Understanding Adversarial Examples from the Mutual Influence of Images and Perturbations

Chaoning Zhang* chaoningzhang1990@gmail.com
Philipp Benz* pbenz@kaist.ac.kr
Tooba Imtiaz timtiaz@kaist.ac.kr
In-So Kweon iskweon@kaist.ac.kr
* indicates equal contribution
Robotics and Computer Vision (RCV) Laboratory
Korea Advanced Institute of Science and Technology (KAIST)
291 Daehak-ro, Yuseong-gu, Daejeon 34141, Korea

Abstract

A wide variety of works have explored the reason for the existence of adversarial examples, but there is no consensus on the explanation. We propose to treat the DNN logits as a vector for feature representation, and exploit them to analyze the mutual influence of two independent inputs based on the Pearson correlation coefficient (PCC). We utilize this vector representation to understand adversarial examples by disentangling the clean images and adversarial perturbations, and analyze their influence on each other. Our results suggest a new perspective towards the relationship between images and universal perturbations: Universal perturbations contain dominant features, and images behave like noise to them. This feature perspective leads to a new method for generating targeted universal adversarial perturbations using random source images. We are the first to achieve the challenging task of a targeted universal attack without utilizing original training data. Our approach using a proxy dataset achieves comparable performance to the state-of-the-art baselines which utilize the original training dataset.

1. Introduction

Deep neural networks (DNNs) have shown impressive performance in numerous applications, ranging from image classification [16, 48] to motion regression [8, 47]. However, DNNs are also known to be vulnerable to adversarial attacks [42, 37]. A wide variety of previous works [14, 43, 44, 21, 33, 3] explore the reason for the existence of adversarial examples, but there is a lack of consensus on the explanation [1]. While the working mechanism of DNNs is not fully understood, one widely accepted interpretation considers DNNs as feature extractors [16], which inspires the recent work [17] to link the existence of adversarial examples to non-robust features in the training dataset.

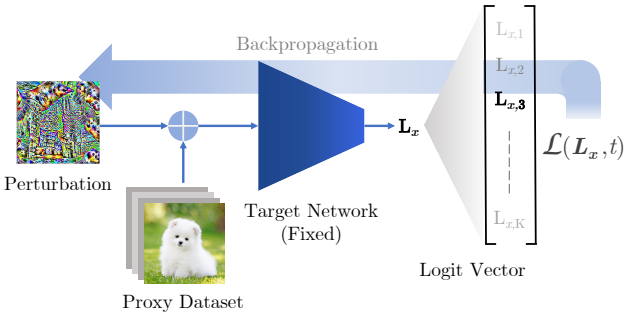

Figure 1. Based on our observation that *adversarial perturbations contain dominant features and images behave like noise to them*, we design a new method of generating targeted universal adversarial perturbations without data, by using a proxy dataset.

Contrary to previous works analyzing adversarial examples as a whole (summation of image and perturbation), we instead propose to analyze adversarial examples by disentangling image and perturbations and studying their mutual influence. Specifically, we analyze the influence of two independent inputs on each other in terms of contributing to the obtained feature representation when the inputs are combined. We treat the network logit outputs as a means of feature representation. Traditionally, only the most important logit values, such as the highest logit value for classification tasks, are considered while other values are disregarded. We propose that all logit values contribute to the feature representation and therefore treat them as a

logit vector. We utilize the Pearson correlation coefficient (PCC) [2] to analyze the extent of linear correlation between logit vectors. The PCC values computed between the logit vectors of each independent input and the input combination gives insight on the contribution of the two independent inputs towards the combined feature representation. Our proposed general analysis framework is shown to be useful for analyzing influence of any two independent inputs, such as images, Gaussian noise, perturbations, etc. In this work, we limit the focus on analyzing the influence of image and perturbation in universal attacks. Our findings show that for a universal attack, the adversarial examples (AEs) are strongly correlated to the UAP, while a low correlation is observed between AEs and input images (see Figure 4). This suggests that for a DNN, UAPs dominate over the clean images in AEs, even though the images are visually more dominant. Treating the DNN as feature extractor, we naturally conclude that the UAP has features that are more dominant compared to the features of the images to attack. Consequently we claim that "UAPs are features while images behave like noise to them". This is contrary to the general perception that treats the perturbation as noise to images in adversarial examples. Our interpretation thus provides a simple yet intuitive insight on the working of UAPs.

The observation, that images behave like noise to UAPs motivates the use of proxy images to generate targeted UAPs without original training data, as shown in Figure 1. Our proposed approach is more practical because the training data is generally inaccessible to the attacker [32]. Our contributions can be summarized as follows:

- We propose to treat the DNN logits as a vector for feature representation. These logit vectors can be used to analyze the contribution of features of two independent inputs when summed towards the output. In particular, our analysis results regarding universal attacks reveal that in an AE, the UAP has dominant features, while the image behaves like noise to them.
- We leverage this insight to derive a method using random source images as proxy dataset to generate targeted UAPs without original training data. To our best knowledge, we are the first to fulfill this challenging task while achieving comparable performance to the state-of-the-art baselines utilizing the original training dataset.

2. Related Work

We summarize previous works with two focuses: (1) explanations of adversarial vulnerability and (2) existing adversarial attack methods.

