Adversarial Vision Challenge

Wieland Brendel, Jonas Rauber, Alexey Kurakin, Nicolas Papernot, Behar Veliqi, Sharada P. Mohanty, Florian Laurent, Marcel Salathé, Matthias Bethge, Yaodong Yu, Hongyang Zhang, Susu Xu, Hongbao Zhang, Pengtao Xie, Eric P. Xing, Thomas Brunner, Frederik Diehl, Jérôme Rony, Luiz Gustavo Hafemann, Shuyu Cheng, Yinpeng Dong, Xuefei Ning, Wenshuo Li, and Yu Wang

Abstract This competition was meant to facilitate measurable progress towards robust machine vision models and more generally applicable adversarial attacks. It encouraged researchers to develop query-efficient adversarial attacks that can successfully operate against a wide range of defenses while just observing the final model decision to generate adversarial examples. Conversely, the competition

W. Brendel (✉) · J. Rauber · B. Veliqi · M. Bethge
University of Tübingen, Tübingen, Germany
e-mail: Wieland.Brendel@bethgelab.org; Jonas.Rauber@bethgelab.org; Behar.Veliqi@bethgelab.org; Matthias.Bethge@bethgelab.org

A. Kurakin · N. Papernot
Google Brain, Mountain View, CA, USA
e-mail: Kurakin@google.com; Papernot@google.com

S. P. Mohanty · F. Laurent · M. Salathé
École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
e-mail: marcel.salathe@epfl.ch

Y. Yu · H. Zhang · P. Xie · E. P. Xing
Petuum Inc., Pittsburgh, PA, USA
e-mail: yy8ms@virginia.edu; pengtao.xie@petuum.com; epxing@cs.cmu.edu

H. Zhang · S. Xu
Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: hongyanz@cs.cmu.edu; susux@andrew.cmu.edu

T. Brunner · F. Diehl
fortiss GmbH, Munich, Germany
e-mail: Brunner@fortiss.org; Diehl@fortiss.org

J. Rony · L. G. Hafemann
Laboratoire d'imagerie de vision et d'intelligence artificielle (LIVIA), ÉTS Montreal, Montreal, QC, Canada

S. Cheng · Y. Dong · X. Ning · W. Li · Y. Wang
Tsinghua University, Beijing, China
e-mail: csy530216@126.com; dyp17@mails.tsinghua.edu.cn; nxf16@mails.tsinghua.edu.cn; lws17@mails.tsinghua.edu.cn; yu-wang@mail.tsinghua.edu.cn

S. Escalera, R. Herbrich (eds.), *The NeurIPS '18 Competition*, The Springer Series on Challenges in Machine Learning, https://doi.org/10.1007/978-3-030-29135-8_5

encouraged the development of new defenses that can resist a wide range of strong decision-based attacks. In this chapter we describe the organisation and structure of the challenge as well as the solutions developed by the top-ranking teams.

1 Motivation

One of the most striking differences between human and machine perception is the susceptibility of modern machine vision algorithms to extremely small and almost imperceptible perturbations of their inputs [2, 7, 20, 26]. A tiny bit of carefully chosen image noise is usually sufficient to derail object recognition with neural networks (e.g. flip the prediction from *dog* to *banana*). Such perturbations are commonly denoted as *adversarial* and algorithms to find them are called *adversarial attacks*. Adversarial perturbations reveal that decision making in current deep neural networks is based on correlational rather than causal features. From a security perspective they are worrisome because they open avenues to manipulate sensory signals in ways that go unnoticed for humans but seriously affect machine decisions. For example, not-safe-for-work filters could be bypassed with minimally invasive image manipulations or content categorization filters could be mislead with minimal changes to a text or an image.

So far, existing attacks (see [4, 21] for a taxonomy) had only limited success to threaten real-world applications like autonomous cars which do not convey internal model information like gradients or confidence values to an attacker. Even current transfer-based attacks can be effectively defended against through ensemble adversarial training. In addition, even if internal model information is available most existing attacks are easily disarmed through simple means like gradient masking or intrinsic noise. An important goal of this competition was to foster the development of stronger and more query-efficient attacks that do not rely on any internal model information but only on the final decision [4, 27]. These so-called *decision-based* attacks have only recently been described for vision models [4, 27] but are highly relevant in real-world scenarios and much harder to defend against than transfer-based or gradient-based attacks.

Adversarial examples highlight that neural networks do not rely on the same causal features that humans use in visual perception. Closing this gap is important for many reasons: it would enable safety-critical applications of neural networks, would make neural networks more interpretable, would provide us with a deeper understanding of the human visual system and would increase the transferability of feature learning. Despite these advantages and many publications there has been little significant progress towards more robust neural networks in complex visual tasks like object recognition, see also [1, 6]. A core problem is the proper evaluation of model robustness: a model might just be perceived to be robust because the attacks deployed against it fail. However, most attacks fail due to trivial side effects like gradient masking. Thus, just like in cryptography, the real test of model

robustness is how well it stands against the scrutiny of attacks that are specifically designed against it. For this reason the competition was set up as a two-player game in which models and attacks have been continuously pitted against each other. This encouraged a co-evolution in which attacks could progressively adapt to the defense strategies of the models, and in which the models could progressively learn to better defend against strong attacks. Attacks have been able to query individual models, thereby enabling them to craft model-specific adversarials.

2 Overview of the Competition

In a robust network no attack should be able to find imperceptible adversarial perturbations. The Adversarial Vision Challenge facilitated an open competition between neural networks and a large variety of strong attacks, including ones that did not exist at the time when the defenses have been proposed. To this end the challenge consisted of one track for robust vision models, one track for targeted and one for untargeted adversarial attacks. Submitted models and attacks were continuously pitted against each other on an image classification task. Attacks were able to observe the decision of models on a restricted number of self-defined inputs in order to craft model-specific minimal adversarial examples.

The Adversarial Vision Challenge was built upon the experience from two related competitions:

1. **NIPS 2017 Competition on adversarial attacks and defenses [16].**
   Organised by Alexey Kurakin, Ian Goodfellow and Samy Bengio.
   This competition pitted models against attacks, but only indirectly: attacks were unable to query the models and hence had to devise generic adversarial examples that would work against as many models as possible. Devising defenses against such unspecific transfer-based attacks is much simpler than becoming robust against model-specific attacks.
2. **Robust Vision Benchmark (RVB)**[1]
   Organised by Wieland Brendel, Jonas Rauber and Matthias Bethge.
   The RVB is a continuously running benchmark (started in August 2017) in which submitted models are evaluated against a wide number of attacks (including submitted ones). Attacks are able to query the model both for confidence scores as well as gradients. This setting is interesting in order to evaluate model robustness but does not represent a realistic security scenario.

The Adversarial Vision Challenge (AVC) sits in between the two challenges: in contrast to the NIPS 2017 challenge, the AVC allows attacks to directly query models and thus to craft model-specific adversarial attacks. In contrast to the Robust

[1] https://robust.vision/benchmark.

Vision Benchmark, attacks are limited in the number of queries and cannot observe internal model variables like gradients or confidence scores (both of which are easily masked and thus rarely useful).

3 Task, Data, Metric and Rules

3.1 Data

While adversarial examples exist in many domains the AVC focused on vision or more specifically on the popular task of object recognition in natural images. ImageNet ILSVRC [25], one of the most wide-spread object recognition data sets, demands a lot of computational resources both for training and evaluation and would have limited access to the competition. Images in CIFAR-10 [15], on the other hand, are too small and are often hard to recognize even for humans. We hence opted to rely on TINY IMAGENET. This data set is derived from the full ImageNet ILSVRC data set and consists of 100,000 images with size 64×64 pixels categorized into one of 200 classes. TINY IMAGENET is freely available and we provided several pre-trained baseline models. For testing and development we collected images ourselves.

