6. Additional related works

In this section, we provide a detailed introduction to the related works that we used as baseline (NI-SI-TI-DI) throughout the work and show how DWP is combined with it.

6.1. Baseline

Momentum and Nesterov Iterative Method (NI) [5, 25] Inspired by Nesterov Accelerated Gradient [32], the Nesterov Iterative Method (NI) modifies Momentum Iterative-FGSM [5] by adding the historical gradients to current adversarial examples x_n and gets x_n^{nes} in advance. Gradients at the ahead x_n^{nes} instead of the current x_n will be used for updating. The scheme helps accelerate convergence by avoiding the local optimum earlier:

$$x_n^{\text{nes}} = x_n + \alpha \cdot \mu \cdot g_{n-1} \tag{7}$$

$$g_n = \mu \cdot g_{n-1} + \nabla_x J(x_n^{\text{nes}}, y^{\text{target}}; \theta) \tag{8}$$

$$x_{n+1} = \text{Clip}_x^{\epsilon}(x_n - \alpha \cdot \text{sign}(g_n)). \tag{9}$$

Here μ is the decay factor of the historical gradients. The gradient computed encourages adversarial examples to increase confidence logit output by the white-box network model θ on the target class through gradient ascent with learning rate α. A clipping operation onto the ϵ-ball centered at the original input image x is at the end of each iteration. To preserve more information about the gradient for attacking [56], we don't include the L1 normalization.

Scale Invariant Method (SI) [25] Neural networks can preserve output even though the input image x goes through scale operations such as $S_m(x) = x/2^m$. With the scale-invariant property, each composite of white-box networks and scale operations becomes different functions. Adversarial examples can enjoy more diverse gradients:

$$g_n = \mu \cdot g_{n-1} + \frac{1}{M} \sum_{m=0}^{M-1} \nabla_x J(S_m(x_n^{\text{nes}}), y^{\text{target}}; \theta). \tag{10}$$

M is the number of scaled versions feeding into the network for each image.

Diverse Input Patterns (DI) [49] Inspired by data augmentation techniques [37] used in network training, DI imposes random resizing and padding on each image before it feeds into network models to avoid overfitting. Straightforward cooperation with NI and SI is as follows:

$$g_n = \mu \cdot g_{n-1} + \frac{1}{M} \sum_{m=0}^{M-1} \nabla_x J(S_m(T(x_n^{\text{nes}}, p_{\text{DI}})), y^{\text{target}}; \theta). \tag{11}$$

The introduced T decides whether to apply random resizing at each iteration with probability p_{DI}, which degenerates when $p_{\text{DI}} = 0$.

Translation Invariant Method (TI) [6] To deal with different discriminative regions [6] of various defense neural networks, TI produces several translated versions for the current image in advance and computes the gradient for each separately. These gradients will then be fused and used to attack the current image. [6] also shows that one can approximate the gradient fusion using convolution. The approximation prevents TI from enduring the costly computation on excessive translated versions for every single image, also yielding the further revised updating procedure:

$$g_n = \mu \cdot g_{n-1} + \boldsymbol{W} * \frac{1}{M} \sum_{m=0}^{M-1} \nabla_x J(S_m(T(x_n^{\text{nes}}, p_{\text{DI}})), y^{\text{target}}; \theta). \tag{12}$$

$\boldsymbol{W}$ is the convolution kernel matrix applied. Some typical options are linear, uniform, or Gaussian kernel.

6.2. Combining DWP with NI-SI-TI-DI

We acquire pruned models at each iteration right before gradient computing and combine with NI-SI-TI-DI:

$$g_n = \mu \cdot g_{n-1} + \frac{\boldsymbol{W}}{M} * \sum_{m=0}^{M-1} \nabla_x J(S_m(T(x_n^{\text{nes}}, p_{\text{DI}})), y^{\text{target}}; P(\theta, r)). \tag{13}$$

where the pruning operation $P(.)$ is obtained in Eq. (5).