Explanation of adversarial vulnerability. Goodfellow *et al*. attribute the reason of adversarial examples to the local linearity of DNNs, and support their claim by their proposed simple yet effective FGSM [14]. However, this linearity hypothesis is not fully compatible with the existence of adversarial examples which violate local linearity [24]. Moreover, it can not fully explain the phenomenon that greater robustness is not observed in less linear classifiers [3, 43, 44]. Another body of works attributes the reason for low adversarial robustness to high-dimensional input properties [40, 10, 25, 13]. However, reasonably robust DNNs of high-dimensional inputs can be trained in practice [24, 36]. One recent work [17] attributes the reason for the existence of adversarial examples to non-robust features in the dataset. Some previous explanations, ranging from limited training data induced over-fitting [39, 44] to robustness under noise [11, 12, 6], are well aligned with their framework [17]. The concept of non-robust features is also implicitly explored in other works [4, 33]. On the other hand, possible reasons for vulnerability against universal adversarial perturbations have been explored in [27, 28, 18, 29]. Their analysis is mainly based on the network decision boundaries, in particular, the existence of universal perturbations is linked to the large curvature of decision boundary. Our work mainly focuses on the explanation of universal adversarial vulnerability. One core aspect that differentiates our analysis framework from previous works is that we explore the influence of images and perturbations on each other, while previous works mainly analyze adversarial example as a whole [27, 28, 18]. We explicitly analyze how the image and perturbations influence each other. Our analysis framework is mainly based on the proposed logit vector interpretation of how DNNs respond to the features in the input, without relying on the curvature property of decision boundaries [27, 28, 18].

Existing adversarial attack methods. The existing attacks are commonly categorized under image-dependent attacks [42, 14, 22, 30, 5] and universal (*i.e*. image-agnostic) attacks [27, 19, 32, 26, 35, 46, 34] which devise one single perturbation to attack most images. Image-dependent attack techniques have been explored in a variety of works ranging from optimization based techniques [42, 5] to FGSM related techniques [14, 22, 7, 45]. Universal adversarial perturbations (UAPs) were first proposed by [27], and deploy the DeepFool attack [30] iteratively on single data samples. Due to the nature of being image-agnostic, universal attacks constitute a more challenging task than image-dependent ones.

Another way to categorize attacks is non-targeted vs. targeted attacks. Generative targeted universal perturbations have been explored by [35]. Targeted attacks can be seen as a special, but more challenging case of non-targeted attacks. Class discriminative (CD) UAPs were proposed in [46], aiming to fool only a subset of classes. The above mentioned universal attacks require utilization of the original

training data. However, in practice the attacker often has no access to the training data [32]. To overcome this limitation, Mopuri *et al*. propose to generate universal perturbation without training data [32]. However, their approach is specifically designed for non-targeted attacks by maximizing the activation scores in every layer, and their performance is inferior to approaches with access to original training data. Another attempt for data-free non-targeted universal attack by training a network to generate proxy images is explored in [38] . No prior work is found to have achieved targeted universal attack without access to the original training data, and our work is the first attempt in this direction.

3. Analysis Framework

3.1. Logit Vector

Following the common consensus that DNNs are feature extractors, we intend to analyze adversarial examples from the feature perspective. The logit values are often used as an indicator of feature presence in an image. Previous works [18, 17], however, mainly focus only on the DNN highest logit output indicating the predicted class, while all other logits are usually neglected. "Logits" refer to the DNN output before the final softmax layer. In this work, we assume that all DNN output logit values represent the network response to features in the input. One concern about this vector interpretation is that only the logits of the ground-truth classes or other semantically similar classes are meaningful, while the other logits might be just random (small) values and thus do not carry important information. We address this concern after introducing the terms and notation used throughout this work.

A deep classifier $\hat{C}$ maps an input image $x \in \mathbb{R}^d$ with a pixel range of $[0, 1]$ to an output logit vector $L_x = \hat{C}(x)$. The vector L_x has K entries corresponding to the total number of classes. The predicted class y_x of an input x can then be calculated from the logit vector as $y_x = \arg\max(L_x)$. We adopt the logit vector to facilitate the analysis of the mutual influence of two independent inputs in terms of their contribution to the combined feature representation. We mainly consider two independent inputs $a \in \mathbb{R}^d$ and $b \in \mathbb{R}^d$, which can be images, Gaussian noise, perturbations, etc., whose corresponding logit vectors are denoted as L_a and L_b, respectively. The summation of these two inputs $c = a + b$, when fed to a DNN, leads to the feature representation L_c. Both inputs a and b contribute partially to L_c. Moreover, it is reasonable to expect that the contribution of each input will be influenced by the other one. Specifically, the extent of influence will be reflected in the linear correlation between the individual logit vector L_a (or L_b) and L_c.

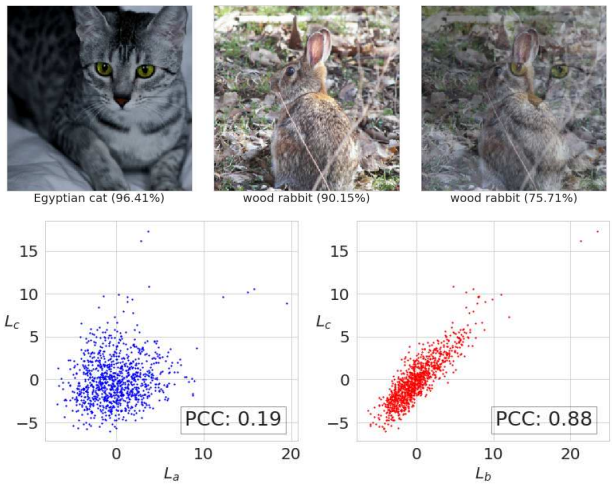

Figure 2. Images and their logit vector analysis. The first row shows the sample images a and b and the resulting image c. The second row shows the plots of logit vector L_c over L_a (left) and L_b (right), with their respective PCC values.