We split the collected images into 500 development images and 600 test images. The 500 development images were released to participants in order to help during the development process. The other test images have been used in the intermediate evaluations as well as the final evaluation of the competition. All images from the test set were kept secret until after the end of the competition.

3.2 Tasks

The competition included three tasks that correspond to the three tracks:

1. **Generate minimum untargeted adversarial examples.** In this task participants were given a sample image and access to a model. The goal was to create an adversarial image that is as similar as possible to the sample image (in terms of L_2 distance) but is wrongly classified by the given model.
2. **Generate minimum targeted adversarial examples.** In this task participants are given a sample image, a target class and access to a model. The goal was to create an adversarial image that is as similar as possible to the sample image (in terms of L_2 distance) but was classified as the target class by the given model.
3. **Increase size of minimum adversarial examples.** In this task participants designed robust object recognition models. For each given sample each adver-

sarial attack proposed a minimum adversarial example for the given model. The goal of this task was to increase the size of the minimum adversarial perturbations (the L_2 distance between the best adversarial example and the sample image) the attacks can find.

Participants were able to submit solutions to all three tasks. All submissions were continuously pitted against each other on a fixed set of samples to encourage a co-evolution of robust models and better adversarial attacks.

3.3 Metrics

We designed the model score such that it represents the expected size of the minimum adversarial perturbation that attackers can generate for a new image sample (larger is better). Similarly, we designed the attack score such that it represents the expected size of the adversarial perturbations the attack can generate (smaller is better).

To be more precise we denote the set of submitted models as M, the set of attacks as A (both targeted and untargeted) and the set of samples as S. The top-5 models and attacks are denoted as M_5 and A_5 respectively. We further denote an adversarial image $\hat{\mathbf{s}}$ for a given sample $\mathbf{s} \in S$ generated by attack $a \in A$ against model $m \in M$ as $\hat{\mathbf{s}}_a(\mathbf{s}, m)$. As a distance metric between to images $\mathbf{s}_1$ and $\mathbf{s}_2$ we use the L_2 distance,

$$d(\mathbf{s}_1, \mathbf{s}_2) = \|\mathbf{s}_1 - \mathbf{s}_2\|_2 . \tag{1}$$

3.3.1 Model Score

For each model m and each sample $\mathbf{s}$ we computed the adversarial $\hat{\mathbf{s}}_a(\mathbf{s}, m)$ for the attacks $a \in A_5$. We then determined the size of the smallest adversarial perturbation,

$$d_m^{min}(\mathbf{s}, A_5) = \min_{a \in A_5} d(\mathbf{s}, \hat{\mathbf{s}}_a(\mathbf{s}, m)). \tag{2}$$

If for a given sample $\mathbf{s}$ no attack was able to generate an adversarial example we set $d_m^{min}(\mathbf{s})$ to a conservative upper bound. Finally, the model score is calculated as the median across the minimum distances,

$$\text{ModelScore}_m = \text{median}\left(\left\{d_m^{min}(\mathbf{s}, A_5) | s \in S\right\}\right). \tag{3}$$

The median is important to make the evaluation robust against outliers.

3.3.2 Attack Score

We run each attack a against the top-5 models $m \in M_5$ and each sample $\mathbf{s} \in S$. For each model and sample we compute the distance

$$d_a(\mathbf{s}, m) = d(\mathbf{s}, \hat{\mathbf{s}}_a(\mathbf{s}, m)). \tag{4}$$

If the attack failed to generate an adversarial we set the corresponding distance to a conservative upper bound $d_a(\mathbf{s}, m)$. The final attack score was then the median size of the adversarial perturbation,

$$\text{AttackScore}_a = \text{median}(\{d_a(\mathbf{s}, m) | \mathbf{s} \in S, m \in M_5\}) \tag{5}$$

For attacks lower scores are better, for models higher scores are better.

Both scores depend on the set of top-5 models and attacks. This focused attacks on the hardest models and made the evaluation feasible but also introduced a recursive dependence between (a) evaluating model/attack scores and (b) determining the top-5 in each track. This does not affect the final evaluation in which we pitted all models against all attacks, which in turn allowed us to reliably determine the top-5 model and attack submissions (see 4.1.4). During the rest of the competition we determined the top-5 models and attacks in two development rounds (in the same way we performed the final evaluation) and all submissions were tested against them until the next evaluation round.

3.4 Baselines and Code Available

3.4.1 Model Baselines

We provided three baseline models: (1) a vanilla ResNet-18 [9], (2) an adversarially trained ResNet-50[13], and (3) a ResNet-18 with intrinsic frozen noise. We provided pretrained weights for all models.

3.4.2 Untargeted Attack Baselines

We provided five baselines: (1) a simple attack using additive Gaussian noise, (2) a simple attack using salt and pepper noise, (3) the Boundary Attack [4] with reduced number of iterations, (4) a single-step transfer attack (where the substitute model is the adversarially trained ResNet-50) and (5) an iterative transfer attack (with the same substitute model) based on the basic iterative method [17].

3.4.3 Targeted Attack Baselines

We provided three baselines: (1) a simple interpolation-based attack (where an image from the target class is blended into the original image until the classifier decision changes), (2) the pointwise attack [23] and (3) the Boundary attack [4].

3.5 Tutorial and Documentation

We released an extensive development package containing test images, tutorials, example submissions and evaluation scripts:

- Example model submission with placeholder for user-defined models (framework agnostic) and a tutorial on how to use it.
- Example attack submission with placeholder for user-defined adversarial attacks (framework agnostic) and a tutorial on how to use it.
- Code of all baseline attacks, including a detailed description.
- Code and model weights of all baseline models, including a detailed description.
- Set of 500 test images which participants can use for development of their models and attacks.
- Tool to evaluate model and attack submissions before the actual submissions. In this way users can test their code and its runtime behaviour before the actual submission.
- A reading list summarizing publications relevant for this competition.[2]

Some of the above mentioned code and tutorials will be reused and adapted from the NIPS 2017 competition, from the Robust Vision Benchmark as well as from previous competitions run by crowdAI.

4 Organizational Aspects

4.1 Protocol

The competition was hosted on crowdAI (https://crowdai.org). Participants submitted their models and attacks as commits to their respective git repositories from which a Docker image was built automatically (see Sect. 4.1.1). Submissions were continuously evaluated throughout the competition (see Sect. 4.1.2).

At the end of the competition we performed a final evaluation to determine the winners in each track (see Sect. 4.1.4). The models and attacks were run in

[2] https://medium.com/@wielandbr/reading-list-for-the-nips-2018-adversarial-vision-challenge-63cbac345b2f.

an isolated Kubernetes environment with local subnetworks that restrict intercommunication to the exchange between a single model and a single attack to prevent cheating. The communication was further restricted via a very limited HTTP API interface.

4.1.1 Submission Process

Given the nature of the challenge participants were expected to package their models as docker images. To decrease the entrance barriers for participants not as comfortable with the Docker ecosystem, we allowed simple code submissions based on Binder (https://mybinder.org/). Binder allows users to distill the software environment of their code as a set of configuration files in their source code from which we can deterministically generate a Docker image using *repo2docker*. In the development package we provided a series of template submission repositories which are already pre-configured with popular libraries of choice like Tensorflow. Participants could use the binder tools to locally test their code before making the submissions. The code repositories have been hosted on a custom GitLab instance operated by crowdAI.

4.1.2 Continuous Evaluation

Participants were allowed to submit their models or attacks at any point in time. The number of submissions was limited to at most five submissions per track within 24 h. The submitted Docker images were evaluated in the backend against the top-5 opponents (either models or attacks depending on track) on 200 validation samples to determine the score for that submission.

