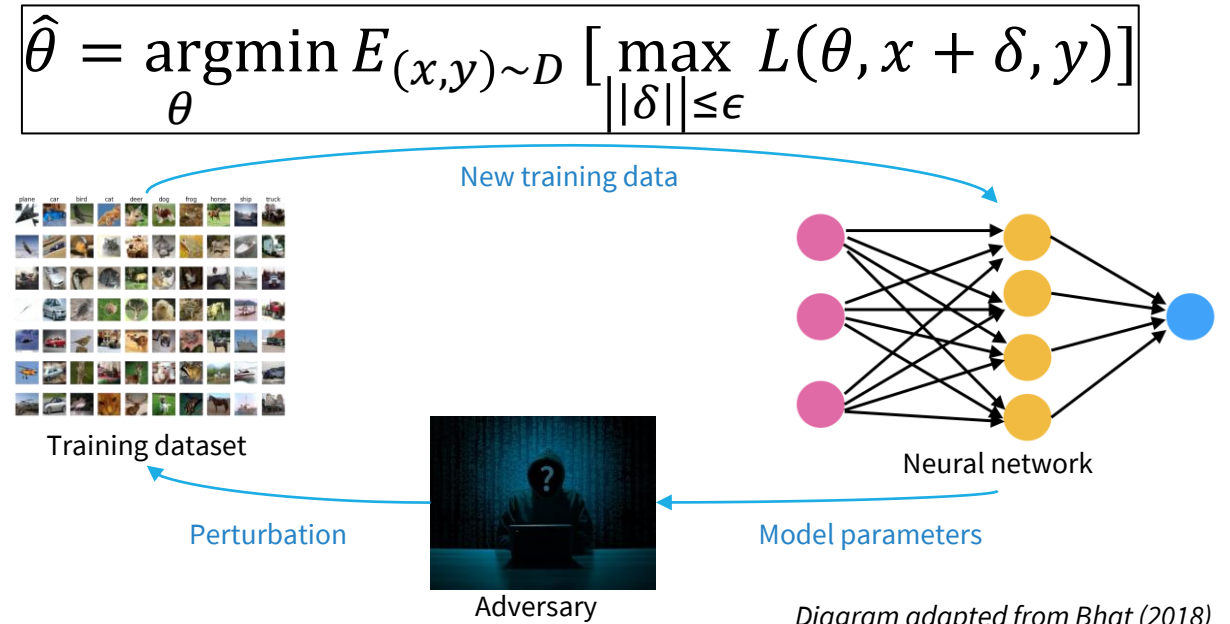


“airliner”

(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...



Robust training

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

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

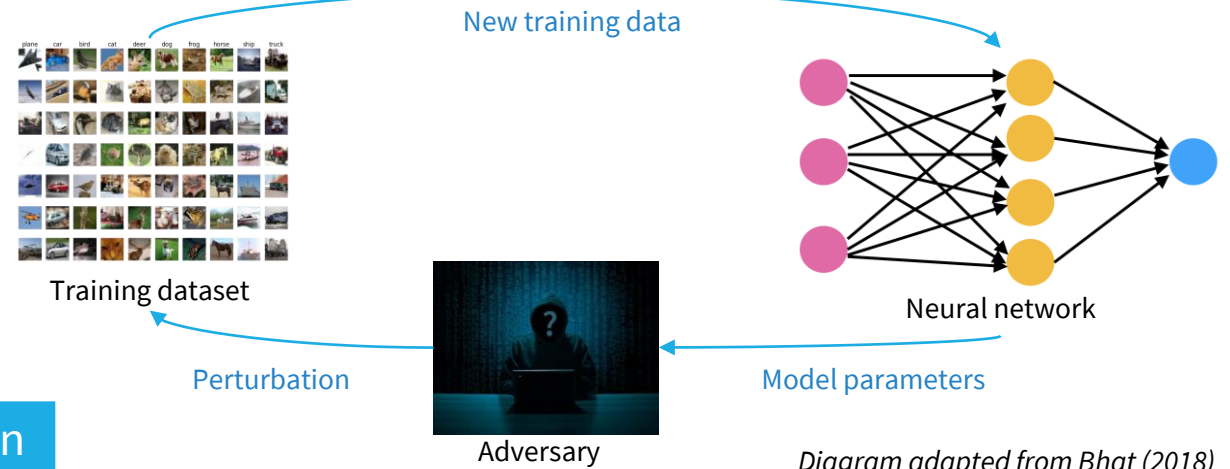


Diagram adapted from Bhat (2018)