3.2. Pearson Correlation Coefficient

In statistics, the Pearson correlation coefficient (PCC) [2] is a widely adopted metric to measure the linear correlation between two variables. In general, this coefficient is defined as

$$\text{PCC}_{X,Y} = \frac{cov(X, Y)}{\sigma_X \sigma_Y}, \tag{1}$$

where cov indicates the covariance and σ_X and σ_Y are the standard deviation of vector X and Y, respectively, and the PCC values range from -1 to 1. The absolute value indicates the extent to which the two variables are linearly correlated, with 1 indicating perfect linear correlation, 0 indicating zero linear correlation, and the sign indicates whether they are positively or negatively correlated. Treating the logit vector as a variable, the PCC between different logit vectors can be calculated. We are mainly concerned about PCC_{L_a,L_c} and PCC_{L_b,L_c}, since PCC_{L_a,L_b} is always close to zero due to independence. Comparing PCC_{L_a,L_c} and PCC_{L_b,L_c} can provide insight about the contribution of the two inputs to L_c, with a higher PCC value indicating the more significant contributor. For example, if PCC_{L_a,L_c} is larger than PCC_{L_b,L_c}, input a's share can be seen as more dominant than input b towards the final feature response. The relationship of two logit vectors, L_a and L_c for instance, can be visualized by plotting each logit pair. The extent of their correlation can be observed and quantified by the PCC.

As a basic example, we show the logit vector analysis of two randomly sampled images from ImageNet [41] in Figure 2. The plot shows a strong linear correlation between L_b and L_c ($\text{PCC}_{L_b,L_c} = 0.88$), while L_a and L_c are practically uncorrelated ($\text{PCC}_{L_a,L_c} = 0.19$). These observations sug-

gest a dominant contribution of input b towards logit vector L_c. As a result, the same label "Wood rabbit" is predicted for c and b. Such combination of images has also been explored in Mixup [49] for training classifiers.

Table 1. PCC analysis for VGG19 using 1000 image pairs randomly sampled from the ImageNet test set. Here, for each image pair, the mean and standard deviations of higher and lower PCC values are reported under PCC_h and PCC_l, respectively.

	$\lvert S\rvert$	PCC_h	PCC_l	$\text{PCC}_h - \text{PCC}_l$	$\mathcal{P}_{\text{PCC}}$
S_m	445	0.74 ± 0.10	0.27 ± 0.23	0.47 ± 0.27	96%
S_n	555	0.63 ± 0.13	0.33 ± 0.20	0.30 ± 0.22	-

To establish the reliability of the PCC value as a metric, we repeat the above experiment with 1000 image pairs and report results on the effectiveness of PCC to predict label c in Table 1. We divide the image pairs into two groups: S_m and S_n. S_m comprises of image pairs having the same predicted class y_c as the prediction y_a or y_b. For S_n, the predicted class y_c is different from both y_a and y_b. Moreover, we use the parameter $\mathcal{P}_{\text{PCC}}$ to show the proportion of predictions correctly inferred from the PCC values relative to the network predictions for c. For the image pairs from set S_m, the $\mathcal{P}_{\text{PCC}}$ is 96%, confirming the reliability of the PCC as our metric. The high gap between PCC_h and PCC_l further provides evidence for the high $\mathcal{P}_{\text{PCC}}$. For the image pairs from S_n, $\text{PCC}_h - \text{PCC}_l$ is smaller, implying that neither of the inputs is significantly dominant.

Recall that there is a concern that most logit values might be just random values, which is partially addressed by observing the correlation between PCC and y_c as shown in Figure 2. If the concern were valid, such that only a few logits are meaningful (*i.e.* only the highest logits or the logits for semantically similar classes), a high divergence should be observed for the less significant logits. However, this assumption does not align well with the results in Figure 2, thus confirming the importance of all logit values. A higher PCC value for the dominant input further rules out the concern that the lower logit values are random.

4. Influence of Images and Perturbations on Each Other

In this section, we analyze the interaction of clean images with Gaussian noise perturbation, universal perturbations and image-dependent perturbations. In doing so, input a is the image and input b the perturbation. The analysis is performed on VGG19 pretrained on ImageNet. For consistency, a randomly chosen a (shown in Figure 2, top left) is used for all experiments. Along the same lines, for targeted perturbations we randomly set 'sea lion' as the target class t. For more results with different images and target classes on different networks, please refer to the supplementary material.

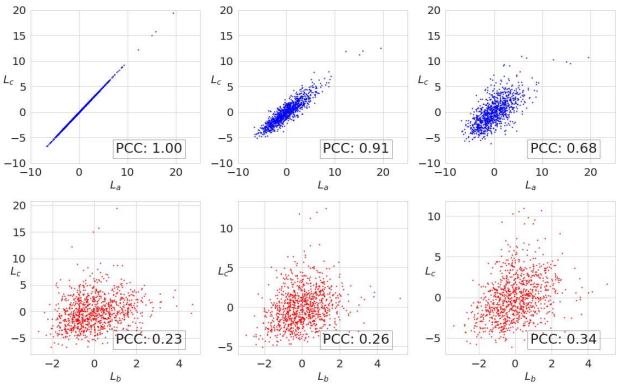

Figure 3. Logit vector analysis for an input image and Gaussian noise $\mathcal{N}(\mu, \sigma)$. The analysis is shown for $\mu = 0$ and $\sigma = 0$ (left), $\sigma = 0.1$ (middle) and $\sigma = 0.2$ (right))

