RaPA: Enhancing Transferable Targeted Attacks via Random Parameter Pruning

Tongrui Su Qingbin Li Shengyu Zhu* Wei Chen* Xueqi Cheng
State Key Lab of AI Safety, Institute of Computing Technology, Chinese Academy of Sciences
University of Chinese Academy of Sciences
{sutongrui25s, liqingbin24z, zhushengyu, chenwei2022, cxq}@ict.ac.cn

Abstract

Compared to untargeted attacks, targeted transfer-based attack still suffers from much lower Attack Success Rates (ASRs), although significant improvements have been achieved by kinds of methods, such as diversifying input, stabilizing the gradient, and re-training surrogate models. In this paper, we find that adversarial examples generated by existing methods rely heavily on a small subset of surrogate model parameters, which limits their transferability to unseen target models. Inspired by this finding, we propose Random Parameter Pruning Attack (RaPA), which introduces parameter-level randomization during the attack process. At each optimization step, RaPA randomly prunes model parameters to generate diverse yet semantically consistent surrogate variants. We show that this parameter-level randomization is equivalent to adding an importance-equalization regularizer, thereby alleviating the over-reliance issue. Extensive experiments across both CNN and Transformer architectures demonstrate that RaPA substantially enhances transferability. In the challenging case of transferring from CNN-based to Transformer-based models, RaPA achieves up to 11.7% higher average ASRs than state-of-the-art baselines (with 33.3% ASRs), while being training-free, cross-architecture efficient, and easily integrated into existing attack frameworks. Code is available on https://github.com/molarsu/RaPA.

1. Introduction

Deep neural networks have become prevalent in computer vision applications [9, 17, 19], but are highly vulnerable to maliciously crafted inputs, called adversarial examples [12, 42]. A major concern is their transferability, that is, adversarial examples generated using a white-box model can directly fool other black-box models, without any access to their architectures, parameters or gradients [14]. Since this type of attacks, usually referred to as transfer-based attacks, do not require any interaction with the target model, they pose severe security risks to real-world machine learning systems. Therefore, studying effective transfer-based attack methods is crucial to understand the vulnerabilities and further enhance model robustness.

This paper focuses on targeted transfer-based attacks with a *single surrogate model*, where the goal is to deceive black-box models to classify input images into a specific incorrect category. Due to the high complexity of decision boundaries, existing methods still have noticeably lower Attack Success Rates (ASRs) in the targeted setting than in the untargeted [2, 60]. A key observation is that the generated adversarial examples tend to overfit the surrogate model but fail to generalize to other models. To improve transferability, various strategies have been proposed. Observing that multiple surrogate models can help enhance transferability but in practice finding proper models for the same task is not easy [7, 30, 31, 59], model self-ensemble [20, 26, 34, 58] tries to create multiple models from an accessible model. Input transformation [1, 2, 27, 49, 51, 52, 56, 61] applies different transformations to inputs and diversify input patterns to reduce overfitting. A notable method is Clean Feature Mixup (CFM) [2], which randomly mixes high-level features with shuffled clean features. Building upon it, Feature Tuning Mixup (FTM) [27] introduces learnable and attack-specific feature perturbations, achieving new state-of-the-art performance in transferability. Despite these progresses, there is still much room for further improvement.

In this work, we take a different perspective and identify a previously overlooked cause of the poor transferability: the generated adversarial perturbations rely excessively on a small subset of parameters in the surrogate model, which limits their generalization to other models that have different parameter configurations. In other words, adversarial perturbation in existing methods tends to exploit a few "shortcut" parameters, leading to strong white-box performance but poor black-box transferability.

To mitigate this issue, we propose Random Parameter

*Corresponding author.

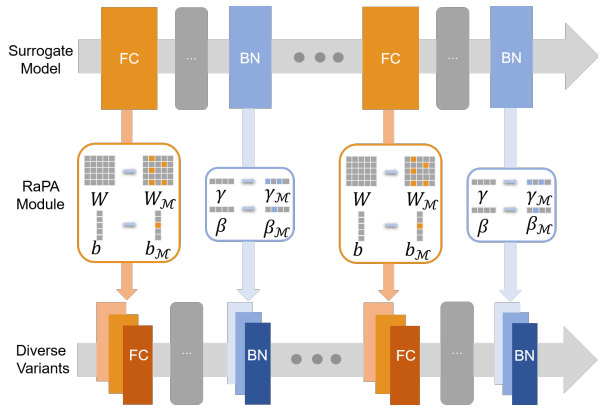


Figure 1. An illustration of the proposed method RaPA. We apply RaPA to selected layers in the surrogate model to create multiple and diverse variants at each iteration.

Pruning Attack (RaPA) that introduces parameter-level randomization into the attack process. At each optimization step, RaPA randomly prunes a subset of parameters in the surrogate model and uses multiple masked variants to update the adversarial example. We show that taking the expectation over such random masks is equivalent to adding an importance regularization term that aims to equalize parameter contributions, thus preventing over-reliance on a few dominant parameters. Conceptually, RaPA can be viewed as a self-ensemble method: each randomly pruned model represents a diverse yet semantically consistent variant of the surrogate. Previous self-ensemble approaches SASD-WS [54], MUP [58], and Ghost Network [26] rely on training-based model enhancement, deterministic pruning metrics, and structural perturbations, respectively. In contrast, RaPA is training-free, cross-architecture efficient, and straightforward to implement.

We evaluate the proposed method across various CNN- and Transformer-based target models, as well as against several defense methods. The experimental results show that RaPA outperforms other state-of-the-art methods. In particular, in the challenging scenario of transferring from CNN-based model to Transformer-based models, RaPA achieves 11.7% and 17.5% higher average ASRs with ResNet-50 [17] and DenseNet-121 [19] as surrogate models, respectively. Moreover, RaPA achieves the highest performance gain when scaling the compute for crafting adversarial examples. Specifically, with ResNet-50 as surrogate model, increasing the optimization iterations from 300 to 500 and number of forward-backward passes per iteration from 1 to 5 boosts the average ASR by 15.9%.

To summarize, our main contributions are as follows:

- We show that adversarial examples from existing transfer-based attacks rely heavily on a tiny subset of parameters in the surrogate model. Alleviating this over-reliance can in turn enhance the transferability of attack.
- We propose the RaPA, which introduces parameter-level randomization during attack optimization. We show, both intuitively and empirically, that random pruning implicitly equalizes parameter importance, acting as a regularizer to mitigate the over-reliance issue.
- Experiments across diverse surrogate and target models demonstrate that RaPA consistently outperforms existing methods. RaPA further benefits from increased computational budget, achieving larger improvements when scaling optimization iterations or inference steps.

2. Preliminary

This section introduces the background of adversarial attacks and briefly reviews related works on targeted transfer-based attacks. See Appendix A for more related work.

2.1. Background

Consider a classification task where the model is defined as a function $f : \mathbb{R}^n \to \mathcal{Y}$ that maps an input $x \in \mathbb{R}^n$ to a label in the set $\mathcal{Y}$ consisting of all the labels. Given a clean image x with its true label $y \in \mathcal{Y}$, untargeted attacks aim to find an adversarial example $x_{\text{adv}} \in \mathbb{R}^n$ that is similar to x but misleads the model to produce an incorrect prediction, i.e. $f(x_{\text{adv}}) \neq y$. Here 'similarity' is usually measured by an ℓ_p-norm, e.g., $\|x_{\text{adv}} - x\|_p \leq \epsilon$ where $\epsilon > 0$ is a predefined perturbation budget. For targeted attacks, the goal is to modify the model prediction to a particular target label y_{tar}, that is, $f(x_{\text{adv}}) = y_{\text{tar}} \neq y$. In this work, we will focus on targeted attacks.

In the white-box setting where the model is fully accessible, adversarial example can be obtained by the following:

$$\arg\max_{x_{\text{adv}}} \mathcal{L}\left(f(x_{\text{adv}})\right), \quad \text{s.t. } \|x_{\text{adv}} - x\|_p \leq \epsilon, \tag{1}$$

where $\mathcal{L}(\cdot)$ is a loss function (e.g., cross-entropy loss). The Fast Gradient Sign Method (FGSM) [12] uses the gradient direction to solve this problem and craft adversarial examples, while Iterative FGSM (I-FGSM) [23] extends this idea to an iterative scheme. In particular, at each iteration t, adversarial example is updated by adding a small perturbation:

$$x_{\text{adv}}^t = x_{\text{adv}}^{t-1} + \alpha \cdot \text{sign}\left(\nabla_{x_{\text{adv}}^{t-1}} \mathcal{L}\left(f(x_{\text{adv}}^{t-1})\right)\right), \tag{2}$$

where $x_{\text{adv}}^0 = x$ and $\alpha > 0$ is a step size. To make the generated adversarial example satisfy the perturbation budget constraint, a straightforward way is to project x_{adv}^t into the ϵ-ball of x.

2.2. Related Work

In black-box settings, the gradient information is not available and we only have limited access to the target model. Transfer-based method assumes that the adversarial examples generated on one model to mislead not only that model but also other models [42].

Method	DI	RDI	SI	Admix	ODI	BSR	CFM	RaPA
No pruning	98.2	98.7	98.7	98.1	98.9	98.5	98.0	98.2
Pruning bottom 0.5%	98.2	98.7	98.7	98.0	98.9	98.5	98.0	98.0
Pruning top 0.5%	16.0	28.6	31.9	29.9	19.9	36.7	51.3	64.5

Table 1. ASRs (%) before and after pruning the selected subsets of parameters in the surrogate model on the ImageNet-compatible dataset. Detailed experimental setting can be found in Section 4.1.