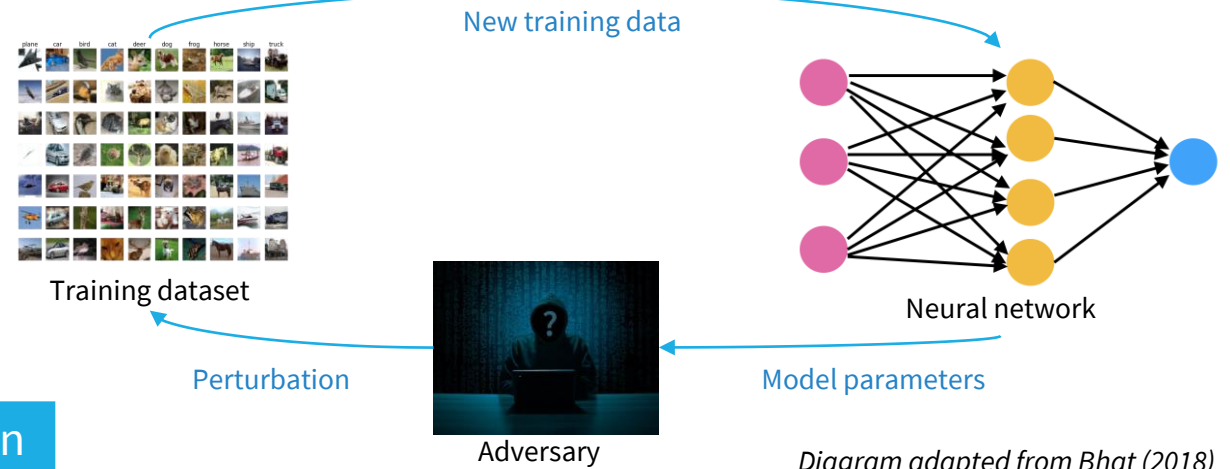
ResNet18 models (He et al. 2015)
trained on CIFAR10

	Natural train	Robust train ($\epsilon=0.5$)
Natural test		
Adv. test ($\epsilon=0.5$)		

Robust training

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

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



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

	Natural train	Robust train ($\epsilon=0.5$)
Natural test	95%	88%
Adv. test ($\epsilon=0.5$)	0%	69%

Metrics

- Assess resistance to adversarial attacks at multiple attack strengths
 - Adversary can choose any arbitrary attack strength against deployed model
- We define AUC metric as

$$AUC(\epsilon_{target}) = \frac{1}{\epsilon_{target}} \int_0^{\epsilon_{target}} \mathcal{A}(\epsilon) d\epsilon.$$

- In practice, evaluate as a Riemann sum
- Use this metric in addition to assessing accuracy at defined attack strengths

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	88%	94%	93%
Adv. test	69% ($\epsilon=0.5$)	0%	30% ($\epsilon=0.02$)

