Geometry-Inspired Top-k Adversarial Perturbations

Nurislam Tursynbek[1], Aleksandr Petiushko[2,3], Ivan Oseledets[1,4]
[1]Skolkovo Institute of Science and Technology, [2]Huawei, [3]Lomonosov Moscow State University,
[4]Institute of Numerical Mathematics, Russian Academy of Sciences
nurislam.tursynbek@gmail.com, petyushko.alexander1@huawei.com, i.oseledets@skoltech.ru

Abstract

The brittleness of deep image classifiers to small adversarial input perturbations has been extensively studied in the last several years. However, the main objective of existing perturbations is primarily limited to change the correctly predicted Top-1 class by an incorrect one, which does not intend to change the Top-k prediction. In many digital real-world scenarios Top-k prediction is more relevant. In this work, we propose a fast and accurate method of computing Top-k adversarial examples as a simple multi-objective optimization. We demonstrate its efficacy and performance by comparing it to other adversarial example crafting techniques. Moreover, based on this method, we propose Top-k Universal Adversarial Perturbations, image-agnostic tiny perturbations that cause the true class to be absent among the Top-k prediction for the majority of natural images. We experimentally show that our approach outperforms baseline methods and even improves existing techniques of finding Universal Adversarial Perturbations.

1. Introduction

Along with revolutionizing a wide range of tasks, Deep Neural Networks (DNNs) are intriguingly vulnerable to imperceptibly perturbed inputs, also known as adversarial examples [40, 19, 12]. These malicious well-designed perturbations are carefully crafted in order to cause neural networks to make mistakes. They may attempt to target a specific wrong class to be a prediction (targeted attack), or to yield a class any different from the true one (untargeted attack). Such adversarial perturbations found potential vulnerabilities of practical safety-critical applications of DNNs in self-driving cars [17, 20], speech recognition systems [6, 13], face identification [38, 25]. Moreover, modern defenses to adversarial attacks are found to be ineffective [7, 41]. These security issues compromise people's confidence in DNNs. Thus, it is crucial to investigate and study different types of adversaries on deep learning models.

Although several adversarial attacks are found to be physically realizable [11, 8], the vast majority of them study high frequency pixel-wise perturbations, which heavily exploit the fact that images are in digital domain. However, in many digital real-world applications of DNNs, such as web search engines, recommendation systems, and computer vision cloud APIs (Google Cloud Vision [1], Amazon Rekognition [2], IBM Watson Visual Recognition [3], Microsoft Azure Computer Vision [4], Clarifai [5]), Top-k prediction is more important and meaningful. A user usually gets k most likely classes corresponding to a particular request and some of them are very similar and difficult to differentiate. Therefore, fooling Top-k prediction in such settings is more relevant. Traditional techniques of computing adversarial examples mainly target fooling the Top-1 prediction of DNNs, sometimes even just swapping classes from the Top-2 prediction. This still makes the true class to be present among Top-k prediction. Only a couple of works [23, 45] study Top-k perturbations, however, they lack practical usability and time efficiency. We fill the gap and provide alternative much faster perturbations. We non-trivially extend simple and accurate Top-1 adversary to Top-k case by formulating a multi-objective optimization problem.

Our method is built upon DeepFool [33], a simple and effective approach of constructing small Top-1 adversarial noise. It analytically finds a perturbation in the direction towards classifier's closest linearized decision boundary, which is computed using first-order Taylor approximation. Based on DeepFool, input-agnostic small universal adversarial perturbations (UAPs) were proposed in [32]. Mere addition of such UAPs of a small norm cause neural networks to make mistakes on majority of natural images. The existence and cross-model transferability of such perturbations show the threats of DNNs deployment in the real-world scenarios, as adversaries can straightforwardly compute and exploit them in a malicious manner. However, no UAPs have been proposed to fool Top-k prediction previously. To fill this gap, we propose a systematic algorithm to find universal Top-k perturbations. A visual illustration of a Top-k UAP is shown in Figure 1.

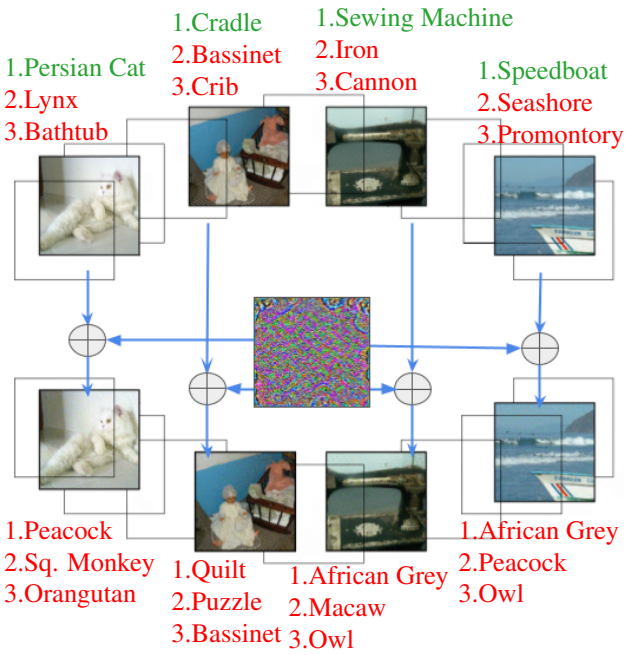


Figure 1. A visual illustration of Top-k Universal Adversarial Perturbation (kUAP) calculated for VGG-16 neural network [39] on ILSVRC2012 dataset. A mere addition of a single small perturbation makes true classes of initial images to be outside of Top-k (here, $k = 3$) prediction of perturbed images for the majority of images, even for unseen images. The $\ell_\infty-$bound (maximal amplitude) of the perturbation is $10/255$.

The main contributions of this paper are following:

- We propose *kFool* - a simple and accurate method to compute a Top-k adversarial perturbation to an image that makes the true class to be absent among the Top-k prediction. Inspired by the idea of DeepFool [33], we linearly approximate decision boundaries and efficiently find such direction that simultaneously push data point maximally closer to classifier's k nearest decision boundaries.

- We show efficacy of *kFool*, by demonstrating that it is possible to construct a Top-k adversarial perturbation of a small magnitude, bounded either in ℓ_2 or ℓ_∞, and compare it to popular existing Top-1 adversarial perturbations crafting techniques.