To improve transferability, many methods have been proposed. The first class, input-transformation techniques, applies a transformation $\mathcal{T}$ and uses $\nabla_{x_{\text{adv}}^{t-1}} \mathcal{L}\left(f(\mathcal{T}(x_{\text{adv}}^{t-1}))\right)$ as the gradient. Diverse Inputs (DI) [56] and its variant Resized DI (RDI) [61] apply random transformations to increase input variation during optimization. Translation-Invariant (TI) [8] averages gradients over translated inputs to reduce location sensitivity. Structure Invariant Attack (SIA) [52] and Block Shuffle and Rotate (BSR) [49] both perform block-level local transformations, with SIA applying diverse transforms and BSR focusing shuffle and rotate operations. Object-based DI (ODI) [1] generates adversarial examples rendered on 3D objects, while Admix [51] mixes inputs with random samples from other classes. CFM [2] extends it to the feature space with competing noises, while FTM [27] further adds learnable, attack-specific perturbations to achieve state-of-the-art transferability.

The second class focuses on stabilizing gradient updates to improve transferability. Momentum Iterative FGSM (MI-FGSM) [7] incorporates a momentum term into I-FGSM to help avoid local optima. Scale-Invariant (SI) optimization [29] improves transferability by applying perturbations across multiple scaled copies of the input, leveraging the scale-invariance property of deep models.

Beyond the above methods, another type of approaches re-train the surrogate model to enhance transferability, e.g., DSM [57] and SASD-WS [54] improve model generalization and transferability through knowledge distillation or sharpness-aware self-distillation.

Closely related to the present work is self-ensemble [20, 26, 34, 58], which creates multiple models from only one surrogate model. The self-ensemble method in [34] specifically considers vision Transformer as surrogate model and is denoted as SE-ViT in this paper. Ghost Network [26] perturbs surrogate model to create a set of new models and then samples one model from the set at each iteration. Masking Unimportant Parameters (MUP) [58] drops out unimportant parameters according to a predefined Taylor expansion-based metric, while Diversity Weight Pruning(DWP) [48] only prunes the parameters with small absolute values.

However, the inherent differences w.r.t. model architecture, parameter setting, and training procedure between the surrogate and target models still limit the effectiveness of transfer-based methods on certain models and datasets. In the next section, we show that there is a key aspect that renders the current over-fitting issue of transfer-based methods.

3. Method

In this section, we first conduct a pilot study to show a key aspect of the overfitting issue in existing transfer-based methods and then propose a random masking based approach. Comparison with related methods is also discussed.

3.1. Motivation

We observe that the adversarial perturbations generated by solving Problem (1) tend to rely heavily on a small subset of parameters in the surrogate model. These parameters may stem from specific training schemes, datasets, or architectural choices. As a result, adversarial examples that strongly depend on these parameters often fail to generalize and mislead other models. Even with state-of-the-art transfer-based attack methods, this issue remains a key factor that can lead to the failure of adversarial example transfer.

To quantify the phenomenon, we conduct a pilot study using the framework of Optimal Brain Damage (OBD) [24, 33], which quantifies the importance of each model parameter from the perspective of sensitivity analysis. Specifically, given an adversarial example x_{adv} and a loss function $\mathcal{L}(\cdot)$. Let θ represent the entire set of model parameters. The importance of a parameter θ_i in the surrogate model f is computed as:

$$\mathcal{I}(\theta_i) = \frac{\partial^2 \mathcal{L}\left(f(x_{\text{adv}})\right)}{\partial \theta_i^2} \times \theta_i^2. \tag{3}$$

This metric reflects how much the loss would change if a parameter θ_i were removed, and can serve as a proxy for its contribution to the effectiveness of the adversarial example.

Next, we consider pruning two distinct subsets of the surrogate model's parameters based on this importance metric: the top 0.5% most important and the bottom 0.5% least important parameters(see Appendix B.1 for details). For each adversarial example, we instantiate the model, prune the selected subset, and evaluate the ASR on the resulting model. Table 1 reports the ASRs after pruning the two subsets of model parameters. Here we use ResNet50 as the surrogate model and detailed setting can be found in Section 4.1.

We observe that pruning the most important parameters leads to a drastic drop in ASR—more than 46%, whereas pruning the least important parameters yields negligible impact. This observation suggests that adversarial examples

generated by existing methods are highly dependent on the most important parameters, validating our observation on the over-reliance issue. As such, how to further alleviate this strong dependence on specific parameters would be a key to improving the transferability of adversarial examples over existing transfer-based attack methods.

3.2. Alleviating Over-reliance via Random Parameter Pruning

As per the pilot study, a direct approach to improving transferability would be to mask the most important parameters at each optimization step, thereby mitigating the over-dependency on them. However, accurately identifying important parameters requires computing second-order derivatives, which is computationally expensive for all parameters. Although we can approximate them with first-order terms, masking the most important parameters typically causes the surrogate model's capacity to degrade rapidly, and the resulting adversarial examples may fail to fool the target model—and even the original surrogate model itself (See Appendix D.1 for further explanations). To address this problem, we propose to apply random parameter pruning to the surrogate model at each optimize step. This approach avoids expensive computations while achieving the goal of reducing over-reliance on specific parameters.

Intuition and Theoretical Explanation Our core idea is that randomly pruning parameters at different optimization steps encourages the generated adversarial examples to be less dependent on particular parameter subsets. This in turn improves transferability across different target models.

We define a random binary mask $\mathcal{M} \in \{0,1\}^{|\theta|}$, where each entry is independently sampled from a Bernoulli distribution: $\mathcal{M}_i \sim \text{Bernoulli}(1-p)$. Here $p \in [0,1]$ is the probability of masking a parameter. With a small p, we would have $\mathbb{E}[\mathcal{M}_i] \approx 1$. Then the parameter of the model used in the forward pass becomes $\mathcal{M} \odot \theta$, where $\odot$ denotes element-wise multiplication.

Under this setup, the expected loss over random masks can be approximated using a second-order Taylor expansion:

$$\begin{aligned}&\mathbb{E}_{\mathcal{M}}[\mathcal{L}(f(x_{\text{adv}}; \mathcal{M} \odot \theta))] \\ &\approx \mathcal{L}(f(x_{\text{adv}}; \theta)) + \frac{p(1-p)}{2} \sum_i \frac{\partial^2 \mathcal{L} f(x_{\text{adv}}; \theta)}{\partial \theta_i^2} \theta_i^2, \quad (4)\end{aligned}$$

which is sum of the original loss plus an importance penalty. Minimizing this objective while resampling the mask at each step would force the adversarial example to distribute the importance over all parameters, making it more robust to different parameters and thus more transferable.

Practical Implementation with DropConnect The above random parameter pruning method is similar to DropConnect method [47] in training neural networks. We notice that DropConnect is mainly effective in terms of linear layers . We thus apply DropConnect to the weight and bias parameters of linear layers as well as the transformation parameters of normalization layers. Both types of layers are widely used in mainstream architectures including Transformer [46]. Empirically, our ablation study in Section 4 validates the effectiveness of this choice, compared with convolutional layers.

We now present our attack method, Random Parameter Pruning Attack (RaPA), as summarized in Algorithm 1. Take linear layer for example. We perform independent random masking onto the weight and bias (if present) parameters using Bernoulli sampling. Specifically, for surrogate model f, let $W \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$ denote the weight matrix and $b \in \mathbb{R}^{d_{\text{out}}}$ the bias vector associated with a linear layer. Here d_{in} and d_{out} represent the input and output dimensions, respectively. The corresponding masks for the weight matrix and bias are

$$\mathcal{M}_w \sim \text{Bernoulli}(1-p_w), \mathcal{M}_b \sim \text{Bernoulli}(1-p_b), \quad (5)$$

where $\mathcal{M}_w \in \{0,1\}^{d_{\text{in}} \times d_{\text{out}}}$ and $\mathcal{M}_b \in \{0,1\}^{d_{\text{out}}}$ are the random masks, and $p_w, p_b \in [0,1]$ are DropConnect probabilities. Then the masked parameters are computed as

$$W_{\mathcal{M}} = \mathcal{M}_w \odot W, \quad b_{\mathcal{M}} = \mathcal{M}_b \odot b, \quad (6)$$

where $\odot$ denotes the element-wise multiplication. For normalization layer, the same operation is applied similarly to the transformation parameters. The random masks are sampled for each selected layer.

The random masks $\mathcal{M}_w$ and $\mathcal{M}_b$ in Eq. (5) are regenerated for each inference, producing diverse variants of the surrogate model with different parameters if we conduct multiple inferences at an iteration. In addition, RaPA can be naturally integrated with existing input transformation and gradient stabilization methods for crafting adversarial examples, as shown in Lines 5 and 8 in Algorithm 1.

Analyzing Parameter Importance with Gini Coefficient To further verify that our random parameter pruning strategy indeed mitigates the over-reliance on a few dominant parameters, we employ the Gini coefficient to measure the distribution of parameter importance across layers. A lower Gini value indicates a more uniform distribution of importance, implying that the adversarial perturbation depends less on specific parameters and generalizes better to unseen models. The formal definition and detailed computation process of the Gini coefficient are provided in Appendix B.3.

We compute the Gini coefficients based on the parameter importance values $\mathcal{I}(\theta_i)$ defined in Eq. (3). The overall and layer-wise results are summarized in Table 2. As

Method	DI	RDI	SI	Admix	ODI	BSR	CFM	RaPA
All Layer Average	0.32	0.30	0.21	0.12	0.33	0.25	0.19	**0.08**
Conv Layer	0.11	0.07	0.03	0.01	0.12	0.05	0.03	**0.00**
Norm Layer	0.51	0.52	0.37	0.22	0.53	0.44	0.34	**0.15**
Linear Layer	1.00	0.86	0.59	0.18	1.00	0.75	0.55	**0.13**

Table 2. Gini coefficients of parameter importance across different layers and methods. Lower values correspond to more uniform parameter importance.

