Introducing Competition to Boost the Transferability of Targeted Adversarial Examples through Clean Feature Mixup

Junyoung Byun Myung-Joon Kwon Seungju Cho Yoonji Kim Changick Kim
Korea Advanced Institute of Science and Technology (KAIST)
{bjyoung, kwon19, joyga, yoonjikim, changick}@kaist.ac.kr

Abstract

Deep neural networks are widely known to be susceptible to adversarial examples, which can cause incorrect predictions through subtle input modifications. These adversarial examples tend to be transferable between models, but targeted attacks still have lower attack success rates due to significant variations in decision boundaries. To enhance the transferability of targeted adversarial examples, we propose introducing competition into the optimization process. Our idea is to craft adversarial perturbations in the presence of two new types of competitor noises: adversarial perturbations towards different target classes and friendly perturbations towards the correct class. With these competitors, even if an adversarial example deceives a network to extract specific features leading to the target class, this disturbance can be suppressed by other competitors. Therefore, within this competition, adversarial examples should take different attack strategies by leveraging more diverse features to overwhelm their interference, leading to improving their transferability to different models. Considering the computational complexity, we efficiently simulate various interference from these two types of competitors in feature space by randomly mixing up stored clean features in the model inference and named this method Clean Feature Mixup (CFM). Our extensive experimental results on the ImageNet-Compatible and CIFAR-10 datasets show that the proposed method outperforms the existing baselines with a clear margin. Our code is available at https://github.com/dreamflake/CFM.

1. Introduction

Although deep neural networks have excelled in various computer vision tasks such as image classification [10, 12] and object detection [19, 22], they are vulnerable to maliciously crafted inputs called *adversarial examples* [8, 37]. These adversarial examples are generated by optimizing imperceptible perturbations to mislead a model to incorrect predictions. Intriguingly, these adversarial examples tend to be transferable between models, and this unique characteristic allows adversaries to attempt adversarial attacks on a black-box model without knowing its interior. However, targeted adversarial attacks, which have a specific target class, still have lower attack success rates due to significant differences in decision boundaries [17, 37]. Nevertheless, targeted attacks can pose more serious risks as they can deceive models into predicting a specific harmful target class. Therefore, preemptive research on developing a novel transfer-based attack is crucial because it can assist service providers in preparing their models for these forthcoming risks and evaluating their models' robustness.

In this work, we aim to further improve the transferability of targeted adversarial examples by introducing competition into their optimization. Our approach involves crafting adversarial perturbations in the presence of two new types of noises: *(a) adversarial perturbations towards different target classes*; and *(b) friendly perturbations towards the correct class*. With these competitors and a source model, even if an adversarial example deceives the source model into extracting certain features that lead to the target class, this disturbance may be suppressed by interference from competitors. Consequently, adversarial perturbations should take various attack strategies, leveraging a wider range of features to overcome interference, which enhances their transferability to different models. In the following, we will further discuss why employing a diverse set of features for attack can boost the transferability of targeted adversarial examples.

In image classification, deep learning models extract a variety of features from images across multiple layers and comprehensively evaluate them to calculate prediction probabilities for each class. As numerous features can contribute to the final output, even when two images are recognized as the same class, the contributing features can significantly differ. Taking this into account, optimizing adversarial examples to utilize as many distinct feature combinations as possible would effectively enhance their transferability.

Conversely, in existing frameworks, an adversarial ex-

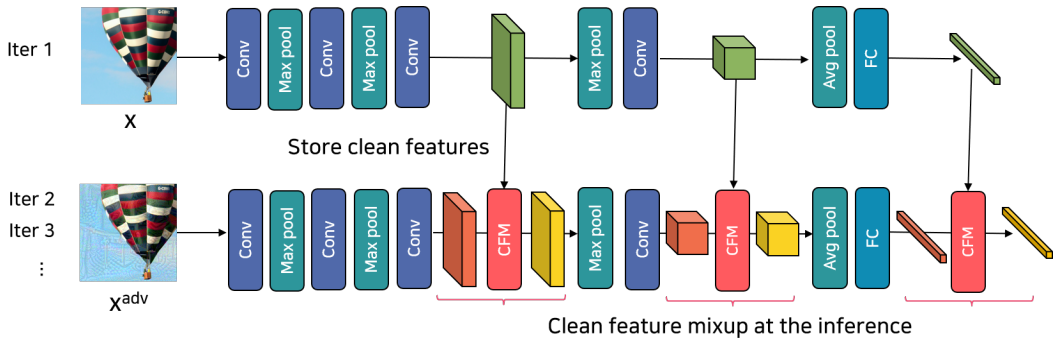

Figure 1. Overview of the Clean Feature Mixup (CFM) method.

ample may be optimized to intensely distract a limited number of features identified in the early stages. However, the target model, unlike the source model, might be insensitive to such feature distractions, leading to the failure of transfer-based attacks.

Meanwhile, if we model the competitors as noises that should also be optimized, they require additional backward passes, substantially increasing the computational burden. To address this challenge and enhance interference diversity, we propose *Clean Feature Mixup* (CFM), a method that efficiently mimics the behaviors of the two types of perturbations in feature space by mixing stored clean features of the images within a batch. A detailed description of their similarities can be seen in Section 3.3.

