Learning Universal Adversarial Perturbation by Adversarial Example

Maosen Li[1], **Yanhua Yang**[1*], **Kun Wei**[1], **Xu Yang**[1], **Heng Huang**[2]

[1]Xidian University, Xi'an 710071, China
[2]Department of Electrical and Computer Engineering, University of Pittsburgh, PA 15260, USA
{maosenli95, yangyanhua.xd, weikunsk, xuyang.xd, henghuanghh}@gmail.com

Abstract

Deep learning models have shown to be susceptible to universal adversarial perturbation (UAP), which has aroused wide concerns in the community. Compared with the conventional adversarial attacks that generate adversarial samples at the instance level, UAP can fool the target model for different instances with only a single perturbation, enabling us to evaluate the robustness of the model from a more effective and accurate perspective. The existing universal attack methods fail to exploit the differences and connections between the instance and universal levels to produce dominant perturbations. To address this challenge, we propose a new universal attack method that unifies instance-specific and universal attacks from a feature perspective to generate a more dominant UAP. Specifically, we reformulate the UAP generation task as a minimax optimization problem and then utilize the instance-specific attack method to solve the minimization problem thereby obtaining better training data for generating UAP. At the same time, we also introduce a consistency regularizer to explore the relationship between training data, thus further improving the dominance of the generated UAP. Furthermore, our method is generic with no additional assumptions about the training data and hence can be applied to both data-dependent supervised and data-independent unsupervised manners. Extensive experiments demonstrate that the proposed method improves the performance by a significant margin over the existing methods in both data-dependent and data-independent settings. Code is is available at https://github.com/lisenxd/AT-UAP.

Introduction

Deep neural networks (DNNs) have achieved remarkable performance in various computer vision tasks, however, recent research works have demonstrated that the existing DNNs are vulnerable to adversarial examples (Wang et al. 2021; Yang et al. 2018, 2020). In general, given a DNN $f(x) : x \in \mathcal{X} \longrightarrow y \in \mathcal{Y}$, which maps input x to label y, adversarial attack is designed to seek an adversarial example x^*, leading to misclassification (Li et al. 2019; Wang 2021). The generation of adversarial examples can be formulated as the following constrained optimization problem:

$$f(x^*) \neq f(x), \quad s.t. \left\|x^* - x\right\|_p \leq \epsilon \,. \tag{1}$$

The perturbation is restricted by ℓ_p-norm $\|\cdot\|_p$ to ensure adversarial examples to be imperceptible for human. The first batch of efforts on adversarial attack have shown that the adversarial example for a given clean example x can be efficiently crafted by performing gradient updates (Papernot et al. 2016; Carlini and Wagner 2017). These approaches, which generate adversarial samples based on a specific instance but often fail to transfer across samples, are categorized as instance-specific adversarial attacks.

Several other parallel studies have shown that the principally curved directions of a deep classifier are also aligned among data points (Moosavi-Dezfooli et al. 2018; Fawzi et al. 2018), which gives rise to the existence of instance-agnostic approach to craft adversarial perturbations. Different from the instance-specific approach, instance-agnostic approach learns the universal adversarial perturbation (UAP) on the data distribution independent of a specific instance, that is, the generated UAP is sufficient to fool most instances drawn from the corresponding data distribution. The UAP approaches are not only effective, but also adaptable for a variety of tasks, such as image segmentation (Hendrik Metzen et al. 2017), natural language processing (Wallace et al. 2019), and automatic check-out (Liu et al. 2020). In addition to conventional data-dependent supervised methods which rely heavily on the quality and quantity of the training data (Khrulkov and Oseledets 2018; Wiyatno et al. 2019), recently, the success of data-independent unsupervised UAP further strengthens the practicality of UAP by alleviating data dependency. Compared to instance-specific methods, data-independent approaches focus on the real vulnerability of the learned model and the lower bound is also reported on the achievable fooling ratios, which provides meaningful theoretical guidance (Mopuri, Ganeshan, and Babu 2018).

In essence, both instance-specific perturbation and UAP are generated features rather than 'bugs' (Ilyas et al. 2019). Although both can fool target models, they differ in that, as a feature, the instance-specific perturbation is 'non-robust' and less dominant, whereas the UAP is 'robust' and more dominant (Zhang et al. 2020). However, most of existing instance-agnostic approaches directly use original data to train UAP, which are often only partially available and cannot guarantee the truly universal, such that the generated UAP contains undesired non-robust features. To this end, the vital issue of generating UAP lies in how to reduce non-

[*] Correspond author.

robust features in UAP and increase its dominance.

Motivated by the above discussions, we propose a new instance-agnostic approach via integrating the advantages of both instance-specific and instance-agnostic attacks. The intuition of our new method is that the non-robust features are brittle and can be destroyed easily by the antagonistic non-robust features (Ilyas et al. 2019), while the robust features are not. And if the generated UAP is robust and dominant, it is reasonable to assume it is resistant to the instance-specific perturbation. We therefore formulate the UAP generation process as a minimax optimization problem. Specifically, in the minimization problem, we adopt the instance-specific approach to craft adversarial examples with more antagonistic non-robust features. Then, the UAP is formulated to maximize the objective with these adversarial examples as inputs. In this process, the instance-specific adversarial examples are cheap, predictive, and flexible (Goodfellow, Shlens, and Szegedy 2015; Ortiz-Jiménez et al. 2020). Therefore, we can even leverage instance-specific adversarial examples to approximate the real data distribution with random noise as initialization in the data-independent setting. Moreover, we design a consistency regularizer by introducing the logarithm normalized volume term to further improve the dominance of the generated UAP. For efficient optimization, we also follow the insight of curriculum learning, in which we first start out with the easy examples and then gradually increase the difficulty of tasks. Our main contributions can be summarized as follows:

1. We investigate and reveal the relationships between instance-specific perturbation and UAP, and formulate the UAP generation task as a minimax optimization problem. By utilizing instance-specific adversarial examples as adversaries, we solve the minimax problem effectively and efficiently.
2. We devise a consistency regularizer to further improve the dominance of generated UAP by exploiting the interactions among samples.
3. Without any real data, the UAPs learned by our method are of a strong semantic pattern, which could be instructive to help the understanding of adversarial examples.
4. We conduct numerical experiments on large-scale ImageNet dataset with five well-known DNN models to confirm the effectiveness of our method in both data-dependent and data-independent settings.