Algorithm 1 Random Parameter Pruning Attack(RaPA)

Input: Classifier f; clean image x; loss function $\mathcal{L}(\cdot)$; max iterations T; ℓ_p bound ϵ; number of inferences per iteration S; DropConnect probabilities p_w, p_b; linear and normalization layers $\mathbb{L}$; input transformation $\mathcal{T}$.

Output: Adversarial example x_{adv}

$x_{\text{adv}}^0 \leftarrow x$
for $t = 1 \rightarrow T$ **do**
  for $s = 1 \rightarrow S$ **do**
    Obtain modified model $f_{\mathcal{M}}$ by applying RaPA to each layer in $\mathbb{L}$ according to Eqs. (5) and (6).
    $g_s^t \leftarrow \nabla_{x_{\text{adv}}^{t-1}} \mathcal{L}\left(f_{\mathcal{M}}(\mathcal{T}(x_{\text{adv}}^{t-1}))\right)$
  end for
  $g^t \leftarrow \frac{1}{S}\sum g_s^t$
  Update x_{adv}^t with gradient g^t using iterative methods (like MI-FGSM [7]).
  Project x_{adv}^t into the ϵ-ball of x.
end for
return x_{adv}^T

shown in Table 2, RaPA achieves the lowest Gini coefficients among all compared methods, suggesting that it effectively flattens the importance distribution and suppresses the over-concentration of sensitivity on a few parameters. This balanced importance allocation leads to improved robustness and transferability across different architectures.

3.3. Discussion

In this section, we compare RaPA with related self-ensemble methods in more details.

RaPA was proposed to reduce over-dependence on specific parameters, and it turns out to be a self-ensemble method that constructs multiple new models at each iteration. Existing self-ensemble method [20] targets object detection task, which is different from ours. SE-ViT [34] is specifically designed for vision Transformer surrogate models; as shown in Section 4, its ASR is lower than RaPA even when using ViT as surrogate model. More closely related are Ghost Network [26] and MUP [58], but are much outperformed by RaPA (c.f. Appendix D.1 and Section 4.2).

We now analyze the effectiveness of RaPA from model ensemble perspective. It has been hypothesized that an adversarial image that remains adversarial for multiple models is more likely to transfer to other models [31]. RaPA generates independent random masks for each selected layer and also at each optimization iteration. In this sense, it brings in more randomness and further diversification than Ghost Network, MUP and DWP. Specifically, Ghost Network perturbs only skip connections for residual networks (like ResNet-50), while MUP and DWP mask unimportant parameters according to a predefined metricor at each iteration. This observation is also in accordance with [3, 20, 30], which show that increasing the number of surrogate models generally enhances the transfer attack performance. On the other hand, from the perspective of ensemble techniques in machine learning, each variant model should also be informative about or useful to the targeted task of image classification [3, 20]. As empirically shown in Appendix D.5, RaPA achieves a good tradeoff in terms of model diversity and utility, thereby enhancing the attack performance.

4. Experiments

This section empirically validates the effectiveness of our method, using both CNN- and Transformer-based models.

4.1. Experimental Settings

Dataset We utilize the ImageNet-compatible dataset [22], served as the official dataset for the NIPS 2017 Attack Challenge. This dataset contains both ground-truth and targeted labels, making it well-suited for targeted-attack.

General Setting We adopt the ℓ_∞-norm as the constraint on perturbation, with budget $\epsilon = 16/255$. The learning rate is chosen as $\alpha = 2/255$. We use the Logit loss [60] as our objective function in Equation (1). By default, we set the maximum number of optimization iterations to $1,000$ and the batch size to 32 for all baseline methods, to ensure sufficient optimization. Furthermore, we notice that different attack methods may take different amounts of computations per iteration. We hence fix the same number of inferences for each optimization iteration to maintain fair comparisons across all attack methods.

Surrogate and Target Models Our experiments choose various models commonly used in the literature [2]. These include **1)** CNN-based models: VGG-16 [41], ResNet-18 (RN18) [17], ResNet-50 (RN50) [17], DenseNet-121 (DN121) [19], Xception (Xcep) [4], MobileNet-v2 (MBv2)

Attack	Source: RN50							Source: DN121						
	ViT	LeViT	ConViT	Twins	PiT	CLIP	Avg.	ViT	LeViT	ConViT	Twins	PiT	CLIP	Avg.
DI	0.4	6.7	0.6	3.8	1.8	0.5	2.3	0.3	4.0	1.2	2.0	2.1	0.3	1.7
RDI	2.8	24.0	4.4	12.6	10.1	1.2	9.2	1.0	12.0	2.1	6.9	8.2	1.4	5.3
SI	8.0	42.9	7.7	25.1	23.3	3.7	18.4	3.9	21.5	4.4	10.2	14.3	2.1	9.4
SIA	3.1	28.4	3.9	16.7	13.5	1.9	11.2	2.4	24.2	2.2	12.9	11.1	1.5	9.0
BSR	6.8	42.4	7.7	25.3	21.9	2.3	17.7	2.7	21.6	2.7	10.6	11.9	1.2	8.5
DWP	3.7	32.0	4.0	17.6	13.3	1.8	12.1	2.7	21.1	2.9	14.3	12.0	2.4	9.2
Admix	7.2	43.4	5.9	22.7	19.4	4.2	17.1	4.2	33.8	4.0	18.0	19.1	3.9	13.8
ODI	15.5	49.3	11.8	31.6	35.1	5.6	24.8	8.5	36.3	8.4	19.8	26.7	4.9	17.4
MUP	7.0	48.3	8.2	30.7	26.5	3.6	20.7	5.2	38.9	5.0	23.2	20.1	4.1	16.1
CFM	17.3	65.8	14.6	47.5	39.9	7.9	32.2	10.4	49.0	8.3	30.1	30.4	5.5	22.3
FTM	18.0	67.6	16.5	47.1	41.5	9.3	33.3	10.7	50.4	9.4	31.1	30.4	4.8	22.8
RaPA	**33.8**$_{\pm 1.0}$	**75.4**$_{\pm 1.8}$	**27.6**$_{\pm 0.2}$	**59.5**$_{\pm 0.4}$	**57.3**$_{\pm 0.7}$	**15.6**$_{\pm 0.1}$	**45.0**	**27.8**$_{\pm 0.1}$	**69.4**$_{\pm 0.6}$	**23.5**$_{\pm 0.2}$	**53.1**$_{\pm 0.6}$	**54.0**$_{\pm 0.1}$	**14.1**$_{\pm 0.0}$	**40.3**

Table 3. ASRs (%) against five Transformer-based target models on the ImageNet-Compatible dataset. All the attack methods are combined with MI-TI. The best results are shown in bold and the second best results are underlined.

[40], EfficientNet-B0 (EFB0) [45], Inception ResNetv2 (IRv2) [44], Inception-v3 (Incv3) [43], and Inception-v4 (Incv4) [44]; and **2)** Transformer-based models: ViT [9], LeViT [13], ConViT [10], Twins [5], and Pooling-based Vision Transformer (PiT) [18]. All the models are pre-trained on the ImageNet dataset [6] . Additionally, we include CLIP [37], trained on 400 million text-image pairs, to evaluate the transferability across different modalities. When using Transformer-based models, the input image is resized to 224×224 to meet the model input requirement.

Baseline Attack Methods We compare RaPA with various existing transfer-based methods, including DI [56], RDI [61], SI [29], Admix [51],SIA [52], BSR [49], ODI [1], and CFM [2]. We also include two existing self-ensemble methods, namely, MUP (whose implementation only handles CNN layers) [58] and SE-ViT (which is specifically designed for vision Transformers) [34]. These methods are primarily used in combination with TI-FGSM [8] and MI-FGSM [7] during the optimization process. It is worth noting that some of these methods, namely, SI, BSR, Admix, CFM, MUP and SE-ViT, are implemented together with RDI, which have been reported to obtain higher transfer ASRs [2]. As previously mentioned, the attack methods are configured with an identical number of inferences per iteration, denoted as S. Specifically, we pick S scaled copies in SI and in the inner loop of Admix, and S transformed images for BSR . For other baselines, we perform S forward-backward passes and use the average gradient on x_{adv} to update the adversarial example in each iteration. In the main experiment, we will set $S = 5$; comparison of other choices is studied in Section 4.3.

RaPA Setting While the DropConnect probabilities can be chosen differently for the weight and bias parameters in a linear layer (or the transformation parameters in the normalization layer) and also across different layers, we choose the same probability for all selected parameters, that is, $p_w = p_b = p$, which greatly simplifies the implementation. Through our ablation study, we find that RaPA performs well across a range of probabilities. For our experiments, we will select the following DropConnect probabilities: 0.05 for ResNet-50, 0.02 for Inception-v3, 0.04 for DenseNet-121, 0.01 for Vision Transformer, and 0.03 for CLIP. By default, RaPA applies DropConnect to all linear and normalization layers in the surrogate model.

4.2. Main Result

We first study the performance of RaPA on the ImageNet-Compatible dataset. We employ ResNet-50, Inception-v3, DenseNet-121, and ViT as surrogate models, and evaluate the obtained adversarial examples on 16 target models.

Table 3 reports the experimental results when adversarial examples are generated using CNN-based models and transferred to Transformer-based neural networks. This task is considered more challenging in the context of transfer-based attacks [32], as the ASRs in this case are relatively low. Our method significantly improves the attack performance over existing methods: it increases the average ASR from 33.3% to 45.0% with ResNet-50 as surrogate model, and from 22.8% to 40.3% with DenseNet-121.