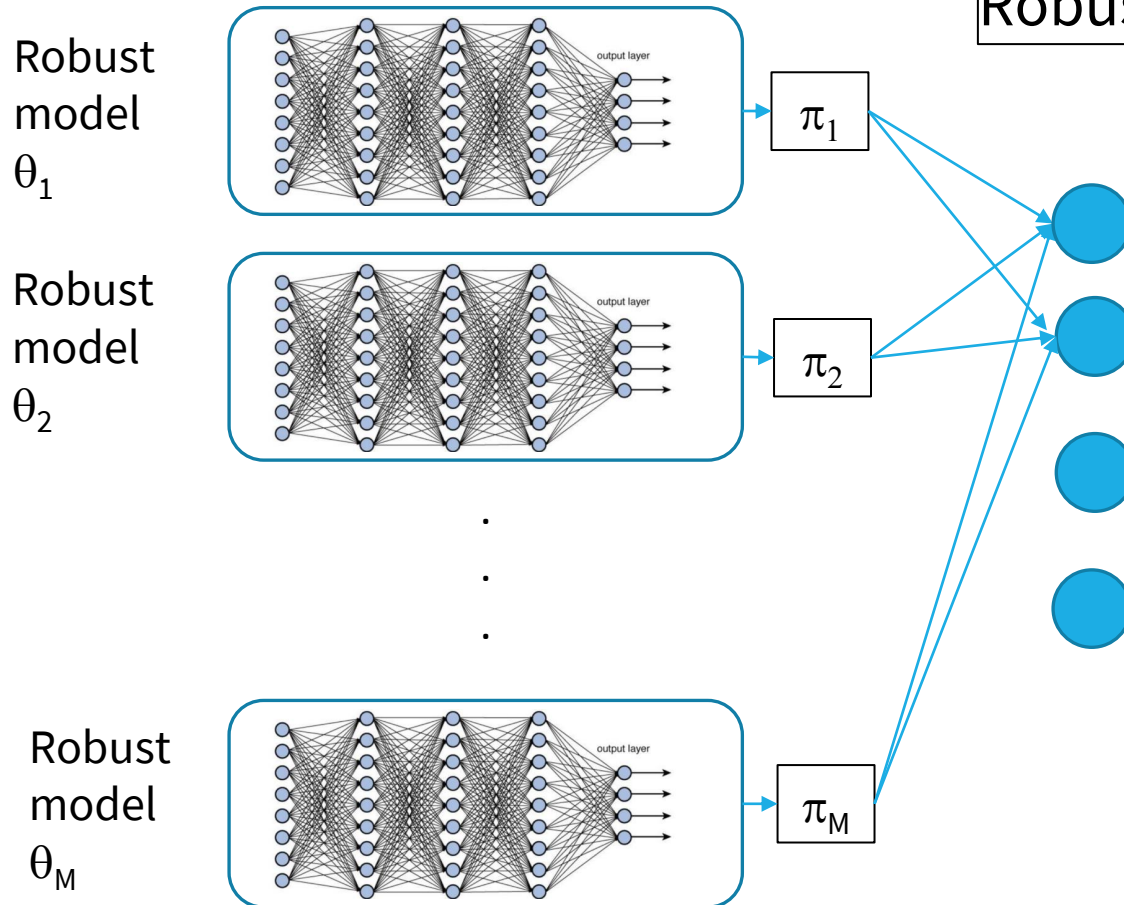
Our proposed methods

Robust ensembling

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

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

Robust training with initialization seed i



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

$\theta_i(x, y)$: 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 $\delta_1 \dots \delta_S$ ($S \rightarrow \infty$) with probability $q_1 \dots q_S$

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

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.

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

Adversary strategy

	θ_1	θ_2	θ_3
δ_1	Loss		
δ_2			
δ_3			

How to understand ensembles?

Value of the game (discrete):

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

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

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.

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

Adversary strategy

	Player strategy		
	θ_1	θ_2	θ_3
δ_1	Loss		
δ_2			
δ_3			

robust ensemble loss \leq single robust model loss
Why? Choose \mathbf{q} to focus on single model

This allows accuracy to increase per model in the ensemble for a given ϵ

Robust and non-robust features

- Images comprised of robust and non-robust features (Ilyas et al. 2019)
- **Key insight 1: Robust features do not have enough info about particular instances**
 - **Non-robust features contain remaining info**

Robust features

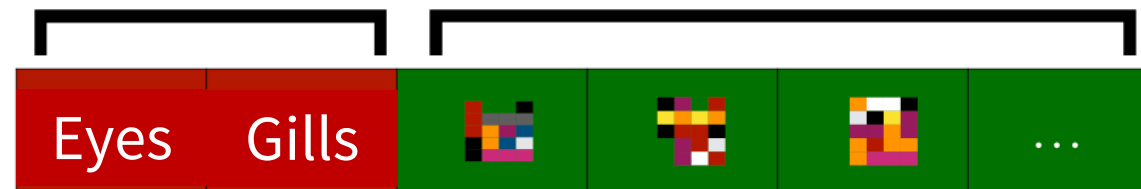


Robust features

Correlated with label even with adversary

Non-robust features

Correlated with label on average, but can be flipped within ℓ_2 ball



Input

(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)
 - Training at lower ϵ means less resistance to non-robust features and better natural accuracy
- **Key insight 1: 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**
- **Key insight 2: Lower train ϵ confers better natural accuracy at the cost of robustness**
 - **Objective: Combine with ensembling to maintain robustness with better natural accuracy**

Robust features

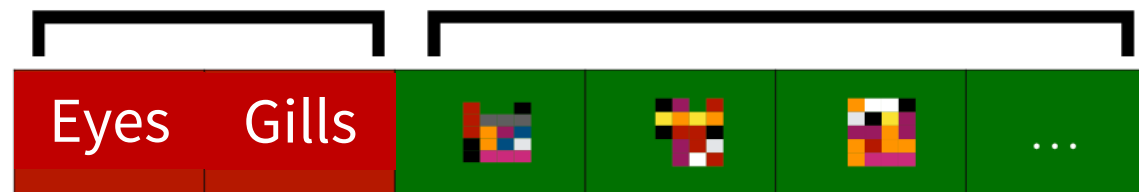


Robust features

Correlated with label even with adversary

Non-robust features

Correlated with label on average, but can be flipped within ℓ_2 ball



Input

(Engstrom et al. 2019)