Related Work

In the following, we will provide a brief overview of the instance-specific and instance-agnostic attack methods.

Instance-specific Attacks: The vulnerability of DNNs to imperceptible perturbation is first observed in (Szegedy et al. 2014), which transforms the box-constrained problem into Lagrangian function and is solved by L-BFGS algorithm. Late on, a series of methods have been proposed to generate the adversarial example. DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard 2016) studies adversarial examples from decision boundary perspective, and shows that almost all samples are very close to their decision boundary. C&W attack (Carlini and Wagner 2017) explores the space of loss functions and breaks many defense strategies. Based on linear hypothesis, Fast Gradient Sign Method (FGSM) (Goodfellow, Shlens, and Szegedy 2015) is proposed to find the adversarial example simply and quickly. Compared with L-BFGS and C&W, FGSM conducts back-propagation only once and then adds a small perturbation along the gradient sign direction to generate adversarial example x^*:

$$x^* = x + \epsilon \cdot \mathrm{sign}(\nabla_x \mathcal{L}(x, y)), \tag{2}$$

where $\nabla_x \mathcal{L}$ is the gradient of the loss function with respect to x, and ϵ is the constant to ensure the perturbation in the ℓ_∞-norm bound. Although much faster than L-BFGS and C&W, it is less effective in terms of fooling ratio (Kurakin, Goodfellow, and Bengio 2016). To address this problem, an iterative extension of FGSM termed I-FGSM (Kurakin, Goodfellow, and Bengio 2017) is proposed. Following I-FGSM, MI-FGSM (Dong et al. 2018) adopts momentum term in iterative algorithms to boost adversarial attack, which achieves a balance between effectiveness and speed, and has been proven to have good transferability across DNN architecture.

Different form previous works, in this paper, we utilize instance-specific attacks from 'non-robust' feature perspective rather than adversarial attack perspective.

Instance-agnostic Attacks: Contrary to instance-specific approach, instance-agnostic approach aims to fool all instance by adding only a single learned universal noise. Now there are two basic categories in instance-agnostic approach: data-dependent and data-independent. As a typically data-dependent method, the vanilla universal attack (V-UAP) accumulates UAP by iteratively executing DeepFool attack (Moosavi-Dezfooli, Fawzi, and Frossard 2016) for each data points. Another algorithm named SV-UAP generates UAP by calculating the singular vectors of the Jacobian matrices of the feature maps (Khrulkov and Oseledets 2018). To alleviate the rather cumbersome and slow procedure of the UAP generation, generative methods are proposed, *i.e.*, NAG (Mopuri et al. 2018) and GAP (Poursaeed et al. 2018). Recently, Feature-UAP (F-UAP) demonstrates UAP has independent semantic feature and a direct optimization can achieve superior performance than training a generative network (Zhang et al. 2020).

As it is not reasonable to assume adversaries have access to original training data, data-independent approach assumes we only have access to the models architecture and parameters, without knowing the training data. Fast Feature Fool (FFF) is believed as the first data-independent method which generates UAP by maximizing the activation of convolutional neurons (Mopuri, Garg, and Venkatesh Babu 2017). Similarly, PD-UA generates UAP by maximizing the model uncertainty (Liu et al. 2019). Different from the above methods, GD-UAP (Mopuri, Ganeshan, and Babu 2018) and AAA (Mopuri, Uppala, and Babu 2018) craft for data-independent UAP based on random noise and compress impressions as proxy data, respectively. Though more effective, AAA requires costly training of multiple compressed impressions for each class and also causes the generation process to be divided into two stages.

Different from the methods mentioned above, we introduce instance-specific attack into instance-agnostic task, which bridges the gap between two adversarial attack tasks.

Feature perspective on adversarial example: The existence of adversarial perturbation is counter-intuitive which motivates numerous works attempting to explain this intriguing phenomenon from a wide spectrum of perspectives, *i.e.,* noise disturbance (Fawzi, Moosavi-Dezfooli, and Frossard 2016; Gilmer et al. 2019), data imitation (Schmidt et al. 2018; Tanay and Griffin 2016), high-dimensionality (Fawzi, Fawzi, and Fawzi 2018; Mahloujifar, Diochnos, and Mahmoody 2019) and local linearity (Goodfellow, Shlens, and Szegedy 2015; Qin et al. 2019). These theories, however, are often limited in explaining this phenomenon as they only focus on explaining one of instance-specific and instance-agnostic adversarial examples.

Recently, feature perspective has been proposed to explain the existence of adversarial perturbation which claims that adversarial vulnerability is a direct result of our models sensitivity to well-generalizing features in the data (Ilyas et al. 2019; Ortiz-Jiménez et al. 2020). In standard ML datasets, there exist both 'robust' features and non-robust features, and DNNs exploit both human-aligned 'robust' features and human-imperceptible 'non-robust' features, as long as they are predictive. Thus, instance-specific perturbations establish spurious input-output associations based purely on 'non-robust' features, while UAP is based on 'robust' features (Liu et al. 2019; Zhang et al. 2020). This key difference leads to UAP dominates over the inputs for the model prediction, whereas instance-specific perturbation does not (See Fig. 2). However, limited by the number of training samples observed, UAPs learned directly from the datasets are not truly universal and still contain non-robust features. This phenomenon is particularly serious in the data-independent setting due to the absence of data.

