FACOLTÀ DI INGEGNERIA DELL'INFORMAZIONE, INFORMATICA E STATISTICA

Corso di Laurea Magistrale in Computer Science

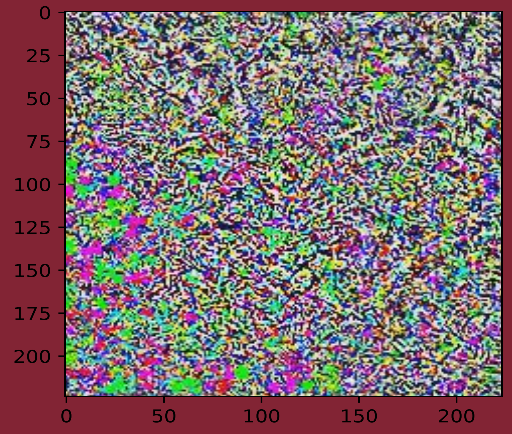
Thesis Defense

Robustness Of Deep Neural Networks Using Trainable Activation Functions

Candidate:
Federico Peconi

Thesis Advisor:
Prof. Simone Scardapane

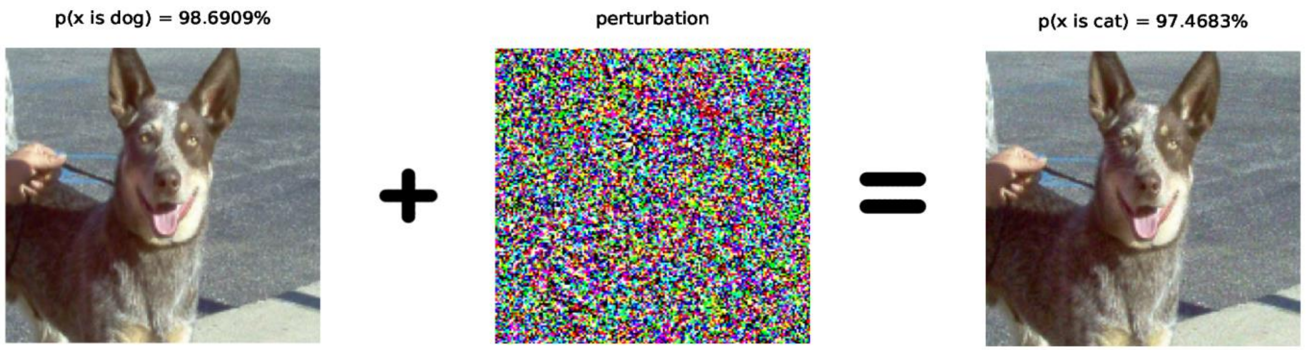
Academic Year:
2019/2020



SAPIENZA
UNIVERSITÀ DI ROMA

Intriguing Properties Of Neural Networks

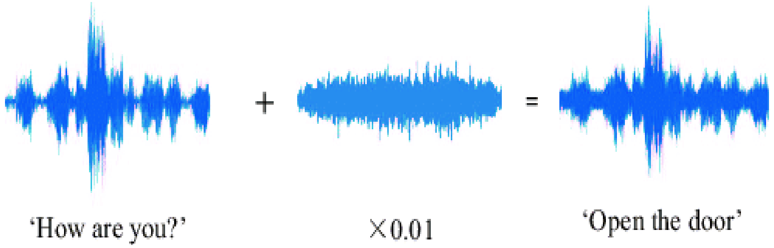
- Neural Networks are brittle w.r.t. *non-random,* small perturbations of the input *[Christian Szegedy, Wojciech Zaremba et al. 2014]*



- *Adversarial Examples* can be found in several different domains other than image classification:

  1. Natural Language Processing *[Moustafa Alzantot et al. 2018]*
  2. Automatic Speech Recognition *[Yao Qin et al. 2019]*

  and so on …



[Yuan Gong, Christian P. 2018]

- Can be computed through *backpropagation* and *gradient ascent,* optimizing:

$$\max_{\delta \in \Delta} \ell\left(f_\theta(x+\delta), y\right)$$

Agenda

- Motivations and Current Effective(?) Defenses
- Kernel-Based Activation Functions (KAFs)
- The role of activation functions in Adversarial Training
- VGG architectures-inspired results
- ResNet20 results
- Contributions and Future Works