- We propose *Top-k Universal Adversarial Perturbations* (*kUAPs*), based on kFool, extending the perturbations to image-agnostic scenarios.

- We experimentally show that *kUAPs* outperform baseline methods and even improve existing techniques of generating UAPs on standard ILSVRC2012 validation dataset.

2. Background

Here, we describe preliminaries of adversarial examples to introduce our method. Given an input image $\mathbf{x}_0 \in \mathcal{R}^m$ and an image classifier $F : \mathcal{R}^m \to \mathcal{R}^C$ for C classes, the (Top-1) adversarial perturbation [40] for an input $\mathbf{x}_0$ is a noise $\mathbf{v} \in \mathcal{R}^m$, such that the norm of the perturbation is small, $\|\mathbf{v}\| \leq \varepsilon$, and the perturbed image is missclassified:

$$\arg\max_i F_i(\mathbf{x}_0) \neq \arg\max_i F_i(\mathbf{x}_0 + \mathbf{v}). \tag{1}$$

where F_i is the output logit corresponding to the class i.

The classical work FGSM [19] proposed a single-step way to craft an adversarial perturbation with small $\ell_\infty-$bound value of ε for an input $\mathbf{x}_0$ with a true label y, using gradient of a loss function $\mathcal{L}$ (typically, cross-entropy) between the prediction $F(\mathbf{x}_0)$ and the true label y:

$$\mathbf{x}_{adv} = \mathbf{x} + \varepsilon \ \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(F(\mathbf{x}_0), y)), \tag{2}$$

An iterative version of FGSM with random initialization is Projected Gradient Descent (PGD) [26]. It finds a smaller perturbation but requires a significant amount of time.

Our work is built upon the *DeepFool* [33], where a geometry-inspired fast way was presented. The method is as following: suppose, we have a linear two-class classifier $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$ with a separating plane $f(\mathbf{x}) = 0$ and an input image $\mathbf{x}_0$. The optimal (minimal norm) perturbation is the distance to the separating plane $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}+b = 0$:

$$\mathbf{r}(\mathbf{x}_0) = -\frac{|f(\mathbf{x}_0)|}{\|\mathbf{w}\|_2^2}\mathbf{w}, \tag{3}$$

and its magnitude is $d(\mathbf{x}_0) = \|\mathbf{r}(\mathbf{x}_0)\|_2 = \frac{|f(\mathbf{x}_0)|}{\|\mathbf{w}\|_2}$. For an arbitrary deep differentiable classifier, the first-order Taylor expansion allows to approximately linearize decision boundary and approximate the "slope" $\mathbf{w}$ as:

$$\mathbf{w} \approx \nabla_{\mathbf{x}} f, \tag{4}$$

For a multi-class classifier, "one-vs-all" scheme is used.

Specifically, for an input $\mathbf{x}_0$ and i-th decision boundary:

$$\begin{aligned} f_i(\mathbf{x}_0) &= F_{true}(\mathbf{x}_0) - F_i(\mathbf{x}_0), \\ \mathbf{w}_i &= \nabla_{\mathbf{x}} F_{true}(\mathbf{x}_0) - \nabla_{\mathbf{x}} F_i(\mathbf{x}_0), \end{aligned} \tag{5}$$

Thus, the ℓ_2-minimal perturbation $\mathbf{r}(\mathbf{x}_0)$ to fool this linearly approximated classifier for $\mathbf{x}_0$ can be computed as:

$$\mathbf{r}(\mathbf{x}_0) = \frac{|f_c(\mathbf{x}_0)|}{\|\mathbf{w}_c\|_2^2}\ \mathbf{w}_c, \text{ where } c = \arg\min_{i \neq true} \frac{|f_i(\mathbf{x}_0)|}{\|\mathbf{w}_i\|_2} \tag{6}$$

Using Holder's inequality, the $\ell_\infty-$minimal perturbation is:

$$\mathbf{r}(\mathbf{x}_0) = \frac{|f_c(\mathbf{x}_0)|}{\|\mathbf{w}_c\|_1}\ \text{sign}\ \mathbf{w}_c, \text{ where } c = \arg\min_{i \neq true} \frac{|f_i(\mathbf{x}_0)|}{\|\mathbf{w}_i\|_1} \tag{7}$$

Since the first-order Taylor expansion is a linear approximation, it may deviate from the actual decision boundary of the classifier. Therefore, the procedure should be repeated in an iterative manner: the original image is perturbed, then a new perturbation vector for the perturbed image is computed and so on. However, only few iterations are needed for DeepFool algorithm to quickly reach an incorrect class, finding an efficient Top-1 adversarial perturbation. It usually swaps classes from Top-2 prediction, consequently, Top-k prediction still contains the correct class.

3. kFool

Our target is different: we need to perturb the initial image such that the true class is not only outside the Top-1 prediction, but it is outside the Top-k prediction. Similarly to (1), we formulate the *Top-k* adversarial perturbation for an input image $\mathbf{x}_0$ as a noise $\mathbf{v} \in \mathcal{R}^m$, such that the norm of the perturbation is small, $\|\mathbf{v}\| \leq \varepsilon$, and the original class is outside of the largest k components of $F(\mathbf{x}_0 + \mathbf{v})$:

$$\arg\max_i F_i(\mathbf{x}_0) \notin \arg\operatorname*{sort}_i F_i(\mathbf{x}_0 + \mathbf{v})[:k], \qquad (8)$$

where $\arg\operatorname*{sort}_i$ is the function that returns indices of sorted elements in decreasing order and [: k] shows the first k components. This notation is used for convenience and readability. However, in formal mathematics it can be written as $\{j \mid F_j(\mathbf{x}_0 + \mathbf{v}) \in \arg\max_{A \subset F(\mathbf{x}_0+\mathbf{v}), |A|=k} \sum_{a \in A} a\}$.

The task is usually to find an *optimal perturbation*: the perturbation that satisfies (1) or (8) and has minimal norm. DeepFool attempts to solve this task for Top-1 efficiently, however it does not intend for Top-k. By considering k nearest decision boundaries we can construct such a perturbation in the same computational cost as the DeepFool.