Table 4 reports the ASRs of various attack methods on ten CNN-based target models. RaPA achieves the best average ASR. Particularly, with Inception-v3 as surrogate model, the ASRs are increased by 14.6% and 20.7% for the challenging target models VGG16 and MBv2, respectively. When transferring from Transformer-based model ViT to CNN-based models, RaPA again attains the best average ASR 51.2%. We also report the results of self-ensemble methods MUP [58] and SE-ViT [34]. RaPA clearly outperforms these two methods by a large margin.

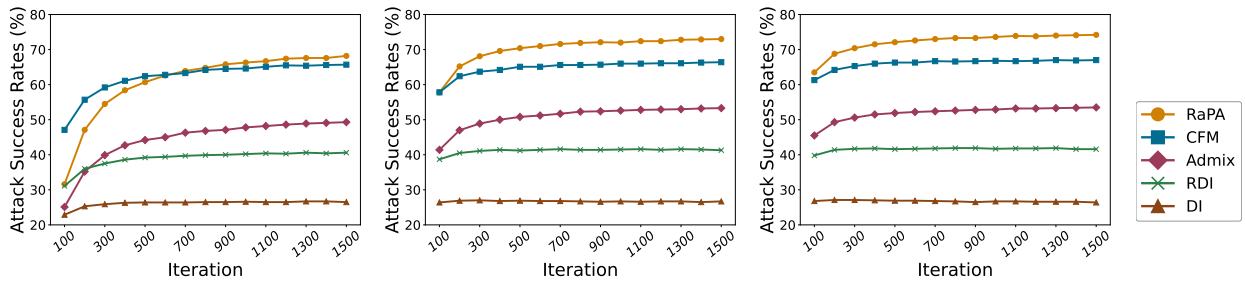
Additional experimental results can be found in Appendix D.4. We also visualize the heatmaps of some adversarial examples in Appendix C for qualitative comparison.

4.3. Ablation Study

In this section, we conduct an ablation study to investigate the impacts of 1) different types of layers where DropConnect is applied, 2) different DropConnect probabilities and 3) more iterations and inferences.

Source : Incv3	Target model										
Attack	RN18	RN50	VGG16	Incv3	EFB0	DN121	MBv2	IRv2	Incv4	Xcep	Avg.
DI	2.2	3.9	3.4	99.1	3.6	5.0	1.2	7.7	8.9	7.0	14.2
RDI	5.8	5.5	3.9	99.0	8.0	8.5	3.8	18.6	18.8	11.1	18.3
SI	6.7	6.7	4.3	98.8	9.7	9.7	4.4	23.3	22.1	13.6	19.9
MUP	13.9	13.6	9.6	98.4	17.7	22.4	8.1	42.2	42.2	26.4	29.5
BSR	15.8	13.9	11.9	98.7	20.5	24.3	9.6	45.7	45.5	30.3	31.6
DWP	17.5	17.2	13.5	99	19.2	29.8	9.7	54.4	52.9	37.5	35.1
Admix	18.5	16.7	13.8	98.1	23.8	27.5	15.9	46.3	47.1	38.9	34.7
SIA	17.4	21.5	16.7	98.8	27.2	32.2	13.1	56.1	59.2	42.9	38.5
ODI	14.4	22.3	22.0	**99.4**	26.0	39.5	13.9	51.8	60.7	44.7	39.5
CFM	37.4	37.9	27.3	97.9	46.1	53.0	27.8	76.9	76.1	68.2	54.9
RaPA	**51.3**$_{\pm 0.6}$	**53.5**$_{\pm 1.0}$	**41.9**$_{\pm 0.6}$	97.4$_{\pm 0.0}$	**60.8**$_{\pm 0.3}$	**68.4**$_{\pm 1.4}$	**48.5**$_{\pm 0.0}$	**86.7**$_{\pm 0.3}$	**87.5**$_{\pm 0.4}$	**84.0**$_{\pm 0.0}$	**68.0**
Source : ViT	Target model										
Attack	RN18	RN50	VGG16	Incv3	EFB0	DN121	MBv2	IRv2	Incv4	Xcep	Avg.
DI	0.5	1.0	0.8	2.1	2.1	1.1	1.3	1.8	1.6	1.4	1.4
RDI	1.7	2.4	1.3	4.1	6.3	4.1	2.7	5.8	4.9	4.6	3.8
SI	2.9	3.9	1.1	8.0	9.2	5.9	2.7	7.9	6.2	6.8	5.5
BSR	5.0	8.2	3.9	11.2	15.0	12.8	5.3	15.5	13.4	11.2	10.2
DWP	13.5	11.7	7.1	15.8	22.2	16.7	10.2	17.4	16.3	16.3	14.7
SE-ViT	8.9	12.0	6.9	21	25.3	20.5	8.3	23.8	22.5	20.3	17.0
Admix	16.4	20.4	13.2	31.8	38.3	31.5	17.8	35.3	31.9	29.8	26.6
SIA	3.6	4.8	3.0	8.1	12.2	8.4	3.5	8.9	10.1	8.3	7.1
ODI	12.4	20.0	10.6	28.3	28.9	30.9	10.4	35.5	34.4	27.9	23.9
CFM	26.1	33.4	18.0	45.2	56.8	47.3	23.2	54.5	49.9	46.2	40.1
RaPA	**37.2**$_{\pm 0.7}$	**42.9**$_{\pm 0.7}$	**28.6**$_{\pm 2.0}$	**57.5**$_{\pm 1.4}$	**68.2**$_{\pm 0.4}$	**56.8**$_{\pm 0.2}$	**36.4**$_{\pm 0.7}$	**62.9**$_{\pm 0.4}$	**62.6**$_{\pm 0.1}$	**58.3**$_{\pm 1.0}$	**51.2**

Table 4. ASRs (%) against ten target models on the ImageNet-Compatible dataset. All the attack methods are combined with MI-TI. The best results are shown in bold and the second best results are underlined.



(a) $S = 1$ (b) $S = 5$ (c) $S = 10$

Figure 2. Average ASRs along optimization iterations. Here S denotes the number of inferences per iteration.

Different Types of Layers We use ResNet-50 and ViT as surrogate models to analyze the impacts of layer types. For ResNet-50, we apply RaPA to different combinations of Batch Normalization (BN) layer, Fully Connected (FC) layer, and Convolutional (Conv) layer. Similarly, we consider Layer Normalization (LN) and FC layers (including linear transformation layer in the attention layer) for ViT.

Table 5 presents the experimental results. Notably, simply applying RaPA to all layers achieves equal or higher ASRs, compared with other baselines. The combination of BN (or LN) and FC layers performs the best, which validates our implementation in Section 3. For ResNet-50, applying DropConnect to Conv layers performs worse than BN layers. We conjecture that Conv layers have sparser weights and may be less affected by over-reliance issue. Applying DropConnect only to FC layers yields particularly low ASRs, as ResNet contains only a single FC layer.

DropConnect Probability We investigate the impact of varying DropConnect probabilities p using ResNet-50 as a surrogate model. RaPA is run with p ranging from 0.01 to 0.09, and we report the average ASRs over 16 models in Appendix D.3. The mean ASR is 66.3% with a standard deviation of 5.9%, peaking at 72.4% when $p = 0.05$. Notably, with $p \in [0.03, 0.07]$, RaPA consistently outperforms baselines by over 2%, underscoring the stability of the proposed method across different choices.

More Iterations and Inferences We study performance under different total iterations and numbers of inferences per iteration, denoted as T and S, respectively. We use ResNet-50 as the surrogate model and evaluate how well adversarial examples transfer to the 16 target models.

Source Model	BN	FC	Conv	ASRs (%)
RN50	✓			72.1
		✓		41.1
			✓	65.1
	✓	✓		**72.4**
	✓		✓	69.1
		✓	✓	64.7
	✓	✓	✓	69.2
Source Model	**LN**	**FC**	**-**	**ASRs (%)**
ViT	✓			58.6
		✓		63.9
	✓	✓		**65.2**

Table 5. Applying DropConnect to different types of layers. The reported ASRs(%) are averages over 16 target models.

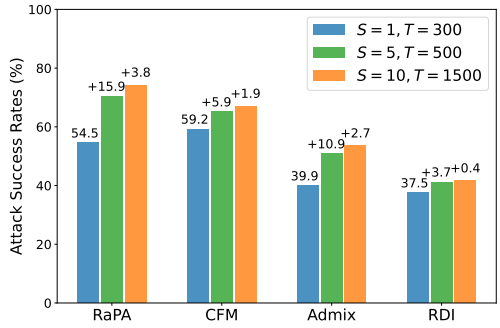


Figure 3. Average ASRs with different iterations (T) and different numbers of inferences per iteration (S).

The results are reported in Figure 2. Although existing methods may also benefit from additional optimization iterations, RaPA and Admix have the best gains when T increases, while RaPA achieves a much higher ASR than Admix. With S increasing, RaPA can outperform CFM even at an early stage of the optimization process. We also depict Figure 3 to ease the comparison of the gains of different methods when both T and S increase. As we observe, RaPA benefits the most from an additional compute budget.

4.4. Attack Performance Against Defenses

We evaluate RaPA against several defenses: adversarially trained ResNet-50 (advRN) [39], Ensemble-Adversarial-Inception-ResNet-v2 (ensIR) [21], High-level representation Guided Denoiser (HGD) [28], Bit Depth Reduction (Bit) [15], JPEG compression [15], R&P [55], and Diffpure [35]. We utilize ResNet-50 as the surrogate model. For Bit, JPEG, and R&P, the target model is ResNet-18 and for Diffpure, the target model is ResNet-50. As shown in Table 6, RaPA outperforms all other baselines. Notably, against the strong defenses ensIR and HGD, RaPA exceeds the second-best ASRs by 29.4% and 10.5%, respectively.