4.1.3 Top-5 Evaluation Round

We performed two more extensive development evaluations of all submissions to determine the new top-5 models and attacks. We used a test set of 200 secret sample images which were different in each evaluation round. The evaluation was performed according the following protocol:

- The submission system is frozen for 72 h during which the evaluation is performed.
- Round 1: All model/attack combinations were evaluated on a small set of 10 samples. From this evaluation we determined a very rough estimate of model and attack scores. Only the best 50% were considered for the next round.
- Round 2: The remaining model/attack combinations were evaluated in the same way as in Round 1 but on a larger set of 20 samples. Again we determined the top 50% of the remaining submissions.

- We iterated these rounds until we ended up with the top-10 models and attacks. For these submissions we evaluated all model/attack combinations on the full test set of 200 samples to rigorously determine the top-5 submissions in each track.
- Scoring round: All submissions were re-scored on the 200 validation images and the leaderboards were updated accordingly.

This somewhat elaborate procedure was necessary to keep the computational resources for evaluation within our budget. The naive evaluation—just pitting all attacks against all models on all models—would have been orders of magnitude more resource intensive. To be concrete, for 100 attacks, 100 models, 200 images and 1000 queries we would have to process 2×10^9 queries. We demanded that each model has to process a query within 40 ms, meaning that the entire procedure would have taken up to 925 GPU-days. The resource-efficient procedure above instead reduces the amount of computation time in the same scenario to a maximum of 82 GPU-days.

4.1.4 Final Evaluation Round

The scoring in the final evaluation round was performed in the same way as in the Top-5 evaluation rounds but this time the final scoring is performed on 500 secret test images. These test images have not been used in any of the evaluation rounds before.

4.2 Rules

- Bethgelab and Google Brain employees can participate but are ineligible for prizes.
- To be eligible for the final scoring, participants are required to release the code of their submissions as open source.
- Any legitimate input that is not classified by a model (e.g. for which an error is produced) will be counted as an adversarial.
- If an attack fails to produce an adversarial (e.g. because it produces an error), then we will register a worst-case adversarial instead (a uniform grey image).
- Each classifier must be stateless and act one image at a time. This rule is supposed to prevent strategies such as memorizing pre-attack images and classifying replayed versions of them at defense time.
- The decision of each classifier must be deterministic. In other words, the classifier decision must be the same for the same input at any point in time.
- Attacks are allowed to query the model on self-defined inputs up to 1000 times/sample. This limit is strictly enforced in the model/attack interface and an error will be returned whenever the attack queries the model more often.

- Each model has to process one image within 40 ms on a K80 GPU (excluding initialization and setup which may take up to 100 s).
- Each attack has to process a batch of 10 images within 900 s on a K80 GPU (excluding initialization and setup which may take up to 100 s).

4.3 Schedule

Timeline of the competition:

- **April 20, 2018.** Launched website with announcement and competition rules. Started active advertisement of the competition.
- **July 18, 2018.** Released development kit for participants.
- **July 18–November 1, 2018.** Competition was running.
- **November 1, 2018.** Deadline for the final submission.
- **November 1–9, 2018.** Organizers evaluated submissions.
- **November 9, 2018.** Announced competition results.

4.4 Competition Promotion

The competition was be promoted via the organisers' Facebook, Twitter, Google+ and Reddit accounts as well as the crowdAI email list and several university mailing lists. We also secured prices from Paperspace (cloud compute credits).

4.5 Resources

Amazon AWS sponsored the necessary cloud computing resources to evaluate submissions ($65,000 worth of cloud compute resources).

5 Competition Results

The competition consisted of three tracks: a robust model track for defenses, an untargeted attacks track and a targeted attacks track. In total, these three tracks received well over three thousand submissions from 352, 96 and 63 participants, respectively. In each track, we invited the three best submissions to describe their approaches (see Sects. 6 and 7).

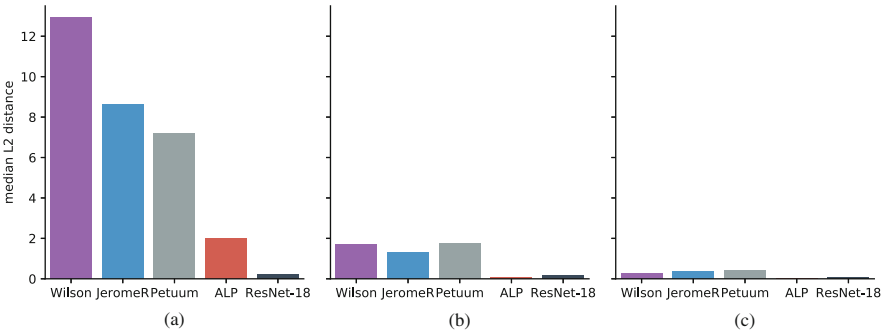

Fig. 1 Robustness of the three winning defenses (Petuum-CMU, Wilson, JeromeR) and two baselines (ResNet-50 with Adversarial Logit Pairing (ALP) and ResNet-18) against the baseline attacks of the first round (**a**), the winning attacks of the final round (**b**) and a set of attacks with with full white-box access to the defense (**c**). This demonstrates a successful co-evolution of attacks and defenses: The winning defenses improved over the baseline models and the winning attacks improved over the baseline attacks

To better understand the progress made in this competition, we analyzed the three winning defenses (Petuum-CMU, Wilson, JeromeR) further and compared them to two baselines (ResNet-50 with Adversarial Logit Pairing (ALP) and ResNet-18). During the competition, their robustness was evaluated against the five attacks winning the untargeted attacks track (Fig. 1b). In addition to that, we now evaluated their robustness against the five baseline attacks that were used in the first round of the competition (Fig. 1a) and in a white-box setting against four gradient-based attacks (Fig. 1c).

The results in Fig. 1a and b show that the new defenses achieve substantial improvements in the constrained black-box setting set out by the competition. Interestingly, their improved robustness in this setting even seems to transfer—to a small extent—to the white-box setting (Fig. 1c).

Comparing the strength of the five winning attacks used for Fig. 1b to the strength of the five baseline attacks used for Fig. 1a, we also see that the new query-constrained decision-based attacks that won the untargeted attacks track (see Sect. 6) have substantially improved the query-efficiency of decision-based attacks. In fact, with just one thousand queries per sample, the winning attacks (Fig. 1b) are now much closer to the performance of gradient-based attacks in a white-box setting (Fig. 1c) than the baseline attacks (Fig. 1a).

Figure 2 illustrates how the smallest adversarial perturbations found by any of the five winning attacks for a given sample look like for the three winning defenses (Petuum-CMU, Wilson and JeromeR). For the lighthouse and the bus, the perturbations are clearly visible but not large enough to affect humans. For the parking meter, the perturbation is hard to see.