To illustrate kFool in Figure 2, for simplicity, we consider $k = 2$ closest linearized decision boundaries. The DeepFool directions ($\mathbf{r}_1$ or $\mathbf{r}_2$, which are opposite to corresponding normal vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ of decision boundaries) bring data point closer to one boundary, and unfortunately might move away data point from another. Thus, to attack Top-k prediction, the adversary needs to find a direction which brings the data point closer to all k (here, $k = 2$) boundaries (green region in Figure 2) and solves following multi-objective optimization problem:

$$\begin{aligned} &\text{minimize} \quad (\|\mathbf{r}_1(\mathbf{x}_0)\|_2, \ldots, \|\mathbf{r}_k(\mathbf{x}_0)\|_2) \\ &\text{subject to} \quad \mathbf{x}_0 \in X \end{aligned} \qquad (9)$$

where X is the feasible space of images. Using (3) and $f_i(\mathbf{x}) = F_{true}(\mathbf{x}) - F_i(\mathbf{x}) > 0$, the distances are:

$$\|\mathbf{r}_i(\mathbf{x}_0)\|_2 = \frac{f_i(\mathbf{x}_0)}{\|\mathbf{w}_i\|_2} = \frac{\mathbf{w}_i^T \mathbf{x}_0 + b_i}{\|\mathbf{w}_i\|_2} \qquad (10)$$

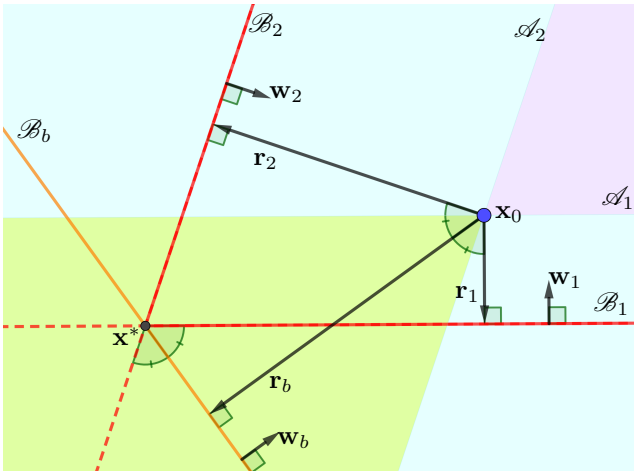


Figure 2. A geometric illustration of a single step of kFool for $k = 2$. The data point $\mathbf{x}_0$ is inside the true class region surrounded by $k = 2$ closest linearized decision boundaries $\mathscr{B}_1$ and $\mathscr{B}_2$ to incorrect classes. Auxiliary planes $\mathscr{A}_1$ and $\mathscr{A}_2$, passing through $\mathbf{x}_0$, are parallel to the boundaries $\mathscr{B}_1$ and $\mathscr{B}_2$ respectively. These planes ($\mathscr{A}_1$ and $\mathscr{A}_2$) split the space into 4 regions. Perturbations in the purple region push away point $\mathbf{x}_0$ from both boundaries. Perturbations in the blue regions bring the point $\mathbf{x}_0$ closer to one boundary, but push away from another (DeepFool). Perturbations in the green region bring the point $\mathbf{x}_0$ closer to k (here, $k = 2$) boundaries (kFool).

We solve (9) in two steps: first we find the direction of the perturbation, then its magnitude. To bring the data point closer to k closest boundaries simultaneously, we need to follow direction that minimizes the sum of distances to them. This is equivalent to opposite of the direction that maximizes the sum. Among all the directions that increase the sum of distances to k nearest boundaries, the gradient (by definition) with respect to the input is the one that increases it the most:

$$\mathbf{w}_b = \arg\max_{\mathbf{x}_0} \sum_{i=1}^{k} \|\mathbf{r}_i\|_2 = \frac{\partial \sum_{i=1}^{k} \|\mathbf{r}_i\|_2}{\partial \mathbf{x}_0} = \sum_{i=1}^{k} \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2} \qquad (11)$$

From basic geometry, the sum of normalized vectors is the direction of the *bisector* between these vectors. The direction of Top-k perturbation $\mathbf{r}_b$, that decreases the most, is exactly opposite to $\mathbf{w}_b$. For $k = 2$, the direction of $\mathbf{r}_b$ is perpendicular to bisector line $\mathscr{B}_b$ of the exterior angle between the boundaries (Figure 2). As we found the direction of the perturbation, next we need the magnitude of $\mathbf{r}_b$. Following the analogy from DeepFool[33] (3), to compute the magnitude of the perturbation $\mathbf{r}_b$, we assume *the most optimal Top-k perturbation is the distance to the bisector line*. For that reason, we need to calculate $f_b = \mathbf{w}_b^T \mathbf{x} + b_b$, for which we need b_b.

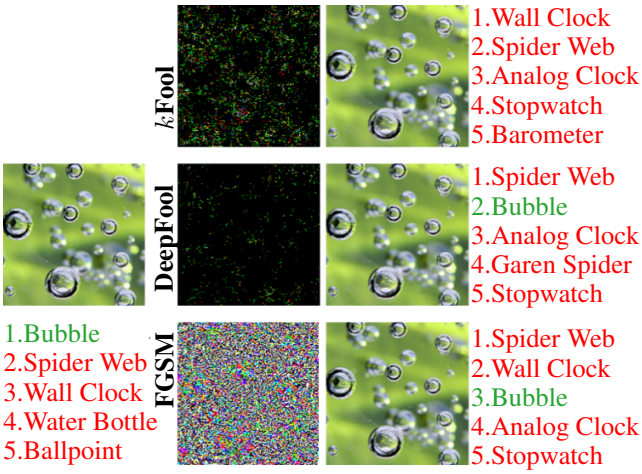


Figure 3. Examples of kFool ($k = 5$), DeepFool [33] and FGSM [19] adversarial perturbations. For the kFool-perturbed image, true class is absent among Top-k prediction, while for the image perturbed by DeepFool and FGSM true class is present among Top-k prediction, which shows the superiority of kFool. Moreover, visually kFool produces perturbation even smaller than FGSM and comparable to DeepFool, however latter two use more simple task statement.

