Data-free Universal Adversarial Perturbation and Black-box Attack

Chaoning Zhang* chaoningzhang1990@gmail.com
Philipp Benz* pbenz@kaist.ac.kr
Adil Karjauv* mikolez@gmail.com
In So Kweon iskweon77@kaist.ac.kr

Korea Advanced Institute of Science and Technology (KAIST)

Abstract

Universal adversarial perturbation (UAP), i.e. *a single perturbation to fool the network for most images, is widely recognized as a more practical attack because the UAP can be generated beforehand and applied directly during the attack stage. One intriguing phenomenon regarding untargeted UAP is that most images are misclassified to a dominant label. This phenomenon has been reported in previous works while lacking a justified explanation, for which our work attempts to provide an alternative explanation. For a more practical universal attack, our investigation of untargeted UAP focuses on alleviating the dependence on the original training samples, from removing the need for sample labels to limiting the sample size. Towards strictly data-free untargeted UAP, our work proposes to exploit artificial Jigsaw images as the training samples, demonstrating competitive performance. We further investigate the possibility of exploiting the UAP for a data-free black-box attack which is arguably the most practical yet challenging threat model. We demonstrate that there exists optimization-free repetitive patterns which can successfully attack deep models. Code is available at* https://bit.ly/3y0ZTIC.

1. Introduction

Deep neural networks [28] are widely known to be vulnerable to adversarial examples [52]. This intriguing phenomenon of human imperceptible perturbation fooling the DNN has inspired active research for studying the model robustness against adversarial attack techniques [18, 36, 3]. More surprisingly, [38] shows that a single perturbation can be generated to attack the model for most images. Due to its image-agnostic nature, it is often termed universal adversarial perturbation (UAP). The existence of UAP is especially worrisome because, unlike image-dependent adversarial perturbations, after being generated beforehand the UAP can be applied directly for performing a real-time attack [38].

Our work revisits UAP by providing an alternative explanation on the phenomenon of dominant label, *i.e.* an *untargeted* UAP causing the model to misclassify a large fraction of images to a dominant label, which has been reported in [38, 43] but still lacks a justifiable explanation. Note that this phenomenon is counter-intuitive because unlike targeted UAP [25, 60], the untargeted UAP is not optimized towards any predefined target label. In the targeted setting, [25, 60] have shown that the UAP alone leads the model to output the target class. We observe a similar phenomenon for the untargeted UAP by perceiving the dominant label, which is the result of the optimization algorithm instead of being predefined, as the pseudo-target class. When added to the images, the UAP causes the average logit values over all adversarial examples to be somewhat proportional to the logit values with the UAP alone as the input. This result is further collaborated by the layer-wise and step-wise model response analysis, suggesting untargeted UAP has a dominant contribution to the model response and there exists a positive correlation between this dominant influence and fooling ratio as the training evolves. This observation motivates to further investigate simple yet effective techniques towards more practical universal attacks under the data-free constraint, in both white-box and black-box settings.

Overall our contributions are shown as follows:

- We revisit the mechanism behind the dominant label phenomenon caused by an untargeted UAP. Specifically, we show that the existing explanation [38] hypothesizing a dominant label occupying a large image space can not justify some observed phenomena based on some reasonable assumptions. We provide an alternative explanation with the observation that untargeted UAP has a dominant contribution to the model response of adver-

*Equal contribution

sarial examples. Nonetheless, the untargeted UAP still does not lead to the misclassification of all images, for which we find that some samples tend to be systematically more robust against the explored untargeted UAP and they tend to have repetitive semantic content.

- Our findings motivate the investigation of untargeted UAP towards a more practical attack by alleviating the dependence on the original training samples in a progressive manner from removing the need for sample labels to limiting the sample size. Specifically, we adopt a self-supervision cosine similarity loss to optimize the untargeted UAP and reduce the sample size by common augmentation techniques. Towards strictly data-free UAP, we propose to adopt artificial jigsaw images of variable frequency as the training samples. Our work suggests the benefit of designing artificial images that mimic the properties of natural images.

- We further investigate whether the UAP can be exploited for facilitating practical data-free black-box attack, also termed no-box attack in [31]. Interestingly, we find that optimization-free repetitive content, such as vertical/horizontal or checkerboard pattern, is sufficient enough for a strong attack. It outperforms an existing sophisticated optimization-based method [31] which is resource-intensive and not strictly data-free. Beyond the deep classifier, we further demonstrate this attack is effective for attacking DNNs in other applications, such as object detection and semantic segmentation.

2. Related Work

Basic Attack Methods. Szegedy *et al*. first found and optimized adversarial examples by using box-constrained L-BFGS [52]. DeepFool [40] exploits the decision boundary to update the perturbation in the direction of minimizing the perturbation budget in each iteration. Incorporating the minimization of the perturbation magnitude into the optimization function, Carlini and Wagner (C&W) introduce a famous attack named after the two authors [3]. The above methods are all cumbersome and slow. To mitigate this, [18] has introduced an efficient one-step attack method, widely known as the Fast Gradient Sign Method (FGSM). [27] has extended the basic FGSM to its iterative variant, *i.e*. I-FGSM, which limits the perturbation update at each iteration to only a fraction of the allowed total perturbation budget. Projected gradient descent (PGD) [36] is another widely adopted effective multi-step attack.

Universal Attack Methods. The above basic attack methods can be easily adapted to UAP. For example, [38] has first discovered the existence of UAP and generating it through applying DeepFool [40] iteratively. Generative Adversarial Perturbations (GAP) were proposed by Poursaeed *et al*. [46], using generative models to craft the UAP. In another variant, UAPs are crafted by leveraging the Jacobian matrices of the networks' hidden layers [26]. Assuming no access to the original training data, Fast Feature Fool has been proposed in [42] to generate data-free UAPs by optimizing the feature change caused by the applied UAP. More follow-up works [42, 41, 47, 34, 32] have attempted at addressing this data-free challenge. Recently, the investigation of UAP has appeared in a wide range of applications [23, 29, 45, 1, 1, 13, 55, 30, 24], which has been summarized in a recent survey [62] on this topic.