Given all this, we introduce instance-specific perturbations as adversaries to destroy brittle non-robust features during the UAP generation, resulting in more robust UAP.

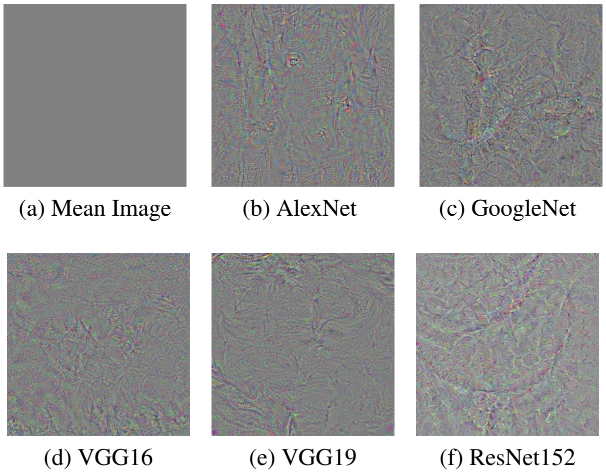
Methodology

In this paper, we reveal the relationship between minimax optimization and UAP attack firstly. Then we illustrate how to apply the instance-specific attack approach to solve this problem. Finally, consistency regularizer is introduced to normalize the response of UAP.

Problem Formulation

In universal adversarial attack, we try to find a single perturbation δ satisfying:

$$\begin{aligned} &\max_{\delta} \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{D}} \left[\mathcal{L}(f(x+\delta),y)\right], \\ &s.t. \quad \|\delta\|_{p_1} \le \epsilon \end{aligned} \tag{3}$$


(a) Mean Image (b) AlexNet (c) GoogleNet

(d) VGG16 (e) VGG19 (f) ResNet152

Figure 1: The visualization of adversarial examples generated for different network architectures. Rescaled to $[0, 255]$.

where $\mathcal{L}(\cdot)$ denotes the perturbation loss over the data x and f is target network. Most of existing methods focus on how to solve Eq. 3 more efficiently while ignoring how limited data effects the optimization process, which results in the generated UAP less robust and dominant. To obtain better training data for generating UAP, we introduce instance-specific attacks to generate training data with more antagonistic non-robust features. So we rewrite the objective as:

$$\begin{aligned} &\arg\max_{\delta} \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{D}} \left[\min_{x^*\sim\Delta(x)} \mathcal{L}(f(x^*,\delta),y)\right], \\ &s.t. \quad \|\delta\|_{p_1} \le \epsilon_1 \quad \|\Delta(\cdot)\|_{p_2} \le \epsilon_2 \end{aligned} \tag{4}$$

where x^* is adversarial example near the initial data x to minimize the loss associated with the corresponding label y. The UAP δ is going to be against x^* by maximizing the loss. Compared with Eq. 3, the key difference is that our object function replaces real data x with adversarial example x^* as training data of δ. Note that the inner problem minimizes individual loss and acts as an adversary for the outer problem. Next, we will discuss how to solve Eq. 4 in detail.

Adversarial Example Generation. Directly generating adversarial example is expensive and difficult, in order to solve it at an acceptable cost, we resort to MI-FGSM method. The MI-FGSM is computation-friendly, and it only takes a few steps rather than hundreds or even thousands of steps to generate an adversarial example. Besides, the adoption of momentum term overcomes poor local optima and can achieve nearly 100% attack success rate. However, the vanilla MI-FGSM also has some problems. First, in the case of non-targeted attacks, the dataset has to be the original dataset. Second, ℓ_∞ constraint focuses only on maximum value, leading to all pixels in perturbation lying in $[-\epsilon_2, +\epsilon_2]$, far from the dynamic range of natural image $[0, 255]$. To address these problems, we use ℓ_2-norm targeted variant of MI-FGSM instead of vanilla one, the function is:

$$g_{t+1} = \mu \cdot g_t - \frac{\nabla_x \mathcal{L}(x_t^*, y)}{\|\nabla_x \mathcal{L}(x_t^*, y)\|_2}, \tag{5}$$

$$x_{t+1}^* = Clip(x_t^* - \alpha \cdot \frac{g_{t+1}}{\|g_{t+1}\|_2}), \tag{6}$$

where $Clip(\cdot)$ is clip function to ensure generated x^* within $[0, 255]$. Following default settings in (Dong et al. 2018), the

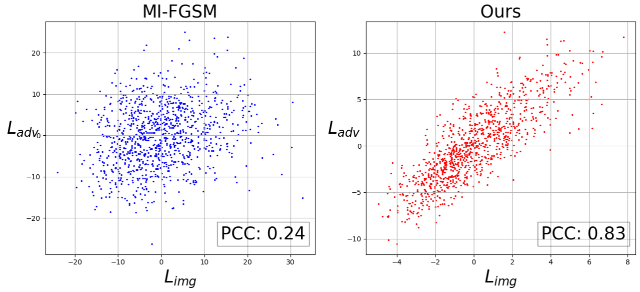


Figure 2: Plots of logit vectors from the perturbed image L_{img} and scaled crafted perturbation L_{adv} of MIM and proposed method with their respective PCC values. The perturbations are crafted on VGG16.

decay factor μ is set to 1, total iteration is set to 10 and step size α is set to ${}^{\epsilon_2}/_{10}$. For data-independent setting, we initialize x^* via a mean image (all pixels are set as 127.5 out of 255) and add uniform distribution $\mathcal{U} \in [-16, 16]$ in the beginning to keep diversity. While for data-dependent setting, x^* starts from real data x and y^* is set to true label y. The difference between our variant and vanilla MI-FGSM is that in Eq. 5 and Eq. 6, we rescale the gradient as a unit vector to constraint the magnitude of perturbation with ℓ_2-ball. And the targeted setting makes x^* look like random label y^* for target model, which is crucial for data-independent setting.