To find b_b we introduce an "intersection" point $\mathbf{x}^*$, where $0 = f_1(\mathbf{x}^*) = f_2(\mathbf{x}^*) = \cdots = f_i(\mathbf{x}^*) = \cdots$. Since there are k equations and $m >> k$ (input dimension) variables, from basic linear algebra such a point exists. Since the bisector line is also passing through this point, then $0 = f_b(\mathbf{x}^*) = \mathbf{w}_b^T\mathbf{x}^* + b_b$ and we have:

$$b_b = -\mathbf{w}_b^T\mathbf{x}^* = -\sum_{i=1}^{k}\frac{\mathbf{w}_i^T\mathbf{x}^*}{\|\mathbf{w}_i\|_2} = \sum_{i=1}^{k}\frac{b_i}{\|\mathbf{w}_i\|_2}. \tag{12}$$

Then:

$$f_b(\mathbf{x}_0) = \mathbf{w}_b^T\mathbf{x} + b_b = \sum_{i=1}^{k}\frac{f_i(\mathbf{x}_0)}{\|\mathbf{w}_i\|_2} \tag{13}$$

Then, using p as an index array of sorted logits $F(\mathbf{x}_0)$ in descending order, setting $f_i(\mathbf{x}_0) = F_{p[i]}(\mathbf{x}_0) - F_{true}(\mathbf{x}_0)$ and $\mathbf{w}_i = \nabla_{\mathbf{x}}F_{p[i]}(\mathbf{x}_0) - \nabla_{\mathbf{x}}F_{true}(\mathbf{x}_0)$, we have:

$$\mathbf{r}_b = -\frac{|f_b(\mathbf{x}_0)|}{\|\mathbf{w}_b\|_2^2}\mathbf{w}_b = \frac{\sum_{i=1}^{k}\frac{f_i(\mathbf{x}_0)}{\|\mathbf{w}_i\|_2}}{\left\|\sum_{i=1}^{k}\frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}\right\|_2^2}\sum_{i=1}^{k}\frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2} \tag{14}$$

Similarly to DeepFool, it might be not enough to add a perturbation only once to satisfy the goal (true class is absent among largest k components), thus we do a few iterations for that (see Algorithm 1).

Extension of (14) to ℓ_∞ is straightforward, as we follow DeepFool's extension in (6) and (7):

Algorithm 1 kFool

INPUT: k, Image $\mathbf{x}$, its label: *true*, classifier F with logits $\{F_1, \ldots, F_C\}$

$p \leftarrow \arg\operatorname{sort}_i(F_i(\mathbf{x}))$ ▷ In descending order
$\mathbf{r} \leftarrow \mathbf{0}$
while $true$ **in** $p[:k]$ **do**
  $\mathbf{w}_b \leftarrow \mathbf{0}$
  $f_b \leftarrow 0$
  for $i = 1$ to $k+1$ **do**:
    $\mathbf{w}_b \leftarrow \mathbf{w}_b + \frac{\nabla_{\mathbf{x}}F_{p[i]}(\mathbf{x}) - \nabla_{\mathbf{x}}F_{true}(\mathbf{x})}{\|\nabla_{\mathbf{x}}F_{p[i]}(\mathbf{x}) - \nabla_{\mathbf{x}}F_{true}(\mathbf{x})\|_2}$
    $f_b \leftarrow f_b + \frac{F_{p[i]}(\mathbf{x}) - F_{true}(\mathbf{x})}{\|\nabla_{\mathbf{x}}F_{p[i]}(\mathbf{x}) - \nabla_{\mathbf{x}}F_{true}(\mathbf{x})\|_2}$
  end for
  $\mathbf{r} \leftarrow \mathbf{r} + \frac{|f_b|}{\|\mathbf{w}_b\|_2^2}\mathbf{w}_b$
  $p \leftarrow \arg\operatorname{sort}_i(F_i(\mathbf{x}+\mathbf{r}))$ ▷ In descending order
end while

OUTPUT: Top-k Adversarial Perturbation $\mathbf{r}$

$$\mathbf{r}_b = \frac{\sum_{i=1}^{k}\frac{f_i(\mathbf{x}_0)}{\|\mathbf{w}_i\|_2}}{\left\|\sum_{i=1}^{k}\frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}\right\|_1}\operatorname{sign}\left(\sum_{i=1}^{k}\frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}\right) \tag{15}$$

Comparative illustration of kFool perturbation is shown in Figure 3. The comprehensive quantitative experimental comparison is presented in the Section 5.1.

4. kUAP

Univesral Adversarial Perturbations (UAPs) [32] solve (1) for most of images simultaneously. To find such a universal direction that fools the majority of images, DeepFool [33] algorithm was applied in an iterative manner over the dataset of images, as it finds a small Top-1 adversarial perturbation efficiently. To satisfy the constraint of smallness of noise, at each time a new perturbation is projected to the ℓ_p-ball, suitable for that.

Inspired by the existence of such directions, we propose Top-k Universal Adversarial Perturbations (kUAPs). Following [32], we apply the kFool algorithm iteratively over a dataset of images, to find a perturbation, mere addition of which to most of natural images makes their true classes to be outside of Top-k prediction. Formally, the goal of kUAP is to find a perturbation $\mathbf{v}$ that satisfies two following conditions:

1. $\mathbb{P}_{\mathbf{x}\sim\mu}\left[\arg\max(F(\mathbf{x})) \notin \arg\operatorname{sort}(F(\mathbf{x}+\mathbf{v}))[:k]\right] \geq \zeta$
2. $\|\mathbf{v}\|_p \leq \varepsilon$

In the above criteria, μ is the distribution of images from which $\mathbf{x}$ is sampled. Adversarial strength ε is the maximum

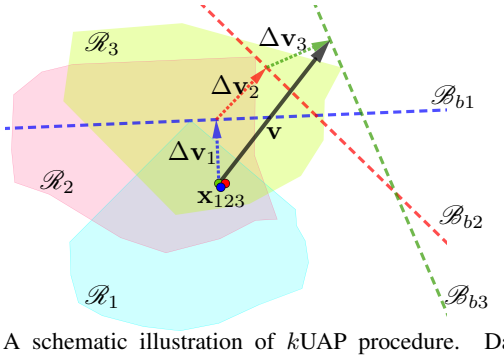


Figure 4. A schematic illustration of kUAP procedure. Data points $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ from different classes (with decision regions $\mathscr{R}_1, \mathscr{R}_2, \mathscr{R}_3$) are super-imposed. Then, kFool is applied iteratively. It first sends points in the direction $\Delta\mathbf{v}_1$ to the bisector line $\mathscr{B}_{b1}$ of the exterior angle between k nearest boundaries for the $\mathbf{x}_1$. Then, in the direction $\Delta\mathbf{v}_2$ to $\mathscr{B}_{b2}$. Then in the direction $\Delta\mathbf{v}_3$ to $\mathscr{B}_{b3}$ and so on. The resulting $\mathbf{v}$ is Top-k UAP.