Motivations

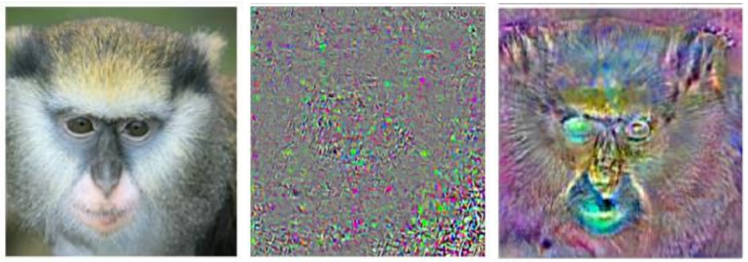
- High Security Concerns:
  - *Safety-critical* applications not yet reliable
  - Malware, intrusion detection
  - Face recognition

  …

[Mahmood Sharif, Sruti Bhagavatula et al. 2016][

[Kevin Eykholt et al. 2018]

- We want to develop human-aligned models:
  - Basic idea: humans do not suffer from such brittleness
  - *Robustness* and *Interpretability* seem to be strictly connected
  - "Adversarial examples are not bugs, they are features" *[Madry et al. 2019]*

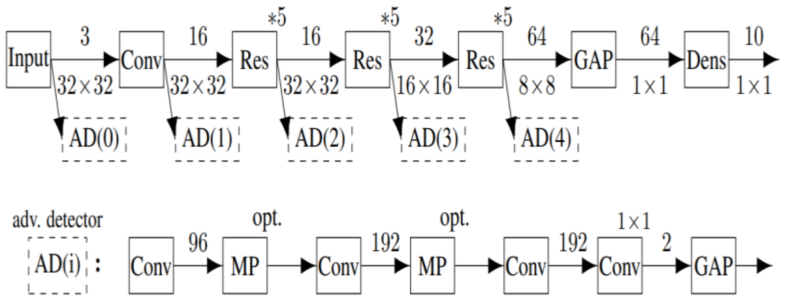


[Dimitris Tsipras, Shibani Santurkar et al. 2019]

Defenses

Detection Methods

- ❑ Add a “patch” to the network which is in turn another neural network.
- ❑ Such NN, is used as a *detector* to spot adversarial perturbations from the analysis of layer’s *statistics*.
- ❑ Training samples extended with a *binary label* denoting the presence of an adversary
- ☺ Easy to perform, tractable computational overhead
- ☹ Tend to *overfit* on the specific attack used during training



[H. Metzen et al. 2017]

Adversarial Training (AT)

- ❑ Train to approximate the *min, max*:

$$\min_{\theta} \frac{1}{|S|} \sum_{x,y \in S} \max_{\delta \in \Delta} \ell\left(f_\theta(x+\delta), y\right)$$

- ❑ Inner maximization is computed crafting adversarial examples on-the-fly
- ❑ *Running-time* boosted with novel techniques: *[E. Wong, L. Rice 2020]*
- ☺ Only *unbroken* known defense, believed to be *truly robust*
- ☹ Robustness accuracy up to 50%
  Std Accuracy ~ 80 %
  (CIFAR10, delta <= 8/255)

Provable Robustness

- ❑ Aims to give a *formal certification* of the robustness of a model.
- ❑ We want to guarantee that, for any allowed perturbation our model will always predict the correct class
- ❑ Problem approached through different techniques:
  - Lipschitz Regularization
  - Linear Programming
  - Semi-Definite Programming
  - Randomized Smoothing
- ☺ Provides provably robust models (up to computationally negl. probability)
- ☹ Currently intractable for large-scale networks, architecture-specific implementations

Kernel-Based Activation Functions

- Along with 'traditional' *fixed* activation functions (e.g. ReLU, ELU, Tanh, Swish, ..), we can devise *adaptive* activation functions whose shape is learned through the optimization of parameters present in their formulation:

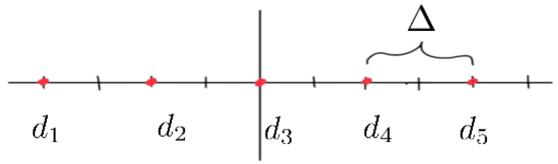
  - Adaptive Piece-Wise Linear Activation Functions (APLs) *[F. Agostinelli 2014]*
  - Spline Activation Functions (SAFs*) [S. Scardapane, M. Scarpiniti 2017]*
  - Maxout layers *[I. Goodfellow et al. 2013]*