Here we set $\epsilon_2 = 20$ and the generated adversarial examples are as shown in Fig. 1. Though target model gives a high confident, these instance-specific adversarial examples do not make any sense in human cognition. From feature perspective, the generated adversarial examples are semantically meaningless but highly predictable. Therefore, it is reasonable to assume they are typical non-robust features.

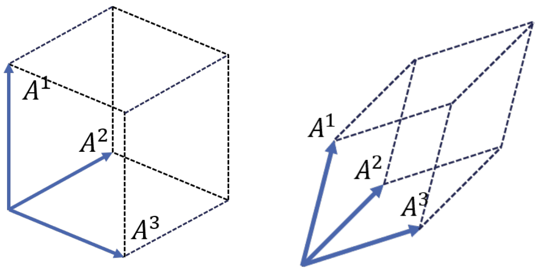
UAP Generation After solving the inner minimization problem, we leverage x^* to train UAP. Specially, we treat UAP δ as unknown weight and apply gradient decent method to optimize it. We adopt the ADAM optimizer with mini-batch training, which is easy to use and has also been adopted in the context of generating UAPs (Mopuri, Ganeshan, and Babu 2018; Zhang et al. 2020). The final generation scheme is defined as follows:

$$\delta_t = Adam(\nabla_\delta \mathcal{L}(x^*, \delta_{t-1}), \gamma), \tag{7}$$

$$\delta_t = \min(\max(\delta_t, -\epsilon_1), \epsilon_1), \tag{8}$$

where γ is the learning rate and min-max operation constrains $\delta \in [-\epsilon_1, \epsilon_1]$.

With the instance-specific adversarial examples as training data, the UAP is designed to learn more dominant pattern that can defeat antagonistic non-robust features. To verify our claim, we plot the Pearson Correlation Coefficient (PCC) analysis, a widely adopted metric to measure the linear correlation between two variables (Anderson 1962). As shown in Fig. 2, our generated UAP is highly correlated with adversarial image while the perturbation generated by MI-FGSM is not, which is consistent with previous claim.



(a) Before

(b) After

Figure 3: Illustration of the consistency regularizer. (a): The logit vector directions are diverse. (b): After the regularization, the logit vector directions are more consistency.

Loss Design

Fooling Loss In the design of the objective function, both the requirements of adversarial example and UAP generation should be considered. A most naively and commonly used loss is cross-entropy. However, cross-entropy loss has been pointed out to be less effective in both targeted instance-specific and UAP scenario (Naseer et al. 2019; Li et al. 2020) because of the following reasons: (a). In cross-entropy loss, the gradient value scale changes with label confidence, which sometimes even leads to gradient vanish; (b). Cross-entropy loss holistically incorporates logits of all classes but not the only target class we want, thus this loss function leads to overall lower fooling ratios. Drawbacks of cross-entropy loss can be resolved by applying logit loss:

$$\mathcal{L}_\ell(x, y) = -l_y(x), \tag{9}$$

where $l_y(x)$ denotes the logit output of input x with respect to the target label. It is also supported by recent works that logit loss has superior performance than cross-entropy loss (Zhao, Liu, and Larson 2020; Zhang et al. 2020).

Consistency Regularizer The logit loss $\mathcal{L}_\ell$ only concerns instance level information, but ignores a common feature shared by all instances in generation, which is UAP δ. Note that a well-trained UAP should be more robust and dominant than normal instances as we mentioned before. Thus, under the influence of UAP, the response of different instances with UAP should align well. Technically, we introduce logarithm normalized volume term (Pang et al. 2019) to measure the correlation between adversarial examples in order to promote consistency of feature responses. In general, our consistency regularizer is defined as:

$$\mathcal{L}_c = \log(\det(A^\intercal A)), \tag{10}$$

where $\det(\cdot)$ denotes determinant and $A = \{A^1, ..., A^m\} \in \mathbb{R}^{c \times m}$ is the output of logit layer. Here output A^i is the logit of i-th instance normalized under ℓ_2-norm. According to the matrix theory, we have:

$$\det(A^\intercal A) = \text{Vol}^2(\{A^i\}_{i \in [m]}), \tag{11}$$

where $\text{Vol}(\cdot)$ denotes the volume spanned by m vectors A^i. As shown in Fig. 3, since A^i is normalized, $\mathcal{L}_c$ achieves

Mthod	AlexNet	GoogleNet	VGG16	VGG19	ResNet152	Avg.
F-UAP*	96.17	88.94	94.30	94.98	90.08	92.89
FFF	80.92	56.44	47.10	43.62	-	-
GD-UAP	87.02	71.44	63.08	64.67	37.30	64.65
AAA	89.04	75.28	71.59	72.84	60.72	73.89
PD-UA	-	67.12	70.69	64.98	46.39	-
Ours(w/o CR)	96.29±0.17	80.08±0.84	93.67±0.24	92.08±0.43	71.26±0.91	86.68
Ours(with CR)	**96.66±0.12**	**82.60±0.72**	**94.50±0.21**	**92.85±0.48**	**73.15±1.15**	**87.95**

Table 1: The evaluation results (FR%) of the proposed method and other data-independent UAPs. To avoid cherry picking, we present mean and standard deviation obtained for 5 runs. 'w/o CR' indicates generating UAP without consistency regularizer and 'with CR' indicates generating UAP with consistency regularizer. * indicates data-dependent method.