Explanation on the Adversarial Vulnerability. Numerous works have attempted at explaining the reason for adversarial vulnerability from various perspectives, such as local linearity [18, 2, 53, 54], high-dimensional input properties [50, 11, 37, 17], over-fitting [49, 54], and robustness under noise [12, 16, 8]. The investigation on the vulnerability to UAP is relatively limited. [38] claims that the significant fooling ratio gap between UAP and random perturbations suggests redundancies in the geometry of the decision boundary. In [39], the authors have further analyzed the UAP existence from the geometry perspective. Specifically, the existence has been attributed to the *positively curved* decision boundary [39]. Through studying the UAP, [25] has found that the predictive power and adversarial vulnerability of the studied deep classifier are intertwined, suggesting any gain in robustness must come at the cost of accuracy. Focusing on the targeted setting, [60] shows that targeted UAP has dominant features of the predefined target class. In contrast, focusing on the more general untargeted setting, our work is motivated to explain why *untargeted* UAP leads the model to fool most images to a dominant label, which is a widely known phenomenon but still lacks a convincing explanation. One predominant explanation [23] hypothezes that dominant label occupies a large image space. Recently, a concurrent work [56] has also analyzed the dominant label phenomenon of UAP, providing an explanation from the geometric and data-feature perspective in the task of speech command classification.

3. Background and Algorithm Comparison

Untargeted UAP Task Definition. Following [38], we adopt X to denote a distribution of images in $\mathbb{R}^d$ and k is used to define a deep classifier as a function that outputs a predicted label $\hat{k}(x)$ for each image $x \in X$. The goal of UAP is to seek a single perturbation vector ν, *i.e*. UAP, such that

$$\hat{k}(x+\nu) \neq \hat{k}(x) \text{ for most } x \sim X \quad \text{s.t.} \quad ||\nu||_p \leq \epsilon. \quad (1)$$

ν obeys the constraint that its l_p-norm is smaller than a predefined magnitude value ϵ for making it quasi-imperceptible. For consistency we follow prior works [38, 46, 41] and adopt $l_\infty = 10/255$. Unless otherwise specified, the UAP in this

work is by default untargeted. Following prior works, the *fooling ratio* is adopted to evaluate the UAP effectiveness, defined as the percentage of the samples that change its prediction under the UAP attack. We evaluate the generated UAPs on the ImageNet validation dataset.

Algorithm Discussion. The vanilla UAP method [38] accumulates image-dependent perturbations to the final universal perturbation. These image-dependent perturbations are iteratively generated via the attack method DeepFool [40]. To differentiate from the generated UAP, we term this vanilla UAP method DeepFool-UAP. Instead of optimizing the perturbation directly, Poursaeed *et al*. proposed a generator-based method (GAP) [46] training a generative network to output a UAP. Compared to the DeepFool-UAP, the generator-based method has the benefit that a batch of images can be used to train the generator network instead of processing each image individually. Concurrent to [46], [20] adopts a similar approach. Combining the merits of the DeepFool-UAP [40] and GAP [46], direct optimization of the UAP with batches of images results in a simple yet effective UAP algorithm. A similar approach has been adopted in [51] for performing universal adversarial training. Algorithm 1 outlines the procedure. Most works adopt the cross-entropy loss that requires the ground-truth labels of the training dataset. In practice, however, the ground-truth labels might not be available. To this end, we propose to optimize the UAP in a self-supervised manner with a new loss to minimize the cosine similarity ($CosSim$) as follows:

$$\mathcal{L} = CosSim(k(x), k(x+\nu)) \tag{2}$$

where $k(\cdot)$ indicates the DNN output logit vector, *i.e*. before the $\arg\max$ operation *vs*. the predicted class indicated by $\hat{k}(\cdot)$ (see Eq. 1). Intuitively, with this loss the perturbation ν can be optimized such that $k(x)$ and $k(x+\nu)$ are far from each other, consequently resulting in a change in the class prediction for x. As a control study, we also experiment with another variant of cosine similarity loss that maximizes the cosine similarity between $k(\nu)$ and $k(x+\nu)$. To differentiate it from the loss in Eq. 2, we term it CosSim-max which is empirically found to achieve inferior performance compared with that in Eq. 2 (see Table 1). The design of these two variants of losses is partly motivated by the results in Figure 2. Assuming full availability of the training dataset, we compare the adopted Cosine-UAP with multiple existing SOTA UAP methods [38, 46, 20, 43, 60], which all use the ground-truth labels, and the results are shown in Table 1. Despite being label-free, the Cosine-UAP achieves competitive performance. Generator-based approaches [46, 20, 43] require training an additional network, thus they are more resource-intensive. On a single GPU, DeepFool-UAP [38] requires multiple hours to craft a UAP, while Cosine-UAP only takes around 1 minute by optimizing the UAP for 1000 iterations. Due to its effectiveness and efficiency, Cosine-UAP with the CosSim loss is adopted in the remainder of this work for addressing the challenge of generating the UAP with limited unlabeled or no training images.

Algorithm 1: Cosine-UAP

Input: classifier k, loss $\mathcal{L}$, batch size m, number of iterations N, allowable magnitude ϵ
Output: perturbation vector ν

$\nu \leftarrow 0$ ▷ initialization
for *iteration* $= 1, \ldots, N$ **do**
  $B \sim \mathcal{X}_\nu$ ▷ samples with $|B| = m$
  $g_\nu \leftarrow \mathop{\mathbb{E}}_{x \sim B}[\nabla_\nu CosSim(k(x), k(x+\nu))]$ ▷ Gradient
  $\nu \leftarrow \texttt{Optim}(g_\nu)$ ▷ ν update
  $\nu \leftarrow \min(\epsilon, \max(\nu, -\epsilon))$ ▷ ν clipping
end

Table 1. UAP algorithm comparison on the ImageNet validation dataset with the metric of fooling ratio (%). The algorithms are trained on the ImageNet training dataset. The results except Cosine-UAP, such as DeepFool-UAP [38] and GAP [46], DF-UAP [60], are reported as in the original papers. DF-UAP term for [60] is adopted following [62].