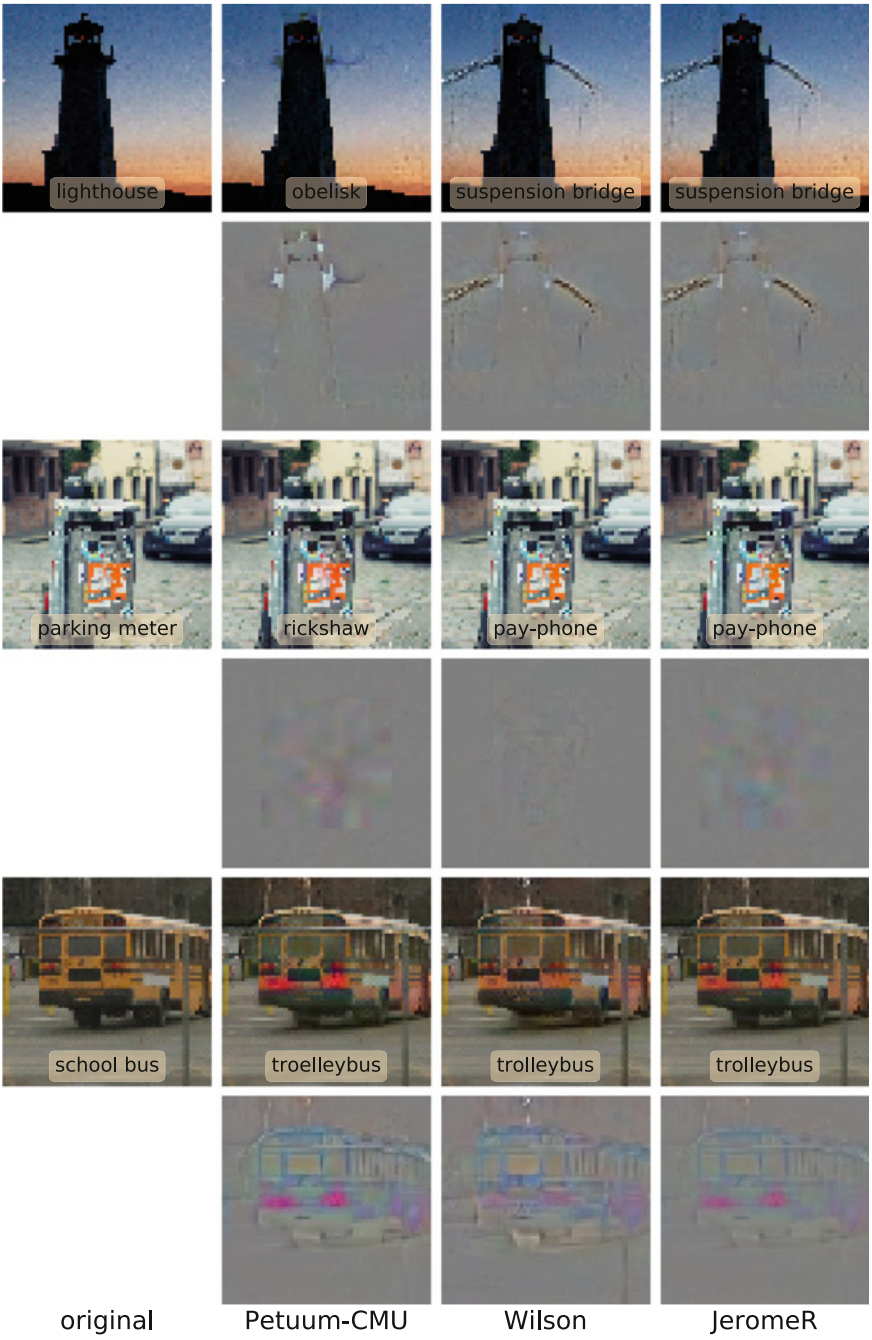

Fig. 2 Minimal adversarial perturbations for three example images (rows) and the three winning defenses (columns): for each defense and image we chose the smallest perturbation found by any of the five winning untargeted attacks. Even rows display the perturbed images while odd rows display the perturbations

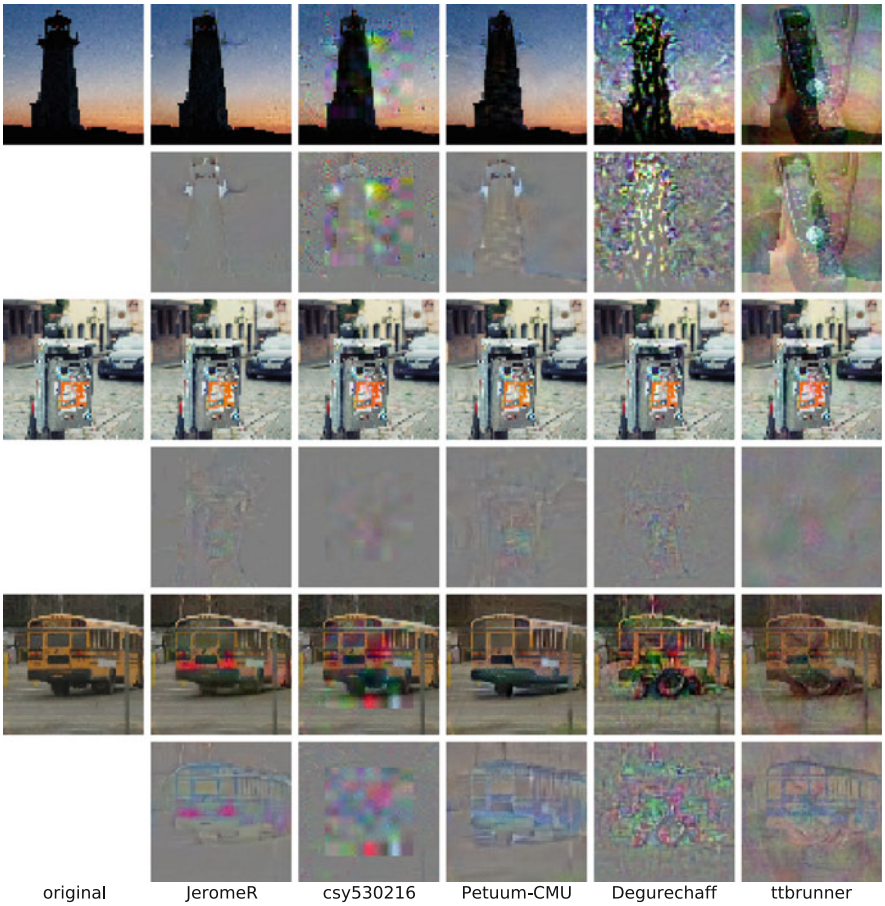

Fig. 3 Adversarial perturbations for three example images (rows) and the five winning targeted attacks (columns) against the winning defense. Even rows display the perturbed images while odd rows display the perturbations

In Fig. 3 we show the same three example images as in Fig. 2. Instead of choosing the best attack, we now plot the perturbation found by each attack against the winning defense. This allows us to visually compare the different attacks which reveals some striking differences in the types of perturbations found by the different attacks.

Finally, we tested the three defenses winning in the robust model track against the five targeted adversarial attacks winning the targeted attacks track. In Fig. 4, we show the smallest adversarial perturbations found by any of the five targeted attacks against the three defenses for three different samples. The perturbations are substantially larger than the untargeted ones and do contain features of the target class, e.g. dog faces or parts of a pizza.