Algorithm 1: Our UAP algorithm

```
Input: Target model $f$, mean image $x_m$, loss function
       $\mathcal{L}$, mini-batch size $m$, iterations $T$, magnitude
       $\epsilon_1,\ \epsilon_2$, constant $a_0$ and zoom factor $\alpha$.
Output: Perturbation vector $v$
Initialize: $\delta \leftarrow 0, x \leftarrow 255.0/2$
for Iteration $i = 0, \dots, T-1$ do
    Update the magnitude of instance-specific
      adversarial examples via Eq. 13.
    Sample $x_i \leftarrow x_m + \mathcal{U}(-16, 16)$ (data-independent)
      or $x_i \leftarrow \mathbb{D}(x)$ (data-dependent).
    Update the $x^*$ via Eq. 5 and Eq. 6.
    Update the $\delta$ via Eq. 7 and Eq. 8.
end
```

maximal value 0 if and only if column vectors of A mutually orthogonal. Note that different from most of the previous efforts to enforce the UAPs dissimilar on the same instance x, we minimize $\mathcal{L}_c$ to keep the responses of the instances containing UAP consistent. Overall, our loss for training UAP is given by:

$$\mathcal{L}(\cdot) = \mathcal{L}_\ell(\cdot) + \lambda\mathcal{L}_c(\cdot), \tag{12}$$

where λ is trade-off factor.

Optimization with Curricula

Since overmighty adversary in the beginning may make it hard to convergence, we optimize our method following the insight of curriculum learning, in which we first start out with only easy examples and then gradually increase the difficulty of task. Specifically, our definition of sample difficulty depends not on the label category but on the strength of adversarial examples, then we can gradually expand the scope and magnitude of adversarial examples directly by controlling ϵ_2. Our protocol of curriculum learning is defined as:

$$\begin{aligned} &D_1 \subset D_2 \subset ... \subset D_T, \\ &D_i = \{x_i^*, y\},\ x_i^* \in \Delta_i(x), \\ &s.t.\ \ \|\Delta_i(\cdot)\| \le Clip(a_0 + \alpha\tfrac{i}{T}(\epsilon_1 - a_0)), \end{aligned} \tag{13}$$

where T is total iteration, α is a zoom factor, and $Clip(\cdot)$ together with constant a_0 jointly ensure the constraint within $[a_0, \epsilon_1]$. We summarize the overall procedure of the proposed method in Algorithm 1.

Experiments

Datasets. Following (Mopuri, Ganeshan, and Babu 2018; Zhang et al. 2020), we evaluate the proposed method to fool a serial of DNNs pretrained on ImageNet, including AlexNet (Krizhevsky, Sutskever, and Hinton 2012), GoogleNet (Szegedy et al. 2015), VGG-16 (Simonyan and Zisserman 2015), VGG-19 (Simonyan and Zisserman 2015), and ResNet152 (He et al. 2016). We use the ImageNet validation set (Russakovsky et al. 2015) containing 50,000 samples to evaluate the performance. We also explore generating supervised data-dependent UAP with the ImageNet training data.

Evaluation metrics. To evaluate the attacking performance of the generated UAP, we use the widely-used fooling ratio (FR) metric (Moosavi-Dezfooli et al. 2017; Mopuri, Garg, and Venkatesh Babu 2017), that is the proportion of images that change labels when perturbed by our UAP.

Comparative Methods. The proposed method is compared with the following data-independent UAPs: FFF (Mopuri, Garg, and Venkatesh Babu 2017), GD-UAP (Mopuri, Ganeshan, and Babu 2018), AAA (Mopuri, Uppala, and Babu 2018), and PD-UA (Liu et al. 2019). We also compare data-independent UAPs: V-UAP (Khrulkov and Oseledets 2018), GAP (Poursaeed et al. 2018), NAG (Mopuri et al. 2018) and F-UAP (Zhang et al. 2020).

Implementation Details. All of our experiments are conducted on Pytorch and run with single NVIDIA TITAN Xp GPU. Following widely-used setting, we set $p_1 = \infty$ and maximum perturbation $\epsilon_1 = 10$ with the pixel value in $[0, 255]$ (Mopuri, Uppala, and Babu 2018; Zhang et al. 2020). The number of iterations T, batch-size m, learning rate γ and trade-off factor λ are set to 1000, 32, 0.5 and 0.05, respectively. ϵ_2, constant a_0 and zoom factor α are set to $20, 14, 20$ respectively for data-independent setting and in data-dependent setting, ϵ_2 is set to 4.

Quantitative Results

Data-independent Setting In this subsection, we firstly utilize our method to attack five DNN models without any real data, and then test the obtained UAP on ImageNet validation set. The fooling ratios of different methods are reported in Table 1. Note that F-UAP is SOTA data-dependent UAP method. As seen, our method consistently outperforms all other data-independent attacks by 7-20%. In most cases, our methods only slightly inferior to data-dependent

Mthod	AlexNet	GoogleNet	VGG16	VGG19	ResNet152	Avg.
V-UAP	93.3	78.9	78.3	77.8	84.0	82.46
SV-UAP	-	-	52.0	60.0	-	-
GAP	-	82.7	83.7	80.1	-	-
NAG	96.44	90.37	77.57	83.78	87.24	87.09
F-UAP	96.17	88.94	94.30	94.98	90.08	92.89
Ours(w/o CR)	96.59±0.17	90.35±0.26	96.94±0.16	97.11±0.04	90.78±0.29	94.35
Ours(with CR)	**97.01±0.11**	**90.82±0.29**	**97.51±0.08**	**97.56±0.04**	**91.52±0.78**	**94.88**

Table 2: The evaluation results (FR%) of the proposed method and other data-dependent UAPs. To avoid cherry picking, we present mean and standard deviation obtained for 5 runs. 'CR' indicates consistency regularizer