The overview of the proposed CFM method is illustrated in Fig. 1. Specifically, this method converts a pre-trained source model by attaching our specially designed CFM modules to convolution and fully-connected layers. After that, the attached CFM modules randomly mix the features of the clean images (i.e., *clean features*) with current input features at each inference. This process can effectively mitigate the overfitting of adversarial examples in their optimization by preventing them from focusing on particular features in their targeted attacks on the source model. Unlike many existing techniques [18, 30, 31] that significantly increase the computational cost by multiplying the required number of forward/backward passes, CFM adds just one additional forward pass for storing clean features and requires a marginal amount of computation for feature mixup at each inference.

Our contributions can be summarized as follows:

- We propose the idea of introducing competition into the optimization of targeted adversarial examples with two types of competitor noises to encourage the utilization of various features in their attacks, ultimately boosting their transferability.
- Motivated by the above idea, we propose the Clean Feature Mixup (CFM) method to improve the transferability of adversarial examples. This method efficiently simulates the competitor noises by randomly mixing up stored clean features of the images in a batch.
- We performed extensive experiments with 20 models, including four defensive models and five Transformer-based classifiers. Our experimental results on the ImageNet-Compatible and CIFAR-10 datasets demonstrate that CFM outperforms state-of-the-art baselines.

2. Background

In this section, we briefly overview the background of adversarial attacks, various techniques to improve adversarial transferability, and defensive models against them.

2.1. Adversarial Attacks

Given a clean image $\mathbf{x}$ with its true label y and a source model f, adversarial attacks inject inconspicuous noise $\boldsymbol{\delta}$ to craft adversarial examples $\mathbf{x}^{adv} = \mathbf{x} + \boldsymbol{\delta}$, causing the model to make erroneous predictions. The imperceptibility constraint of adversarial perturbation can be set differently, but the ℓ_∞ norm is widely used in existing works [1, 5, 30, 34, 37, 38]. Under the white-box setting, adversaries can utilize the gradient of the loss function (e.g., the cross-entropy loss) with respect to the input image for the optimization of an ℓ_∞-norm constrained targeted adversarial example towards a given target class y_t as follows:

$$\arg\min_{\mathbf{x}^{adv}} \mathcal{L}(f(\mathbf{x}^{adv}), y_t) \; s.t \; ||\mathbf{x} - \mathbf{x}^{adv}||_\infty \leq \epsilon, \quad (1)$$

where $\mathcal{L}$ is the adversary's objective loss for targeted attacks. One of the most fundamental attack methods for the above optimization is the Fast Gradient Sign Method (FGSM) [8], which utilizes the sign of the gradient. For targeted attacks, it updates $\mathbf{x}$ towards the direction of minimizing the classification loss to the given target class y_t in a single step as follows:

$$\mathbf{x}^{adv} = \mathbf{x} - \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(f(\mathbf{x}), y_t)), \quad (2)$$

where ϵ represents the perturbation bound. It can be further optimized by updating the image iteratively with a smaller step size, η, as in Iterative-FGSM [16].

2.2. Transfer-based Black-Box Attacks

Under the black-box setting, the target model's interior cannot be accessed, so the gradient of the image cannot be directly computed via the back-propagation technique. Therefore, adversaries need to craft adversarial examples on white-box surrogate source models that mimic the target model's function. After that, the attackers can attempt black-box attacks by feeding the generated adversarial examples to the target model.

However, the transfer success rate varies significantly depending on the difference between the source and target models, such as architectural differences. Therefore, for successful targeted attacks on black-box models, it is essential to improve the transferability of adversarial examples generated on surrogate models by preventing them from overfitting the source models. To this end, various techniques have been proposed to improve transferability based on the fundamental adversarial attacks explained in Section 2.1. These techniques include input diversification [1, 34, 38], gradient stabilization [4, 18], and use of different loss functions [13, 17, 32, 37]. In the following, we briefly introduce these approaches.

One of the representative methods for input diversification is the Diverse-Inputs (DI) method [34]. For each inference in iterative optimization, it randomly expands and pads the image with the probability p. The Resized-Diverse-Inputs (RDI) method [38] extends the DI technique by shrinking the expanded image to its original size at the end of the DI transform. Unlike DI, RDI always applies the image transform (i.e., $p = 1$).

The Translation-Invariant (TI) attack [5] blurs image gradients, approximating a weighted average of gradients from a set of translated images within a certain range. This technique provides a degree of translation invariance to the adversarial examples, making them more transferable between models. The Admix [31] further improves the transferability by mixing different images in the image domain. Specifically, according to the official implementation, Admix takes randomly shuffled images of the current batch, diminishing their pixel values by multiplying a mixing weight w, and adds them to the batch's images. Admix repeats the above addition-based mixup for N times and computes the average gradients.

The Object-based Diverse Input (ODI) method [1] is a recent technique that naturally diversifies inputs. This approach draws an input image on the surface of a randomly chosen 3D object and renders this painted object in various rendering environments. Empirical results demonstrate that ODI significantly improves the transferability of targeted adversarial examples, achieving state-of-the-art performance.