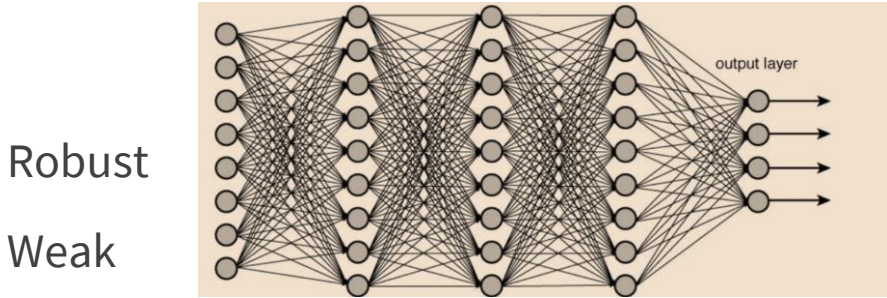
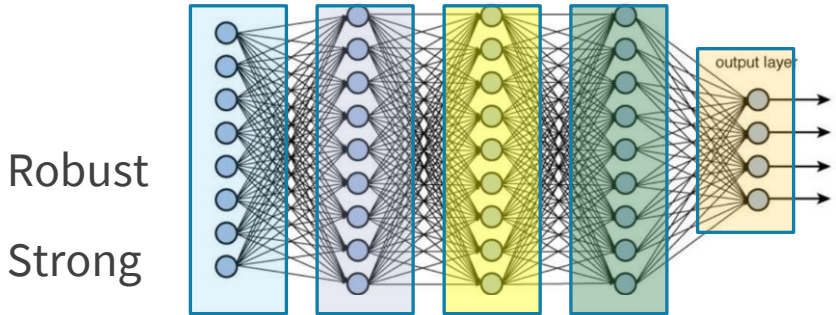
Robust + non-robust features



Robust ensembling: Results

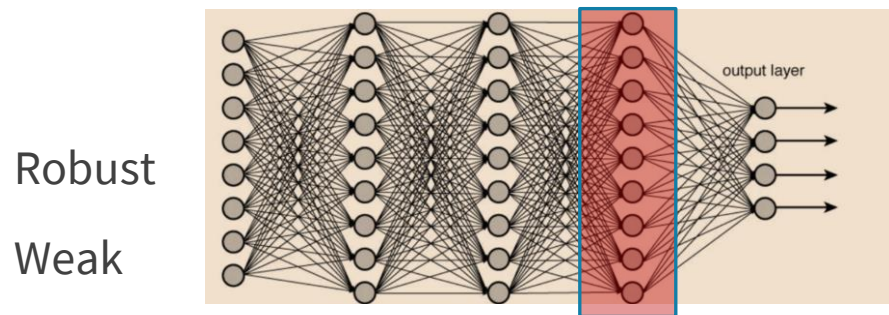
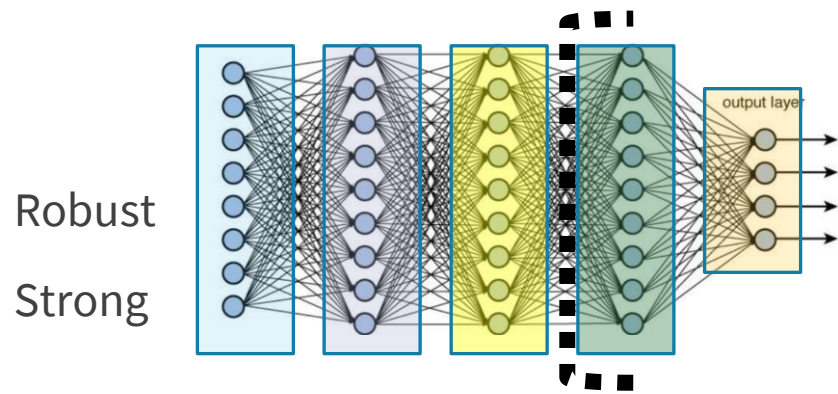
Number of models (train $\epsilon = 0.5$)	Natural accuracy	Adversarial accuracy ($\epsilon = 0.5$)		Single non-robust model	Single robust model (train $\epsilon = 0.5$)	Robust ensemble (8 models, train $\epsilon = 0.22$)
1	88.30%	68.73%	Natural test	94.6%	88.3%	94.0%
2	88.92%	71.19%	Adv. Test ($\epsilon = 0.5, k = 7$)	0.4%	68.7%	68.8%
4	89.07%	72.53%	AUC(0.5) w/4 increments	0.067	0.767	0.781
8	89.36%	73.08%				
12	89.28%	73.34%				
16	89.18%	73.37%				