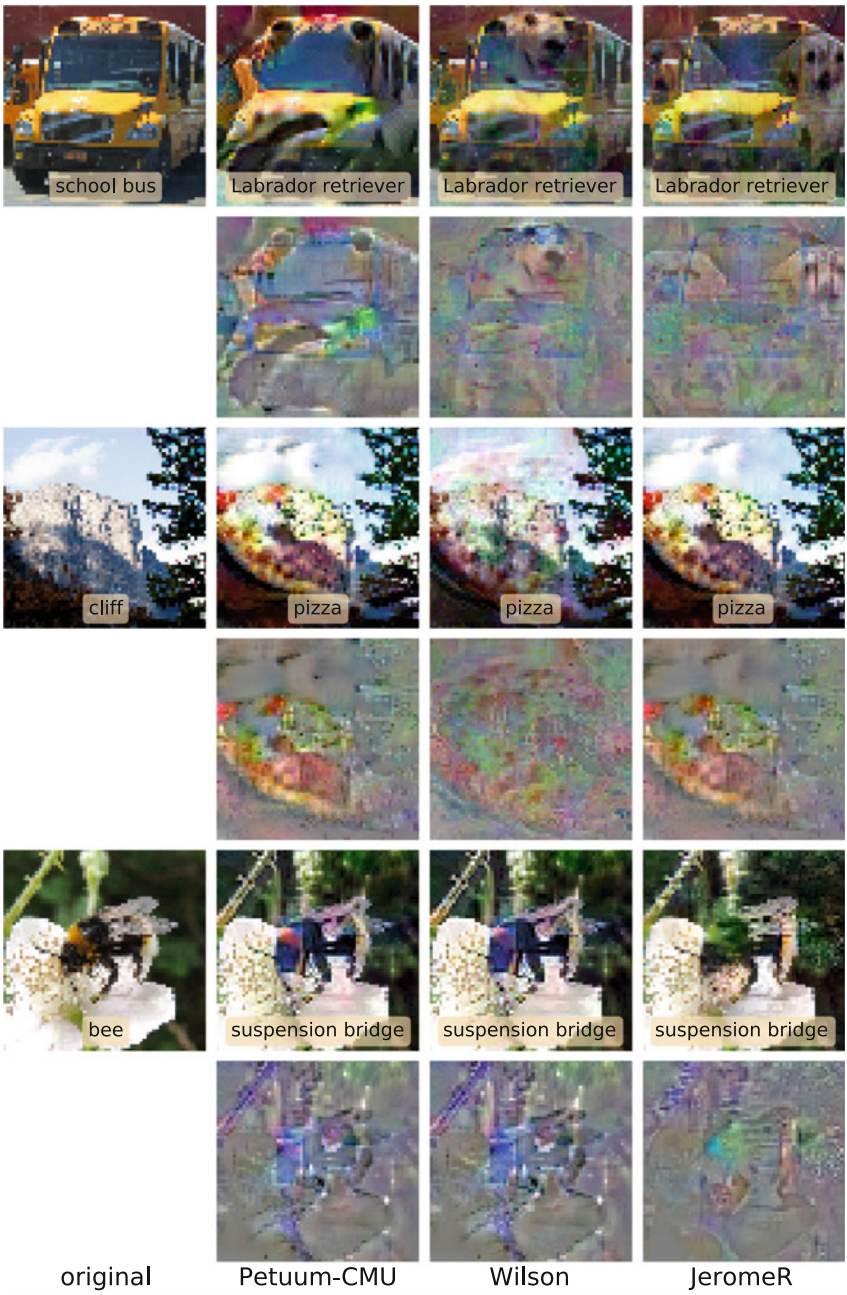

Fig. 4 Minimal targeted adversarial perturbations for three example images (rows) and the three winning defenses (columns): for each defense and image we chose the smallest perturbation found by any of the five winning targeted attacks. Note that the defenses won in the robust model track which measures robustness against untargeted attacks

6 Top Submissions in the Attack Track

All presentations of the top attacks and defenses have been provided by the teams.

6.1 *Team `JeromeR`: First Place Untargeted and Third Place Targeted Attack Track*

By Jérôme Rony and Luiz Gustavo Hafemann.

The strategy for both attacks relied on the *transferability* of adversarial images [3]. To this end, we trained our own (surrogate) model, generate an adversarial example using this surrogate model and use it to attack the actual system. As observed by Szegedy et al. [26], attacks generated for one model often transfer to other models, even with different architectures and trained with different subsets of the data.

For these submissions, a collection of surrogate models was considered, and two attacks were used: Decoupled Direction and Norm (DDN) [24] and Fast Gradient Method (FGM) [7]. For each surrogate model and attack, a direction $g = \frac{\delta}{||\delta||_2}$ is used to find the decision boundary in the model under attack. Algorithm 1 formalizes the attack. The boundary search was performed as follows: given an attack direction g, it iteratively searchs a norm λ such that $x + \lambda g$ is adversarial. The algorithm

Algorithm 1 Adaptive Ensemble black-box attack

Input: x: Original image
Input: y: True label (untargeted) or Target label (targeted)
Input: m: Model under attack
Input: S: Surrogate models
Input: targeted: Targeted or untargeted attack
Output: $\tilde{x}_{\text{best}}$: Adversarial Image

$\tilde{x}_{\text{best}} \leftarrow x$
$d_{\text{best}} \leftarrow \infty$
for each $s \in S$ **do** ▷ For each surrogate model
  $\delta_{\text{DDN}} \leftarrow \text{DDN}(s, x, y, \text{targeted})$
  $\delta_{\text{FGM}} \leftarrow \text{FGM}(s, x, y, \text{targeted})$
  for each $\delta \in \{\delta_{\text{DDN}}, \delta_{\text{FGM}}\}$ **do** ▷ For each attack
    $g \leftarrow \frac{\delta}{||\delta||_2}$
    $\tilde{x} \leftarrow \text{boundary\_search}(m, x, g, y)$ ▷ Find the decision boundary of model m, starting from image x, in direction g
    if $\tilde{x}$ is adversarial and $||\tilde{x} - x||_2 < d_{\text{best}}$ **then**
      $\tilde{x}_{\text{best}} \leftarrow \tilde{x}$
      $d_{\text{best}} \leftarrow ||\tilde{x} - x||_2$
    end if
  end for
end for
$\tilde{x}_{\text{best}} \leftarrow \text{boundary\_attack}(m, x, y, \tilde{x}_{\text{best}})$ ▷ Refine with a boundary attack

starts with a small λ, and increases it exponentially until the boundary is found, and refines it with a binary search. Finally, the best adversarial perturbation is further refined with a boundary attack [4]. We observed that this attack performed very well for untargeted attacks, but in many cases fails to find an adversarial image for targeted attacks. In the final targeted-attack submission, in case of failure to find an adversarial image, the attack returned a training image of the target class.

A critical decision for this attack is which surrogate models to consider. In this work we observed that attacks based on adversarially trained models have better transferability. This is illustrated in Fig. 5, that plots the decision space around an example x. We notice that the direction of the gradient given by the adversarially trained surrogate model (b) reaches the decision boundary of the actual model under attack with much smaller norm than if we follow the direction given by a non-adversarially trained model. For this figure, the model under attack is an adversarially trained DenseNet-161, and the attacks used a ResNet-101.

Our submission used Algorithm 1 with ResNet [10], ResNeXt [29] and DenseNet [11] surrogate models: **(1) non-adversarially trained**: ResNet-18, ResNet-34, ResNet-50, ResNet-101, DenseNet161; **(2) adversarially trained with DDN** [24]: ResNet-18, ResNet-34, Resnet-101, densenet161; **(3) Ensemble adversarial trained** [27]: ResNet-18, ResNeXt50-32x4d.

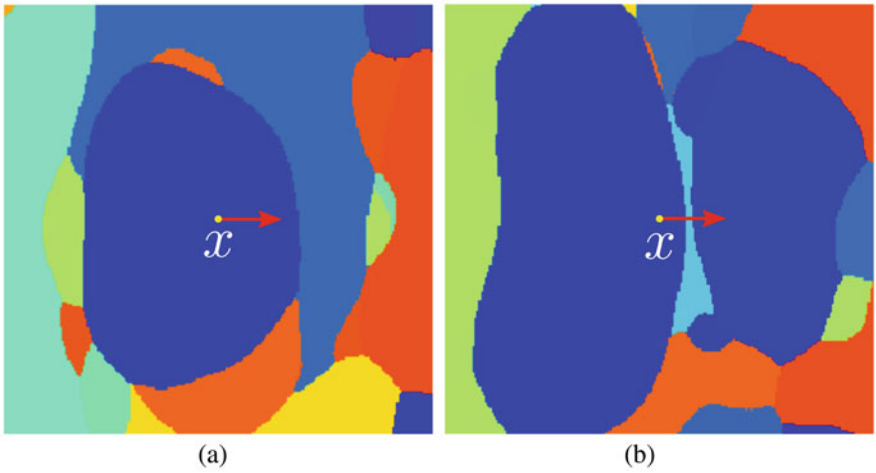

Fig. 5 Decision space of a model under attack. Each color represents a different class, and the axes are two directions in input space. The x direction is given by the gradient of a surrogate model: **(a)** normally trained and **(b)** adversarially trained

6.2 Team `Petuum-CMU`: First Place Targeted and Third Place Untargeted Attack Track

By Yaodong Yu, Hongyang Zhang, Susu Xu, Hongbao Zhang, Pengtao Xie and Eric P. Xing.