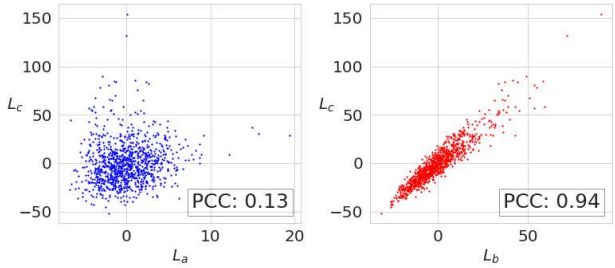

Figure 4. Logit vector analysis for input image (a) and targeted UAP (b). The targeted UAP was trained for target class 'sea lion' and loss function $\mathcal{L}^t_{CL2}$

4.1. Analysis of Gaussian Noise

To facilitate the interpretation of our main experiment of performing analysis for perturbations, we first show the influence of noise (Gaussian noise) on images. The Gaussian noise is sampled from $\mathcal{N}(\mu, \sigma)$ with $\mu = 0$ and different standard deviations. The relationship between L_a, L_c is visualized in Figure 3. As expected, by adding zero magnitude Gaussian noise (*i.e.* no Gaussian noise) to the image, L_a and L_c are perfectly linearly correlated ($\text{PCC}_{L_a, L_c} = 1$). If the Gaussian noise magnitude is increased ($\sigma = 0.1$ for instance), L_a and L_c still show a high linear correlation ($\text{PCC}_{L_a, L_c} = 0.91$). Investigating the relationship between L_b and L_c, a low correlation can be observed for all noise inputs b indicating a low contribution to the final prediction.

4.2. Analysis of Universal Perturbations

Universal perturbations come in two flavors: targeted and non-targeted. We use Algorithm 1 with loss function $\mathcal{L}^t_{CL2}$ to generate targeted universal perturbations, and generate non-targeted universal perturbations using Equation 4 as the loss function. The results of this analysis are shown for a targeted and non-targeted UAP in Figure 4 and Figure 5, respectively. For the targeted scenario, two major

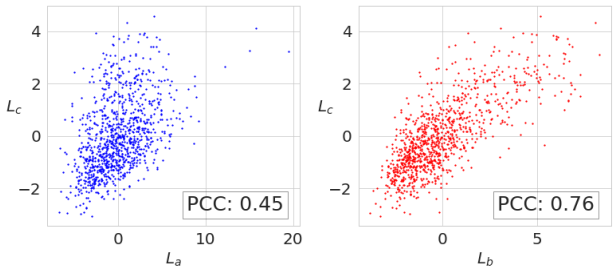

Figure 5. Logit vector analysis for input image (a) and non-targeted UAP (b). The UAP was trained with loss function Equation 4

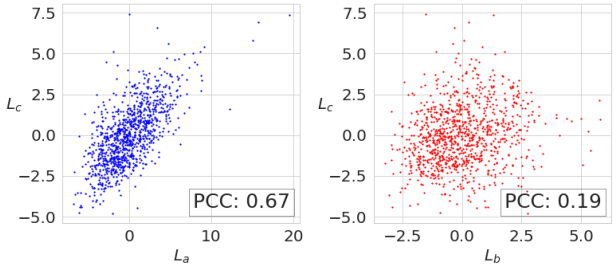

Figure 6. Logit vector analysis for input image (a) and targeted image-dependent perturbation (b). The perturbation was crafted with PGD [24], with target class 'sea lion'

observations can be made: First, PCC_{L_a,L_c} is smaller than PCC_{L_b,L_c}, indicating a higher linear correlation between L_c and L_b than L_c and L_a. In other words, the features of the perturbation are more dominant than that of the clean image. Second, PCC_{L_a,L_c} is close to 0, indicating that the influence of the perturbation on the image is so significant that the clean image features are seemingly unrecognizable to the DNN. In fact, comparing the logit analysis of L_a and L_c in Figure 4 with that of Gaussian noise and image in Figure 3 (bottom), a striking similarity is observed. This offers a novel interpretation of targeted universal perturbations: **Targeted universal perturbations themselves (independent of the images to attack) are features, while images behave like noise to them.** We further explore the non-targeted perturbations, and report the results in Figure 5. Similar to targeted universal perturbations, the PCC_{L_a,L_c} is smaller than PCC_{L_b,L_c} for the non-targeted perturbation. However the dominance of the non-targeted perturbation is not as significant as that of the targeted perturbation.

4.3. Analysis of Image-Dependent Perturbations

The logit vector analysis results for targeted and non-targeted image-dependent perturbations are reported in Figure 6 and Figure 7, respectively. Contrary to the universal perturbations, the image-dependent perturbations are weakly correlated to c, and have a noise-like behaviour (Figure 3). However, the image gets misclassified even

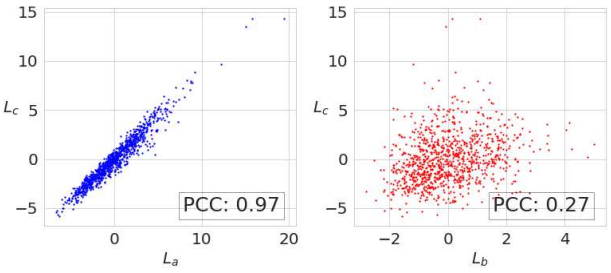

Figure 7. Logit vector analysis for input image (a) and non-targeted image-dependent perturbation (b). The perturbation was crafted with PGD [24]

though the image features appear to be more dominant than the perturbation. This is because the image features are more strongly corrupted through the image-dependent perturbation than Gaussian noise. This special behavior appears due to the fact that the image-dependent perturbations are crafted to form concrete features only in combination with the image. Such image-dependent behavior violates our assumption of independent inputs. However, we include these results since they offer additional insight into adversarial examples.