Composite ensembling



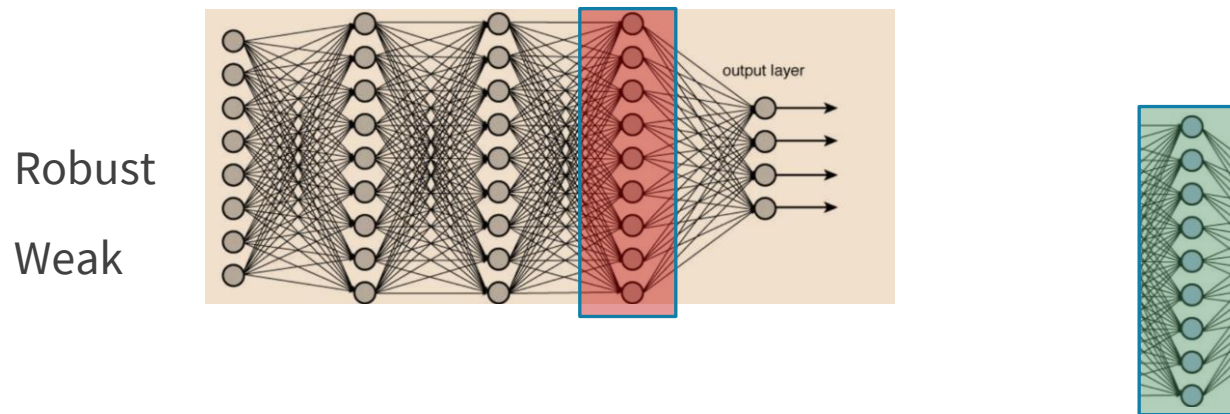
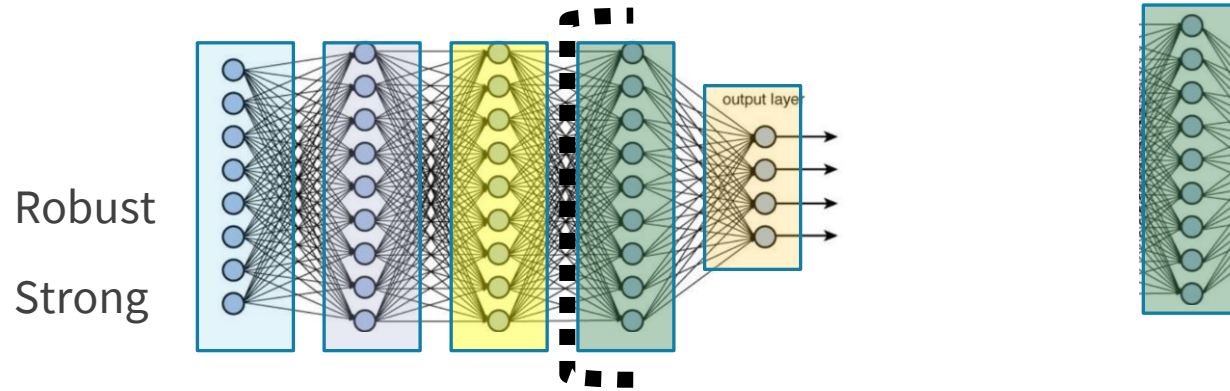
Composite ensembling

