Generating Transferable Adversarial Examples via Smooth Max Ensembling

Yuchen Zhang[1], Yi Sun[2], Jacob Steinhardt[1], Adithya Ganesh[1], Hugh Zhang[1], Philip Hwang[1], Florian Tramer[1], Percy Liang[1]

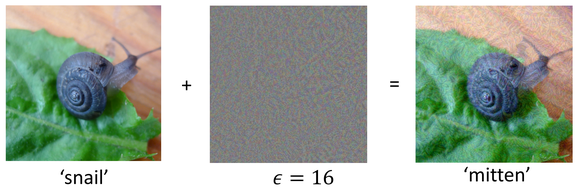
Stanford University[1], Columbia University[2]

Overview

In this work, we present an approach to generating adversarial images. We achieved 77.4% black box transfer against image recognition models specifically designed to resist adversarial attacks. Key design choices included:

- Using a log-sum-exp loss with a margin γ
- Using the RMSProp optimization algorithm instead of FGSM
- Average pooling the first layer to improve generalization

Introduction



Computer vision models can now correctly classify images with surprising accuracy, but add a little bit of adversarially chosen noise and you can break nearly all of them.

NIPS Adversarial Competition.

- Given an image I, generate an image I' such that $||I - I'||_\infty < \epsilon$, where ϵ is chosen from $4 - 16$ for each batch of images. Note the attacker has no access to the defending models.
- L_∞ norm means that you can change each pixel in the image by at most ϵ.
- Attackers generate adversarial images for a holdout set which are then tested against submissions in the defense track.

Attacker Score. Final score is evaluated as:

$$\mathrm{S}_{\text{attack}} = \sum_D \sum_{k=1}^{N} [D(A(\text{Image}_k)) \neq \text{TrueLabel}_k]$$

= avg. % of images misclassified among all defenders

- A, D: functions to execute attack and defense

Attack Generation

Strategy.

- We construct attacks against a family of pretrained ImageNet models and ensure that our attack simultaneously reduces accuracy on all of them.
- Adversarial examples are known to often transfer across neural network architectures.
- We partition pretrained models into two categories: *Easy* and *Hard*.
- *Easy* models have good blackbox transfer performance when attacked; *Hard* models do not transfer as well (typically adversarially trained models).
- Generate an attack for each family; final attack is computed as a weighted linear combination

$$A_{\text{final}} = \alpha A_{\text{easy}} + (1 - \alpha) A_{\text{hard}}$$

Average pooling.

- Many defense algorithms used image smoothing (e.g. JPEG compression / low pass filtering)
- Adding an average pooling layer to all models serves as an approximation of all possible blurring algorithms.
- Serves as an effective regularizer to improve performance since it prevents overfitting.

Objective Function

- **Typical objective:** Maximize sum of defender model errors.
- **Issue:** Defender accuracy is more correlated with the strongest model accuracy in the ensemble rather than the average performance.

Log-sum-exp objective function.

- Use a "smooth-max" loss where the best performing models dominate
- Let N be the number of models in the family. Further let $\mathbf{L}_i$ denote the logits of the ith model. Then our loss function is:

$$\mathcal{L} = \log\left(\sum_{i=1}^{N} \sigma(\mathbf{L}_i)_{\text{true}} + \gamma\right), \qquad (1)$$

where $\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$ is the softmax function, and $\sigma(\mathbf{L}_i)_{\text{true}}$ denotes the softmax probability of the true class.

- Margin γ improves stability as the loss nears minimum, since it causes the gradient to level off instead of increase exponentially in magnitude.

Optimizer.

- RMSProp optimization algorithm improves blackbox transfer performance over vanilla stochastic gradient descent

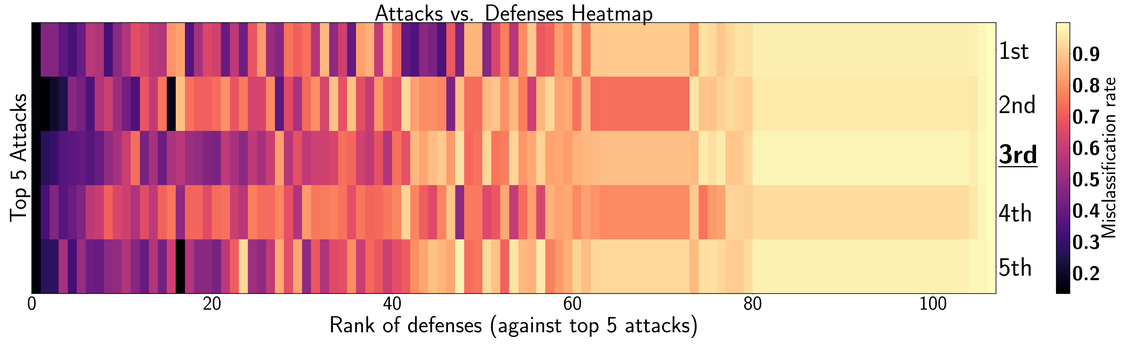


Figure 1: A heatmap of the performances of defenses against the top 5 attacks. Our attack is shown under 3rd place.

Blackbox Transfer Performance

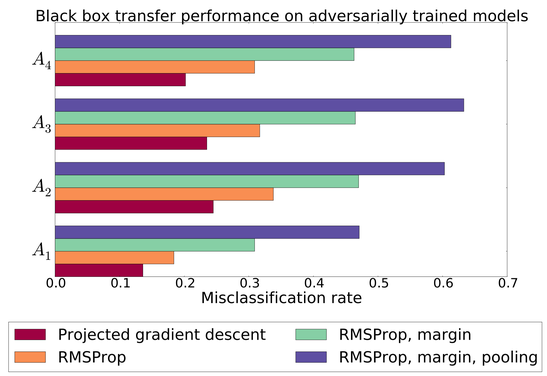
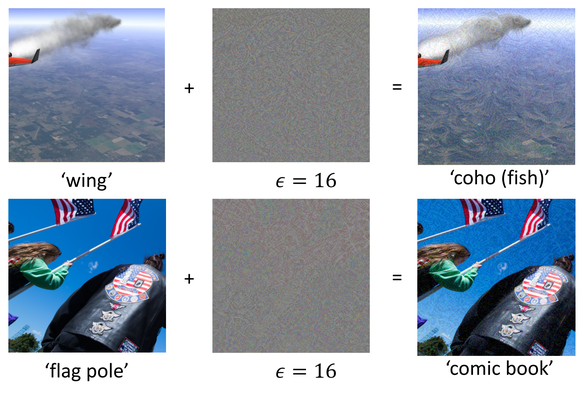


Figure 2: Black box transfer attack hit rate on adversarially trained models using the various improvements described above. A_1: EnsAdvInceptionResNetV2; A_2: AdvInceptionV3; A_3: Ens3AdvInceptionV4; A_4: Ens4AdvInceptionV3.

Adversarial Images



Discussion

- Attacking common models using our methods creates transferable adversarial examples.
- Using log-sum-exp loss improves blackbox transfer performance over an average loss baseline.
- Various other optimizations were used, e.g. the RMSProp optimization algorithm, margin coefficient in the objective function, and an average pooling layer.
- Our attack ranked 3rd out of 161 attacks.