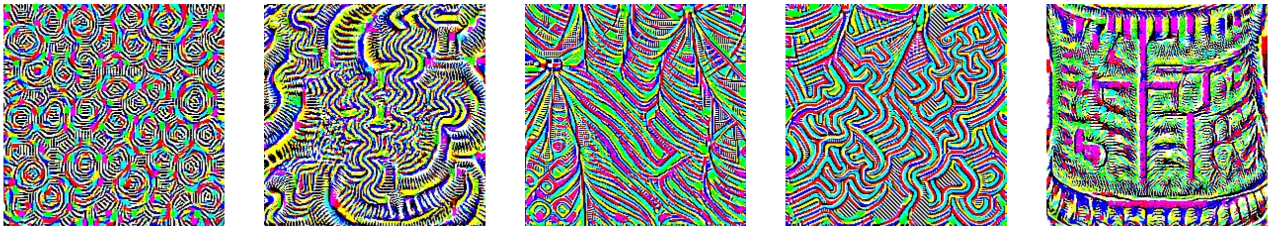

(a) AlexNet (b) GoogleNet (c) VGG16 (d) VGG19 (e) ResNet152

Figure 4: The visualization of unsupervised UAPs learned by the proposed framework for different network architectures. Rescaled to $[0, 255]$. (Best viewed in color.)

method. It is worth noting that in some DNN models, *i.e.,* AlexNet and VGG16, our method even outperforms data-dependent method without requiring any data. As shown in Fig. 4, our generated UAPs look more semantically meaningful whereas the perturbation from instance-specific perturbations performs like the noise (see Fig. 1 for reference), which suggests we manage to generate a more robust and dominant feature. And we also report the attacking performance of directly using logit loss without using consistency regularizer. Results show that consistency regularizer brings a performance gain of 0.4-2.9%, which confirms the effectiveness of consistency regularizer. Samples of clean images and perturbed images are shown in Figure 5.

Data-dependent Setting Similar to data-independent setting, in this subsection, we further validate the effectiveness of the proposed method with data-dependent setting. We set learning rate $\gamma = 0.005$ and $\epsilon_2 = 4$ here, as ϵ_2 is small, we do not use curriculum learning in this part. In this scenario, the instance-specific adversarial examples reinforce the non-robust features in data. If our learned UAP can defeat these examples with more non-robust features, it means that the UAP is more robust and dominant. The fooling ratios are reported in Table 2. With real data as initial, our method can further improve the fooling ratio and consistently outperforms comparison data-dependent method by 1.0-3.0%, which demonstrates the universality of proposed method.

Cross-model Generalizability

While we have shown the our generated UAP are universal across unseen data points, we now examine their cross-model generalizability. To quantitatively study the cross-model generalizability, we study to what extent the generated UAP for a given architecture (*e.g.*, AlexNet) is also valid for another architecture (*e.g.*, ResNet152). Table 3 shows the generalizability of the generated UAP across five different architectures. Here we denote data-dependent setting as 'supervised' and data-independent setting as 'unsupervised'. We observe that the generated UAP can achieve considerable FR in other architectures, which proves it can generalize well across both data points and architectures. This result is also consistent with previous claim that such UAP are of practical relevance and more robust. It is also worth noting that in some cases, data-independent approach shows better generalizability than data-independent approach, *i.e.,* AlexNet as source model. The cause of this phenomenon may be that the data dependence approach overfits the training data, which indicates an additional benefit of our data-independent method.

Ablation Study

In this subsection, we conduct a series of ablation experiments to study the effectiveness of our method.

We first validate the necessity of training UAP with instance-specific adversarial examples. We adopt mean image and Gaussian noise instead of adversarial example to craft UAP. For fair comparison, we rescale Gaussian noise with the same ℓ_2-norm of adversarial example. The results are presented in Table 4. In both scenarios, the performance of Gaussian noise is inferior to our method. We can also observe that in data-independent setting, the generated UAPs have little semantical patterns, indicating that the UAP is not able to learn semantically meaningful features with only Gaussian noise as proxy data. This phenomenon may be caused by the following reasons: 1. Mean-image or random noise has no correlation with training data at the feature level thus can not provide effective information for optimization. 2. These random data are lack of diversity, while targeted adversarial example are meaningful at the feature level and

	Mthod	AlexNet	GoogleNet	VGG16	VGG19	ResNet152
AlexNet	Unsupervised	96.66±0.12	62.01±0.32	72.33±0.50	67.24±0.18	43.63±0.29
	Supervised	97.01±0.11	47.31±1.65	62.37±1.37	57.72±0.62	33.40±0.77
GoogleNet	Unsupervised	55.65±0.37	82.60±0.72	71.38±0.83	68.25±0.59	43.03±0.42
	Supervised	55.90±0.62	90.82±0.29	78.71±0.67	76.01±0.45	54.49±0.29
VGG16	Unsupervised	54.15±0.70	48.53±1.32	94.50±0.21	86.65±0.70	36.96±1.03
	Supervised	45.58±0.29	53.63±0.90	97.51±0.08	91.53±0.22	47.16±0.95
VGG19	Unsupervised	62.05±1.01	60.99±1.41	88.96±0.50	92.85±0.48	42.72±0.51
	Supervised	46.04±0.58	52.58±0.81	93.49±0.17	97.56±0.04	43.53±0.57
ResNet152	Unsupervised	49.78±0.68	48.37±0.49	62.78±0.71	60.54±0.49	73.15±1.15
	Supervised	47.33±0.89	61.32±0.98	81.93±0.94	78.72±0.91	91.52±0.78

Table 3: Transferability results for the proposed universal adversarial attack. The rows indicate the source model and the columns indicate the target model. The values in each column are reported in the FR (%). To avoid cherry picking, we present mean and standard deviation obtained for 5 runs.

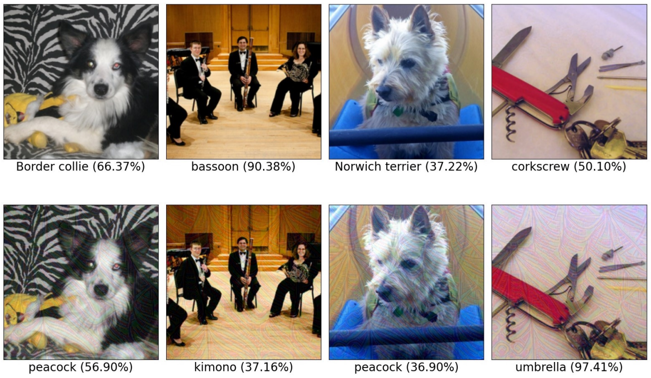
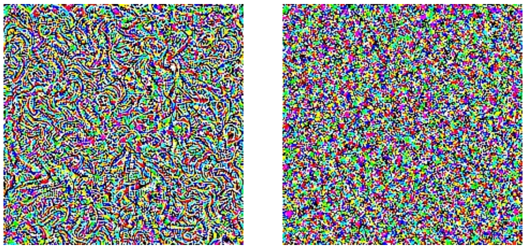


Figure 5: The visualization of clean images (top) and perturbed images (bottom) for VGG16 (Best viewed in color.)