Extract Last Layers



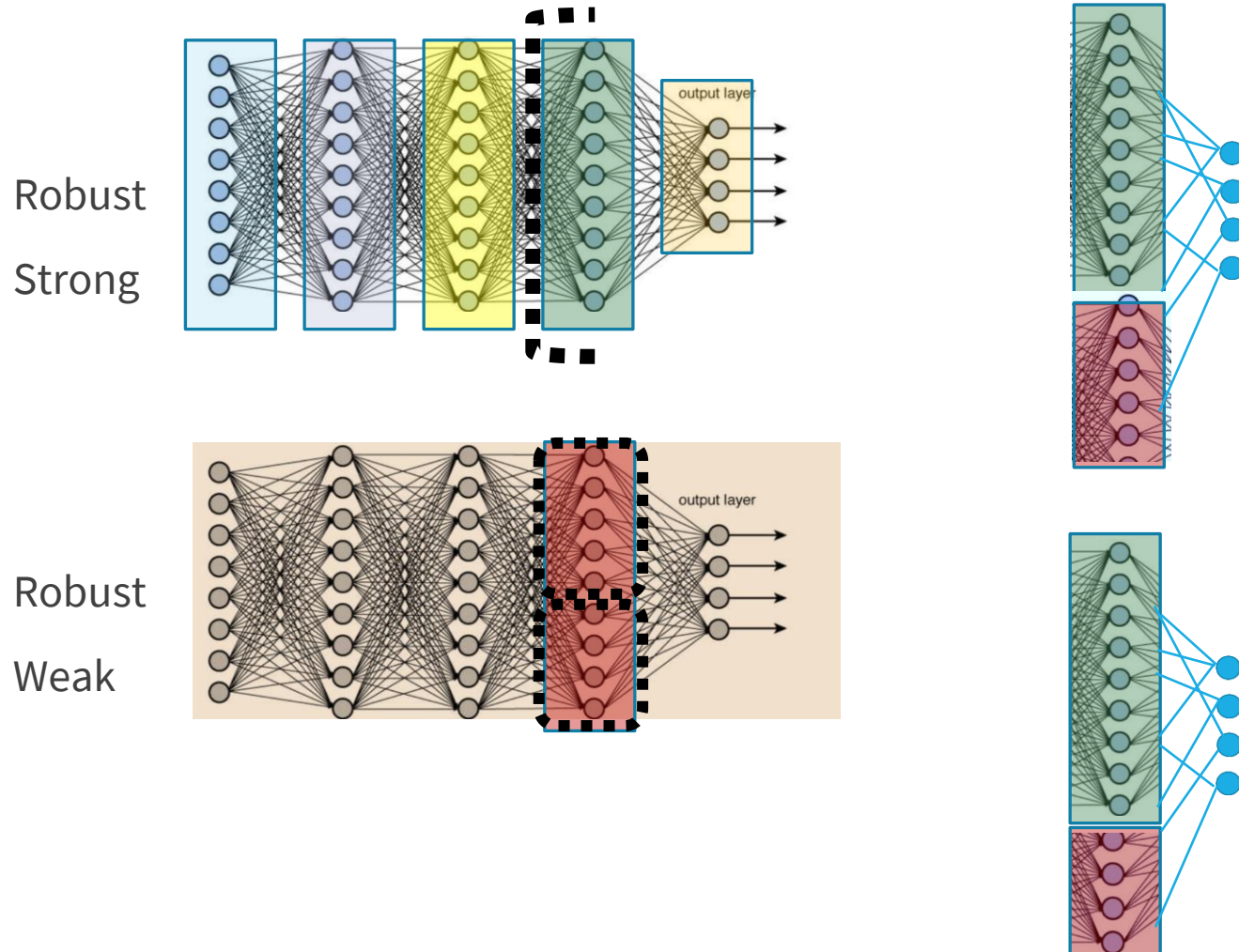
Composite ensembling

Replicate Last Robust Layer



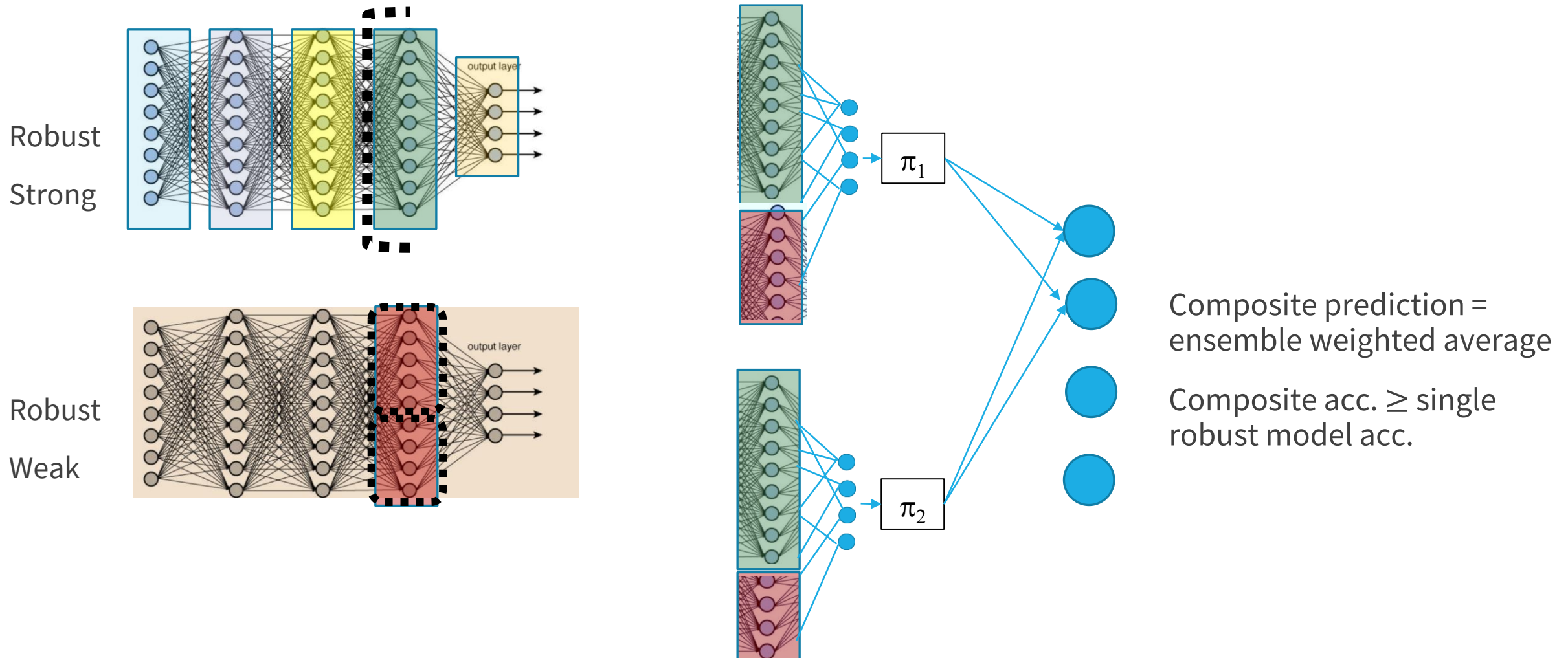
Composite ensembling

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



Composite ensembling?