- In *[S. Scardapane, S. V. Vaerenbergh 2017],* authors introduce a novel class of trainable activation functions called *kernel-based activation functions (KAFs):*

$$\mathrm{KAF}(x) = \sum_{i=1}^{D} \alpha_i \kappa\left(x, d_i\right)$$

  - Where $\{\alpha_i\}_1^D$ are the weights to train, called *mixing coefficients.*
  - $\kappa : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is a 1-dimensional kernel method.
  - $\{d_i\}_1^D$ are the kernel's dictionary elements and D the dictionary size, which is user-defined.
  - First derivatives: $\frac{\partial\,\mathrm{KAF}(x)}{\partial \alpha_i} = \kappa\left(x, d_i\right)$, $\quad \frac{\partial\,\mathrm{KAF}(x)}{\partial x} = \sum_{i=1}^{D} \alpha_i \frac{\partial \kappa(x, d_i)}{\partial x}$

KAF Design Details

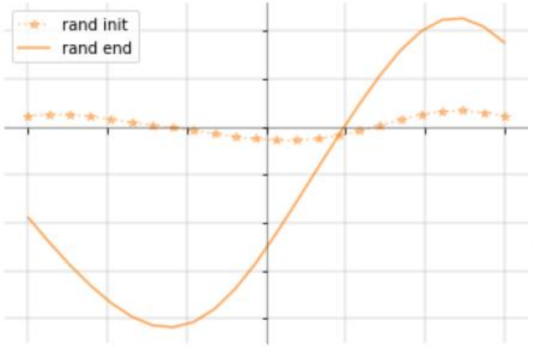


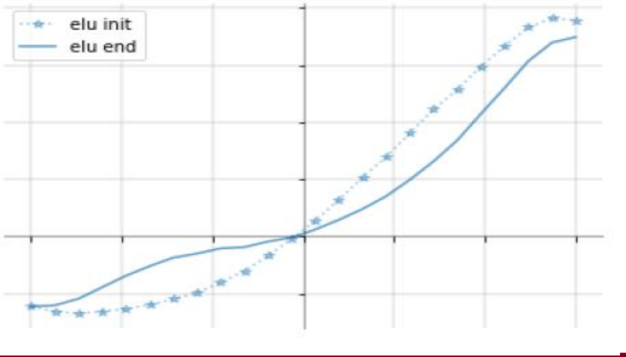
- Dictionary elements uniformly sampled around 0 with step size Δ
- *Gaussian* kernel: $k(x, d_i) = \exp\{-\gamma(x - d_i)^2\}$
  - Heuristic $\gamma := \frac{1}{6\Delta^2}$
  - Guarantees a *local* effect of the mixing coefficients w.r.t. the overall shape $\Rightarrow$ ease optimization
- Mixing coefficients are *initialized* either:
  - *Randomly* from a normal distribution
  - With *ridge-regression* if we want to *approximate* any fixed activation function $\sigma : \mathbb{R} \to \mathbb{R}$

$$\sigma(\mathbf{d}) = \mathbf{t} \implies \alpha = (\mathbf{K} - \varepsilon \mathbf{I})^{-1}\mathbf{t}$$

with: $K_{i,j} = \kappa(d_i, d_j)$

- Such constructed KAFs satisfy:
  - *Universal* approximability
  - *Smoothness* over the entire domain
  - Only *linear number of weights* introduced in the model
  - Mixing coefficients admit common *regularization techniques*





Smooth Activation Functions and AT

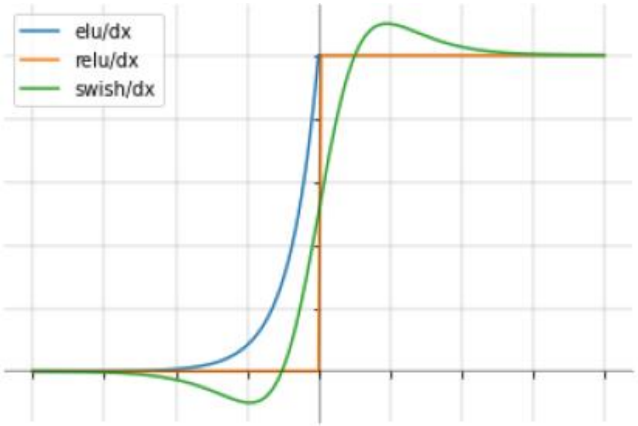
Can activation functions play a role in making neural networks more robust?