(a) Mean

(b) Gaussian

Figure 6: The visualization of UAPs generated for VGG16 with mean images (left) and Gaussian noise images (right). Rescaled to $[0, 255]$. (Best viewed in color.)

Mthod	AlexNet	GoogleNet	VGG16	VGG19	ResNet152
Mean	72.07	35.83	51.14	40.49	26.14
Gaussian-I	56.80	24.27	30.56	27.75	19.47
Gaussian-D	96.26	89.13	94.49	95.35	90.46

Table 4: Our data-independent approach with mean image and Gaussian noise as proxy datasets for crafting UAP. We denote data-independent setting as ‘Gaussian-I’ and data-dependent setting as ‘Gaussian-D’.

Method	AlexNet	GoogleNet	VGG16	VGG19	ResNet152
Base	0.3742	0.0503	0.3978	0.4409	0.1394
Ours (w/o CR)	0.7408	0.4613	0.7666	0.6992	0.5582
Ours (with CR)	0.7423	0.4980	0.8123	0.7477	0.5974
Base	0.7304	0.5522	0.7433	0.7096	0.6285
Ours (w/o CR)	0.7597	0.5873	0.8032	0.8515	0.6337
Ours (with CR)	0.8041	0.6934	0.8429	0.8997	0.6508

Table 5: PCC analysis for 5 different network architectures. The results are divided into data-independent setting (upper) and data-dependent setting (lower). ‘Base’ indicates generating UAP with mean image (data-independent setting) or clean image (data-dependent setting).

have enough diversity (corresponding to target label).

Next, we evaluate the impact of each component on UAP dominance. As shown in Table 1 and Table 2, our method improves performance in both data-independent and data-dependent setting. However, it remains unknown whether the UAP we learned is more dominant. To verify our claim, we calculate PCC analysis to show the correlation between adversarial images and UAP. The absolute value of PCC analysis indicates the extent to which the two variables are linearly correlated, with 1 indicating perfect linear correlation, 0 indicating zero linear correlation. We use ImageNet validation set to perform experiments here and report average PCC values for 50000 images. The higher the average PCC value, the more dominant our UAP is. As shown in Table 5, we observe that the PCC values are relatively higher for UAPs with adversarial example and consistency regularizer, which supports our claim.

Conclusion

In this paper, we propose a novel universal attack method, which mainly considers the feature nature of UAP to craft a more dominant UAP. First, we build a bridge between instance-specific and universal attacks by minimax optimization, so that we can learn a more dominant UAP. Then to further improve the dominant of generated UAP, interactions among samples are utilized through a consistency regularizer, which can effectively improve the attacking performance. Benefiting from the bridge we bailed between instance-specific and universal attacks, we manage to learn a more dominant UAP. Extensive experiments verify the effectiveness of our method in both data-dependent as well as data-independent settings.

Ethics Statement

Due to the fact that DNN models have been widely deployed in real world applications, the potential privacy problems are also growing(Lyu et al. 2020; Xu and Lyu 2020, 2021). Stronger UAP can obviously benefit applications of adversarial images for social good, such as protecting user privacy. In addition, our paper can assist researchers to perform more thorough evaluations and designing stronger defenses. We firmly believe that the positives of our work outweigh the negatives.

Acknowledgements

Maosen Li, Yanhua Yang, Kun Wei and Xu Yang were partially supported by the National Natural Science Foundation of China under Grant 62132016, Grant 62171343, and Grant 62071361 in part by Key Research and Development Program of Shaanxi under Grant 2021ZDLGY01-03, and in part by the Fundamental Research Funds for the Central Universities ZDRC2102.

References

Anderson, T. W. 1962. An introduction to multivariate statistical analysis. Technical report, Wiley New York.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *S&P*, 39–57. IEEE.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *CVPR*, 9185–9193.

Fawzi, A.; Fawzi, H.; and Fawzi, O. 2018. Adversarial vulnerability for any classifier. In *NeurIPS*.

Fawzi, A.; Moosavi-Dezfooli, S.-M.; and Frossard, P. 2016. Robustness of classifiers: from adversarial to random noise. *arXiv preprint arXiv:1608.08967*.

Fawzi, A.; Moosavi-Dezfooli, S.-M.; Frossard, P.; and Soatto, S. 2018. Empirical study of the topology and geometry of deep networks. In *CVPR*, 3762–3770.

Gilmer, J.; Ford, N.; Carlini, N.; and Cubuk, E. 2019. Adversarial examples are a natural consequence of test error in noise. In *International Conference on Machine Learning*, 2280–2289. PMLR.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

Hendrik Metzen, J.; Chaithanya Kumar, M.; Brox, T.; and Fischer, V. 2017. Universal adversarial perturbations against semantic image segmentation. In *CVPR*, 2755–2764.

Ilyas, A.; Santurkar, S.; Tsipras, D.; Engstrom, L.; Tran, B.; and Madry, A. 2019. Adversarial Examples Are Not Bugs, They Are Features. *NeurIPS*, 32: 125–136.

Khrulkov, V.; and Oseledets, I. 2018. Art of singular vectors and universal adversarial perturbations. In *CVPR*, 8562–8570.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 25: 1097–1105.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2017. Adversarial machine learning at scale. In *ICLR*.

Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2016. ADVERSARIAL EXAMPLES IN THE PHYSICAL WORLD. *arXiv preprint arXiv:1607.02533*.

Li, C.; Gao, S.; Deng, C.; Xie, D.; and Liu, W. 2019. Cross-modal learning with adversarial samples. In *NeurIPS*.

Li, M.; Deng, C.; Li, T.; Yan, J.; Gao, X.; and Huang, H. 2020. Towards transferable targeted attack. In *CVPR*, 641–649.

Liu, A.; Wang, J.; Liu, X.; Cao, B.; Zhang, C.; and Yu, H. 2020. Bias-Based Universal Adversarial Patch Attack for Automatic Check-Out. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J.-M., eds., *ECCV*, 395–410. Cham: Springer International Publishing. ISBN 978-3-030-58601-0.

Liu, H.; Ji, R.; Li, J.; Zhang, B.; Gao, Y.; Wu, Y.; and Huang, F. 2019. Universal adversarial perturbation via prior driven uncertainty approximation. In *ICCV*, 2941–2949.

Lyu, L.; Yu, H.; Zhao, J.; and Yang, Q. 2020. Threats to Federated Learning. In *Federated Learning*, 3–16. Springer.

Mahloujifar, S.; Diochnos, D. I.; and Mahmoody, M. 2019. The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure. In *AAAI*, volume 33, 4536–4543.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; Fawzi, O.; and Frossard, P. 2017. Universal adversarial perturbations. In *CVPR*, 1765–1773.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; Fawzi, O.; Frossard, P.; and Soatto, S. 2018. Robustness of classifiers to universal perturbations: A geometric perspective. In *ICLR*.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, 2574–2582.

Mopuri, K.; Garg, U.; and Venkatesh Babu, R. 2017. Fast feature fool: A data independent approach to universal adversarial perturbations. In *BMVC*. BMVA Press.

Mopuri, K. R.; Ganeshan, A.; and Babu, R. V. 2018. Generalizable data-free objective for crafting universal adversarial perturbations. *IEEE transactions on pattern analysis and machine intelligence*, 41(10): 2452–2465.

Mopuri, K. R.; Ojha, U.; Garg, U.; and Babu, R. V. 2018. NAG: Network for adversary generation. In *CVPR*, 742–751.

Mopuri, K. R.; Uppala, P. K.; and Babu, R. V. 2018. Ask, acquire, and attack: Data-free uap generation using class impressions. In *ECCV*, 19–34.

Naseer, M. M.; Khan, S. H.; Khan, M. H.; Shahbaz Khan, F.; and Porikli, F. 2019. Cross-domain transferability of adversarial perturbations. *NeurIPS*, 32: 12905–12915.

Ortiz-Jiménez, G.; Modas, A.; Moosavi-Dezfooli, S.-M.; and Frossard, P. 2020. Optimism in the face of adversity: Understanding and improving deep learning through adversarial robustness. *arXiv preprint arXiv:2010.09624*.

Pang, T.; Xu, K.; Du, C.; Chen, N.; and Zhu, J. 2019. Improving adversarial robustness via promoting ensemble diversity. In *ICML*, 4970–4979. PMLR.

Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z. B.; and Swami, A. 2016. The limitations of deep learning in adversarial settings. In *EuroS&P*, 372–387. IEEE.

Poursaeed, O.; Katsman, I.; Gao, B.; and Belongie, S. 2018. Generative adversarial perturbations. In *CVPR*, 4422–4431.

Qin, C.; Martens, J.; Gowal, S.; Krishnan, D.; Dvijotham, K.; Fawzi, A.; De, S.; Stanforth, R.; and Kohli, P. 2019. Adversarial Robustness through Local Linearization. *NeurIPS*, 32: 13847–13856.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252.

Schmidt, L.; Santurkar, S.; Tsipras, D.; Talwar, K.; and Madry, A. 2018. Adversarially Robust Generalization Requires More Data. In *NeurIPS*.

Simonyan, K.; and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *CVPR*, 1–9.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR*.

Tanay, T.; and Griffin, L. 2016. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*.

Wallace, E.; Feng, S.; Kandpal, N.; Gardner, M.; and Singh, S. 2019. Universal Adversarial Triggers for Attacking and Analyzing NLP. In *EMNLP-IJCNLP*.

Wang, J. 2021. Adversarial examples in physical world. In *Proc. Workshop Track Int. Conf. Learn. Represent.(ICLR)*.

Wang, J.; Liu, A.; Yin, Z.; Liu, S.; Tang, S.; and Liu, X. 2021. Dual Attention Suppression Attack: Generate Adversarial Camouflage in Physical World. In *CVPR*, 8565–8574.

Wiyatno, R. R.; Xu, A.; Dia, O.; and de Berker, A. 2019. Adversarial examples in modern machine learning: A review. *arXiv preprint arXiv:1911.05268*.

Xu, X.; and Lyu, L. 2020. Towards building a robust and fair federated learning system. *arXiv preprint arXiv:2011.10464*.

Xu, X.; and Lyu, L. 2021. A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning. In *Proc. ICML Workshop on Federated Learning for User Privacy and Data Confidentiality*.

Yang, E.; Liu, T.; Deng, C.; and Tao, D. 2018. Adversarial examples for hamming space search. *IEEE transactions on cybernetics*.

Yang, X.; Deng, C.; Wei, K.; Yan, J.; and Liu, W. 2020. Adversarial learning for robust deep clustering. In *NeurIPS*, volume 33, 9098–9108.

Zhang, C.; Benz, P.; Imtiaz, T.; and Kweon, I. S. 2020. Understanding adversarial examples from the mutual influence of images and perturbations. In *CVPR*, 14521–14530.

Zhao, Z.; Liu, Z.; and Larson, M. 2020. On success and simplicity: A second look at transferable targeted attacks. *arXiv preprint arXiv:2012.11207*.