Method	AlexNet	GoogleNet	VGG16	VGG19	ResNet152
DeepFool-UAP [38]	93.3	78.9	78.3	77.8	84.0
GAP [46]	-	82.7	83.7	80.1	-
UAN [20]	-	-	-	84.6	88.1
NAG [43]	-	90.37	77.57	83.78	87.24
DF-UAP [60]	96.17	88.94	94.30	94.98	90.08
Cosine-UAP (CosSim-max)	95.7	90.7	94.6	91.1	80.2
Cosine-UAP (CosSim)	**96.5**	**90.5**	**97.4**	**96.4**	**90.2**

4. Intriguing Phenomenon of Untargeted UAP

We attempt to analyze the intriguing dominant label phenomenon observed in untargeted UAP [38, 43]. Unless otherwise specified, we adopt a VGG16 network as the victim model for performing the analysis.

UAP Leading Most Images to a Dominant Label. The authors of [38] first report this phenomenon and note that the dominant label is found by their algorithm automatically without any prior. For explaining this intriguing phenomenon, the authors of [38] stated:

"*We hypothesize that these dominant labels occupy large regions in the image space, and therefore represent good candidate labels for fooling most natural images.*"

Their explanation hypothesizes that the frequency of misclassified prediction class is correlated to the region of the corresponding label. Here, we *assume* that (a) input space region is the only reason for the explanation and (b) the percentage of a certain new class is proportional to its corresponding input space region. Note that the authors of [38] do not explicitly make the above assumptions and the assumptions are

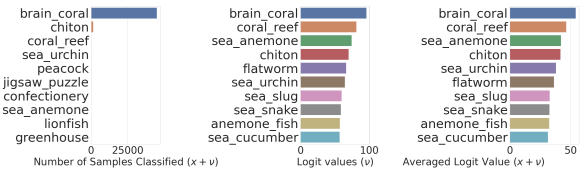


Figure 1. Number of adversarial examples classified as a certain class (left), top logit values of the used UAP ν (middle), and top average logit values over all adversarial examples $x+\nu$ (right).

made here to simplify our following reasoning. ImageNet dataset has 1000 classes and thus the random chance is 0.1% for a sample being any arbitrary class if we assume each class has an identical region in the image space. We find that on VGG16, under the attack of UAP (with an overall fooling ratio of 97.4%), 90.1% of all samples are classified as the dominant label "brain coral", see Figure 1 (left). Given the above hypothesis [38], the value 90.1% implies that in the image space, the label "brain coral" occupies a significantly larger region than the region occupied by the remaining 999 classes combined. If this would be the case, most (random) perturbations would consequently result in a misclassification to this dominant label. Under a certain random noise, we observe a fooling ratio of 42.7% and only 0.1% (50 samples) of all samples are classified as the label of "brain coral", suggesting that "brain coral" does not necessarily occupy a significantly larger region in the image space.

We empirically find that one factor that determines the dominant label is the class of the training samples (see relevant results in the supplementary). We optimize a UAP with our Cosine-UAP but only on samples from a single class. We find that limiting the training samples to a single class only leads to a marginal performance gap compared with utilizing all classes. The resulting UAP with different runs in most cases leads to the same dominant label. We repeat the experiment by changing that single class for sampling the training images and, accordingly, we observe a new dominant label. This phenomenon further suggests that large space regions are not the major reason for the dominant label phenomenon; otherwise, the dominant label should not change with the chosen single class.

An Alternative Explanation. The above analysis shows that the hypothesis of "occupying large regions in the image space" can not explain why the UAP causes many samples to a certain dominant label. This phenomenon is difficult to explain by focusing on the behavior of images under the influence of UAP, *i.e.* comparing $k(x)$ and $k(x+\nu)$. We find that this phenomenon can be intuitively explained by comparing $k(\nu)$ and $k(x+\nu)$. Specifically, we evaluate the UAP on the ImageNet validation images and report the ordered logits of $k(\nu)$ in Figure 1 (middle) as well as the ordered averaged logits of $k(x+\nu)$ in Figure 1 (right). We find that the class distribution for $k(\nu)$ and $k(x+\nu)$ almost matches exactly, indicating that the logit distribution of the perturbation dominates that of the images x. Further underlining this phenomenon, most of the classes of the UAP's top logit values $k(\nu)$ are also found among the most classified samples. With such dominant influence, it is not surprising that most of the samples are classified as the dominant label even though the UAP is untargeted without any predefined label. Despite having a much smaller magnitude, the untargeted UAP overshadows the contribution of images for DNN response.

We further perform depth-wise and step-wise analysis of dominant label influence caused by Cosine-UAP. Feeding an input to the model, we calculate its channel-wise average for a certain feature layer (*i*-th layer for instance). The resulting feature vector $k_i(\cdot)$ layer is the model response triggered at *i*-th layer by the input. Image (x) and UAP (ν) trigger their corresponding model responses, *i.e.* $k_i(x)$ and $k_i(\nu)$. When they are combined as an adversarial example $x+\nu$, they trigger a joint response $k_i(x+\nu)$. Here, we calculate the cosine similarity between $k_i(x)$ and $k_i(x+\nu)$ denoted by $cos_i(x, x+\nu)$ and that between $k_i(\nu)$ and $k_i(x+\nu)$ denoted by $cos_i(\nu, x+\nu)$. We investigate and measure such similarity by adopting the widely used cosine similarity metric. Here, the cosine similarity value ranges from 0 to 1, with a value close to 0 (1) indicating a small (large) contribution to the jointly triggered response.