ℓ_p norm of the perturbation $\mathbf{v}$. The $\arg\operatorname{sort}(F_i(\cdot))[:k]$ operator gets the first k indices of sorted output logits F_i (i.e. the Top-k prediction). The parameter ζ quantifies the desired fooling rate — i.e. the fraction of images Top-k prediction of which should be fooled.

Algorithm. Given a dataset $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \sim \mu$, our proposed algorithm kUAP searches for a direction $\|\mathbf{v}\|_p \leq \varepsilon$, addition of which to $(1-\delta)$ fraction of images makes their true label $(\arg\max_i(F_i(\mathbf{x})))$ to be outside of Top-k prediction $(\arg\operatorname{sort}(F_i(\mathbf{x}+\mathbf{v}))[:k])$. Following [32], we propose to apply kFool (which finds the normal vector to the "bisector of an exterior angle between the nearest k decision boundaries" (see Algorithm 1)) iteratively over data samples from X. The illustrative schematic of the procedure is demonstrated in Figure 4. First, all images are super-imposed into one starting point and $\mathbf{v}$ is initialized as a zero vector. At each iteration i, Algorithm finds kFool direction $\Delta\mathbf{v}_i$ for a given data point $\mathbf{x}_i + \mathbf{v}$, which fools the Top-$k$ prediction for the current image $\mathbf{x}_i$, and updates the current universal perturbation $\mathbf{v}$ simply by $\mathbf{v} = \mathcal{P}_\varepsilon(\mathbf{v} + \Delta\mathbf{v}_i)$. The projector operator $\mathcal{P}_\varepsilon$ controls the criteria $\|\mathbf{v}\|_p \leq \varepsilon$. For example, for $p = \infty$:

$$\mathcal{P}_\varepsilon(\mathbf{v}) = Clip(\mathbf{v}, -\varepsilon, \varepsilon) \tag{16}$$

To improve the quality of kUAP the iterative procedure over X needs to be repeated several times until the desired universal fooling rate $(1-\delta)$ is reached, as in [32] (see Algorithm 2). The universal fooling rate for Top-k prediction is similar to (18), except that $\mathbf{v}$ does not depend on $\mathbf{x}$:

$$\mathrm{UFR}_k[X] = \frac{1}{N}\sum_{i=1}^{N} 1_{\arg\max F(\mathbf{x}_i) \notin \arg\operatorname{sort} F_i(\mathbf{x}_i+\mathbf{v})[:k]} \tag{17}$$

Algorithm 2 kUAP

INPUT: k, ℓ_p-bound ε, fooling rate δ, dataset $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, classifier F

$\mathbf{v} \leftarrow \mathbf{0}$
while $\mathrm{UFR}_k[X] \leq 1-\delta$ **do**
  for $\mathbf{x}_j \in X$ **do**:
    if $\arg\max_i F_i(\mathbf{x}_j) \in \arg\operatorname{sort}_i(F_i(\mathbf{x}_j + \mathbf{v}))[:k]$ **then**:
      $\Delta\mathbf{v}_j = k\mathrm{Fool}(k, \mathbf{x}_j + \mathbf{v}, F)$ ▷ Algorithm 1
      $\mathbf{v} \leftarrow \mathcal{P}_\varepsilon(\mathbf{v} + \Delta\mathbf{v}_j)$
    end if
  end for
end while

OUTPUT: Top-k Universal Adversarial Perturbation $\mathbf{v}$

5. Experiments

5.1. Experiments with kFool

Here, we experimentally show the effectiveness and speed of the kFool algorithm. Different values of k lead to different presentation of the perturbations. In the experiments below we present results for a fixed k, however, the numerical results for other values of k are always similar (see Table 1).

For the experiments below we use following neural network architectures: LeNet [28] for MNIST test dataset, ResNet-20 [22] for CIFAR-10 test dataset and ResNet-18 [22] for ILSVRC2012 [16] validation dataset. To show the effectiveness of kFool ($k = 3; 5$ for MNIST and CIFAR10, $k = 5; 10; 15; 20$ for ILSCVRC2012, for other values of k we got similar results), we compare the Top-k fooling rate with DeepFool [33] and FGSM [19] (90% Top-1 fooling rate). Results shown in Table 1 illustrate that kFool is indeed effective in terms of Top-k fooling rate. The metric to compare fooling rates is:

$$\mathrm{FR}_k[X] = \frac{1}{N}\sum_{i=1}^{N} 1_{\arg\max F(\mathbf{x}_i) \notin \arg\operatorname{sort} F_i(\mathbf{x}_i+\mathbf{v}(\mathbf{x}_i))[:k]} \tag{18}$$

Figure 3 illustrates examples of a kFool adversarial perturbation for $k = 5$, DeepFool [33] perturbation, and FGSM [19] perturbation. It can be observed that kFool produces a hardly perceptible adversarial noise of a small norm. To quantitatively measure the efficiency (smallness) of kFool perturbations, we compare it to existing techniques of generating adversarial examples: FGSM [19] and DeepFool [33]. Following [33], the numerical metric (the lesser - the better) to compare norms of adversarial perturbations for a dataset $\mathcal{D}$ is:

$$\rho_p = \frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}} \frac{\|\mathbf{r}(\mathbf{x})\|_p}{\|\mathbf{x}\|_p} \tag{19}$$

	DF [33]	FGSM [19]	kFool k=3	kFool k=5
FR_1	**1.0**	0.9009	**1.0**	**1.0**
FR_2	0.0	0.4299	**0.9994**	**0.9998**
FR_3	0.0	0.2206	**0.9988**	**0.9994**
FR_4	0.0	0.1181	0.2819	**0.9987**
FR_5	0.0	0.0620	0.0935	**0.9984**