With query-based access to the target models, our main strategy was to train substitute models to generate adversarial examples, which is based on the transferability phenomenon [20]. For our black-box attacks submissions, the design of both untargeted and targeted attack strategies were based on three basic insights:

- Both the natural accuracy and the robust accuracy (i.e. accuracy on norm-constrained adversarial examples) of the substitute model are important for black-box attacks. We found that by balancing the natural accuracy and the robust accuracy of substitute model could lead to better transferabilities for black-box attacks.
- The performance of our black-box attacks could be improved by relying on an ensemble of different models as the substitute model.
- The target defense models may have different vulnerabilities on classifying different classes of images. To deal with the different vulnerabilities, leveraging various attack strategies jointly can increase attack success.

Based on the above three insights, we constructed adversarial examples as described in Algorithm 2. The training of the substitute models, i.e., the robust models, is described in Sect. 7.1.

With the trained substitute (robust) models, we combined different types of gradient-based attack methods to improve the performance of our black-box attacks, including Fast Gradient Sign Method [7], Projected Gradient Descent Attack [18],

Algorithm 2 Adversarial attack

Input: Original example $\boldsymbol{x}$, label for untargeted attack/target label for targeted attack $\boldsymbol{y}$, target model M, substitute model $\hat{M}$:
Output: $\boldsymbol{x}'$: Adversarial examples

Train different defense models and ensemble models as the substitute model $\hat{M}$
$\boldsymbol{x}' \leftarrow \boldsymbol{x}$; $D_{\text{best}} \leftarrow \infty$ ▷ Initialize the example and best distance
$\tilde{\boldsymbol{x}}_1 \leftarrow$ Projected Gradient Descent Attack($\boldsymbol{x}$, $\boldsymbol{y}$, $\hat{M}$, Linfinity Distance)
$\tilde{\boldsymbol{x}}_2 \leftarrow$ Fast Gradient Sign Method($\boldsymbol{x}$, $\boldsymbol{y}$, $\hat{M}$, Mean Squared Distance)
$\tilde{\boldsymbol{x}}_3 \leftarrow$ DeepFool($\boldsymbol{x}$, $\boldsymbol{y}$, $\hat{M}$, Linfinity Distance) ▷ Construct different types of adversarial examples
for $\tilde{\boldsymbol{x}} \in \{\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \tilde{\boldsymbol{x}}_3\}$ **do**
  $D \leftarrow \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_2$
  if $\tilde{\boldsymbol{x}}$ is an adversarial example and $D < D_{\text{best}}$ **then**
    $\boldsymbol{x}' \leftarrow \tilde{\boldsymbol{x}}$
    $D_{\text{best}} \leftarrow D$
  end if
end for
output $\boldsymbol{x}'$

and DeepFool Attack [19] (all implementations taken from `foolbox` [23]). The ensemble of different attack strategies can further reduce the perturbation distances while ensuring effective attacks.

In our final submission, for the targeted attack track, we independently trained two deep models, ResNet-34 and ResNet-152 [10], based on Algorithm 4. We averaged the logits of the above two substitute models. For the untargeted attack track, we trained a ResNet-34 as the substitute model.

6.3 *Team `csy530216`: Second Place Untargeted Attack Track*

By Shuyu Cheng and Yinpeng Dong

We designed an evolutionary attack method to improve the efficiency of decision-based attacks over existing methods (such as boundary attack [4]). The evolutionary attack is similar to the boundary attack since both of them are based on random walk (by adding perturbation) along the decision boundary, but our algorithm can reduce the dimension of the search space and model the local geometry of the search directions. The algorithm is outlined in Algorithm 3.

Features of our algorithm include:

- **Reduce the dimension of the search space**: Based on the assumptions that (1) the perturbations applied to the main objects in images are more useful for generating adversarial examples and that (2) smooth noise leads to better search directions [8], we could reduce the dimension of the search space. Specifically, we first sample a random noise in $\mathbb{R}^{10\times10\times3}$, then upscale it to $\mathbb{R}^{40\times40\times3}$ by bilinear interpolation. We next zero-pad it to $\mathbb{R}^{64\times64\times3}$ in the input space.

Algorithm 3 The evolutionary attack algorithm

Require: The original image $\mathbf{x}$; the initial adversarial image $\tilde{\mathbf{x}} \in \mathbb{R}^n$; the dimension $n \in \mathbb{N}_+$ of the input space; the dimension $l \in \mathbb{N}_+$ of the foreground; the dimension $m \in \mathbb{N}_+$ of the search space. (In our solution, $n = 64 \times 64 \times 3$, $l = 40 \times 40 \times 3$, $m = 10 \times 10 \times 3$. Initial $\tilde{\mathbf{x}}$ is found by transferring from the ResNet50 ALP baseline (single model) provided by the organizers.)
Require: The total number of queries T.
Initialize $\mathbf{C} = \mathbf{I}_m$, $\mathbf{p}_c = \mathbf{0}$, $\sigma, \mu \in \mathbb{R}_+$;
for $t = 1$ to T **do**
  Sample $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$;
  Upscale $\mathbf{z}$ to $\mathbb{R}^l$ by bilinear interpolation and pad it to $\mathbb{R}^n$ with zeros to obtain $\tilde{\mathbf{z}}$;
  $\tilde{\mathbf{x}}_{\text{line}} \leftarrow \tilde{\mathbf{x}} + \mu(\mathbf{x} - \tilde{\mathbf{x}})$; $\tilde{\mathbf{x}}_{\text{new}} \leftarrow \tilde{\mathbf{x}}_{\text{line}} + \sigma \frac{||\tilde{\mathbf{x}}-\mathbf{x}||_2}{||\tilde{\mathbf{z}}||_2}\tilde{\mathbf{z}}$; $\tilde{\mathbf{x}}_{\text{new}} \leftarrow (\tilde{\mathbf{x}}_{\text{new}} - \mathbf{x}) \cdot \frac{||\tilde{\mathbf{x}}_{\text{line}}-\mathbf{x}||_2}{||\tilde{\mathbf{x}}_{\text{new}}-\mathbf{x}||_2} + \mathbf{x}$;
  if $\tilde{\mathbf{x}}_{\text{new}}$ is adversarial **then**
    $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}}_{\text{new}}$; Update $\mathbf{p}_c$ and $\mathbf{C}$ by $\mathbf{z}$ according to Eq. (6);
  end if
  (Periodically) $\mu \leftarrow \mu \cdot \exp(P_{\text{success}} - 1/5)$; P_{success} is the success rate of several past trials.
end for
return $\tilde{\mathbf{x}}$.

- **Model the local geometry of the search directions**: The adaptation of covariance matrix $\mathbf{C}$ is suitable for solving non-separable optimization problems since it can model the local geometry of the search directions [12]. We use a diagonal covariance matrix as $\mathbf{C}$, which is updated after each successful trial as

$$\mathbf{p}_c = (1 - c_c)\mathbf{p}_c + \sqrt{c_c(2 - c_c)}\frac{\mathbf{z}}{\sigma}; c_{ii} = (1 - c_{cov})c_{ii} + c_{cov}(\mathbf{p}_c)_i^2, \tag{6}$$

  where $\mathbf{p}_c \in \mathbb{R}^m$ is called the evolution path as it stores the exponentially decayed successful search directions; for $i = 1, \ldots, m$, c_{ii} is the diagonal element of $\mathbf{C}$ and $(\mathbf{p}_c)_i$ is the i-th element of $\mathbf{p}_c$. c_c and c_{cov} are two hyper-parameters of CMA. In our submission, $c_c = 0.01$, $c_{cov} = 0.001$.