4.4. Why Do Adversarial Perturbations Exist?

A wide variety of works have explored the existence of adversarial examples as discussed in section 2. Based on our previous analyses, we arrive at the following explanation for the existence of UAPs:

Universal adversarial perturbations contain features independent of the images to attack. The image features are corrupted to an extent of being unrecognizable to a DNN, and thus the input images behave like noise to the perturbation features.

The finding in [18] that universal perturbations behave like features of a certain class aligns well with our statement. Jetley *et al*. argue that universal perturbations exploit the high-curvature image-space directions to behave like features, while our finding suggests that universal perturbations themselves contain features independent of the images to attack. Utilizing the perspective of positive curvatures of decision boundaries, Jetley *et al*. adopt the decision boundary-based attack DeepFool [30]. However, our explanation does not explicitly rely on the decision boundary properties, but focuses on the occurrences of strong features, robust to the influence of images. We can therefore deploy the PGD algorithm to generate perturbations consisting of target class features similar to [17].

If universal perturbations themselves contain features independent of the images to attack, do image-dependent perturbations behave in a similar way? As previously discussed, the analysis results in Figure 6 reveal that the behavior of image-dependent perturbations is not like features,

but noise. On the other hand, the original image features are retained to a high extent. Ilyas *et al*. [17] revealed that image-dependent adversarial examples include the features of the target class. However, as seen from the analysis in subsection 4.4, the isolated perturbation seems not to retain independent features due its low PCC value, but rather interacts with the image to form the adversarial features.

5. Targeted UAP with Proxy Data

Our above analysis demonstrates that images behave like noise to the universal perturbation features. Since the images are treated like noise, we can exploit proxy images as background noise to generate targeted UAPs without the original training data. The proxy images do not need to have any class object belonging to the original training class and their main role is to make the targeted UAP have strong background-robust target class features.

5.1. Problem Definition

Formally, given a data distribution $\mathcal{X} \in \mathbb{R}^d$ of images, we compute a single perturbation vector v that satisfies

$$\begin{aligned} &\hat{C}(x+v) = t \quad \text{for } \textit{most } x \sim \mathcal{X} \\ &||v||_p \leq \epsilon. \end{aligned} \tag{2}$$

The magnitude of v is constrained by ϵ to be imperceptible to humans. $||\cdot||_p$ refers to the l_p-norm and in this work, we set $p = \infty$ and $\epsilon = 10$ for images in range $[0, 255]$[1] as in [27]. Specifically, we assume having no access to original training data. Thus, the training data $\mathcal{X}_v$ for v generation can be different from the original dataset $\mathcal{X}$. We denote the proxy dataset as $\mathcal{X}_v$.

To evaluate targeted UAPs, we use the targeted fooling ratio metric [35], *i.e*. the ratio of samples fooled into the target class to the number of all data samples. We also use the non-targeted fooling ratio [35, 27], calculating the ratio of misclassified samples to the total number of samples, for evaluation.

5.2. Loss Function and Algorithm

To achieve the desired objective Eq. 2 most naively, the commonly used cross-entropy loss function $\mathcal{L}_{\text{CE}}$ can be utilized. Since cross-entropy loss holistically incorporates logits of all classes, this loss function leads to overall lower fooling ratios. This behavior can be resolved by using a loss function $\mathcal{L}_{\text{L}}$ that only aims to increase the logit of the target class.

Since we consider universal perturbations, to balance the above objective between different samples in training, we extend $\mathcal{L}_{\text{L}}$ by clamping the logit values as follows:

$$\mathcal{L}^t_{CL1} = \max(\max_{i \neq t} \hat{C}_i(x_v + v) - \hat{C}_t(x_v + v), -\kappa) \tag{3}$$

[1]For images in the range $[0, 1]$, $\epsilon = \frac{10}{255}$

Algorithm 1: UAP algorithm

Input: Proxy data $\mathcal{X}_v$, Classifier $\hat{C}$, Loss function $\mathcal{L}$, mini-batch size m, Number of iterations I, perturbation magnitude ϵ
Output: Perturbation vector v

```
v ← 0                                   ▷ Initialize
for iteration = 1, ..., I do
    B ∼ X_v: |B| = m                    ▷ Randomly sample
    g_v ← E_{x∼B}[∇_v L]                ▷ Calculate gradient
    v ← Optim(g_v)                      ▷ Update
    v ← ε v / ||v||_p                   ▷ Norm projection
end
```

where κ indicates the confidence value, x_v are samples from the proxy data $\mathcal{X}_v$ and $\hat{C}_i$ indicates the i-th entry of the logit vector. In this case, the proxy data can be either a random source dataset or the original training data, depending on data availability. Note that similar techniques of clamping the logits have also been used in [5], however, their motivation is to obtain minimum-magnitude (image-dependent) perturbations. While the target logit in loss function $\mathcal{L}^t_{CL1}$ is increased, the logit values of $\max \hat{C}_i(x_v + v)$ are decreased simultaneously during the training process. This effect is undesirable for generating a UAP with strong target class features, since other classes except the target classes will be included in the optimization, which might have negative effects on the gradient update. To prevent manipulation of logits other than the target class, we exclude the non-targeted class logit values in the optimization step, such that these values are only used as a reference value for clamping the target class logit. We indicate this loss function as $\mathcal{L}^t_{CL2}$. We report an ablation study of the different loss function performances in Table 2. The results suggest that $\mathcal{L}^t_{CL2}$, in general, outperforms all other discussed loss functions. We further provide a loss function resembling $\mathcal{L}^t_{CL2}$ for the generation of non-targeted UAPs.

$$\mathcal{L}^{nt} = \max(\hat{C}_{gt}(x_v + v) - \max_{i \neq gt} \hat{C}_i(x_v + v), -\kappa) \tag{4}$$

In the special case of crafting non-targeted UAPs, the proxy dataset has to be the original training dataset.