(a) MNIST (LeNet)

	DF [33]	FGSM [19]	kFool k=3	kFool k=5
FR_1	**1.0**	0.8919	**1.0**	**1.0**
FR_2	0.0	0.7851	**0.9972**	**0.9999**
FR_3	0.0	0.6615	**0.9941**	**0.998**
FR_4	0.0	0.5348	0.1928	**0.9962**
FR_5	0.0	0.4367	0.0502	**0.9958**

(b) CIFAR10 (ResNet-20)

	DF [33]	FGSM [19]	kFool k=5	kFool k=10	kFool k=15	kFool k=20
FR_1	**1.0**	0.892	**1.0**	**1.0**	**1.0**	**1.0**
FR_5	0.0	0.538	**0.995**	**0.998**	**1.0**	**1.0**
FR_{10}	0.0	0.428	0.062	**0.998**	**0.997**	**0.999**
FR_{15}	0.0	0.366	0.007	0.201	**0.996**	**0.997**
FR_{20}	0.0	0.328	0.0	0.053	0.301	**0.996**

(c) ILSVRC2012 (ResNet-18)

Table 1. Comparison of fooling rates (18) of DF (DeepFool) [33], FGSM [19], and kFool (ours) for different datasets and architectures.

	Metric	kFool (ℓ_∞)	DF (ℓ_∞)	FGSM (90%)
MNIST	ρ_2	0.6659	0.3277	0.5598
	ρ_∞	0.2456	0.1116	0.1836
CIFAR10	ρ_2	0.0279	0.0122	0.3536
	ρ_∞	0.0165	0.0061	0.1533
ILSVRC2012	ρ_2	0.0061	0.0024	0.0095
	ρ_∞	0.0033	0.0012	0.0042

Table 2. Comparison of average relative ℓ_p-norms (19) of adversarial perturbations by kFool ($k = 3$ for MNIST and CIFAR10, $k = 5$ for ILSVRC2012), FGSM [19] and DeepFool [33] algorithms.

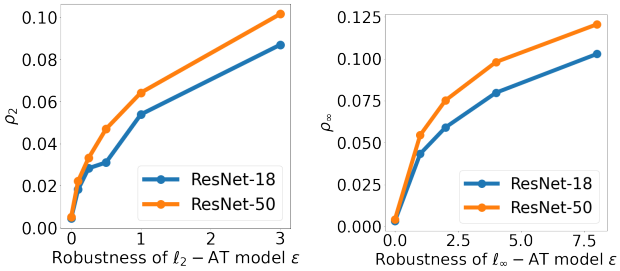
Since FGSM [19] targets the ℓ_∞-bounded perturbation, we use the ℓ_∞ version of DeepFool and kFool for fair comparison (see Table 2). In the case of DeepFool and kFool we reach our desired fooling condition (either Top-1 or Top-k) for 100% of images, however for FGSM increasing ε even to very large values, we cannot reach 100% fooling rate. For this reason, we use such values of ε for FGSM, that guarantee the fooling for some specific number of images (90% Top-1 fooling rate).

Based to the quantitative results in Table 2, it can be seen that kFool generates very efficient perturbation both in terms of ℓ_2 and ℓ_∞ norms. kFool either reaches the same average relative norms ρ_p (19) as FGSM, or outperforms it, and has average relative norms comparable to DeepFool, however the goal of kFool is more challenging, as it targets to perturb input data point such that rue class is outside of Top-k prediction.

We also show the efficiency of kFool in terms of running time. We compared kFool to Top-k PGD attack, which is extension of PGD [26, 31] and Top-k CW [45] attack [45], which is extension of CW [12], for CIFAR-10 ($k = 3$) and Imagenet ($k = 5$). PGD [26, 31] and CW [12] are known to find minimal Top-1 perturbations. To extend PGD to Top-k scenario, we maximize losses of Top-k classes other than the true. As we see in Table 3, kFool 60 times quickly finds Top-k adversarial perturbation compared to Top-k CW [45] for CIFAR-10, and 42 times more quickly for ILSVRC2012, though Top-k CW finds perturbation of lesser norm.

	Top-k CW[45]	Top-k PGD	kFool
Time (CIFAR-10)	30.4s	**0.6s**	**0.5s**
Time (ILSVRC2012)	33.3s	**0.68s**	**0.68s**
FR (CIFAR-10)	**0.994**	0.5	**0.9941**
FR (ILSVRC2012)	**0.999**	0.99	**0.9984**
ρ_2 (CIFAR-10)	0.0094	0.1	0.017
ρ_2 (ILSVRC2012)	0.0022	0.07	0.0043

Table 3. Comparison of sample processing time, fooling rate, ℓ_2 norms of kFool, Top-k CW [45], and Top-k PGD for CIFAR-10 ($k = 3$) and ILSVRC2012 ($k = 5$).



(a) L_2 (b) L_∞

Figure 5. Average relative norms of kFool ($k = 5$) of adversarially trained models over ILSVRC2012 validation dataset

Adversarial training (AT) [19, 31] has been recently proposed as an empirical defense to make models robust to Top-1 adversarial perturbations. AT models are trained on Top-1 PGD adversarial examples instead of clean samples. This models have been shown to be prone to Top-1 adversarial perturbations, however, it is interesting how adversarial training affects norms of Top-k perturbations. To explore this, we tested kFool on AT-models (pretrained from [36]) trained at different robustness strengths ε. The results are shown in Figure 5. As we see from the plots, adversarial training helps to resist not only Top-1 perturbations, but also for Top-k perturbations.

1.Shopping Cart
2.Sleeping Bag
3.Shopping Basket

1.Spider Web
2.Peacock
3.Sock

1.Pillow
2.Quilt
3.Brain Coral

1.Brain Coral
2.Sea Urchin
3.Spider Web

1.Pajama
2.Quilt
3.Umbrella

Figure 6. Examples of perturbed images with a single quasi imperceptible Top-k Universal Adversarial Perturbation generated for MobileNetV2 and $k = 3$. Under each image the wrong Top-3 prediction is shown, when the perturbation is added.