Stabilizing image gradients is another approach that can improve adversarial transferability by preventing adversarial examples from falling into local optima. The Momentum Iterative FGSM (MI-FGSM) [4] incorporates a momentum term in the iterative attacks. The Variance Tuning (VT) method [30] highlights gradient variance, the difference between the image's gradients and the average gradients of adjacent images. By minimizing gradient variance, VT can stabilize the update direction in the optimization process. Similarly, the Scale-Invariant (SI) attack method [18] scales down the pixel values of the input image in several steps and computes the gradients from the set of images. These techniques help to alleviate the overfitting of adversarial examples and thus improve transferability.

As another direction to improve transferability, several studies [17, 37] have suggested different loss functions for targeted attacks. Zhao *et al*. [37] point out that previous works use insufficient iterations to reach the optimal point in crafting adversarial examples. They empirically show that with large enough steps, significant performance improvement can be achieved with the following simple logit loss to increase the target class's logit.

$$\mathcal{L}_{logit}(f(\mathbf{x}^{adv}), y_t) = -\ell_t(f(\mathbf{x}^{adv})), \tag{3}$$

where ℓ_t is the logit value corresponding to the target class y_t.

2.3. Defensive Models

Several studies have also been conducted to build more robust models against transfer-based black-box attacks. One of the representative methods for defending against adversarial attacks is adversarial training [20, 29], which directly utilizes adversarial examples in training models. Another effective defense strategy is constructing an ensemble of individual networks, where adversaries should fool multiple models in the ensemble instead of a single model, thus improving robustness. To this end, an ensemble model is trained using the usual cross-entropy loss and some regularizing terms to consider the interaction among individual networks. Pang *et al*. [21] introduce the Adaptive Diversity Promoting (ADP) regularizer, which enhances the diversity among non-maximal predictions of individual members to make adversarial examples more difficult to transfer among them. Kariyappa *et al*. [14] propose the Gradient Alignment Loss (GAL) regularizer to misalign loss gradients, reducing the dimensionality of the shared adversarial subspace. DVERGE [35] trains sub-models to utilize a distinct set of features, isolating the adversarial vulnerability in each submodel.

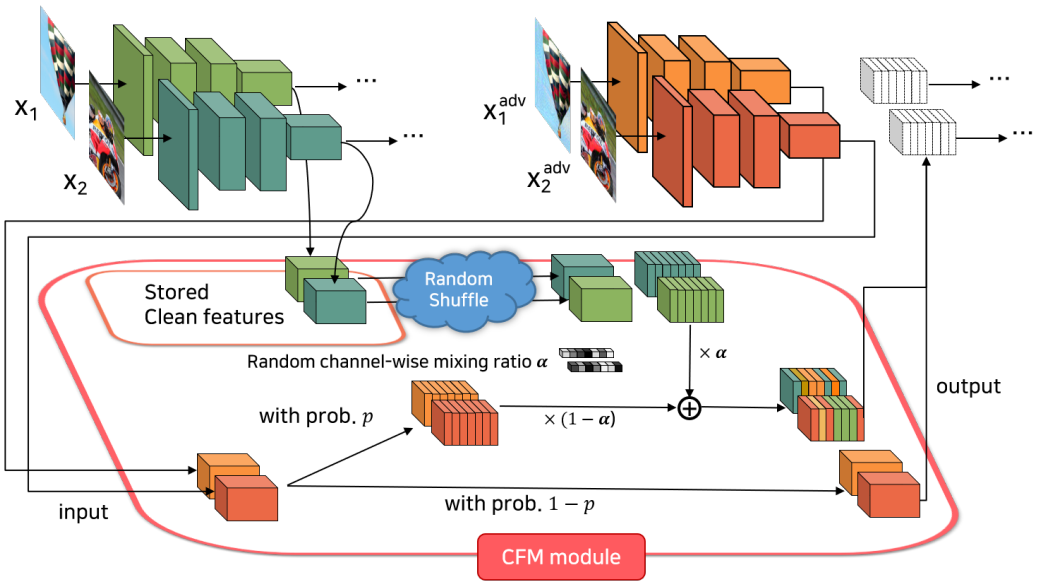

Figure 2. The detailed illustration of the internal process of the CFM module.

3. Clean Feature Mixup

The proposed Clean Feature Mixup is a method designed to enhance the transferability of targeted adversarial examples by efficiently emulating two types of competitors in the optimization process, as motivated in Section 1. In this section, we will first describe the implementation of CFM and then explain how CFM can simulate the two competitor noises in Section 3.3.

As an overview, the proposed CFM technique transforms the feature maps in their optimization to prevent adversarial examples from overfitting the source model. However, the feature space is much broader than the image space, and each model can have a different architecture, making it challenging to decide where and how to transform the features. Naively applying conventional image transforms to feature maps may excessively impede the optimization due to the significant domain gap between images and features. Instead, we transform the features by randomly mixing clean features with the input features. Specifically, for convolution and fully connected layers, it mixes the layers' outputs (i.e., features) with the stored clean features via linear interpolation [36].