Finally, with K white-box models participating in longitudinal ensemble, our final DWP attack procedure is shown as follows:

$$g_n = \mu \cdot g_{n-1} + \frac{\boldsymbol{W}}{M} * \sum_{m=0}^{M-1} \sum_{k=1}^{K} \beta_k \nabla_x J(S_m(T(x_n^{\text{nes}}, p_{\text{DI}})), y^{\text{target}}; P(\theta_k, r)), \tag{14}$$

where β_k are the ensemble weights, $\sum_{k=1}^{K} \beta_k = 1$.

7. Untargeted attack for single model attack transferability

We provide untargeted attack results transferring from a single source model in Tab. 8. The untargeted attack's goal is to minimize the overall accuracy of the victim model without considering which class to predict. As a result, the untargeted success rate is higher than the targeted one on average. In this situation, DWP still prevail NI-SI-TI-DI for

	Source Model: Res-50			Source Model: VGG-16		
	→VGG-16	→Den-121	→Inc-v3	→Res-50	→Den-121	→Inc-v3
NI-SI-TI-DI	92.3	96.3	79.7	80.1	83.4	74.8
+GN	93.2	96.7	80.8	82.1	86.4	79.1
+DWP	**95.5**	**98.2**	**85.0**	**83.6**	**86.7**	**79.8**
	Source Model: Den-121			Source Model: Inc-v3		
	→Res-50	→VGG-16	→Inc-v3	→Res-50	→VGG-16	→Den-121
NI-SI-TI-DI	87.0	86.9	71.8	71.8	74.6	69.7
+GN	90.4	89.3	77.4	58.1	72.3	62.0
+DWP	**91.7**	**92.2**	**81.7**	**72.7**	**80.4**	**75.0**

Table 8. Untargeted success rates of transferring to naturally trained CNNs without the ensemble strategy. The "→" prefix stands for the black-box network. Results with targeted / untargeted attack success rates are reported.

Attack Method	NI-SI-TI-DI	+GN	+DWP
Inc-v3ens3	80.3	84.1	**88.0**
IncRes-v2ens	52.7	66.0	**67.5**
Average	66.5	75.05	**77.75**

Table 9. The untargeted success rates of transferring to adversarially trained models. DWP outperforms GN and DSNE over 10%.

3.47% when transferring from Res-50, 3.93% from VGG-16, 6.63% from Den-121, and 4% from Inc-v3, on average. When comparing with GN, DWP obtains 2.67%, 0.83%, 2.83% and 11.9% improvement for Res-50, VGG-16, Den-121 and Inc-v3, respectively. We can observe a similar phenomenon mentioned in Fig. 2 that the extent of improvement brought by DWP is affected by the network redundancy. When the model is more sensitive to the parameter drops, DWP exhibits better performance.

8. Untargeted attack for ensemble transfer to adversarially trained model

We report the untargeted attack success rate for ensemble transferring to the adversarially-trained model in Tab. 9. DWP suppress NI-SI-TI-DI by a notable 11.25%. When comparing to the related model augmentation methods, DWP is 2.7% higher in untargeted success rate than GN.

9. Transferring to multi-step adversarially trained models

Transferable targeted attacks from naturally-trained CNNs to multi-step adversarially trained networks remain an open problem. Recent attacks only show non-targeted results [34]. Even the resource-intensive attack [31] fails to achieve satisfied targeted success rates. We choose four naturally-trained networks (Res-50, VGG-16, Den-121, Inc-v3) as white-box source models to generate the adversarial examples, transferring to the multi-step adversarially trained networks provided by Salman *et al*. [36]. Tab. 10 shows the