RN50	Defense methods						
Method	advRN	ensIR	JPEG	Bit	R&P	HGD	Diffpure
DI	10.6	0.0	26.7	50.5	41.1	0.1	0.0
RDI	39.6	0.8	58.8	75.8	71.3	0.8	0.0
SI	61.8	9.4	75.1	80.7	78.8	2.1	0.4
Admix	68.9	5.7	76.9	81.4	78.6	2.4	0.7
BSR	60.4	3.0	76.8	85.2	83.4	2.6	0.1
ODI	58.6	5.1	71.7	73.8	76.1	0.3	0.3
CFM	84.1	13.8	88.2	91.5	89.3	15.2	0.5
RaPA	**88.2**	**43.2**	**91.2**	**92.2**	**92.7**	**25.7**	**4.0**

Table 6. ASRs (%) against six defense methods.

Attack	ViT	LeViT	ConViT	Twins	PiT	CLIP	Avg.
RaPA	33.8	75.4	27.6	59.5	57.3	15.6	45.0
DSM	8.1	49.5	7.8	31.6	23.3	3.1	20.6
DSM-RaPA	50.8	84.2	42.2	74.6	72.6	25.1	58.3
SASD-WS	30.1	74.1	25.6	52.1	52.9	18.1	42.2
SASD-RaPA	42.9	79.5	35.3	63.0	63.1	25.9	51.6

Table 7. ASRs (%) of adversarial examples generated by ResNet50 against Transformer-based targets, comparing RaPA with training-based methods (DSM and SASD-WS) and their combinations. Note that when RaPA is combined with SASD-WS, we did not apply Weight Scaling.

4.5. Training-enhanced Frameworks

We compare RaPA with two training-dependent approaches, DSM[57] and SASD-WS[54], which involve additional optimization to enhance surrogate models for better adversarial transferability. Specifically, DSM trains a surrogate model with dark knowledge extracted from a teacher model and enriched by mixing augmentation. SASD-WS improves transferability via sharpness-aware self-distillation and weight scaling, refining the loss landscape and model generalization. Table 7 shows that under a fully training-free setting, RaPA already surpasses these training-dependent methods. Furthermore, when integrated with such training-based frameworks, RaPA continues to deliver consistent gains. For instance, combining with DSM increases the average ASR from 20.6% to 58.3%, highlighting its compatibility with existing training-enhanced frameworks.

5. Concluding Remarks

In this paper, we reveal the over-reliance issue in existing transfer-based attacks, where adversarial examples depend excessively on a small subset of model parameters. To alleviate this, we propose RaPA, which randomly prunes surrogate parameters during optimization. We show that the expected effect of random pruning equals adding an importance-equalization regularizer, thereby reducing parameter over-reliance and improving transferability. Extensive experiments on CNN and Transformer architectures confirm the effectiveness and stability of RaPA.

6. Acknowledgement

This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB0680101); the National Key Research and Development Program of China (Grant No. 2023YFA1011602); the CAS Project for Young Scientists in Basic Research (Grant No. YSBR-034); the Xiaomi Young Talents Program; and the Innovation Project of Institute of Computing Technology, Chinese Academy of Sciences (Grant No. E561130).

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 3, 6, 12

[2] Junyoung Byun, Myung-Joon Kwon, Seungju Cho, Yoonji Kim, and Changick Kim. Introducing competition to boost the transferability of targeted adversarial examples through clean feature mixup. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 1, 3, 5, 6, 12

[3] Huanran Chen, Yichi Zhang, Yinpeng Dong, and Junyi Zhu. Rethinking model ensemble in transfer-based adversarial attacks. In *International Conference on Learning Representations*, 2024. 5, 12

[4] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017. 5

[5] 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. In *Advances in Neural Information Processing Systems*, 2021. 6

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2009. 6

[7] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 1, 3, 5, 6, 12

[8] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 3, 6, 12

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, G Heigold, S Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020. 1, 6

[10] 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*, 2021. 6

[11] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 13

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2015. 1, 2, 12

[13] 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 *International Conference on Computer Vision*, 2021. 6

[14] Jindong Gu, Xiaojun Jia, Pau de Jorge, Wenqian Yu, Xinwei Liu, Avery Ma, Yuan Xun, Anjun Hu, Ashkan Khakzar, Zhijiang Li, et al. A survey on transferability of adversarial examples across deep neural networks. *Transactions on Machine Learning Research*, 2024. 1

[15] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018. 8

[16] Ali Hatamizadeh and Jan Kautz. Mambavision: A hybrid mamba-transformer vision backbone. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 15

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016. 1, 2, 5

[18] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *International Conference on Computer Vision*, 2021. 6

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2, 5

[20] Hao Huang, Ziyan Chen, Huanran Chen, Yongtao Wang, and Kevin Zhang. T-sea: Transfer-based self-ensemble attack on object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 1, 3, 5, 12

[21] Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, et al. Adversarial attacks and defences competition. In *The NIPS'17 Competition: Building Intelligent Systems*, 2018. 8

[22] Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, et al. Adversarial attacks and defences competition. In *The NIPS'17 Competition: Building Intelligent Systems*, pages 195–231. Springer, 2018. 5

[23] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial Intelligence Safety and Security*, pages 99–112. Chapman and Hall/CRC, 2018. 2, 12

[24] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989. 3

[25] Maosen Li, Cheng Deng, Tengjiao Li, Junchi Yan, Xinbo Gao, and Heng Huang. Towards transferable targeted attack. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 12

[26] Yingwei Li, Song Bai, Yuyin Zhou, Cihang Xie, Zhishuai Zhang, and Alan Yuille. Learning transferable adversarial examples via ghost networks. In *AAAI Conference on Artificial Intelligence*, 2020. 1, 2, 3, 5, 12, 15

[27] Kaisheng Liang, Xuelong Dai, Yanjie Li, Dong Wang, and Bin Xiao. Improving transferable targeted attacks with feature tuning mixup. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25802–25811, 2025. 1, 3, 12

[28] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 8

[29] 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*, 2020. 3, 6, 12

[30] Chuan Liu, Huanran Chen, Yichi Zhang, Yinpeng Dong, and Jun Zhu. Scaling laws for black box adversarial attacks. *arXiv preprint arXiv:2411.16782*, 2024. 1, 5, 12

[31] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Xiaodong Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations*, 2017. 1, 5, 12

[32] Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. In *International Conference on Computer Vision*, 2021. 6

[33] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *IEEE/CVF conference on computer vision and pattern recognition*, 2019. 3, 13

[34] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Fahad Shahbaz Khan, and Fatih Porikli. On improving adversarial transferability of vision transformers. In *International Conference on Learning Representations*, 2022. 1, 3, 5, 6, 12

[35] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. In *International Conference on Machine Learning (ICML)*, 2022. 8

[36] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 15

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. 6

[38] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pages 5389–5400. PMLR, 2019. 15

[39] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? In *Advances in Neural Information Processing Systems*, 2020. 8

[40] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 6

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 5

[42] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. 1, 2

[43] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016. 6

[44] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI Conference on Artificial Intelligence*, 2017. 6

[45] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, 2019. 6

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. 4

[47] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, 2013. 4

[48] Hung-Jui Wang, Yu-Yu Wu, and Shang-Tse Chen. Enhancing targeted attack transferability via diversified weight pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 3

[49] Kunyu Wang, Xuanran He, Wenxuan Wang, and Xiaosen Wang. Boosting adversarial transferability by block shuffle and rotation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 1, 3, 6, 12

[50] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 12

[51] Xiaosen Wang, Xuanran He, Jingdong Wang, and Kun He. Admix: Enhancing the transferability of adversarial attacks. In *International Conference on Computer Vision*, 2021. 1, 3, 6, 12

[52] Xiaosen Wang, Zeliang Zhang, and Jianping Zhang. Structure invariant transformation for better adversarial transferability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 1, 3, 6

[53] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. *arXiv preprint arXiv:2301.00808*, 2023. 15

[54] Han Wu, Guanyan Ou, Weibin Wu, and Zibin Zheng. Improving transferable targeted adversarial attacks with model self-enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2, 3, 8, 12

[55] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018. 8

[56] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 3, 6, 12

[57] Dingcheng Yang, Zihao Xiao, and Wenjian Yu. Boosting the adversarial transferability of surrogate models with dark knowledge. In *2023 IEEE 35th International Conference on Tools with Artificial Intelligence*, 2023. 3, 8, 12

[58] Dingcheng Yang, Wenjian Yu, Zihao Xiao, and Jiaqi Luo. Generating adversarial examples with better transferability via masking unimportant parameters of surrogate model. In *International Joint Conference on Neural Networks*, 2023. 1, 2, 3, 5, 6, 12

[59] Zhuolin Yang, Linyi Li, Xiaojun Xu, Shiliang Zuo, Qian Chen, Benjamin Rubinstein, Ce Zhang, and Bo Li. Trs: Transferability reduced ensemble via encouraging gradient diversity and model smoothness. In *Advances in Neural Information Processing Systems*, 2021. 1

[60] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. On success and simplicity: A second look at transferable targeted attacks. In *Advances in Neural Information Processing Systems*, 2021. 1, 5, 12

[61] 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*, 2020. 1, 3, 6, 12

Supplementary Material

A. More Related Work

One of the most fundamental attack methods is Fast Gradient Sign Method (FGSM) [12], which uses the direction of gradient to craft adversarial examples. Iterative-FGSM (I-FGSM) [23] extends FGSM into an iterative framework to enhance the attack performance. However, while the obtained adversarial examples achieve high success rates for white-box attacks, they do not perform well when transferred to other black-box models. To improve transferability, many methods have been proposed, mainly from the following perspectives: input transformation, gradient stabilization, loss function refinement, and model ensemble.