Classifier	**Metric**	**UAP**	**kUAP** ($k = 3$)
ResNet-18	Top-1	0.7725	**0.7789**
	Top-2	0.7015	**0.7109**
	Top-3	0.6598	**0.6720**
VGG-16	Top-1	0.7909	**0.8231**
	Top-2	0.7265	**0.7661**
	Top-3	0.6882	**0.7320**
MobileNetV2	Top-1	0.8851	**0.9154**
	Top-2	0.8373	**0.8791**
	Top-3	0.8033	**0.8550**

Table 4. Universal fooling rates (17) of different architectures

	ResNet-18	VGG-16	MobileNetV2
ResNet-18	**0.6720**	0.2688	0.3040
VGG-16	0.3448	**0.7320**	0.4211
MobileNetV2	0.2465	0.1500	**0.8550**

Table 5. Cross-network transferability of kUAPs ($k = 3$). The rows indicate the network for which the kUAP is computed, and the columns indicate the network for which the fooling rate is reported.

5.2. Experiments with kUAP

For our experiments with ILSVRC2012 [16] dataset we used the following pre-trained architectures: VGG-16 [39], ResNet-18 [22], MobileNetV2 [37].

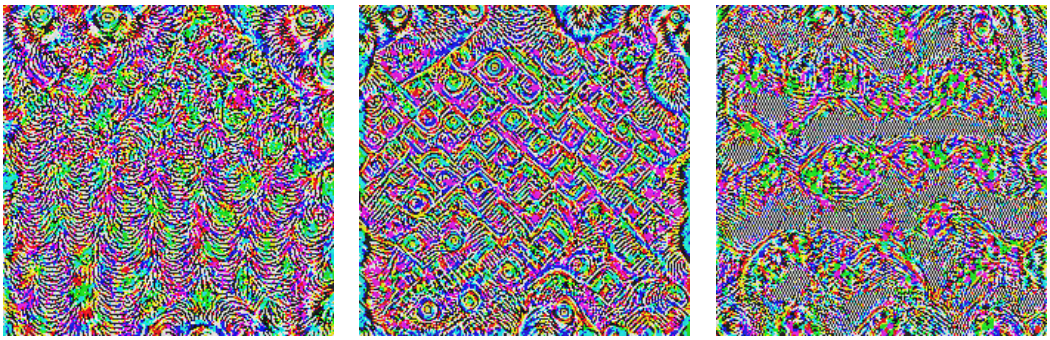
To generate Top-k universal adversarial perturbation we use 10000 images from validation set of ILSVRC2012 [16] dataset, such that each of 1000 classes are represented by 10 samples, as the train set. The remaining 40000 images from ILSVRC2012 validation set is used as the test set. We constraint the universal perturbation $\mathbf{v}$ by ℓ_∞ norm bounded by $\varepsilon = 10$, which is significantly smaller than the average ℓ_∞ norm of the validation set: $\frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}} \|\mathbf{x}\|_\infty \approx 250$. These criteria produces quasi-imperceptible Top-k Universal Adversarial Perturbations. Examples of such perturbed images from test set are shown in Figure 6. In Figure 6 one single Top-3 universal adversarial perturbation, generated using kUAP algorithm for MobileNetV2 [37] architecture, was added to natural images.

We also generate Top-k Universal Adversarial Perturbations using kUAP for different deep neural networks. Figure 7 shows generated kUAPs ($k = 3$) corresponding to ResNet-18 [22], VGG-16 [39], MobilenetV2 [37] for ILSVRC2012 dataset. Similarly to [32], these perturbations contain visually structured patterns, which might reveal some interesting information about DNNs. We report their fooling rates on test set and compare to UAP in Table 4. Even UAP's target is not Top-k prediction, it shows good fooling rate, however kUAP outperforms.

It is well-known that the UAPs [32] have property to transfer across networks, which make them 'doubly-universal'. It is interesting to check if proposed kUAPs are also transferable. It is expected that they are more network-specific, which is indeed confirmed by Table 5, however, the constructed perturbations give fooling rate sufficiently higher than random perturbation.

It should be mentioned that Top-k Universal Adversarial Perturbations shown in Figure 7 are not unique perturbations and there are a numerous perturbations satisfying above criteria. The diversity for example might be reached by changing the training batch of images, however, it is interesting to see how fooling rate depends on the size of training set.

To explore that we select $1, 2, 3, 4$ samples from each class from previous training set (10000 images) which corresponds to $1000, 2000, 3000, 4000$ size values and construct universal perturbation using UAP [32] and our proposed kUAP ($k = 3$). We test all perturbations on the same test set of 40000 images that was used before. Figure 8 demonstrates the Top-3 fooling rate for UAP and kUAP using different sizes of training set. As it can be seen, kUAP generates much stronger Top-k universal adversarial perturbations than UAP [32] for the same size of training dataset.



(a) ResNet-18 (b) VGG-16 (c) MobileNetV2

Figure 7. Result of kUAP ($k = 3$) to different deep neural networks for ILSVRC2012

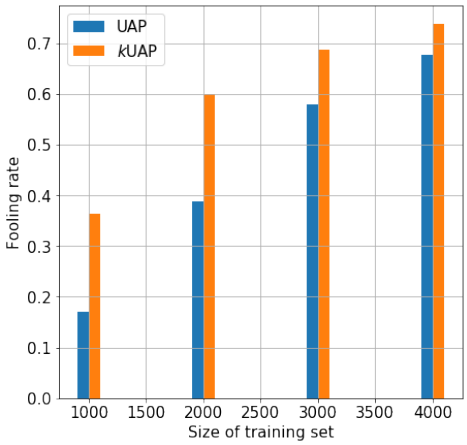


Figure 8. The test set fooling rate on the size of training set

6. Related Work

In the task of image classification, class ambiguity is a common problem especially when the number of classes increases. Thus, it makes sense to allow making k guesses and it motivates to evaluate classifiers based on the Top-k error, instead of the typical Top-1 error. This problem is computationally easier to solve (scales better), and produces the better accuracy score. Several Top-k losses were suggested recently to yield the better Top-k accuracy score [27, 10, 18, 14, 44, 30].

Initially found in [40], adversarial examples have gained significant attention [33, 19, 26, 31, 12, 15, 46]. Goodfellow et.al [19] first proposed a single-step way of constructing adversarial perturbations, and its iterative extension was proposed in [26]. DeepFool [33] is an efficient geometric approach of finding small perturbations. These attacks investigate Top-1 vulnerability of deep learning models.