In the untargeted setting, we visualize $cos_i(x, x+\nu)$ and $cos_i(\nu, x+\nu)$ in Figure 2 (left), where we randomly sample 100 samples and report their mean and standard deviation. We observe that $cos_i(x, x+\nu)$ is larger for the first few shallow layers , while $cos_i(\nu, x+\nu)$ is significantly larger for most layers, especially the deep layers. Similar behavior has also been reported in the targeted setting [61]. It is worth mentioning that the Cosine-UAP adopting the loss in Eq.2 does not directly encourage a maximization of $cos_i(\nu, x+\nu)$. Intuitively, we can perceive the dominant label as a pseudo-target class and directly maximize $cos_i(\nu, x+\nu)$ as discussed in Sec. 3, however, it leads to inferior performance. We confirm that combining the two losses together works worse than only adopting the loss in Eq.2. In other words, the dominant label influence of untargeted UAP is a natural choice of the optimization algorithm, not necessarily subject to the loss choice. For example, we observe that untargeted GD-UAP [41] also shows overall similar behavior but with some nuanced difference, *i.e.* relatively lower $cos_i(\nu, x+\nu)$ compared with that in Cosine-UAP especially in the very last few layers (see relevant results in the supplementary). This might be explained by the fact that GD-UAP optimizes the loss at all layers. We further investigate how the influence of UAP/image on the triggered joint response during the training as well as their relationship with the fooling ratio. Specifically, we only visualize the response of the final logit

layer for simplicity and the results are shown in Figure 2 (right). We observe that $cos_i(\nu, x+\nu)$ increases as training evolves. Moreover there is a positive correlation between the fooling ratio and $cos_i(\nu, x+\nu)$ and negative correlation between the fooling ratio and $(cos_i(x, x+\nu))$.

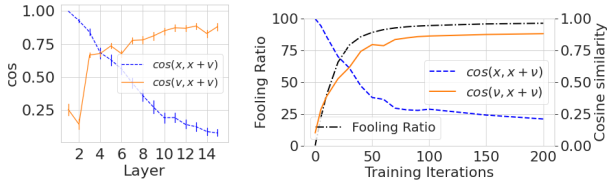


Figure 2. Layer-wise (left) model and step-wise (right) analysis of the UAP on the model response in the untargeted setting.

Existence of Robust Samples. [61] reports that there exists a class-wise robustness gap under the attack of targeted UAP. Here, we perform sample-wise UAP robustness in the untargeted setting. Due to the image-agnostic property, UAP fools the model for most but not all images. We term those samples that keep the original prediction, *i.e.* $\hat{k}(x+\nu) = \hat{k}(x)$, *robust samples*. Here, we first investigate whether these robust samples just happen to be robust to a certain UAP on a specific model or are they inherently more robust to the UAP regardless of the evaluated UAP or model. We first analyze the different UAPs generated on the same model (VGG16) with different runs and the results are available in Table 2. We observe that the overlapping ratio is very high. Take UAP#3 and UAP#4 as an example. Out of 50,000 evaluation images, 1282 of them are robust samples for UAP#3, and 1307 of them are robust samples for UAP#4. The over-lapping number between them is 1028, suggesting a very high over-lapping ratio taking a total of 50,000 samples into account. A cross-model analysis is shown in Table 3, where also a high overlapping ratio across samples can be observed. This suggests that whether a sample is robust or vulnerable is not random, instead, there exists a systematic factor(s) that affects the sample robustness. A preliminary check of the difference between robust and vulnerable samples shows that robust samples tend to have more edge or contrast content, such as repetitive patterns (see the relevant results in the supplementary). Our results align well with [61] that identifies frequency as a factor for the class-wise robustness gap against targeted UAP. Our results can also be explained from the perspective of DNNs being more biased to texture, *i.e.* content with HF property, instead of shape [15, 7, 57].

5. Crafting UAP with Limited or No Data.

In practice, due to security or secrecy concerns, a model manager is unlikely to open-source their training dataset. Thus, a line of works have attempted to craft UAP with limited [26, 34] or no [42, 41, 47, 34] data. Overall, there is a general consensus among the UAP researchers that crafting UAP with limited or no data is challenging.

Table 2. The number of robust samples overlapping across different UAPs. Diagonal entries indicate the total number of robust samples for the corresponding UAP, while other numbers indicate the over-lapping number between any two UAPs (generated on the same target model VGG16).

	UAP #1	UAP #2	UAP #3	UAP #4	UAP #5
UAP #1	**1438**	1096	1076	1056	1080
UAP #2	1096	**1352**	1062	1056	1068
UAP #3	1076	1062	**1282**	1028	1044
UAP #4	1056	1056	1028	**1307**	1051
UAP #5	1080	1068	1044	1051	**1329**

Table 3. The number of robust samples overlapping across the UAPs generated on different networks. Diagonal entries indicate the total number of robust samples for the corresponding network, while other numbers indicate the overlapping number between any two network-specific UAPs.

	AlexNet	GoogleNet	VGG16	VGG19	ResNet152
AlexNet	**1748**	1047	658	799	1047
GoogleNet	1047	**4745**	1046	1344	2749
VGG16	658	1046	**1309**	962	1035
VGG19	799	1344	962	**1816**	1343
ResNet152	1047	2749	1035	1343	**4890**

5.1. UAP With Limited Training Samples.

With GoogleNet as the target model, [38] showed that limiting the data sample size to 500 achieves a fooling ratio only slightly above 30%. [26, 34] have also investigated how to craft an effective UAP with limited training data and their performance is still far from the performance when the full training dataset is given. We identify the main reason that reduces the attack success rate as being over-fitting to the limited images. One straightforward way to improve the UAP generalization capability is to perform data augmentation. We adopt heavy augmentation techniques, such as random rotation (5 degrees), random crop, random horizontal and/or vertical flips. After data augmentation, the main object in the image is often not recognizable in the image. This might be an issue for algorithms that depend on the ground-truth labels. This is not an issue in the adopted Cosine-UAP algorithm due to the self-training mechanism. Table 4 shows that Cosine-UAP achieves competitive results. This further motivates us to adopt jigsaw images as the alternative training dataset for crafting UAP with no data.