We provide a simple, yet effective algorithm in Algorithm 1. Our gradient based method adopts the ADAM [20] optimizer and mini-batch training, which have also been adopted in the context of data-free universal adversarial perturbations [38]. Mopuri *et al*. train a generator network for crafting UAPs with this configurations, which can be considered more complex.

Table 2. Ablation study on the performance of different loss functions, for the proposed targeted UAP. The values in each column represent mean and standard deviation of the non-targeted fooling ratio (%) and targeted fooling ratio (%) obtained for 5 runs and target class 'sea lion'.

Loss	AlexNet		GoogleNet		VGG16		VGG19		ResNet152	
$\mathcal{L}_{\text{CE}}$	$\mathbf{90.5 \pm 0.6}$	55.4 ± 1.0	70.8 ± 1.5	55.2 ± 2.2	89.1 ± 0.3	75.9 ± 0.9	87.9 ± 0.5	70.8 ± 1.1	78.2 ± 0.9	66.5 ± 1.3
$\mathcal{L}_{\text{L}}$	89.2 ± 0.4	47.1 ± 1.1	71.6 ± 0.8	56.9 ± 1.1	91.0 ± 0.3	79.0 ± 0.6	90.8 ± 0.2	73.1 ± 0.8	80.1 ± 0.8	69.1 ± 0.4
$\mathcal{L}^t_{CL1}$	90.2 ± 0.3	$\mathbf{57.6 \pm 1.4}$	71.7 ± 1.4	57.9 ± 2.3	90.1 ± 0.4	80.3 ± 0.5	88.2 ± 0.3	$\mathbf{75.5 \pm 0.6}$	80.2 ± 0.3	$\mathbf{71.4 \pm 0.5}$
$\mathcal{L}^t_{CL2}$	$\mathbf{90.5 \pm 0.3}$	49.4 ± 1.2	$\mathbf{73.0 \pm 1.5}$	$\mathbf{58.4 \pm 2.2}$	$\mathbf{93.5 \pm 0.3}$	$\mathbf{82.8 \pm 0.7}$	$\mathbf{92.7 \pm 0.1}$	72.3 ± 2.5	$\mathbf{81.3 \pm 1.1}$	70.6 ± 2.1

Table 3. Results for targeted UAPs trained on four different datasets. The values in each column represent mean and standard deviation of the non-targeted fooling ratio (%) and targeted fooling ratio (%) obtained for 8 different target classes.

Proxy Data	AlexNet		GoogleNet		VGG16		VGG19		ResNet152	
ImageNet [41]	89.9 ± 2.2	48.6 ± 13.3	77.7 ± 3.2	59.9 ± 6.6	92.5 ± 1.3	75.0 ± 7.8	91.6 ± 1.3	71.6 ± 6.9	80.8 ± 2.6	66.3 ± 7.0
COCO [23]	89.9 ± 2.6	47.2 ± 13.1	76.8 ± 3.7	59.8 ± 7.5	92.2 ± 1.7	75.1 ± 12.3	91.6 ± 1.5	68.8 ± 9.4	79.9 ± 2.9	65.7 ± 7.8
VOC [9]	88.9 ± 2.6	46.9 ± 12.7	76.7 ± 3.2	58.9 ± 6.0	92.2 ± 1.6	74.7 ± 7.9	90.5 ± 2.3	68.8 ± 8.2	79.1 ± 3.3	65.2 ± 7.1
Places365 [50]	90.0 ± 2.1	42.6 ± 16.4	76.4 ± 3.7	60.0 ± 5.4	92.1 ± 1.5	73.4 ± 9.6	91.5 ± 1.6	64.5 ± 17.0	78.0 ± 3.2	62.5 ± 9.9

Table 4. Comparison of the proposed method to other methods. The results are divided in universal attacks with access to the original ImageNet training data (upper) and data-free methods (lower). The metric is reported in the non-targeted fooling ratio (%))

Method	AlexNet	GoogleNet	VGG16	VGG19	ResNet152
UAP [27]	93.3	78.9	78.3	77.8	84.0
GAP [35]	-	82.7	83.7	80.1	-
Ours(ImageNet)	**96.17**	**88.94**	**94.30**	**94.98**	**90.08**
FFF [32]	80.92	56.44	47.10	43.62	-
AAA [38]	89.04	75.28	71.59	72.84	60.72
GD-UAP [31]	87.02	71.44	63.08	64.67	37.3
Ours (COCO)	**89.9**	**76.8**	**92.2**	**91.6**	**79.9**

Table 5. Transferability results for the proposed targeted universal adversarial attack. The attack was performed for target class 'sea lion' and proxy dataset MS-COCO. The rows indicate the source model and the columns indicates the target model. The values in each column are reported in the non-targeted fooling ratio (%) and targeted fooling ratio (%)

	AlexNet		GoogleNet		VGG-16		VGG19		ResNet152	
AlexNet	**90.45**	**49.61**	54.77	0.01	60.43	0.13	58.66	0.09	47.02	0.02
GoogleNet	53.25	0.02	**75.47**	**62.06**	50.51	0.17	48.79	0.14	34.94	0.34
VGG16	53.71	0.03	41.26	0.02	**93.62**	**82.90**	82.99	13.69	36.73	0.01
VGG19	53.67	0.02	39.78	0.02	83.40	44.53	**92.53**	**75.61**	35.36	0.01
ResNet152	54.46	0.03	42.43	0.07	55.05	1.63	55.12	1.05	**80.47**	**70.20**

Table 6. Results for Transferability measured with PCC values. Generated with COCO as background, for target class sea lion. The rows indicate the source model and the columns indicates the target model.