The first class applies transformations to the input images, thereby diversifying the input patterns [1, 2, 8, 49, 51, 56, 61]. Diverse Inputs (DI) [56] is one of the representative methods. At each iteration, it applies random and differentiable transformations (e.g., random resizing) to the input image and then maximizes the loss function w.r.t. the transformed inputs. Resized Diverse Inputs (RDI) [61] extends DI by resizing the transformed image back to its original size. The Translation-Invariant (TI) attack method [8] optimizes perturbations across a set of translated images, making the adversarial example less sensitive to the surrogate model. Block Shuffle and Rotate (BSR) [49] divides the input image into blocks, followed by shuffling or rotation, and Object-based Diversity Inputs (ODI) [1] generates an adversarial example on a 3D object and leads the rendered image to being classified as the target class. Admix [51] mixes the input image with random samples from other classes. Clean Feature Mixup (CFM) [2] extends Admix to high-level feature space and introduces two types of competing noises to guide adversarial perturbations. Building upon it, Feature Tuning Mixup (FTM) [27] further introduces learnable and attack-specific feature perturbations that combine random and optimized noises in the feature space, achieving SoTA performance in transfer-based targeted attacks.

The second class of methods stabilize the gradient updates to enhance transferability during adversarial example generation. Momentum Iterative FGSM (MI-FGSM) [7] incorporates a momentum term into I-FGSM, helping avoid local optima. The Variance-Tuned (VT) method [50] further improves stability by not only accumulating the gradients at current iteration but also incorporating the variance of gradients from previous iterations to adjust the current update. Scale-Invariant (SI) optimization [29] leverages the scale-invariance property of deep learning models by applying perturbations across multiple scaled copies of the input image, in order to reduce overfitting to the white-box model.

Another direction is to devise tailored loss functions, particularly for targeted attacks. The Po+Trip method [25] introduces Poincaré distance as a similarity measure and dynamically adjusts gradient magnitudes to mitigate the "noise curing" issue. The Logit method [60], which directly maximizes the logit output of the target class, alleviates the gradient vanishing problem and has demonstrated significant improvement w.r.t. transferability of targeted attack.

Additionally, surrogate model ensemble is also useful to enhancing transferability, where one utilizes the average of losses, predictions, or logits of multiple models to craft adversarial examples [7, 31]. A recent work [3] proposes the common weakness attack method composed of sharpness aware minimization and cosine similarity encourager, beyond the averaging. As shown in [20, 30], the transferability generally improves with an increasing number of surrogate classifiers. However, the number of surrogate models in practice is usually small and picking proper surrogate models for the same task is also not easy.

Another line of work focuses on improving the *surrogate model itself* through additional training to enhance the adversarial transferability. Dark Surrogate Model (DSM) [57] trains a surrogate model using *dark knowledge* distilled from a teacher model and further augments the training data with *mixing augmentations* such as CutMix and Mixup, thereby enriching soft supervision and improving transferability. SASD-WS [54] introduces *Sharpness-Aware Self-Distillation (SASD)* to flatten the loss landscape and combines it with *Weight Scaling (WS)* to approximate model ensembling. While these approaches require retraining surrogate models, our RaPA works in a *training-free* manner, yet achieves comparable or even superior transferability, and can be seamlessly integrated with such training-based frameworks.

Closely related to the present work is self-ensemble [20, 26, 34, 58], which creates multiple models from only one surrogate model. Transfer-based self-ensemble (T-sea) [20] locally apply various transformations onto the input image to improve such diversity while preserving the structure of imagetargets the task of object detection, and the self-ensemble method in [34] specifically considers vision Transformer as surrogate model and is denoted as SE-ViT in this paper. Ghost Network [26] perturbs surrogate model to create a set of new models and then samples one model from the set at each iteration. Masking Unimportant Parameters (MUP) [58] drops out unimportant parameters according to a predefined Taylor expansion-based metric.

B. Estimation and Approximation

B.1. Parameter Importance Estimation and Pruning by Importance

Computing the second-order derivative defined in Eq. (3) is generally costly. A more compact approximation is obtained via first-order expansion [33], which reduces to

$$\mathcal{I}(\theta_i) = \frac{\partial^2 \mathcal{L}\left(f(x_{\text{adv}})\right)}{\partial \theta_i^2}\theta_i^2 \approx \left(\frac{\partial \mathcal{L}\left(f(x_{\text{adv}})\right)}{\partial \theta_i}\theta_i\right)^2. \tag{7}$$

In our pilot study, we adopt the method in DepGraph [11] to perform the pruning, which explicitly models inter-layer dependencies and comprehensively groups coupled parameters for pruning.

B.2. Derivation of the Masked Loss Approximation in Eq. (4)

Let θ represent the entire set of model parameters. We define a random binary mask $\mathcal{M} \in \{0,1\}^{|\theta|}$, where each entry is independently sampled from a Bernoulli distribution: $\mathcal{M}_i \sim \text{Bernoulli}(1-p)$. A second-order Taylor expansion gives

$$\mathcal{L}\big(f(x_{\text{adv}}; \mathcal{M} \odot \theta)\big) \approx \mathcal{L}\big(f(x_{\text{adv}}; \theta)\big) + \sum_i g_i \Delta_i + \tfrac{1}{2}\sum_{i,j} H_{ij}\Delta_i\Delta_j,$$

where $\Delta = (\mathcal{M} - \mathbf{1}) \odot \theta$, $g_i = \partial \mathcal{L}\big(f(x_{\text{adv}}; \theta)\big)/\partial \theta_i$ and $H_{ij} = \partial^2 \mathcal{L}\big(f(x_{\text{adv}}; \theta)\big)/(\partial \theta_i \partial \theta_j)$ evaluated at θ. Since p is taken to be small and close to zero, we have $\mathbb{E}[\Delta_i] \approx 0$. And the mask entries are independent, so we further have $\mathbb{E}[\Delta_i \Delta_j] \approx 0$ for $i \neq j$ while $\mathbb{E}[\Delta_i^2] = p(1-p)\theta_i^2$. Taking expectations yields

$$\mathbb{E}_{\mathcal{M}}\left[\mathcal{L}\big(f(x_{\text{adv}}; \mathcal{M} \odot \theta)\big)\right] \approx \mathcal{L}\big(f(x_{\text{adv}}; \theta)\big) + \frac{p(1-p)}{2}\sum_i H_{ii}\,\theta_i^2, \tag{8}$$

which is the expression in Eq. (4).

B.3. Detailed Gini Coefficients Computation Procedure

We compute the Gini coefficients based on the parameter importance values $\mathcal{I}(\theta_i)$ defined in Eq. (3). For each layer l in the surrogate model, we first collect all parameter importance values $\{\mathcal{I}^l(\theta_i)\}_{i=1}^n$. To better highlight the disparity among parameters, we further apply an exponential scaling to these importance values before computing the Gini coefficient:

$$\tilde{\mathcal{I}}^l(\theta_i) = \exp\left(\mathcal{I}^l(\theta_i)\right), \tag{9}$$

where $\tilde{\mathcal{I}}^l(\theta_i)$ denotes the scaled importance. This transformation magnifies the relative differences among parameters, enabling a clearer assessment of importance inequality.

Given the scaled importance values $\{\tilde{\mathcal{I}}^l(\theta_1), \tilde{\mathcal{I}}^l(\theta_2), \ldots, \tilde{\mathcal{I}}^l(\theta_n)\}$ in layer l, we compute the Gini coefficient as:

$$\text{Gini}^l = \frac{2\sum_{i=1}^n i\tilde{v}_{(i)}}{n\sum_{i=1}^n \tilde{v}_{(i)}} - \frac{n+1}{n}, \tag{10}$$

where $\tilde{v}_{(i)}$ represents the i-th smallest value after sorting in ascending order. A smaller Gini coefficient indicates a more balanced distribution of parameter importance within the layer.

Finally, we aggregate the Gini coefficients across layers to obtain both the type-wise and overall average results:

$$\text{Gini}_{\text{avg}} = \frac{1}{|\mathbb{L}|}\sum_{l \in \mathbb{L}} \text{Gini}^l, \tag{11}$$

where $\mathbb{L}$ denotes the set of all layers containing learnable parameters. This process allows us to quantitatively evaluate the degree of importance imbalance and verify that the proposed random parameter pruning strategy effectively reduces the model's over-reliance on a small subset of dominant parameters.

C. Visualization of Adversarial Examples

Figs. 4 and 5 visualize the attention heatmap of the adversarial examples, with surrogate model ResNet-50 and target model ResNet-18. We observe that the attention heatmaps w.r.t. RaPA on the surrogate model are more similar to the heatmaps on target model, compared with other attack methods. This in part explains that the generated adversarial example of MCD is more transferable to the target model.