5.2. Strictly Data-free Untargeted UAP

Motivation for Artificial Images. For some applications where the data involves high security, even containing a small number of training samples might be challenging for an attacker. [60] shows that proxy dataset can be exploited for

Table 4. Fooling ratio for UAPs crafted with limited samples.

Method	# samples	VGG16	VGG19	ResNet50
Singular Fool [26]	64	52.0	60.0	44.0
GD-UAP [41]	49	72.80	67.60	56.40
PD-UA (49) [34]	49	70.60	73.30	65.80
Cosine-UAP	64	**96.0**	**94.9**	**91.8**
Cosine-UAP	32	**93.5**	**93.5**	**91.8**

generating targeted UAP, however, in the untargeted setting, this proxy dataset is still required to be the original training dataset. Towards strictly data-free untargeted UAP, we aim to approximate those training samples by imitating their characteristics with artificial images without the need for a proxy dataset. Due to the domain gap between training samples and artificial images, a performance drop is expected. To reduce the performance drop, the artificial images need to have two properties for resembling the training samples: (a) locally smooth for resembling natural images; (b) mixed frequency pattern for improving diversity. To prove the concept without losing generality, we propose an artificial jigsaw image as a simple solution to fulfill the above two criteria. Alternative sophisticated artificial patterns might lead to superior performance, however, optimizing such patterns is beyond the scope of this work.

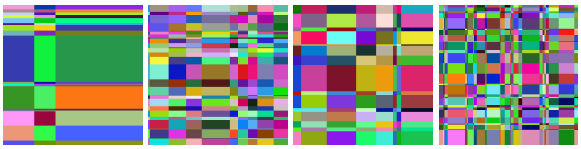
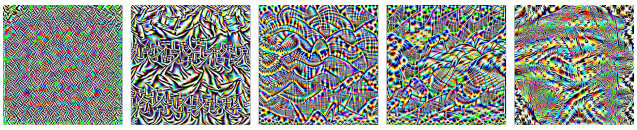


Figure 3. Four examples of jigsaw images.



AlexNet GoogleNet VGG16 VGG19 ResNet152

Figure 4. UAPs trained on jigsaw images for different networks.

Experimental Results with Jigsaw Images. To this end, a large body of works has explored data-free UAPs[42, 41, 47, 34]. We generate the jigsaw images with random frequency patterns as shown in Figure 3. The resulting UAPs are shown in Figure 4. Our work is not the first attempt towards strictly data-free UAP and the comparison with existing methods is show shown in Table 5. PD-UA [34] deploys a Monte Carlo sampling method to increase the model uncertainty. Despite its delicate design, the performance improvement over GD-UAP [41], is around 5% points. The performance of AAA [47] is better than that of other methods but still worse than ours. Note that their AAA approach

Table 5. Comparison of the proposed method to other data-free methods with the metric of fooling ratio (%). The first row reports the DeepFool-UAP [38] that uses the training samples. Thus it is not data-free; however, we use it as a benchmark to indicate the gap between the data-free methods and DeepFool-UAP [38] with data. The gap is indicated in the bracket for the "average" column. "Prior" denotes the range prior in [41].

Method	AlexNet	GoogleNet	VGG16	VGG19	ResNet152	average
DeepFool-UAP (**with data**)	93.3	78.9	78.3	77.8	84.0	82.46
FFF [42]	80.92	56.44	47.10	43.62	29.78	51.27(−30.89)
GD-UAP (w/o Prior) [41]	84.88	58.62	45.47	40.68	29.78	51.59(−30.57)
GD-UAP (with Prior) [41]	87.02	71.44	63.08	64.67	37.3	64.40(−17.76)
AAA [47]	89.04	75.28	71.59	72.84	60.72	73.59(−8.57)
PD-UA (w/o Prior) [34]	–	67.12	53.09	48.95	53.51	–
PD-UA (with Prior) [34]	–	–	70.69	64.98	46.39	–
Ours (Jigsaw images)	**91.07**	**87.57**	**89.48**	**86.81**	**65.35**	**84.08**(+**1.60**)

Table 6. Ablation results for different artificial images.

Artificial images	AlexNet	GoogleNet	VGG16	VGG19	ResNet152	Average
Uniform Random Noise	82.6	40.3	72.3	64.4	47.2	61.4
Gaussian Noise [41]	89.5	48.7	76.1	75.4	49.9	67.9
Flat Images	80.0	39.9	81.3	79.5	29.9	62.1
Jigsaw with fixed frequency	89.1	78.6	85.6	80.9	62.9	79.4
Jigsaw with variable frequency	**91.1**	**87.6**	**89.5**	**86.8**	**65.4**	**84.08**

Table 7. Fooling ratio for regular and adaptively trained UAP when evaluated on validation data of different input sizes. Regular UAP is over-fitting to the input size of 224 × 224. Adaptive UAP can mitigate this over-fitting to some extent.

UAP Type	112	168	224	280	336
Regular UAP	58.57	54.13	89.48	64.81	62.90
Adaptive UAP	62.05	64.34	86.16	70.00	71.51

Table 8. Transferability results for various UAP methods with VGG16 as the source model.

Target Model	Technique	VGG16	VGG19	ResNet50	ResNet152	GoogleNet
VGG16	GD-UAP	45.47	38.20	27.70	23.80	34.13
	GD-UAP+P	51.63	44.07	32.23	28.78	36.79
	UA	48.46	41.97	29.09	24.90	35.52
	PD-UA	53.09	49.30	33.61	30.31	39.05
	Ours (Regular UAP)	**89.48**	**76.84**	**44.11**	**38.37**	**48.97**
	Ours (Adaptive UAP)	**86.16**	**77.88**	**49.30**	**44.27**	**56.96**

requires to emulate the effect of data samples by optimizing a large number of samples of class impression [47]. Thus their approach is much complex and resource-intensive. Our Jigsaw images can be directly designed on-the-fly. Overall, our simple approach outperforms other methods by a non-trivial margin. It is worth mentioning that our data-free approach achieves comparable (marginally better) performance as DeepFool-UAP that utilizes the training dataset. Overall, our proposed jigsaw solution outperforms the existing approaches by a significant margin. The ablation study is provided in Table 6 to justify the choice of jigsaw images with variable frequency.