To do this, it is necessary to store clean features in memory to mix them at inference. However, the structures of deep neural networks are not uniform, and they may have different modules. Our particular interest is to devise an off-the-shelf implementation to mix clean features while minimizing the effort of modifying the existing codes for the network architectures. Taking this into consideration, we design the CFM module that performs the above two functions (i.e., storing clean features and mixing them in the input features) and simply append the CFM modules to deeper convolution and fully connected layers as shown in Fig. 1. In the following, we describe each function of the CFM module in more detail.

3.1. Storing Clean Features

To mix the clean features in model inferences, the CFM modules require storing the clean features in the memory at first. To do this, it first converts a pre-trained source model f to the CFM-attached model f' by attaching the CFM modules to selected *conv* layers and all *fc* layers, and passes the clean image $\mathbf{x}$ to the converted model f'. Each CFM module stores the clean features in its memory at this first inference.

To avoid excessive disturbance in the optimization process, we do not append CFM modules to all convolution layers but only to deeper layers where the output size is significantly smaller than the input image. Mixing larger-sized, low-level features can cause excessive disturbance in the optimization process as they can vary significantly based on the input transforms. Therefore, we apply CFM modules only when the output's spatial size is less than or equal to $\frac{1}{16}$ of the original input size, which typically occurs after passing two pooling layers. We store pre-activated features since features can lose some information after passing through ReLU activations and apply feature mixup for these features.

3.2. Mixing Stored Clean Features with Input Features

As an overview, the internal process of a CFM module at the inference is depicted in Fig. 2. We will explain the internal functions in order.

Stochastic activation. Deep neural networks usually have tens to hundreds of layers, and applying clean feature mixup in all these layers at once can disrupt the inference process excessively. To address this issue, each CFM module stochastically applies the clean feature mixup with a probability p. This allows features to be mixed in a certain percentage of the total layers while maintaining randomness. Furthermore, this approach helps to obtain consistent performance gains regardless of the number of layers, thus reducing the effort required for hyperparameter tuning. From the perspective of competitor noises, this stochastic approach causes the influence of competitors to occur at random layers of the network.

Random feature shuffle. We also randomly shuffle the stored clean features on an image-wise basis within a batch. This allows the clean features of the image itself or those of another image to be mixed. Consequently, this enables the selection of competitor noises as either adversarial perturbations towards another target class or friendly perturbations towards the correct class, which is described in detail in Section 3.3.

Random channel-wise mixing ratio. Each CFM module mixes the stored clean features and the input features via linear interpolation, and for more randomness, the mixing ratio is randomly sampled for each channel. This allows the effects of competitor noises to vary arbitrarily across channels, enabling more diverse interference.

Mathematically, given a batch of B images, each CFM module stores B clean feature maps, denoted as $\mathbf{f}_1^c, \ldots, \mathbf{f}_B^c$, where $\mathbf{f}_i^c \in \mathbb{R}^{C\times H\times W}$ for $i = 1, \ldots, B$. The variables C, H, and W represent the number of channels, height, and width of the feature map, respectively. Then, each CFM module randomly mixes them with the input feature maps $\mathbf{f}_1, \ldots, \mathbf{f}_B \in \mathbb{R}^{C\times H\times W}$ at each inference as follows:

$$\mathbf{f}'_i = (1 - \boldsymbol{\alpha}_i) \odot \mathbf{f}_i + \boldsymbol{\alpha}_i \odot \mathbf{f}^c_{s_i},\ i = 1, \ldots, B, \quad (4)$$

where $\odot$ denotes element-wise multiplication and s_i is the i-th element of randomly shuffled indices (for image-level feature shuffling) and $\boldsymbol{\alpha}_i \in \mathbb{R}^{C\times 1\times 1}$ is the random channel-wise mixing ratio vector for the i-th image, where $\boldsymbol{\alpha}_i \sim \mathcal{U}(0, \alpha_{max})$, and $0 \leq \alpha_{max} \leq 1$. The channel-wise mixing ratios $\{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_B\}$ are sampled at each inference.

3.3. How Can CFM Improve the Transferability of Targeted Adversarial Examples?

The CFM modules randomly mix clean features of the image itself or another image into the input features, and this interference can further improve the transferability of the targeted adversarial examples in the following ways.

First, when an image's own clean features are mixed, this mixup suppresses the feature disturbance caused by the current targeted adversarial perturbations and guides the model's prediction back to the true class. In other words, it has the *opposite effect* on the targeted adversarial attack, encouraging adversarial perturbations to explore alternative feature disturbances for successful attacks.

Second, when clean features of another image are mixed, this introduces the effect of *targeted attacks on a different target class*[1]. Consequently, the adversarial examples should be optimized to induce the model to predict the given target class in the presence of other targeted attacks on different classes, prompting the adversarial perturbations to explore robustly adversarial feature disturbances to succeed in the attacks.

In summary, the interference from clean feature mixup can *effectively* and *efficiently* mitigate overfitting in the optimization of adversarial examples by preventing them from concentrating on specific features during their targeted attacks on the source model. The CFM method is compatible with many existing attack methods, and as an example, the pseudo-codes of the CFM-RDI-MI-TI method are described in Appendix.