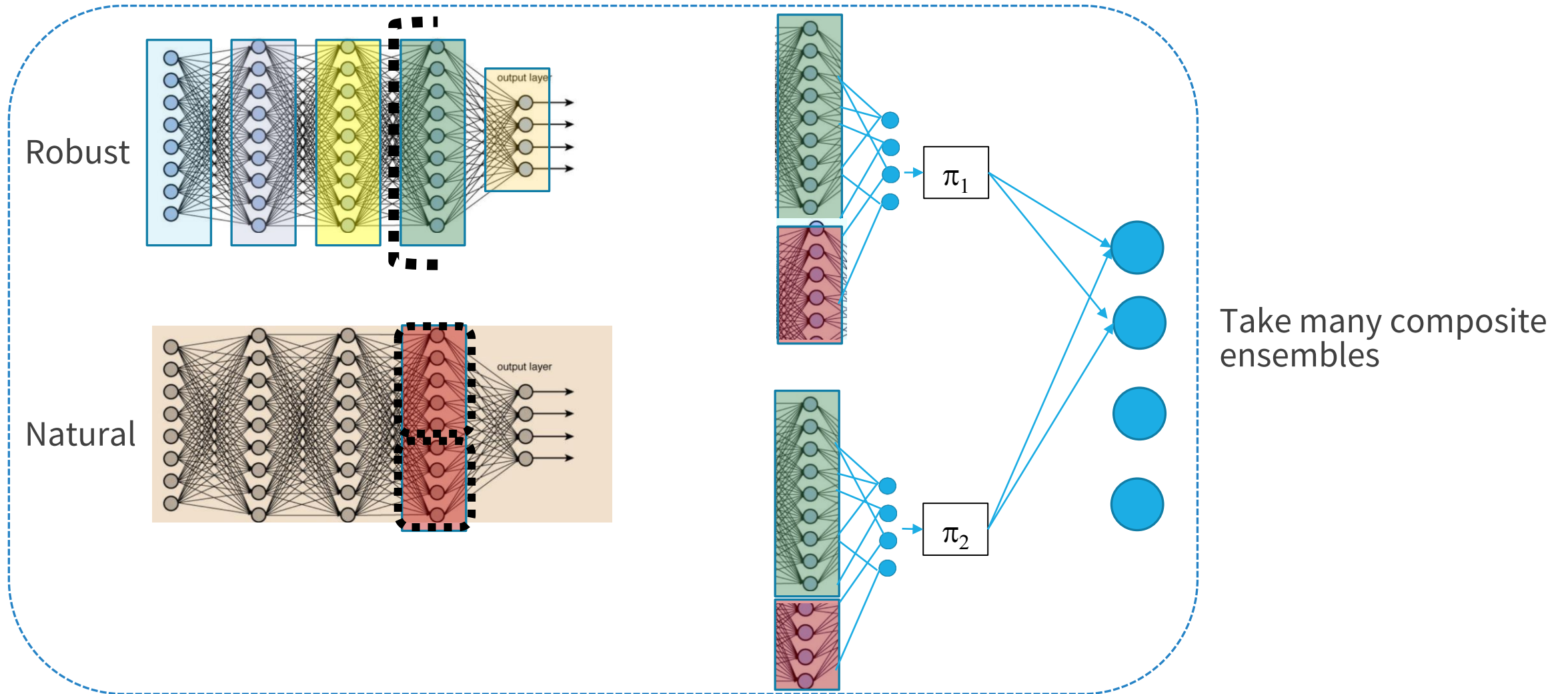
Replicate Last Robust Layer + Attach Natural Last Layer + Train Last Composite Layer Independently