6.4 *Team `ttbrunner`: Second Place Targeted Attack Track*

By Thomas Brunner and Frederik Diehl.

Our submission is based on the Biased Boundary Attack [5], which uses biased sampling as a means to improve the efficiency of the Boundary Attack [4] (and, by extension, any attack based on random search). Over the years, much intuition has been gathered about the patterns that are formed by adversarial examples, and about the geometry of adversarial regions in the input space. Why not formulate these intuitions as prior beliefs for a random sampling procedure, and concentrate samples towards directions with a higher chance of success? In our submission, we use two such biases:

- **Low-frequency perturbations**: We observed that last year's competition winners [16] used denoisers and random transforms, both of which amount to a low-pass filter effect. Therefore, low-frequency perturbations should have a higher chance of success. For our attack, we sample from a distribution of Perlin noise patterns, instead of a normal distribution, and observe a great increase in query efficiency.
- **Projected gradients**: As a second bias, we seek to exploit transferability of gradients from a surrogate model, however limited it may be. At each step of the attack, we calculate the gradient and project it so it lies on the same hyperplane as the samples which are drawn for the Boundary Attack. We then bias the sample directions slightly towards it. This allows our attack to not directly follow the gradient (as PGD does), but instead sample large regions around it. As a consequence, even very simple surrogate models deliver a great speed-up.

For the gradient bias, we simply combined the ResNet18 and ResNet50 baselines provided by the organizers. This demonstrates the flexibility of our approach: Most attack submissions relied on ensembles of carefully-trained surrogates, while we

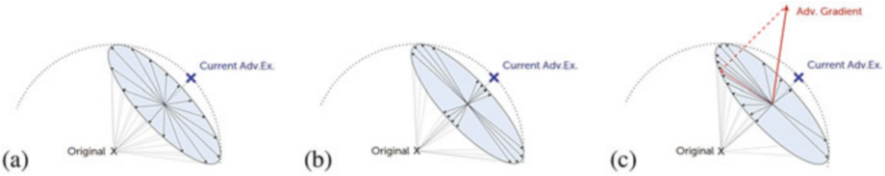

Fig. 6 Distribution of sampling directions in the input space. (**a**) unbiased, (**b**) low-frequency bias, (**c**) gradient bias

achieved a high competitive score with much simpler models. For an in-depth description of the attack, please refer to our publication [5] (Fig. 6).

7 Top Submissions in the Robust Model Track

7.1 Team `Petuum-CMU`: First Place Defense Track

By Yaodong Yu, Hongyang Zhang, Susu Xu, Hongbao Zhang, Pengtao Xie and Eric P. Xing.

For our submission, we applied a new formulation for adversarial training, **TRADES** (TRadeoff-inspired Adversarial DEfense via Surrogate-loss minimization) [30], which is able to characterize the trade-off between the natural accuracy and the robustness of the defense model. More specifically, we studied the gap between the *natural error* and the *robust error*, and give tight upper bounds on this gap for a wide class of differentiable surrogate losses. Inspired by our theoretical analysis, we proposed a new formulation for adversarial defenses which consists of two terms, an empirical risk minimization term and a regularization term.

To start with, we use *bold capital* letters, $\boldsymbol{X}$ and $\boldsymbol{Y}$, to represent random vectors, *bold lower-case* letters, $\boldsymbol{x}$ and $\boldsymbol{y}$, to represent the realization of random vectors, *capital* letters, X and Y, to represent random variables, and *lower-case* letters, x and y, to represent realization of random variables. We next define the *natural error* and the *robust error*. The natural error is defined as $\mathcal{R}_{\text{nat}}(f) = \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\left[\arg\max f(\boldsymbol{X}) \neq Y\right]$. The robust error is defined as $\mathcal{R}_{\text{rob}}(f) = \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\left[\exists \boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X}, \epsilon) \text{ s.t. } \arg\max f(\boldsymbol{X}) \neq Y\right]$, where $f(\boldsymbol{X})$ is the output vector of the classification model. Note that the above two terms satisfy $\mathcal{R}_{\text{rob}}(f) \leq \mathcal{R}_{\text{nat}}(f)$, and $\mathcal{R}_{\text{rob}}(f) = \mathcal{R}_{\text{nat}}(f)$ when $\epsilon = 0$. We introduce our new formulation as follows,

$$\min_f \mathbb{E}\Big[\underbrace{\ell(f(\boldsymbol{X}), \boldsymbol{Y})}_{\text{for accuracy}} + \underbrace{\max_{\boldsymbol{X}' \in \mathbb{B}(\boldsymbol{X},\epsilon)} \ell(f(\boldsymbol{X}), f(\boldsymbol{X}'))/\lambda}_{\text{for robustness}}\Big], \tag{7}$$

Algorithm 4 Adversarial training of network

Input: Step sizes η_1 and η_2, batch size m, number of iterations K for perturbation, deep neural network parametrized by θ
Output: Robust network f_θ

Initialize network f_θ
repeat
  Read mini-batch $B = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ from training set
  for $i = 1, \ldots, m$ (in parallel) **do**
    $\boldsymbol{x}'_i \leftarrow \boldsymbol{x}_i + 0.001 \cdot \mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\mathcal{N}(\mathbf{0}, \mathbf{I})$ is a normal Gaussian distribution
    for $k = 1, \ldots, K$ **do**
      $\boldsymbol{x}'_i \leftarrow \Pi_{\mathbb{B}(\boldsymbol{x}_i, \epsilon)}(\eta_1 \mathsf{sign}(\boldsymbol{x}'_i + \nabla_{\boldsymbol{x}'_i} \ell(f_\theta(\boldsymbol{x}_i), f_\theta(\boldsymbol{x}'_i))))$, where Π is the projection operator
    end for
  end for
  $\theta \leftarrow \theta - \eta_2 \sum_{i=1}^{m} \nabla_\theta [\ell(f_\theta(\boldsymbol{x}_i), \boldsymbol{y}_i) + \ell(f_\theta(\boldsymbol{x}_i), f_\theta(\boldsymbol{x}'_i))/\lambda]/m$
until converge

where $\ell(\cdot, \cdot)$ is a multi-class calibrated loss [22], $\boldsymbol{Y}$ is the one-hot vector for indicating the label Y, $\mathbb{B}(\boldsymbol{X}, \epsilon)$ is the neighbourhood of $\boldsymbol{X}$, i.e., $\{\boldsymbol{X}' \in \mathcal{X} : \|\boldsymbol{X}' - \boldsymbol{X}\| \leq \epsilon\}$, and $\lambda > 0$ is a regularization parameter to balance the natural error and the robust error. The first term in (7) is supposed to minimize the loss for natural examples and improve the natural accuracy, the second term in (7) is supposed to push the decision boundary away from the natural examples and improve the robustness of the model. We describe our adversarial training procedure in Algorithm 4.

In our final submission, we applied our new formulation to independently train three deep models, ResNet-18, ResNet-50, ResNet-152 [10], based on Algorithm 4. The logits of the above three defense models were summed together and the weight vector is $\boldsymbol{w} = [w_{\text{res}_{18}}, w_{\text{res}_{50}}, w_{\text{res}_{152}}] = [0.25, 0.25, 0.5]$. The final prediction is the class with the highest score.

7.2 *Team `Wilson`: Second Place Defense Track*

By Xuefei Ning, Wenshuo Li and Yu Wang.