4. Experiments

4.1. Experimental Settings

Datasets. Following previous works [1, 17, 37], we utilized the widely used ImageNet-Compatible dataset[2], which was released for the NIPS 2017 adversarial attack challenge. It has 1,000 299×299-sized images with their true and target classes for targeted attacks. We also leveraged the CIFAR-10 dataset [15] for targeted attacks on defensive models against transfer-based attacks. Specifically, we randomly sampled 1,000 images (100 images per class) from the test set and performed targeted attacks on randomly chosen incorrect target classes.

General settings. Most of our experimental settings followed the recent study [1, 37]. Specifically, we employed the commonly used ℓ_∞-norm perturbation constraint with $\epsilon = 16/255$ and set the step size $\eta = 2/255$ for the iterative attacks following [37]. All the methods, including CFM and baselines, optimize the adversarial examples based on the simple logit loss [1, 37]. To give sufficient iterations to optimize adversarial examples, we set the total iterations T to 300, which is also used in [1, 37].

Source and target models. We employed ten pre-trained neural networks as target networks: VGG-16 [24], ResNet-18 (RN-18) [10], ResNet-50 (RN-50) [10], DenseNet-121 (DN-121) [12], Xception (Xcep) [2], MobileNet-v2 (MB-v2) [23], EfficientNet-B0 (EF-B0) [28], Inception ResNet-v2 (IR-v2) [26], Inception-v3 (Inc-v3) [27], and Inception-v4 (Inc-v4) [26]. Additionally, we included an adver-

[1] Exceptionally, if the shuffled features to be mixed come from an image of the same class, it has the opposite effect on the targeted attack.

[2] https://github.com/cleverhans-lab/cleverhans/tree/master/cleverhans_v3.1.0/examples/nips17_adversarial_competition/dataset

Source : RN-50	Target model										
Attack	VGG-16	RN-18	RN-50	DN-121	Xcep	MB-v2	EF-B0	IR-v2	Inc-v3	Inc-v4	Avg.
DI	62.5	56.6	**98.9**	72.3	5.7	28.2	29.3	4.5	9.2	9.9	37.7
RDI	65.4	71.8	98.0	81.3	13.1	46.6	46.6	16.8	30.7	23.9	49.4
SI-RDI	70.5	79.8	98.8	88.9	29.5	56.2	66.2	37.9	56.4	43.6	62.8
VT-RDI	68.8	78.7	98.2	82.5	27.9	54.5	56.1	32.8	45.8	37.9	58.3
Admix-RDI	74.2	80.7	98.7	86.8	20.9	59.4	56.1	26.7	42.7	34.1	58.0
ODI	78.3	77.1	97.6	87.0	43.8	67.3	70.0	49.5	65.9	55.4	69.2
CFM-RDI	**84.7**	**88.4**	98.4	**90.3**	**51.1**	**81.5**	**78.8**	**48.0**	**65.5**	**59.3**	**74.6**
Source : Inc-v3	Target model										
Attack	VGG-16	RN-18	RN-50	DN-121	Xcep	MB-v2	EF-B0	IR-v2	Inc-v3	Inc-v4	Avg.
DI	2.9	2.4	3.4	5.0	1.9	1.8	3.7	3.0	**99.2**	4.2	12.8
RDI	3.5	3.8	4.0	7.0	3.1	3.0	5.9	6.3	98.7	7.1	14.2
SI-RDI	4.0	5.2	5.7	11.0	6.3	4.6	8.2	11.6	98.8	12.1	16.8
VT-RDI	5.9	8.9	9.4	13.2	7.4	5.9	9.8	12.3	98.7	14.7	18.6
Admix-RDI	6.3	6.5	8.8	12.8	6.0	6.1	10.9	12.2	98.7	13.6	18.2
ODI	14.3	14.9	16.7	32.3	20.3	13.7	25.3	26.4	95.6	31.6	29.1
CFM-RDI	**22.9**	**26.8**	**26.2**	**39.1**	**34.1**	**27.1**	**38.6**	**36.2**	95.9	**44.8**	**39.2**

Table 1. Targeted attack success rates (%) against ten target models on the ImageNet-Compatible dataset.

sarially trained RN-50 network (adv-RN-50) [33], which was trained with small ℓ_2-norm-constrained adversarial examples ($||\boldsymbol{\delta}||_2 \leq 0.1$), as it is effective in boosting the transfer success rate when used as a source model [25]. We also added five Transformer-based classifiers: Vision Transformer (ViT) [6], LeViT [9], ConViT [7], Twins [3], and Pooling-based Vision Transformer (PiT) [11]. For the CIFAR-10 dataset, we used various ensemble models composed of three ResNet-20 [10] networks (ens3-RN-20). They are trained under four defensive settings: standard training, ADP [21], GAL [14], and DVERGE [35]. The sources of the pre-trained model weights are described in Appendix.