Adaptive Input Size. In practice, a CNN trained on a fixed input size can work for different input sizes during the inference stage [22]. Prior works on UAP do not take an adaptive input size into account and thus the fooling

Table 9. Performance comparison on ImageNet under the threat model of data-free black-box attack. Beyonder indicates a baseline approach with a full training dataset. Following [31], the prediction accuracy on adversarial examples under $\epsilon = 0.1$ is reported (lower $\downarrow$ is better). "None" indicates that no HF removing method is used to craft an adversarial example, "SVD" means that singular value decomposition of the image is utilized for HF removal, and "FT" shows that Fourier transform is applied to perform HF removal.

Method	VGG-19	Inception v3	ResNet	DenseNet	SENet	WRN	PNASNet	MobileNet v2	Average
Beyonder	24.9%	51.1%	30.3%	27.1%	43.7%	33.9%	51.8%	27.0%	36.2%
Naïve‡ [31]	45.9%	63.9%	60.6%	56.4%	65.5%	58.8%	73.1%	37.7%	57.7%
Jigsaw [31]	31.5%	50.2%	46.2%	42.3%	59.0%	51.2%	62.3%	25.2%	46.0%
Rotation [31]	31.1%	48.1%	47.4%	41.2%	58.2%	50.7%	59.9%	26.0%	45.3%
Naïve† [31]	76.2%	80.8%	83.7%	78.9%	87.0%	84.1%	86.9%	72.4%	81.2%
Prototypical [31]	19.7%	36.4%	37.9%	29.1%	44.5%	37.2%	48.5%	17.7%	33.9%
Prototypical* [31]	18.7%	33.6%	34.7%	26.0%	42.3%	33.1%	45.0%	16.3%	**31.2%**
Ours (None, checkerboard)	3.1%	34.5%	32.6%	14.0%	50.5%	39.3%	26.3%	2.3%	**25.3%**
Ours (FT, checkerboard)	5.5%	31.3%	26.7%	8.3%	39.2%	32.2%	19.5%	3.7%	**20.8%**
Ours (SVD, checkerboard)	5.3%	32.4%	30.8%	12.0%	43.6%	33.0%	20.9%	3.7%	**22.7%**

performance might decrease if the test images do not have the same input size as the trained UAP. To mitigate this problem, we propose to augment the UAP by resizing the UAP during the training stage. A commonly chosen input size on VGG16 on ImageNet is 224×224. During training, we randomly resize the UAP to a size ranging from 112 to 336. We find that resizing the UAP during the training reduces the fooling ratio for the original input size by a small margin; however, it achieves superior performance for other input sizes (see Table 7). Moreover, it also improves transferability across models (see Table 8).

Table 10. Ablation study on different patterns: uniform noise pattern (UNP), horizontal pattern (HP), and the vertical pattern (VP). Average accuracy is reported over the same models as in Table 9.

None			FT			SVD		
UNP	HP	VP	UNP	HP	VP	UNP	HP	VP
67.3%	35.9%	32.3%	48.1%	33.1%	28.4%	51.2%	30.1%	26.7%

6. Practical Data-free Black-Box Attack

In general, adversarial attack methods can be divided into two threat models: white-box, black-box. Black-box attacks [44, 6, 4] are more practical since it only requires only forward queries but is resource-intensive. To this end, another transfer-based variant of black-box generates the adversarial example on a substitute model [9, 10, 59, 14], for which the original training dataset is necessary. Towards a practical attack, one recent work [31], has recently investigated the possibility of conducting an effective transfer-based black-box attack with a very small number of images, which is denoted as a no-box attack. Due to the no-query limitation, their approach still exploits the transferability by training a substitute model. They have investigated a total of 7 methods, among which Prototypical* trains one encoder together with 20 decoders is their best model. Their Prototypical* consists of three stages: (1) collecting a small number of images; (2) training a substitute model; (3) white-box I-FGSM attack on the substitute model to craft transferable adversarial examples. For a detailed description of their various methods, we refer the readers to [31]. Taking the task challenge into account, their well-engineered approach has achieved reasonable performance, even outperforming another baseline "Beyonder" that utilizes the whole training dataset for training a substitute model. However, their image-specific approach requires repeating the above three steps for attacking another image from a new class. To this end, we investigate an alternative approach by applying a universal attack.

Our universal attack approach is optimization-free, requiring none of the above three steps in [31]. We apply a universal adversarial pattern that can be directly added to any image. Specifically, we adopt three common patterns: horizontal repetitive lines, and vertical repetitive lines, checkerboard patterns. Additionally, we also experiment with removing the high-frequency content in the original images, which can bring additional performance gain. Note that the finally added perturbation, *i.e.* change to the original images, is clipped with the strict constraint of $l_\infty = \epsilon$ for a fair comparison with [31]. Due to the optimization-free nature, the attack success rate might be low. Surprisingly, we find that our simple, data-free, model-free, and optimization-free approach even outperforms the best well-engineered model [31] by a non-trivial margin (see Table 9). Simply adding a checkerboard pattern already reduces the average accuracy to 25.3%, outperforming the best model (31.2%) in [31]. Together with Table 10, we observe that checkerboard performs better than uniform noise, horizontal repetitive lines, and vertical repetitive lines. Removing the high-frequency content with either FT or SVD is beneficial for further boosting performance.