Three techniques are used in our solution:

- Adversarial training using white-box and return-early black-box examples;
- Mutual adversarial training, an extension of Mutual Learning [31];
- Ensemble of models, this mainly helps with the clean accuracy, which suffers from a significant drop after adversarial training, widely known as the Robustness-VS-Accuracy trade-off phenomenon [18, 27, 28].

More specifically, for white-box adversarial examples, we use projected gradient descent (PGD) with random start [18]. To generate black-box adversarial examples we follow three steps:

1. Train another two baselines, VGG11, Inception V4.
2. To generate black-box adversarial examples, we follow the gradients of VGG11, Inception V4, and the supplied Resnet18 baselines to transfer attack on our white-box defended model, which is trained using only white-box adversarial examples. Note that different from the strategy used in [27], in which black-box adversarial examples are generated by randomized FGSM variants (at most 2-step attack is used in the generation), a return-early iterative attack is used for further boosting black-box robustness.
3. We merged these pre-generated black-box adversarial examples together with white-box examples in every training batch.

We find that "return early" in step 2 of the black-box generation is important. Intuitively, by returning early based on the criterion of making a white-box adversarially trained model misclassify, we can generate adversarial examples whose strength is adaptive instead of being the same for all data points. We extensively tried generating non-return-early black-box examples for training but failed to increase performance.

We also developed Mutual Adversarial Training, which is an extension to the Mutual Learning framework [31]: Among the N models that are being mutually trained, we change the inputs of one model to be adversarial examples ADV[x], while using the corresponding clean augmented inputs x for other models. Using the KL divergence KL(MEAN[$p_{-i}(x)$]$||p_i$(ADV[x])) as a regularization term, we update the weights of the i-th model.[3] The above procedure is applied to each model alternately for every training batch. We expect this technique can further increase the invariance of the model to augmentation and adversarial perturbation. In our experiments, we find promising results for black-box robustness on CIFAR-10 but only marginal improvement on Tiny ImageNet.

In our experiments, we fix the hyper-parameters of white-box PGD to eps $=$ 4, step $=$ 1. For black-box adversarial examples generation, we use return-early L_2 iterative transfer attack with step $=$ 10. All experiments were run using batch size $50 \times K$ on a single GeForce GTX 1080Ti, in which $K \leq 4$ implies the number of different adversarial types, including white-box generated one and black-box generated ones. We found that training using a bigger batch size can further improve the performance [14]. Our final submission is an ensemble of three models: two ResNet18 models (trained with mutual and normal adversarial training respectively) and one Inception-V4 model (trained with mutual adversarial training).

[3] $p_i(\cdot)$ denotes the softmax output of the i-th model.

7.3 *Team JeromeR: Third Place Defense Track*

By Jérôme Rony and Luiz Gustavo Hafemann.

This submission used a newly proposed Adversarial Training method based on the Decoupled Direction and Norm (DDN) attack [24]. This approach is similar to the Madry defense [18]: in each training iteration, a strong iterative attack is used to obtain $\tilde{x}$. Training consists in minimizing the cross-entropy of these perturbed samples:

$$\tilde{J}(x, y, \theta) = J(\tilde{x}, y, \theta) \tag{8}$$

While in the Madry defense $\tilde{x}$ is optimized to obtain the largest loss in an $\epsilon-$ball around the input x, in the defense using DDN $\tilde{x}$ is optimized to be the closest adversarial examples:

$$\begin{gathered} \min_{\delta} \|\delta\|_2 \quad \text{subject to} \quad \arg\max_{j} P(y_j | x + \delta, \theta) \neq y_{\text{true}} \\ \text{and} \quad 0 \leq x + \delta \leq M \end{gathered} \tag{9}$$

Where x is the sample, y_{true} is the true label and M defines the bounds of each pixel (e.g. 255). We restrict the perturbation to have a maximum norm, by re-assigning $\delta \leftarrow \epsilon \frac{\delta}{\|\delta\|_2}$ if the norm is larger than ϵ.

The submission used a ResNeXt50-32x4d [29] model pre-trained on Imagenet, that was adversarially trained with the DDN attack for 9 epochs using the TinyImageNet dataset, using and $\epsilon = 2.0$.

8 Conclusion

The goal of the Adversarial Vision Challenge was to stimulate the development of more effective and generally applicable decision-based adversarial attacks as well as models more robust against optimization-based attacks. The winning attacks and defenses employed a wide range of techniques to reach this goal, ranging from more effective adversarial training, model ensembling, attack ensembling and dimensionality reduction. The baseline attacks and defenses, some of which have been state-of-the-art before the challenge, have been vastly outperformed by the winning entries within the attacks scenario of this challenge.

This progress is important and dearly needed to ensure that robustness evaluations of newly proposed defenses can be evaluated more accurately and without manipulations of the underlying architecture (e.g. in the case of gradient masking). It also highlights that deployed machine learning systems should not consider themselves save from adversarial examples just because they do not allow white-box

access to the model internals or limit the number of queries. Much more work is needed towards even better attacks and defenses but the Adversarial Vision Challenge proved to be an important stepping stone towards this goal.

Acknowledgements This work has been funded, in part, by the German Federal Ministry of Education and Research (BMBF) through the Verbundprojekt TUEAI: Tübingen AI Center (FKZ: 01IS18039A) as well as the German Research Foundation (DFG CRC 1233 on "Robust Vision"). The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Rony; Rony acknowledges support by the Bosch Forschungsstiftung (Stifterverband, T113/30057/17); Brendel and Bethge were supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DoI/IBC) contract number D16PC00003.

References

1. Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, July 2018.
2. Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases*, pages 387–402. Springer, 2013.
3. Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, December 2018.
4. Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018.
5. Thomas Brunner, Frederik Diehl, Michael Truong Le, and Alois Knoll. Guessing Smart: Biased Sampling for Efficient Black-Box Adversarial Attacks. *arXiv preprint arXiv:1812.09803*, 2018.
6. Nicholas Carlini, Guy Katz, Clark Barrett, and David L. Dill. Provably minimally-distorted adversarial examples. *arXiv preprint arXiv:1709.10207*, 2017.
7. Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
8. Chuan Guo, Jared S Frank, and Kilian Q Weinberger. Low frequency adversarial perturbation. *arXiv preprint arXiv:1809.08758*, 2018.
9. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
10. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
11. Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.
12. Christian Igel, Thorsten Suttorp, and Nikolaus Hansen. A computational efficient covariance matrix update and a (1+ 1)-cma for evolution strategies. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 453–460. ACM, 2006.
13. Harini Kannan, Alexey Kurakin, and Ian J. Goodfellow. Adversarial logit pairing. *CoRR*, abs/1803.06373, 2018.
14. Kurakin Kannan and Goodfellow. Adversarial logits pairing. *arXiv preprint arXiv:1803.06373*, 2018.

15. Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).
16. Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, Alan Yuille, Sangxia Huang, Yao Zhao, Yuzhe Zhao, Zhonglin Han, Junjiajia Long, Yerkebulan Berdibekov, Takuya Akiba, Seiya Tokui, and Motoki Abe. Adversarial Attacks and Defences Competition. *arXiv preprint arXiv:1804.00097*, 2018.
17. Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016.
18. Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.
19. Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016.
20. Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.
21. Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS' 17, pages 506–519, New York, NY, USA, 2017. ACM.
22. Bernardo Ávila Pires and Csaba Szepesvári. Multiclass classification calibration functions. *arXiv preprint arXiv:1609.06385*, 2016.
23. Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. *arXiv preprint arXiv:1707.04131*, 2017.
24. Jérôme Rony, Luiz G Hafemann, Luiz S Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. *arXiv preprint arXiv:1811.09600*, 2018.
25. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
26. Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
27. Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018.
28. Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.
29. Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5987–5995. IEEE, 2017.
30. Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.
31. Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.