Baseline attacks. We composed the baseline attacks using various combinations of eight existing techniques: DI [34], RDI [38], MI [4], TI [5], SI [18], VT [30], Admix [31], and ODI [1]. We applied MI and TI techniques to all attack methods, so we omitted 'MI-TI' when denoting them. Iteratively feeding fixed-size images can easily result in the overfitting of adversarial examples. Consequently, we opted for RDI as a common baseline technique in most cases. It is worth noting that, due to the computational intensity of ODI, we considered RDI as our primary baseline. We followed [1] for the detailed setup of DI, RDI, TI, and ODI. Specifically, the scale multipliers of image sizes were $1 \sim \frac{330}{299}$ and $\frac{340}{299}$ for DI and RDI, respectively. We set the convolution kernel size for TI to 5×5, the transformation probability for DI to 0.7, and the decay factor μ for MI to 1.0. For VT and SI, we set the number of samples and scales to 5, and β of VT to 1.5. For Admix, we set the mixing weight $w = 0.2$ and the number of images to be mixed $N = 3$ (i.e., $m_2 = 3$ in [31]) following the experimental settings of [31]. The original Admix settings utilize SI with five scale copies in its internal loops. However, using SI in internal loops of Admix makes it difficult to directly compare the performance improvement of an image-level mixup in Admix and a feature-level mixup in CFM. For that reason, we basically set the number of scale copies of SI inside Admix to 1 (i.e., $m_1 = 1$ in [31]). However, for comprehensive comparisons, we also included the results of Admix with the number of scale copies of 5 (i.e., $m_1 = 5$ in [31]) in the Appendix. For both Admix and CFM, we used a batch size of 20 for fair comparisons.

Settings for the CFM method. We set the channel-wise mixing ratio $\boldsymbol{\alpha}$ to be randomly sampled from $\mathcal{U}(0, 0.75)$. We set the mixing probability p to 0.1 and 0.25 for the ImageNet and CIFAR-10 datasets, respectively. Since the CIFAR-10 dataset has only ten classes, a larger value of p is required to maximize the effectiveness of CFM. In addition, since the CFM method consumes one inference for storing clean features, we deducted the available remaining iterations to 299 for strictly fair comparisons. However, adversaries may also eliminate the need for this one additional inference for CFM by omitting the input transform at the first iteration.

4.2. Experimental Results

First, we conducted targeted attack experiments with the ImageNet-Compatible dataset. We used pre-trained RN-50 and Inc-v3 as source models and evaluated the targeted attack success rates on the ten target models.

Transfer success rates. Table 1 shows the targeted attack success rates against the ten non-robust models of targeted adversarial examples generated from each source model. As shown in Table 1, CFM outperforms all baselines with a clear margin in all the source models. In particular, when

Source : RN-50	Target model							
Attack	adv-RN-50	ViT	LeViT	ConViT	Twins	PiT	Avg.	Computation time per image (sec)
DI	10.9	0.1	3.6	0.3	1.3	1.5	2.9	3.73
RDI	34.8	0.7	13.1	1.9	5.9	6.8	10.5	3.29
SI-RDI	59.9	2.9	29.4	6.3	15.5	17.9	22.0	16.16
VT-RDI	64.2	2.9	28.1	5.2	15.0	14.0	21.6	19.83
Admix-RDI	52.4	1.3	22.5	2.5	8.5	8.4	15.9	9.73
ODI	64.7	**5.1**	37.0	**10.7**	20.1	**29.1**	27.8	9.05
CFM-RDI	**75.5**	4.3	**46.1**	8.9	**25.2**	24.7	**30.8**	3.72
Source : adv-RN-50	Target model							
Attack	adv-RN-50	ViT	LeViT	ConViT	Twins	PiT	Avg.	Computation time per image (sec)
DI	**98.9**	5.7	36.9	10.1	19.2	20.5	31.9	3.77
RDI	98.8	10.8	49.5	19.9	29.4	35.8	40.7	3.29
SI-RDI	98.7	19.4	57.6	35.3	35.2	52.1	49.7	16.34
VT-RDI	98.5	10.6	46.3	20.0	27.1	34.4	39.5	19.83
Admix-RDI	**98.9**	12.1	55.5	23.1	32.4	38.9	43.5	9.86
ODI	97.3	22.2	57.7	38.8	40.0	54.9	51.8	9.04
CFM-RDI	98.3	**29.5**	**69.8**	**41.8**	**52.7**	**59.8**	**58.6**	3.74

Table 2. Targeted attack success rates (%) against a robust model and five Transformer-based classifiers with the ImageNet-Compatible dataset. We also report the average computation time to construct an adversarial example.

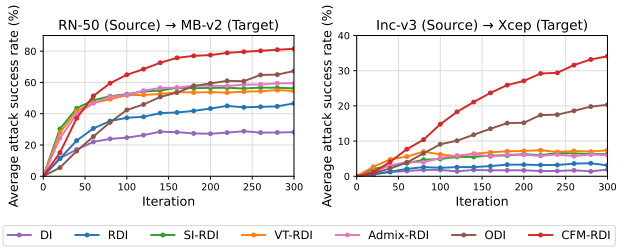

Figure 3. Targeted attack success rates (%) based on the number of iterations. Best viewed in color.

the source model is Inc-v3, the performance improvement of CFM is remarkable, increasing the average attack success rate by more than 10% over the second-best method.