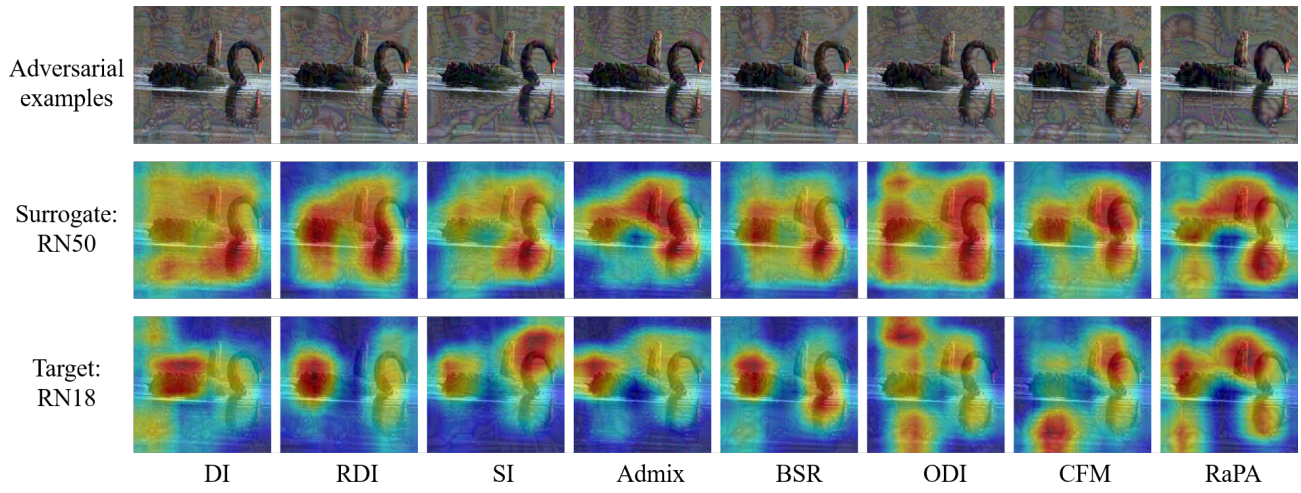


Figure 4. Attention heatmap of adversarial example. The true label is 'black swan' and the target label is 'weasel'. The intensity of red indicates the level of importance assigned to each area, influencing the classifier prediction towards the target label.

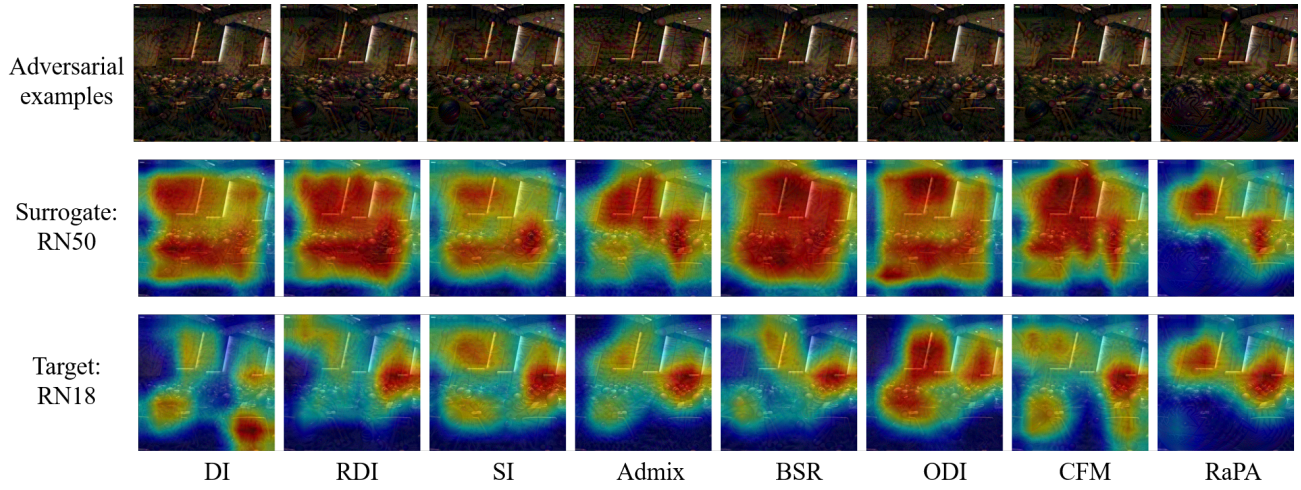


Figure 5. Attention heatmap of adversarial example. The true label is 'cinema' and the target label is 'croquet ball'. The intensity of red indicates the level of importance assigned to each area, influencing the classifier prediction towards the target label.

D. Additional Experimental Results

D.1. Result of masking the most important parameters

Method	DI	GN	DWP	RDI	MMP(1%)	MMP(0.1%)	MMP(0.01%)	MMP(0.001%)
ASR(%)	26.8	29.7	48.0	42.0	0.2	27.7	36.7	38.9

Table 8. Average ASRs when masking the most important parameters (MMP) in the surrogate model at mask ratios of 1 %, 0.1 %, 0.01 %, and 0.001 %. Results are the average over 16 target models on the ImageNet-compatible dataset, with ResNet-50 as surrogate model. Detailed experimental setting can be found in Section 4.1.

Using the approximation in Eq. (7), we mask the most important parameters. The results are reported in Table 8. As analyzed in Section 3, masking the most important parameters degrades the model's attacking capability. A high masking

ratio (e.g., 1 %) causes the attack to fail, whereas a low ratio (e.g., 0.001 %) leaves transferability unaffected, so the ASRs converge to the baseline RDI.

D.2. Result of other related work

We also include the empirical result of Ghost Networks (GN) [26]. It perturbs only skip connections for residual networks and also uses one new model at each iteration. For a fair comparison, we combine it with RDI and pick a larger iteration number for optimization. However, its performance is still much lower.

D.3. Ablation Study on DropConnect Probability

Experimental results of varying DropConnect probabilities are shown in Fig. 6, indicating a stable performance of RaPA across a range of different parameter values.

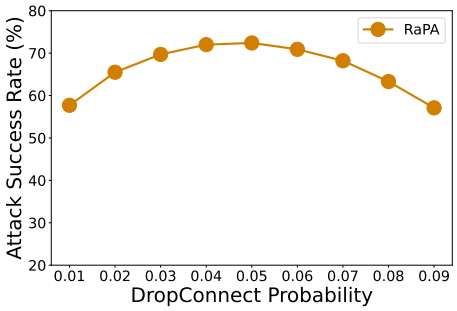


Figure 6. ASRs(%) averaged over 16 models with varying DropConnect probabilities, using ResNet-50 as surrogate .

D.4. Additional Attack Results

Evaluation on Same Architecture Tab. 9 and Tab. 10 present additional attack results under the same-architecture transfer setting, following the configuration in Sec. 4.1. Our method RaPA consistently outperforms baselines by a large margin. Beyond these results, we further evaluate RaPA in more challenging scenarios, including various surrogate model architectures, multi-model ensembles, and untargeted attack settings.

Evaluation on Newer and Larger Surrogate Models. To further validate the scalability of RaPA, we extend our evaluation to more recent and significantly larger surrogate models, including ConvNeXtV2-L [53] (2023, 198M parameters), DINOv2-L [36] (2023, 300M), and the SSM-based MambaVision [16] (2025). As shown in Tab. 11, RaPA consistently outperforms the state-of-the-art CFM. Notably, the high source-to-source (Src→Src) scores of CFM compared to RaPA further validate that CFM tends to overfit the surrogate model, whereas RaPA maintains better transferability.

Evaluation on Cross-Dataset Generalization. To verify the generalization of RaPA across different data distributions, we conduct additional experiments on the ImageNet-V2 dataset[38]. We evaluate the average ASRs across 10 CNN-based and 6 Transformer-based target models. As shown in Tab. 12, RaPA consistently outperforms all baseline methods. Notably, our method achieves a significant 16.4% improvement in the challenging ViT-to-CNN transfer scenario. These results further validate that RaPA maintains its superior transferability with different evaluation dataset.

Evaluation on Multi-Model Ensembles. We also investigate the performance of RaPA in a multi-model ensemble setting. We employ an ensemble of five distinct source models (ResNet50, ResNet18, VGG16, DenseNet121, and Inception-v3) and test them against six Transformer-based target models. RaPA demonstrates superior performance in both single-model (45.0% vs. 32.2% for CFM) and multi-model ensemble settings (63.3% vs. 60.7% for CFM), proving its robustness across diverse source distributions.

Evaluation on Untargeted Attacks. While our primary focus is on the more challenging targeted attack task, we also evaluated RaPA in an untargeted setting. We observed that existing methods already achieve near-saturated performance; for instance, CFM reaches an average ASR of 98.2% across 16 target models. In this regime, while RaPA yields similar ASRs,

the performance gap between top-tier methods becomes statistically indistinguishable. Therefore, the targeted setting serves as a more effective benchmark for demonstrating the advantages of our approach.

D.5. Model Diversity and Utility

We show the diversity and utility of constructed models. Here diversity is measured by first computing the mean KL-divergence between the output distributions of each pair of model variants for an input and then averaging these values over the ImageNet-compatible dataset. Similarly, the utility is quantified by the mean top-1 accuracy over the model variants and the dataset. We use the same surrogate model ResNet-50 as in the ablation study.

The results are presented in Figure 7. As DropConnect probability p increases, the diversity among different variants improves while the utility decreases. As shown in Figure 6 in the supplementary material, the average ASR using the same surrogate model increases with $p \in [0.01, 0.05]$ and then decreases with $p \in [0.05, 0.09]$, while the highest ASR is obtained at $p = 0.05$. This verifies our discussion in Section 3.3 that the attack performance is affected by both diversity and utility.