Our work studies the robustness of Top-k classification. Recently, Jia et al. [23] provided tight bounds of certified robustness for a Top-k adversarial perturbation in ℓ_2 norm, however existing adversarial perturbations are mostly concerned only with Top-1 prediction. In [45] ordered Top-k attack was suggested, however, their method relies on C&W attack [12], which is not an efficient way of constructing adversarial perturbation, as requires a lot of time.

With the discovery of Universal Adversarial Perturbations [32], several other methods were proposed [24, 42, 35, 34, 21, 29, 9]. In [35, 34], it was proposed to craft data-free UAPs, using different objectives. In [24], it was proposed to use $(p, q)-$singular vectors to craft UAPs with a few data samples. Several works proposed to attack images with UAPs in a black-box manner, using Fourier basis[42] or Turing Patterns [43]. In [21], generative models were used to construct UAPs.

7. Conclusion

In this work, we make a step towards geometric understanding of decision boundaries of deep classifiers. We propose an efficient way of constructing Top-k adversarial perturbations and Top-k universal adversarial perturbations. We find our method as a simple, fast and accurate technique. Our method kFool outperforms existing techniques in Top-k fooling rate and finds Top-k adversarial perturbations of small norm. Based on our proposed algorithm kFool we propose kUAPs: single perturbations mere addition of which to most of images pushes away the correct class outside of Top-k prediction. Our method kUAP outperforms UAP both in Top-1 and Top-k fooling rates.

The 'bisector' direction, that simultaneously brings closer several decision boundaries, has interesting interpretation. It normalizes the vectors towards each boundary and sums them up. Similar approaches can be helpful in multi-task learning, when the goal is to solve several tasks simultaneously.

Acknoledgements

This work was supported by the Ministry of Science and Higher Education of the Russian Federation (Grant № 075-15-2020-801).

References

[1] https://cloud.google.com/vision.

[2] https://aws.amazon.com/rekognition/.

[3] https://cloud.ibm.com/catalog/services/visual-recognition.

[4] https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision/.

[5] https://www.clarifai.com/.

[6] Moustafa Alzantot, Bharathan Balaji, and Mani Srivastava. Did you hear that? adversarial examples against automatic speech recognition. *arXiv preprint arXiv:1801.00554*, 2018.

[7] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[8] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018.

[9] Philipp Benz, Chaoning Zhang, Tooba Imtiaz, and In So Kweon. Double targeted universal adversarial perturbations. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[10] Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. Smooth loss functions for deep top-k classification. *arXiv preprint arXiv:1802.07595*, 2018.

[11] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

[12] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[13] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 1–7. IEEE, 2018.

[14] Xiaojun Chang, Yao-Liang Yu, and Yi Yang. Robust top-k multiclass svm for visual category recognition. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 75–83, 2017.

[15] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020.

[16] Jia Deng, Alex Berg, Sanjeev Satheesh, H Su, Aditya Khosla, and L Fei-Fei. Imagenet large scale visual recognition competition 2012 (ilsvrc2012). *See net. org/challenges/LSVRC*, page 41, 2012.

[17] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[18] Yanbo Fan, Siwei Lyu, Yiming Ying, and Baogang Hu. Learning with average top-k loss. In *Advances in neural information processing systems*, pages 497–505, 2017.

[19] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[20] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[21] Jamie Hayes and George Danezis. Learning universal adversarial perturbations with generative models. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 43–49. IEEE, 2018.

[22] 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.

[23] Jinyuan Jia, Xiaoyu Cao, Binghui Wang, and Neil Zhenqiang Gong. Certified robustness for top-k predictions against adversarial perturbations via randomized smoothing. *arXiv preprint arXiv:1912.09899*, 2019.

[24] Valentin Khrulkov and Ivan Oseledets. Art of singular vectors and universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8562–8570, 2018.

[25] Stepan Komkov and Aleksandr Petiushko. Advhat: Real-world adversarial attack on arcface face id system. *arXiv preprint arXiv:1908.08705*, 2019.

[26] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[27] Maksim Lapin, Matthias Hein, and Bernt Schiele. Loss functions for top-k error: Analysis and insights. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1468–1477, 2016.

[28] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[29] Hong Liu, Rongrong Ji, Jie Li, Baochang Zhang, Yue Gao, Yongjian Wu, and Feiyue Huang. Universal adversarial perturbation via prior driven uncertainty approximation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2941–2949, 2019.

[30] Jing Lu, Chaofan Xu, Wei Zhang, Ling-Yu Duan, and Tao Mei. Sampling wisely: Deep image embedding by top-k precision optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7961–7970, 2019.

[31] 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.

[32] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

[33] 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.

[34] Konda Reddy Mopuri, Aditya Ganeshan, and R Venkatesh Babu. Generalizable data-free objective for crafting universal adversarial perturbations. *IEEE transactions on pattern analysis and machine intelligence*, 41(10):2452–2465, 2018.

[35] Konda Reddy Mopuri, Utsav Garg, and R Venkatesh Babu. Fast feature fool: A data independent approach to universal adversarial perturbations. *arXiv preprint arXiv:1707.05572*, 2017.

[36] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? In *ArXiv preprint arXiv:2007.08489*, 2020.

[37] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[38] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

[39] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[40] 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.

[41] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

[42] Yusuke Tsuzuku and Issei Sato. On the structural sensitivity of deep convolutional networks to the directions of fourier basis functions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 51–60, 2019.

[43] Nurislam Tursynbek, Ilya Vilkoviskiy, Maria Sindeeva, and Ivan Oseledets. Adversarial turing patterns from cellular automata. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2683–2691, 2021.

[44] Forest Yang and Sanmi Koyejo. On the consistency of top-k surrogate losses. In *International Conference on Machine Learning*, pages 10727–10735. PMLR, 2020.

[45] Zekun Zhang and Tianfu Wu. Adversarial distillation for ordered top-k attacks. *arXiv preprint arXiv:1905.10695*, 2019.

[46] Tianhang Zheng, Changyou Chen, and Kui Ren. Distributionally adversarial attack. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2253–2260, 2019.