Table 2 shows the attack success rates with two source models, RN-50 and adv-RN-50, against an adversarially trained and five Transformer-based models. The proposed CFM technique overwhelms all the baseline techniques. In particular, the adversarial examples generated from adv-RN-50 record an average targeted attack success rate approaching 60%. Figure 3 shows the average attack success rate according to iterations for two cases. It can be observed that CFM outperforms other techniques in average attack success rates and takes longer to saturate.

More experimental results with different source models can be found in Appendix. We also provide visualizations of the generated adversarial examples in the Appendix for qualitative comparisons.

Transfer-based attacks on the CIFAR-10 dataset. We also conducted transfer-based targeted attacks with the CIFAR-10 dataset. The primary purpose of this experiment is to evaluate the attack performance against four different ensembles of ResNet-20 models that were trained to be robust against transfer-based attacks. Table 3 reports the transfer-attack success rates for five non-robust models along with the four ensemble models. Due to the small size of CIFAR-10 images, we could not apply ODI for this experiment, and we excluded the image size reduction at the end of RDI. The defensive model trained with DVERGE obviously lowers the average attack success rate of RDI to 14.9%. Nevertheless, the CFM technique boosts the attack success rate for the DVERGE model from 14.9% to 59.3% and records an average attack success rate of 89.3%.

Computational cost. In addition to the transfer success rates, computation time is also an important factor to consider, as it indicates the efficiency of a technique. To demonstrate the efficiency of CFM, we describe the average computation time for generating an adversarial example in the rightmost column of Table 2 and Table 3. Since CFM modules add a marginal amount of computation, CFM-RDI increases only a small amount of computation time compared to other baselines. Note that each iterative attack was performed using a single NVIDIA Titan Xp GPU.

Combination with existing techniques. The CFM technique is compatible with many existing attack techniques. To demonstrate this, we attached Admix, SI, and VT to the CFM-RDI method. Due to space limitations, the experimental results are included in Appendix, but CFM showed further improved attack performance combined with other existing techniques.

Source : RN-50	Target model										
Attack	VGG-16	RN-18	MB-v2	Inc-v3	DN-121	ens3-RN-20				Avg.	Computation time per image (sec)
						Baseline	ADP	GAL	DVERGE		
DI	66.4	71.5	62.7	71.1	84.2	77.9	56.5	14.3	15.6	57.8	0.64
RDI	66.4	70.9	64.1	73.4	82.8	76.3	55.8	13.5	14.9	57.6	0.59
SI-RDI	72.9	76.3	77.1	77.0	84.7	81.2	65.5	20.0	22.4	64.1	3.17
VT-RDI	89.8	87.1	92.6	92.9	93.7	94.4	82.3	24.3	31.3	76.5	3.82
Admix-RDI	74.2	78.8	76.2	82.7	89.2	85.2	66.4	17.3	18.4	65.4	1.98
CFM-RDI	**98.3**	**97.7**	**99.0**	**99.0**	**99.2**	**98.8**	**97.2**	**54.9**	**59.3**	**89.3**	0.72

Table 3. Targeted attack success rates (%) against nine target models, including four ensemble-based defensive models on the CIFAR-10 dataset. We also evaluated the average computation time for crafting an adversarial example.

Ablation			Target model										
Mixing clean features	Channel-wise mixing ratio	Shuffle	Xcep	MB-v2	EF-B0	IR-v2	Inc-v4	ViT	LeViT	ConViT	Twins	PiT	Avg.
✓	✓	✓	**67.1**	82.4	83.4	64.7	67.4	**29.5**	**69.8**	**41.8**	**52.7**	**59.8**	**61.9**
✓	✓		57.8	76.1	77.8	58.2	60.1	23.4	64.9	38.9	46.6	52.8	55.7
✓		✓	65.0	**83.1**	**83.6**	**65.3**	**68.1**	26.6	69.4	40.5	50.6	57.5	61.0
✓			58.1	76.3	78.7	59.5	61.2	23.9	63.6	39.2	48.5	54.0	56.3
	✓	✓	63.2	81.2	81.9	61.2	66.9	25.9	67.0	39.2	48.8	56.9	59.2
		✓	63.1	82.0	82.9	62.4	65.9	24.4	68.0	39.1	46.8	56.1	59.1

Table 4. Targeted attack success rates (%) of CFM-RDI by ablating inner functions of the CFM modules. The source model is adv-RN-50.

4.3. Ablation Study

For an extensive ablation study, we investigated the range of mixing ratio $\boldsymbol{\alpha}$, mixing probability p, and the effect of internal functions of the CFM modules. For these ablation experiments, we carefully selected ten target models that are more difficult to disturb.

First, we evaluated how the transfer success rates vary by changing the values of the mixing probability p and the upper bound of mixing ratios α_{max}. In this experiment, we used adv-RN-50 as the source model and evaluated the transfer success rates on the ten target models, and detailed tabular experimental results can be seen in Appendix. CFM achieves the highest success rate when $p = 0.1$ and $\alpha_{max} = 0.75$, but it also achieves comparable attack success rates at other values. This indicates that CFM is not very sensitive to changes in hyperparameters and can consistently improve performance.