Defense Models. We also test the proposed optimization-

Table 11. Comparison of the error rate (%) after attack against defense methods between our approach and RHP [32]. Attacked network is Inception v3 and $\epsilon = 16/255$. The error rates of RHP are calculated based on the values provided in Table 1 and Table of [32].

Method	TVM	HGD	R&P
RHP [32]	70.4	45.4	43.2
Ours	72.8	53.4	57.1

free attack against three defense methods: TVM [19], HGD [33], and R&P [58]. Following [32], we set the ϵ to 16/255. For evading the defense more effectively, we set the checkerboard's square size to 16×16 in this setup. Table 11 shows that our optimization-free approach outperforms [32] by a visible margin.

Object Detection and Semantic Segmentation. Our simple optimization-free attack is also found to be effective against *object detection* and *semantic segmentation*. The results for object detection are demonstrated quantitatively in Table 12 and qualitatively in Figure 5. The results show that the detection performance significantly decreases. We also show the efficacy of our attack on the segmentation task. From Table 13, we can observe a significant drop in mean intersection over union (mIoU) metric indicating the success of the attack. The qualitative examples shown in Figure 6 demonstrates that the predicted segmentation labels are severely damaged under our optimization-free attack.

Table 12. Performance of our attack in object detection task.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN w/ FPN [48]	37.0%	58.5%	39.8%	21.1%	40.3%	48.2%
Faster R-CNN w/ FPN [48] + Attack	18.3%	31.7%	18.7%	8.7%	20.2%	25.4%
Mask R-CNN w/ FPN [21]	37.9%	59.2%	41.1%	21.5%	41.4%	49.3%
Mask R-CNN w/ FPN [21] + Attack	18.0%	30.7%	18.3%	9.2%	20.3%	23.5%

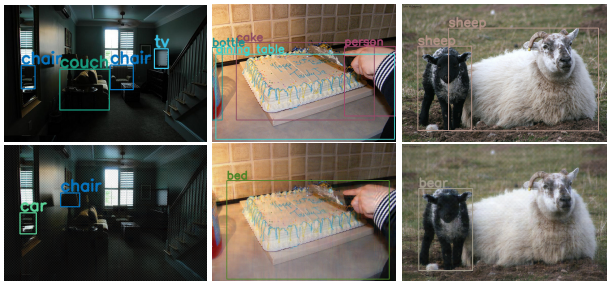


Figure 5. Examples of our attack for the object detection task. The first row shows a detection result for clean images, while detection results under our attack are in the second row. Our simple attack fools the detection network successfully.

7. Conclusion

Our work analyzes the mechanism behind an intriguing dominant label phenomenon caused by the untargeted UAP.

Table 13. Performance of our attack in segmentation task.

Method	mIoU
FCN ResNet101 [35]	63.7
FCN ResNet101 [35] + Attack	26.9
DeepLabV3 ResNet101 [5]	67.4
DeepLabV3 ResNet101 [5] + Attack	20.3

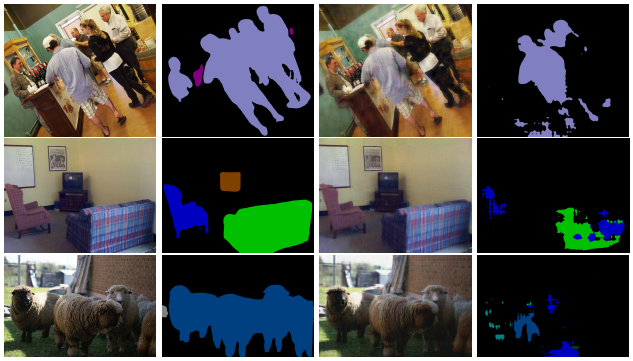


Figure 6. Examples of our attack in semantic segmentation task. From left to right: the first column is the clean images, the second column is the respective segmentation maps, the third column is the attacked images, and the final column is their segmentation maps. Our perturbation significantly worsens the performance quality of the segmentation model (2nd column vs 4th column).

Based on some reasonable assumptions, the existing explanation for this phenomenon is not compatible with some observed phenomena, including the dominant label changing with the class of training samples. Our work provides an alternative explanation with the observation untargeted UAP has a dominant contribution to the model response of adversarial examples. Our analysis motivates us to investigate untargeted UAP towards a more practical attack under the data-free constraint. We adopt a new loss to alleviate the need for ground-truth labels, simple yet effective augmentation techniques to reduce sample size, and proxy jigsaw images for crafting strictly data-free UAP. Under the data-free constraint, our investigation in the black-box setting shows that an optimization-free simple repetitive pattern like checkerboard is sufficient enough for being a strong attack. Given such success, one interesting direction of future work is to explore more effective patterns.

Acknowledgement

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korea Government (MSIT) (Artificial Intelligence Innovation Hub) under Grant 2021-0-02068.

References

[1] Sajjad Abdoli, Luiz G Hafemann, Jerome Rony, Ismail Ben Ayed, Patrick Cardinal, and Alessandro L Koerich. Universal adversarial audio perturbations. *arXiv preprint arXiv:1908.03173*, 2019. 2

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018. 2

[3] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *SP*, 2017. 1, 2

[4] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *ieee symposium on security and privacy (sp)*, 2020. 7

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 8

[6] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *ACM workshop on artificial intelligence and security*, 2017. 7

[7] Kenneth T Co, Luis Muñoz-González, Leslie Kanthan, Ben Glocker, and Emil C Lupu. Universal adversarial perturbations to understand robustness of texture vs. shape-biased training. *arXiv preprint arXiv:1911.10364*, 2019. 5

[8] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019. 2

[9] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, 2018. 7

[10] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *CVPR*, 2019. 7

[11] Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. Adversarial vulnerability for any classifier. In *NeurIPS*, 2018. 2

[12] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *NeurIPS*, 2016. 2

[13] Hang Gao and Tim Oates. Universal adversarial perturbation for text classification. *arXiv preprint arXiv:1910.04618*, 2019. 2