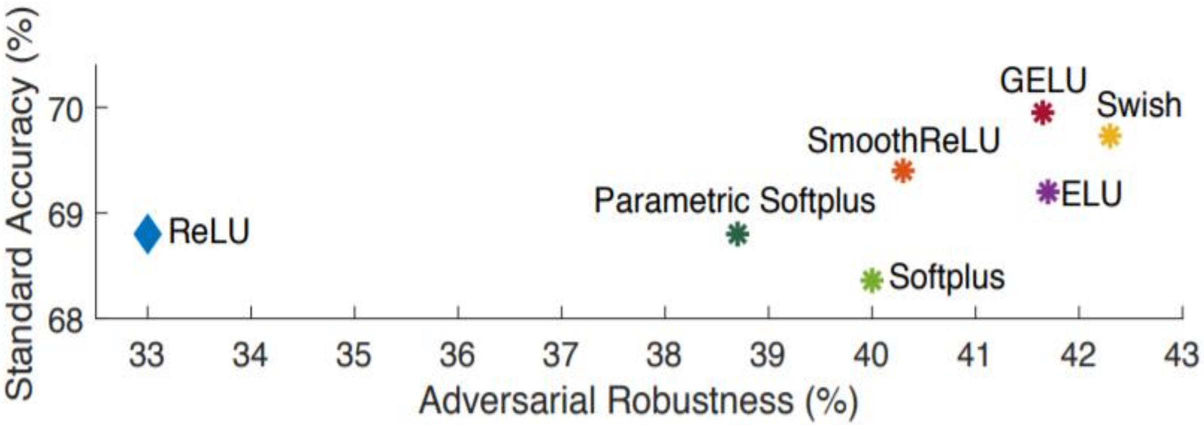
- Remark: to perform adversarial training we compute *twice* the number of *gradient computations* used for standard training, that is, both for the *inner maximization* and the *outer minimization*.

$$\min_{\theta} \frac{1}{|S|} \sum_{x,y \in S} \max_{\delta \in \Delta} \ell\left(f_\theta(x+\delta), y\right)$$

- In *[Chiang Xie, M. Tan, B. Gong et al, 2020]* authors conjectured that the widely-used ReLUs activation functions *weaken AT* due to their *non-smooth* nature

- Empirical results: by *enforcing smoothness*, replacing ReLUs with smooth counterparts, it is possible to train the network against stronger adversarial examples, which eventually leads to a *significant improvement* in the robustness





Hypothesis

procedure FBF_AT($epochs = T$, $dataset\_size = M$, ϵ)

  for $t \leftarrow 1, T$ **do** ▷ For each epoch

    for $i \leftarrow 1, M$ **do** ▷ For each minibatch

Smoothness

      $\delta \leftarrow \mathcal{U}(-\epsilon, \epsilon)$ ▷ Random init

      $\delta \leftarrow \delta + 1.25\epsilon \cdot sign(\nabla_\delta \mathrm{LS}\,(f_\theta(x+\delta), y))$ ▷ FGSM

      $\delta \leftarrow \mathcal{P}(\delta)$

      $\theta = \theta - \nabla_\theta \ell\,(f_\theta\,(x_i + \delta)\,, y_i)$ ▷ Update model weights