Next, we evaluated how the attack success rate varies by ablating each of the three internal functions of the CFM modules. Table 4 shows the results of this ablation experiment. It can be seen that each internal function helps to improve the transferability of adversarial examples. Mixing clean features without shuffling means mixing only the clean features of the image itself, and even this improves the average attack success rate by more than 10% compared to Admix (37.0%), which mixes different images in the image domain. Without using the channel-wise mixing ratio, $\boldsymbol{\alpha}_i$ becomes a scalar (i.e., α_i) rather than a vector. Not mixing clean features means that each CFM module uses the features of other images being optimized in the batch without storing clean features. Since the features of the images in the batch are already perturbed by other targeted attacks, utilizing them for feature mixup degrades performance improvement, demonstrating the importance of mixing clean features.

Lastly, we also investigate the impact of batch size when applying CFM. We evaluated CFM-RDI with several batch sizes, but we could not observe significant differences. Specifically, the average success rates of CFM-RDI with batch sizes 5, 10, 20, and 30 over the ten target models in Table 4 are 61.4%, 61.6%, 61.9%, and 61.0%, respectively.

5. Conclusion

In this paper, we proposed a novel approach to improve the transferability of targeted adversarial examples by introducing competition through the use of two types of competitor noises, which encourage the utilization of various features in attacks. Building upon this idea, we developed the Clean Feature Mixup (CFM) method, which efficiently simulates competitor noises in feature space by randomly mixing clean features of images in a batch. As CFM modules do not require extra backward passes, they require minimal computation, and this off-the-shelf model conversion-based method is easy to apply and compatible with many existing attacks. Our extensive experiments on the ImageNet-Compatible and CIFAR-10 datasets demonstrate that CFM outperforms existing baselines by a significant margin, highlighting the effectiveness and versatility of our proposed method.

References

[1] Junyoung Byun, Seungju Cho, Myung-Joon Kwon, Hee-Seon Kim, and Changick Kim. Improving the transferability of targeted adversarial examples through object-based diverse input. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15244–15253, 2022. 2, 3, 5, 6

[2] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 5

[3] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *Advances in Neural Information Processing Systems*, 34, 2021. 6

[4] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 3, 6

[5] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4312–4321, 2019. 2, 3, 6

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 6

[7] Stéphane d'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning*, pages 2286–2296. PMLR, 2021. 6

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1, 2

[9] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12259–12269, 2021. 6

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 5, 6

[11] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11936–11945, 2021. 6

[12] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 1, 5

[13] Qian Huang, Zeqi Gu, Isay Katsman, Horace He, Pian Pawakapan, Zhiqiu Lin, Serge Belongie, and Ser-Nam Lim. Intermediate level adversarial attack for enhanced transferability. *arXiv preprint arXiv:1811.08458*, 2018. 3

[14] Sanjay Kariyappa and Moinuddin K Qureshi. Improving adversarial robustness of ensembles with diversity training. *arXiv preprint arXiv:1901.09981*, 2019. 3, 6

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[16] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016. 3

[17] Maosen Li, Cheng Deng, Tengjiao Li, Junchi Yan, Xinbo Gao, and Heng Huang. Towards transferable targeted attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 641–649, 2020. 1, 3, 5

[18] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *International Conference on Learning Representations*, 2019. 2, 3, 6

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 1

[20] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 3

[21] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, pages 4970–4979. PMLR, 2019. 3, 6

[22] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1

[23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 5

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[25] Jacob Springer, Melanie Mitchell, and Garrett Kenyon. A little robustness goes a long way: Leveraging robust features for targeted transfer attacks. *Advances in Neural Information Processing Systems*, 34, 2021. 6

[26] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017. 5

[27] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE con-*

ference on computer vision and pattern recognition, pages 2818–2826, 2016. 5

[28] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 5

[29] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018. 3

[30] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1924–1933, 2021. 2, 3, 6

[31] Xiaosen Wang, Xuanran He, Jingdong Wang, and Kun He. Admix: Enhancing the transferability of adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16158–16167, 2021. 2, 3, 6

[32] Zhibo Wang, Hengchang Guo, Zhifei Zhang, Wenxin Liu, Zhan Qin, and Kui Ren. Feature importance-aware transferable adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7639–7648, 2021. 3

[33] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020. 6

[34] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2730–2739, 2019. 2, 3, 6

[35] Huanrui Yang, Jingyang Zhang, Hongliang Dong, Nathan Inkawhich, Andrew Gardner, Andrew Touchet, Wesley Wilkes, Heath Berry, and Hai Li. Dverge: diversifying vulnerabilities for enhanced robust generation of ensembles. *Advances in Neural Information Processing Systems*, 33:5505–5515, 2020. 3, 6

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

[37] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. On success and simplicity: A second look at transferable targeted attacks. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 2, 3, 5

[38] Junhua Zou, Zhisong Pan, Junyang Qiu, Xin Liu, Ting Rui, and Wei Li. Improving the transferability of adversarial examples with resized-diverse-inputs, diversity-ensemble and region fitting. In *European Conference on Computer Vision*, pages 563–579. Springer, 2020. 2, 3, 6