	AlexNet	GoogleNet	VGG-16	VGG19	ResNet152
AlexNet	**1.00**	0.09	0.24	0.14	−0.05
GoogleNet	0.24	**1.00**	0.24	0.14	0.00
VGG16	0.36	0.09	**1.00**	0.48	−0.11
VGG19	0.19	0.07	0.55	**1.00**	−0.09
ResNet152	0.28	0.11	0.36	0.30	**1.00**

5.3. Main Results

We generate the targeted UAPs for four different datasets, the ImageNet training set as well as three proxy datasets. In Algorithm 1, we set the number of iterations to 1000, use loss function $\mathcal{L}^t_{CL2}$ and a learning rate of 0.005 with batch-size 32. As the proxy datasets, we use images from MS-COCO [23] and Pascal VOC [9], two widely used object detection datasets, and Places365 [50], a large-scale scene recognition dataset. We generated targeted UAPs with the 4 datasets for 8 different target classes and evaluate them on the ImageNet test dataset. The average over the 8 target scenarios are reported in Table 3. Two major observations can be made: First, a significant difference can not be observed for the three different proxy datasets. Moreover, there is only a marginal performance gap between training with the proxy datasets and training with the original ImageNet training data. The results support our assumption that the influence of the input images on targeted UAPs is like noise.

We also explored generating targeted UAPs with white images and Gaussian noise as the proxy dataset. In both scenarios, inferior performance was observed. We refer the reader to the supplementary material for a discussion about possible reasons and further results.

Targeted perturbations for different networks are shown in Figure 8. Since the target class is sea lion, we can notice the existence of sea lion-like patterns by taking a closer look. Samples of clean images and perturbed images misclassified as sea lion are shown in Figure 9.

5.4. Comparison with Previous Methods

To the best of our knowledge, this is the first work to achieve targeted UAP without original training data, thus we can only compare our performance with previous works on related tasks. The authors of [35] report a targeted fool-

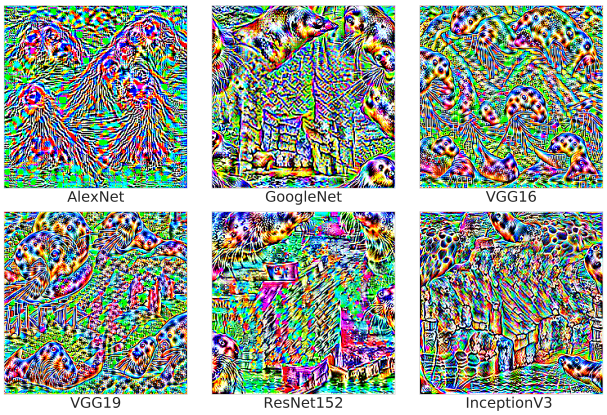

Figure 8. Targeted universal perturbations (target class 'sea lion') for different network architectures.

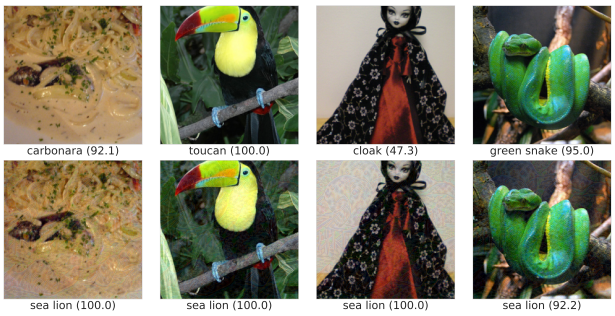

Figure 9. Qualitative Results. Clean images (top) and perturbed images (bottom) for VGG19

ing ratio of 52% for Inception-V3 with access to the ImageNet training dataset. We use COCO as the proxy dataset and achieve a superior performance of 53.4%. We can not find any other targeted UAP method available in the literature but other previous works report the (non-targeted) fooling ratio and we compare our performance with them and the results are available in Table 4. We distinguish between methods with and without data availability. To compare with the methods with data-availability we trained a non-targeted UAP on ImageNet utilizing our introduced non-targeted loss function from Equation 4. Note that we do not block the gradient for $\max_{i \neq gt} \hat{C}_i(x_v + v)$ to let the algorithm automatically search a dominant class for an effective attack. We observe that our approach achieves superior performance than both UAP [27] and GAP [35]. For the case without access to the original training dataset, we use the COCO dataset to generate the UAP, and report the averages of performance on 8 target classes. Note that our method still generates a targeted UAP, but we use the non-targeted metric for performance evaluation. This setting is in favor of other methods, since ideally, we could report the best performance of a certain target class. Without bells and whistles, our method achieves comparable performance to the state-of-the-art data-free methods, constituting evidence that our simple approach is efficient.

5.5. Transferability

The transferability results are available in Table 5. We observe that the non-targeted transferability performs reasonably well, while targeted transferability does not. We find no previous work reporting the targeted transferability for universal perturbations. For image-dependent perturbations, the targeted transferability has been explored in [15], which reveals that the targeted transferability is unsatisfactory when source network and target network belong to different network families. When the networks belong to the same network family, relatively higher transferability can be observed [15]. This aligns well with our finding that VGG16 and VGG19 transfer reasonably well between each other as presented in Table 5. We further report the PCC of the two network UAPs in Table 6. We observe that the PCC values are relatively higher between VGG16 and VGG19 than other networks, indicating an additional benefit of PCC to provide insight to network transferability.