    end for

  end for

end procedure

Flexibility of KAFs

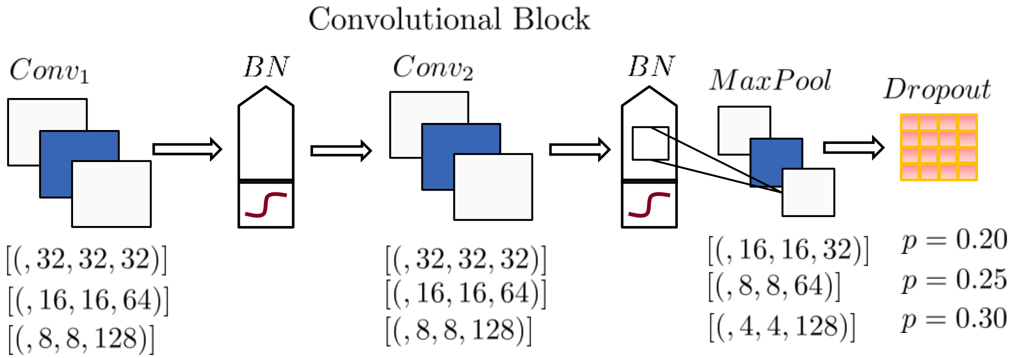
- **KAFs might be able to enhance even more the robustness and the accuracy of adversarially-trained models.**
  - By leveraging *flexibility*, KAFs proved to outperform fixed activation functions in a variety of (adv. free) tasks
  - Using the Gaussian kernel we can enforce *smoothness*
  - Formally this translates to: potential to improve *both outer and inner optimization*, respectively

Experiments Set-up

- **Training**:
  - Fast Adversarial Training *[E. Wong, L. Rice 2020]* until convergence
  - *One-Cycle* learning rate policy *[Leslie N. Smith 2018]* and *mixed-integer precision*
  - *CIFAR10* image classification dataset
  - *SGD* optimizer and *batch size* of 128
  - Categorical *cross-entropy* loss
  - *Dictionary size* for KAFs is D = 20
- **Adversarial examples** computed through *Projected-Gradient Descent (PGD)*
  - Any perturbation lies in the L_∞ norm-ball bounded by $\epsilon = 8/255$
  - 50 iterations sufficient for convergence
  - 10 random restarts
- **Evaluation**
  - *Robustness* i.e. accuracy of the model over PGD-perturbed test set
  - *Accuracy* i.e. standard accuracy over clean test set

VGG Results

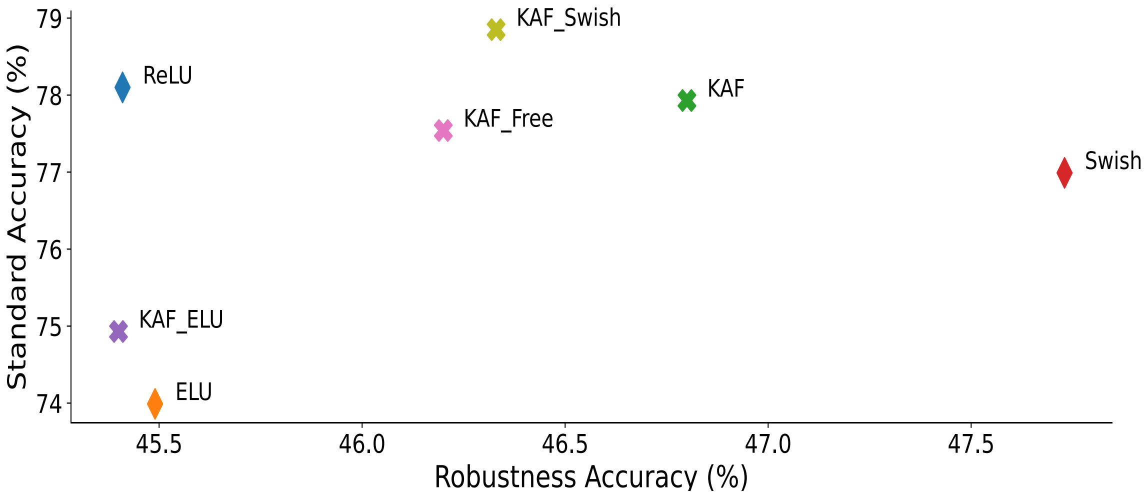
- VGG-inspired **architecture**: *three* consecutive *convolutional blocks*
- Followed by: *flatten* layer, *fully-connected* layer w/ 128 units, *dropout* and a decision layer with *softmax.*



- Train and evaluate seven different architectures that differ *only* on the activation function employed:

Activation	Accuracy	Robustness
ReLU	78.1%	45.41%
ELU	73.99%	45.49%
KAF	<u>77.93%</u>	<u>46.8%</u>
KAF_ELU	74.93%	45.4%
Swish	76.99%	**47.73%**
KAF_Swish	**78.85%**	46.33%
KAF_Free	77.54%	46.2%