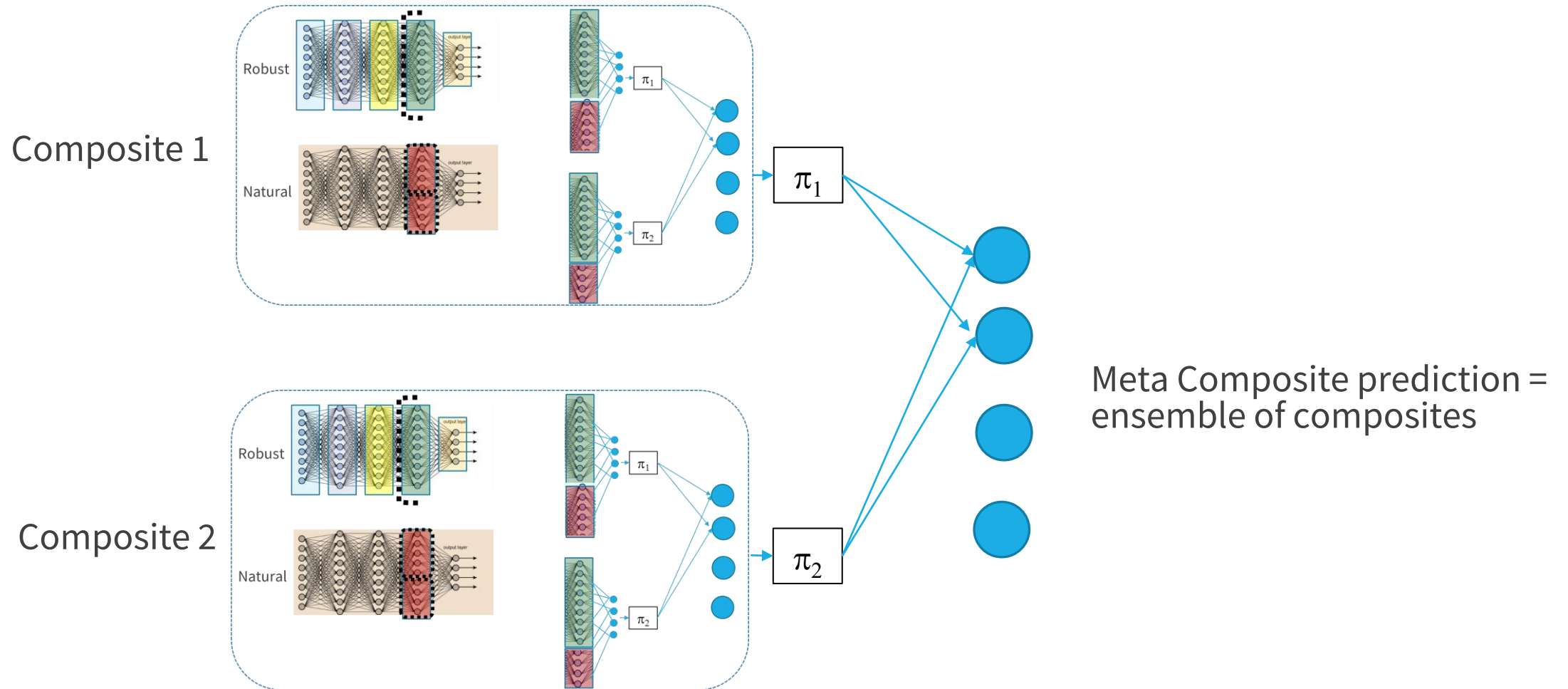
Composite ensembling: Results

	Single non-robust model	Single robust model (train $\epsilon = 0.5$)	Robust ensemble (8 models, train $\epsilon = 0.22$)	1-composite (train $\epsilon = 0.4, 0.05$ trained at $\epsilon = 0.4$)
Natural test	94.6%	88.3%	94.0%	91.4%
Adv. Test ($\epsilon = 0.5, k = 7$)	0.4%	68.7%	68.8%	68.0%
AUC(0.5) w/4 increments	0.067	0.767	0.781	0.769

Meta-composite ensembling



Meta-composite ensembling



Meta-composite ensembling

- Combine M independently trained composite models

	Single non-robust model	Single robust model (train $\epsilon = 0.5$)	Robust ensemble (8 models, train $\epsilon = 0.22$)	1-composite (train $\epsilon = 0.4, 0.05$ trained at $\epsilon = 0.4$)	2x 1-composite Weighted average
Natural test	94.6%	88.3%	94.0%	91.4%	91.6%
Adv. Test ($\epsilon = 0.5, k = 7$)	0.4%	68.7%	68.8%	68.0%	70.0%
AUC(0.5) w/4 increments	0.067	0.767	0.781	0.769	0.783

Key insights and Conclusions

- AUC metric to evaluate robustness of models
 - Allows us to assess robustness at multiple attack strengths
- 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
 - Allows us to decrease train ϵ , therefore increasing natural accuracy at a given level of robustness
- Proposed composite and meta-composite models
 - Re-incorporate non-robust features
 - Improves on AUC metric compared to single models while using less models than robust ensembling

Future work

- Validation with other adversarial attacks such as Carlini-Wagner (Carlini and Wagner 2017)
- Use meta-composite framework to improve natural accuracy outside adversarial context

Acknowledgements

- Guillaume Leclerc and Prof. Aleksander Mądry
- Logan Engstrom, Andrew Ilyas, and the rest of Mądry Lab
- Dr. Slava Gerovitch and Prof. Srinivas Devadas, for supporting the PRIMES program and this research