6. Conclusion

In this work, we treat the DNN logit output as a vector to analyze the influence of two independent inputs in terms of contributing to the combined feature representation. Specifically, we demonstrate that the Pearson correlation coefficient (PCC) can be used to analyze relative contribution and dominance of each input. Under the proposed analysis framework, we analyze adversarial examples by disentangling images and perturbations to explore their mutual influence. Our analysis results reveal that universal perturbations have dominant features and the images to attack behave like noise them. This new insight yields a simple yet effective algorithm, with a carefully designed loss function, to generate targeted UAPs by exploiting a proxy dataset instead of the original training data. We are the first to achieve this challenging task and the performance is comparable to state-of-the-art baselines utilizing the original training dataset.

7. Acknowledgement

We thank Francois Rameau and Dawit Mureja Argaw for their comments and suggestions throughout this project. This work was supported by NAVER LABS and the Institute for Information & Communications Technology Promotion (2017-0-01772) grant funded by the Korea government.

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 2018. 1

[2] TW Anderson. An introduction to multivariate statistical analysis (wiley series in probability and statistics). 2003. 2, 3

[3] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018. 1, 2

[4] Sebastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In *International Conference on Machine Learning (ICML)*, 2019. 2

[5] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Symposium on Security and Privacy (SP)*, 2017. 2, 6

[6] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*, 2019. 2

[7] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[8] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision (ICCV)*, 2015. 1

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 2010. 7

[10] Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. Adversarial vulnerability for any classifier. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 2

[11] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 2

[12] Justin Gilmer, Nicolas Ford, Nicholas Carlini, and Ekin Cubuk. Adversarial examples are a natural consequence of test error in noise. In *International Conference on Machine Learning (ICML)*, 2019. 2

[13] 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

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015. 1, 2

[15] Jiangfan Han, Xiaoyi Dong, Ruimao Zhang, Dongdong Chen, Weiming Zhang, Nenghai Yu, Ping Luo, and Xiaogang Wang. Once a man: Towards multi-target attack via learning multi-target adversarial network once. In *International Conference on Computer Vision (ICCV)*, 2019. 8

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[17] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1, 2, 3, 5, 6

[18] Saumya Jetley, Nicholas Lord, and Philip Torr. With friends like these, who needs adversaries? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 2, 3, 5

[19] Valentin Khrulkov and Ivan Oseledets. Art of singular vectors and universal adversarial perturbations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 6

[21] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, 2017. 1

[22] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations (ICLR)*, 2017. 2

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014. 7

[24] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 5

[25] 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 Conference on Artificial Intelligence (AAAI)*, 2019. 2

[26] Jan Hendrik Metzen, Mummadi Chaithanya Kumar, Thomas Brox, and Volker Fischer. Universal adversarial perturbations against semantic image segmentation. In *International Conference on Computer Vision (ICCV)*, 2017. 2

[27] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 6, 7, 8

[28] 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

[29] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, and Stefano Soatto. Robustness of classifiers to universal perturbations: A geometric perspective. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[30] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 5

[31] Konda Reddy Mopuri, Aditya Ganeshan, and Venkatesh Babu Radhakrishnan. Generalizable data-free

objective for crafting universal adversarial perturbations. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018. 7

[32] Konda Reddy Mopuri, Utsav Garg, and R. Venkatesh Babu. Fast feature fool: A data independent approach to universal adversarial perturbations. In *British Conference on Machine Vision (BMVC)*, 2017. 2, 3, 7

[33] Preetum Nakkiran. A discussion of 'adversarial examples are not bugs, they are features': Adversarial examples are just bugs, too. *Distill*, 2019. https://distill.pub/2019/advex-bugs-discussion/response-5. 1, 2

[34] Muhammad Muzammal Naseer, Salman H Khan, Muhammad Haris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-domain transferability of adversarial perturbations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[35] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 6, 7, 8

[36] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[37] Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J Black. Attacking optical flow. In *International Conference on Computer Vision (ICCV)*, 2019. 1

[38] Konda Reddy Mopuri, Phani Krishna Uppala, and R Venkatesh Babu. Ask, acquire, and attack: Data-free uap generation using class impressions. In *European Conference on Computer Vision (ECCV)*, 2018. 3, 6, 7

[39] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 2

[40] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104*, 2018. 2

[41] Ilya Sutskever, Geoffrey E Hinton, and A Krizhevsky. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 3, 7

[42] 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

[43] Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016. 1, 2

[44] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016. 1, 2

[45] Lei Wu, Zhanxing Zhu, Cheng Tai, et al. Understanding and enhancing the transferability of adversarial examples. *arXiv preprint arXiv:1802.09707*, 2018. 2

[46] Chaoning Zhang, Philipp Benz, Tooba Imtiaz, and In-So Kweon. Cd-uap: Class discriminative universal adversarial perturbation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 2

[47] Chaoning Zhang, Francois Rameau, Junsik Kim, Dawit Mureja Argaw, Jean-Charles Bazin, and In So Kweon. Deepptz: Deep self-calibration for ptz cameras. In *Winter Conference on Applications of Computer Vision (WACV)*, 2020. 1

[48] Chaoning Zhang, Francois Rameau, Seokju Lee, Junsik Kim, Philipp Benz, Dawit Mureja Argaw, Jean-Charles Bazin, and In So Kweon. Revisiting residual networks with nonlinear shortcuts. In *British Machine Vision Conference (BMVC)*, 2019. 1

[49] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018. 4

[50] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017. 7