Source : RN50	Target model										
Attack	RN18	RN50	VGG16	Incv3	EFB0	DN121	MBv2	IRv2	Incv4	Xcep	Avg.
DI	60.1	98.7	64.6	8.1	28.6	77.3	28.4	13.5	21.1	15.1	41.5
RDI	80.6	98.7	75.4	32.8	52.6	87.4	49.6	47.8	51.1	40.7	61.7
SI	82.9	98.8	73.7	60.8	66.0	90.0	59.4	66.9	68.7	60.6	72.8
BSR	87.9	98.5	84.1	54.9	70.0	91.6	65.3	66.5	68.9	63.3	75.1
Admix	83.2	98.1	78.1	57.1	67.8	86.8	70.3	61.9	64.4	64.5	73.2
SIA	84.5	98.3	79.3	39.0	67.1	88.4	70.1	46.0	53.4	50.9	67.7
ODI	77.5	**98.9**	83.0	63.9	69.9	89.8	61.2	71.0	71.1	69.2	75.5
MUP	90.3	97.7	85.6	61.8	75.0	91.6	79.3	70.9	71.8	68.7	79.3
CFM	92.0	98.0	89.5	74.8	85.6	93.4	85.2	**82.6**	82.5	81.7	86.5
RaPA	**93.0**	98.2	**91.0**	**82.6**	**88.5**	**93.6**	**90.9**	82.1	**85.5**	**85.0**	**89.0**
Source : DN121	Target model										
Attack	RN18	RN50	VGG16	Incv3	EFB0	DN121	MBv2	IRv2	Incv4	Xcep	Avg.
DI	28.1	36.4	35.8	6.5	14.5	**99.0**	10.1	9.8	13.9	10.1	26.4
RDI	50.9	54.4	46.7	19.3	26.9	98.6	20.6	29.8	30.5	24.7	40.2
SI	54.8	60.9	43.0	35.6	37.9	98.6	26.7	43.6	43.5	34.6	47.9
BSR	61.6	70.3	57.3	34.1	41.7	98.3	30.2	42.7	44.5	37.5	51.8
Admix	72.4	74.3	65.8	46.9	55.1	97.9	51.2	55.5	56.6	53.9	63.0
SIA	69.2	79.3	66.2	34.9	54.8	98.1	45.1	40.6	51.6	44.1	58.4
ODI	59.6	71.5	69.6	48.1	52.2	98.9	35.5	58.1	60.0	52.7	60.6
MUP	79.9	86.9	75.2	50.3	60.5	97.7	51.4	65.7	66.5	61.0	69.5
CFM	82.4	88.9	78.6	60.9	69.9	98.5	60.5	73.0	72.7	70.3	75.6
RaPA	**89.6**	**90.2**	**85.1**	**77.8**	**84.4**	97.4	**84.4**	**83.2**	**84.1**	**83.7**	**86.0**
Source : CLIP	Target model										
Attack	RN18	RN50	VGG16	Incv3	EFB0	DN121	MBv2	IRv2	Incv4	Xcep	Avg.
DI	0.1	0.1	0.1	0.0	0.0	0.1	0.2	0.0	0.0	0.1	0.1
RDI	0.5	0.4	0.2	0.4	0.9	0.4	0.3	0.4	0.2	0.7	0.4
SI	0.6	0.5	0.6	1.4	1.6	1.4	1.0	1.0	0.8	1.2	1.0
BSR	1.7	1.6	0.9	1.9	3.5	2.4	1.5	1.9	2.3	2.2	2.0
Admix	2.7	2.2	1.6	4.1	6.0	3.5	2.8	2.9	3.8	3.6	3.3
SIA	0.9	1.0	0.8	1.8	3.5	1.6	1.9	1.3	1.8	2.6	1.7
ODI	7.7	7.0	**5.1**	11.0	13.7	13.0	6.5	11.3	11.8	11.5	9.9
CFM	3.1	3.3	1.4	4.9	7.8	4.3	2.5	5.4	4.1	4.8	4.2
RaPA	**8.8**	**9.5**	4.9	**14.6**	**17.5**	**14.9**	**8.5**	**12.5**	**12.8**	**12.4**	**11.6**

Table 9. Additional experimental results against ten target models on the ImageNet-Compatible dataset, using RN50, DN121 and CLIP as surrogate model, respectively. All the attack methods are combined with MI-TI. The best results are shown in bold and the second best results are underlined.

Source : Incv3	Target model						
Attack	ViT	LeViT	ConViT	Twins	PiT	CLIP	Avg.
DI	0.1	0.8	0.1	0.3	0.5	0.2	0.3
RDI	0.1	3.1	0.6	2.0	1.9	0.0	1.3
SI	1.0	5.0	1.0	2.0	3.8	0.4	2.2
SIA	1.5	16.8	1.7	5.2	6.8	0.4	5.4
BSR	1.3	15.3	1.9	5.6	7.3	0.5	5.3
Admix	2.0	17.4	1.9	6.7	8.5	1.4	6.3
ODI	3.6	21.6	4.4	8.9	15.1	2.0	9.3
MUP	1.4	11.9	2.3	4.4	5.4	0.4	4.3
CFM	9.7	43.4	8.8	23.2	26.4	3.9	19.2
RaPA	**14.7**	**57.6**	**16.7**	**34.5**	**38.0**	**5.3**	**27.8**
Source : ViT	Target model						
Attack	ViT	LeViT	ConViT	Twins	PiT	CLIP	Avg.
DI	**100.0**	11.3	15.6	7.4	14.8	3.2	25.4
RDI	**100.0**	31.5	38.9	23.6	36.2	7.7	39.6
SI	**100.0**	48.2	59.2	28.3	51.4	14.8	50.3
SIA	**100.0**	43.0	77.9	44.8	56.2	12.0	55.6
BSR	**100.0**	57.8	63.0	45.7	67.2	18.5	58.7
Admix	99.8	78.2	81.5	66.4	79.2	39.6	74.1
SE	100	65.6	75.7	56.4	72	20.4	65.0
ODI	**100.0**	74.8	68.3	63.1	81.7	31.6	69.9
CFM	99.9	90.9	94.2	82.5	91.6	55.3	85.7
RaPA	99.6	**92.4**	**95.7**	**87.4**	**94.4**	**63.5**	**88.8**
Source : CLIP	Target model						
Attack	ViT	LeViT	ConViT	Twins	PiT	CLIP	Avg.
DI	0.0	0.0	0.1	0.0	0.0	**99.8**	16.6
RDI	1.3	1.5	0.6	0.4	1.0	**99.8**	17.4
SI	3.1	4.5	2.5	1.7	6.2	99.6	19.6
SIA	7.7	7.9	5.3	3.6	8.3	99.7	22.1
BSR	7.0	9.2	4.9	2.9	8.5	99.5	22.0
Admix	7.3	11.9	5.5	4.2	11.1	99.0	23.2
ODI	19.1	24.3	13.1	11.7	22.7	**99.8**	31.8
CFM	13.3	15.4	8.6	4.7	13.5	**99.8**	25.9
RaPA	**32.5**	**35.5**	**25.1**	**16.7**	**33.0**	98.4	**40.2**

Table 10. Additional experimental results against five transformer-based target models on the ImageNet-Compatible dataset, using RN50, DN121 and CLIP as surrogate model, respectively. All methods are combined with MI-TI. The best results are shown in bold, and the second best results are underlined.

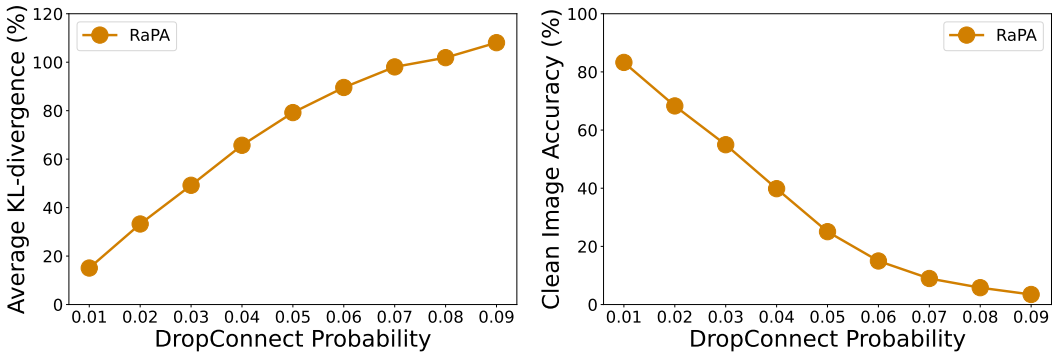
Source Model	**Method**	**CNNs Avg**	**ViTs Avg**	**Src→Src**
MambaVision-T	CFM	12.5	40.2	91.9
	RaPA	**63.8**	**64.9**	89.7
DINOv2-Large	CFM	20.0	27.7	100.0
	RaPA	**41.4**	**49.3**	98.7
ConvNeXtV2-L	CFM	20.6	28.4	100.0
	RaPA	**46.4**	**56.7**	100.0

Table 11. Comparisons on newer and larger source models. ASR (%) is reported. Src→Src prove that CFM overfit to surrogate model

We conclude that the better performance of RaPA is due to the improved diversification across the variants while keeping each variant sufficiently useful.

Method	Source: RN50		Source: ViT	
	CNN	ViT	CNN	ViT
CFM	81.9	52.9	56.6	86.9
FTM	83.2	54.7	20.3	35.6
RaPA (Ours)	**86.3**	**58.7**	**73.0**	**91.0**

Table 12. ASR (%) on ImageNet-V2. RaPA vs. baselines (CFM, FTM) using RN50 and ViT surrogates transferred to 10 CNN and 6 Transformer models.



(a) Diversity (b) Utility

Figure 7. Diversity and Utility with increasing mask ratio.

Ethical Statement

While our work is intended for research and defense purposes, it could be misused by malicious actors to craft black-box adversarial attacks against safety-critical systems such as autonomous-driving or medical models. By demonstrating that these systems remain vulnerable even when attackers have no knowledge of their internal details, we aim to alert service providers and researchers to this threat and to accelerate the development of more robust deep-learning defenses.