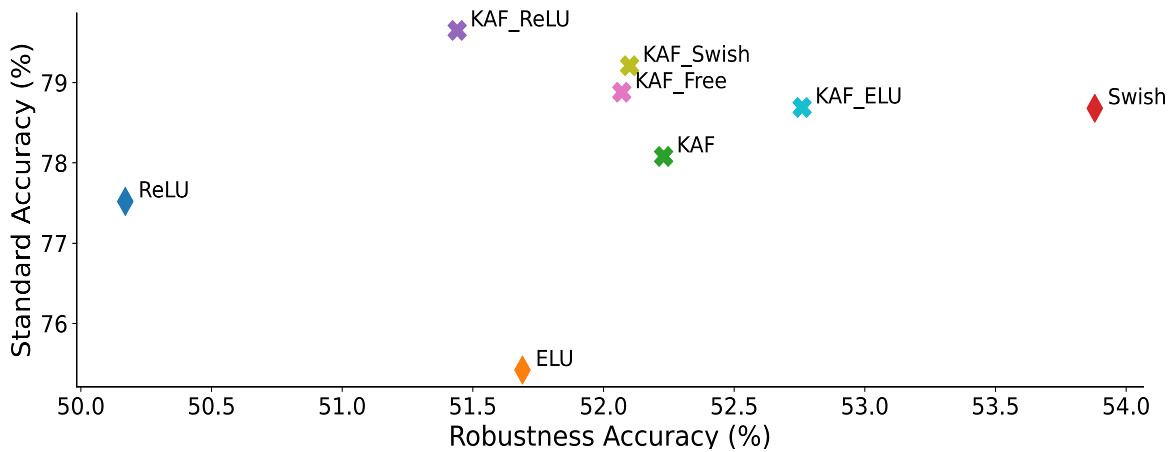
Tab1: Bold denotes best result, underline best trade-off



ResNet20 Results

- *Max-pooling* layers are *non-smooth*, which makes our hypothesis misplaced in the first place. We need to enforce smoothness throughout the whole model to perform a meaningful evaluation.
- Due to its relatively simple implementation and strong performances we decided to repeat the experiment using a *residual neural network* (ResNet).
- We experienced an *exploding gradient* issue using KAFs in deep (>50 layers) scenarios, thus we chose a ResNet20 architecture as described in the original paper *[Kaiming He, X. Zhang et al. 2015]*
- *Data augmentation:* 4 pixels are 0-padded on each side of the image and a 32 × 32 crop is randomly sampled from the resulting image or its horizontal flip.

Activation	Accuracy	Robustness
ReLU	77.52%	50.17%
ELU	75.42%	51.69%
KAF	78.08%	52.23%
KAF_ReLU	**79.65%**	51.44%
KAF_ELU	78.69%	52.76%
Swish	78.68%	**53.88%**
KAF_Swish	79.21%	52.10%
KAF_Free	78.88%	52.07%



- Overall robustness improved wrt VGG case $\Longrightarrow$ ResNet better suited for adversarial training (approx. same number of weights)
- KAFs reach *best std accuracies*, nevertheless, *Swish* function allows again for *strongest robustness*

Conclusions and Contributions

Sub-optimality in terms of robustness might be explained with *the introduction of new parameters* which comes *with KAFs* and ultimately with an *increased non-linearity* of the model's landscape. Despite being leveraged by AT, it can also be exploited by any *gradient-based* attack.

Fix: Try to use *Madry's* original *adversarial training* procedure instead *[A. Madry, A. Makelov, L. Schmidt et al. 2017]*

- ❑ Presented results can also be seen as a *contribution of evidences to the smoothness thesis [Chiang Xie, M. Tan, B. Gong et al, 2020]*

- ❑ We gave a working implementation in TF2 of cutting-edge procedures such as *[E. Wong, L. Rice 2020]* currently lacking even in third- party libraries. Moreover, we extended the *KAF layer* implementation for Keras.

- ❑ **Future Works:**

  - Test KAFs robustness also in *provable* scenarios such *as Lipschitz-regularization training*. *[Y. Tsuzuku, I. Sato et al. 2018]*

  - *Scale* the proposed defense towards larger datasets such as *ImageNet* to see if results are coherent.

  - Assess robustness using more-sophisticated attacks: *C&W*, *DeepFool* or, more importantly, *adaptive attacks [F. Tramer, N. Carlini et al. 2020]*

QUESTIONS?

(Code and Adversarially-trained Models)

THANK YOU!