[14] Lianli Gao, Qilong Zhang, Jingkuan Song, Xianglong Liu, and Heng Tao Shen. Patch-wise attack for fooling deep neural network. In *ECCV*, pages 307–322. Springer, 2020. 7

[15] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019. 5

[16] Justin Gilmer, Nicolas Ford, Nicholas Carlini, and Ekin Cubuk. Adversarial examples are a natural consequence of test error in noise. In *ICML*, 2019. 2

[17] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018. 2

[18] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 1, 2

[19] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017. 8

[20] Jamie Hayes and George Danezis. Learning universal adversarial perturbations with generative models. In *IEEE Security and Privacy Workshops (SPW)*, 2018. 3

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 8

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *T-PAMI*, 2015. 6

[23] Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer. Universal adversarial perturbations against semantic image segmentation. In *ICCV*, 2017. 2

[24] Hokuto Hirano, Akinori Minagi, and Kazuhiro Takemoto. Universal adversarial attacks on deep neural networks for medical image classification. *BMC medical imaging*, 2021. 2

[25] Saumya Jetley, Nicholas Lord, and Philip Torr. With friends like these, who needs adversaries? In *NeurIPS*, 2018. 1, 2

[26] Valentin Khrulkov and Ivan Oseledets. Art of singular vectors and universal adversarial perturbations. In *CVPR*, 2018. 2, 5, 6

[27] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *ICLR2017 workshop*, 2016. 2

[28] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 2015. 1

[29] Jie Li, Rongrong Ji, Hong Liu, Xiaopeng Hong, Yue Gao, and Qi Tian. Universal perturbation attack against image retrieval. In *ICCV*, 2019. 2

[30] Jiguo Li, Xinfeng Zhang, Chuanmin Jia, Jizheng Xu, Li Zhang, Yue Wang, Siwei Ma, and Wen Gao. Universal adversarial perturbations generative network for speaker recognition. In *ICME*, 2020. 2

[31] Qizhang Li, Yiwen Guo, and Hao Chen. Practical no-box adversarial attacks against dnns. *NeurIPS*, 2020. 2, 7

[32] Yingwei Li, Song Bai, Cihang Xie, Zhenyu Liao, Xiaohui Shen, and Alan L Yuille. Regional homogeneity: Towards learning transferable universal adversarial perturbations against defenses. In *ECCV*, 2020. 2, 8

[33] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *CVPR*, 2018. 8

[34] Hong Liu, Rongrong Ji, Jie Li, Baochang Zhang, Yue Gao, Yongjian Wu, and Feiyue Huang. Universal adversarial perturbation via prior driven uncertainty approximation. In *ICCV*, 2019. 2, 5, 6

[35] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 8

[36] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. 1, 2

[37] Saeed Mahloujifar, Dimitrios I Diochnos, and Mohammad Mahmoody. The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure. In *AAAI*, 2019. 2

[38] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *CVPR*, 2017. 1, 2, 3, 4, 5, 6

[39] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Analysis of universal adversarial perturbations. *arXiv preprint arXiv:1705.09554*, 2017. 2

[40] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, 2016. 2, 3

[41] Konda Reddy Mopuri, Aditya Ganeshan, and Venkatesh Babu Radhakrishnan. Generalizable data-free objective for crafting universal adversarial perturbations. *TPAMI*, 2018. 2, 4, 5, 6

[42] Konda Reddy Mopuri, Utsav Garg, and R. Venkatesh Babu. Fast feature fool: A data independent approach to universal adversarial perturbations. In *BMVC*, 2017. 2, 5, 6

[43] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R. Venkatesh Babu. Nag: Network for adversary generation. In *CVPR*, 2018. 1, 3

[44] Nina Narodytska and Shiva Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *CVPRW*, 2017. 7

[45] Paarth Neekhara, Shehzeen Hussain, Prakhar Pandey, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. Universal adversarial perturbations for speech recognition systems. *arXiv preprint arXiv:1905.03828*, 2019. 2

[46] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *CVPR*, 2018. 2, 3

[47] Konda Reddy Mopuri, Phani Krishna Uppala, and R Venkatesh Babu. Ask, acquire, and attack: Data-free uap generation using class impressions. In *ECCV*, 2018. 2, 5, 6

[48] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 8

[49] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *NeurIPS*, 2018. 2

[50] Ali Shafahi, W. Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? In *ICLR*, 2019. 2

[51] Ali Shafahi, Mahyar Najibi, Zheng Xu, John P Dickerson, Larry S Davis, and Tom Goldstein. Universal adversarial training. In *AAAI*, 2020. 3

[52] 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. 1, 2

[53] Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *IJCNN*, 2016. 2

[54] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016. 2

[55] Jon Vadillo and Roberto Santana. Universal adversarial examples in speech command classification. *arXiv preprint arXiv:1911.10182*, 2019. 2

[56] Jon Vadillo, Roberto Santana, and Jose A Lozano. Analysis of dominant classes in universal adversarial perturbations. *arXiv preprint arXiv:2012.14352*, 2020. 2

[57] Haohan Wang, Xindi Wu, Pengcheng Yin, and Eric P Xing. High frequency component helps explain the generalization of convolutional neural networks. In *CVPR*, 2020. 5

[58] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *ICLR*, 2018. 8

[59] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *CVPR*, 2019. 7

[60] Chaoning Zhang, Philipp Benz, Tooba Imtiaz, and In-So Kweon. Understanding adversarial examples from the mutual influence of images and perturbations. In *CVPR*, 2020. 1, 2, 3, 5

[61] Chaoning Zhang, Philipp Benz, Adil Karjauv, and In So Kweon. Universal adversarial perturbations through the lens of deep steganography: Towards a fourier perspective. *AAAI*, 2021. 4, 5

[62] Chaoning Zhang, Philipp Benz, Chenguo Lin, Adil Karjauv, Jing Wu, and In So Kweon. A survey on universal adversarial attack. *IJCAI*, 2021. 2, 3