Attack Method	NI-SI-TI-DI	+GN	+DWP
Res-18 ($\|\epsilon\|_\infty = 1$)	0.2	0.2	0.2
Res-50 ($\|\epsilon\|_\infty = 1$)	0.0	0.6	0.3
WideRes-50-2 ($\|\epsilon\|_\infty = 1$)	0.0	0.2	0.1
Res-18 ($\|\epsilon\|_2 = 3$)	0.0	0.1	0.0
Den-121 ($\|\epsilon\|_2 = 3$)	0.0	0.0	0.0
VGG-16 ($\|\epsilon\|_2 = 3$)	0.0	0.0	0.0
Resnext-50 ($\|\epsilon\|_2 = 3$)	0.0	0.0	0.0

Table 10. The targeted success rates of transferring to three-step adversarially trained networks from naturally trained CNNs.

failure of transferring targeted attacks from the ensemble of naturally-trained CNNs. The attack success rates approach 0% in all cases. All the existing methods fail to effectively attack such a scenario and DWP is not an exception. It requires sophisticated investigation into this difficult setting.

10. Untargeted attack for ensemble transfer to non-CNN architectures

The result of the untargeted attack success rate transferring from four naturally-trained CNNs (Res-50, VGG-16, Den-121, Inc-v3) to non-CNNs (ViT-S-16-224, ViT-B-16-224, Swin-S-224, Swin-B-224, MLP-Mixer, ResMLP, gMLP) is presented in Tab. 11. DWP exceeds the NI-SI-TI-DI by a notable 9.04% on average and also suppress GN by 0.52% in untargeted attack success rate. The results further validate the efficacy of DWP.

11. Time cost of DWP

To ascertain the practical feasibility of DWP without imposing excessive computational overhead, we present a time cost analysis in Tab. 12. The results are obtained using a batch size of 16 images and 100 attack iterations, with each cell representing the average from five different runs on a single RTX A5000 GPU. Remarkably, with an equivalent number of forward passes, DWP introduces minimal overhead in comparison to the NI-SI-TI-DI.

Attack Method	NI-SI-TI-DI	+GN	+DWP
ViT-S-16-224	48.1	**57.7**	55.0
ViT-B-16-224	52.5	61.4	**64.8**
Swin-S-224	57.6	65.1	**66.5**
Swin-B-224	53.9	**62.9**	62.1
MLP-Mixer	50.1	57.7	**59.1**
ResMLP	72.7	78.5	**80.6**
gMLP	44.3	**55.5**	54.4
Average	54.17	62.69	**63.21**

Table 11. The untargeted success rates of transferring to Non-CNN architectures. Our DWP maintains higher success rates stably.

Time (sec.)	Res-50	Den-121	VGG16	Inc-v3
NI-SI-TI-DI	10.50	12.26	17.64	13.19
+DWP	10.86	15.87	18.72	15.62

Table 12. Time cost of NI-SI-TI-DI and DWP on a single CNN.

12. Perturbations diversity from auxiliary models

Recent works [6, 25] have improved transferability with output-preserving operations. Despite the model exhibiting similar output given an example, gradients calculated through backward operations differ as some randomness is introduced. The diverse gradients participating in the attack prevent overfitting to local optimal, yielding better-targeted attack transferability. Motivated by the finding that gradient diversity benefits transferability, we examine the diversity between perturbations from the pruned auxiliary models generated in DWP.

Liu *et al*. [27] first studied the effectiveness of ensemble attacks in enhancing transferability. They demonstrate the diversity of the ensemble by showing near-zero cosine similarities between perturbations from different white-box networks. Following Liu *et al*. [27], we calculate cosine similarities between perturbations generated from the additional auxiliary models produced by DWP. From each of our four naturally trained CNNs, we acquire five auxiliary models with different connections pruned. We term the cosine similarity between perturbations of pruned models from an identical CNN as an intra-CNN similarity. The case from different CNNs is termed as inter-CNN similarity. To avoid cherry-picking, both intra-CNN and inter-CNN similarities come from the average of the first ten images in the ImageNet-compatible dataset. Furthermore, we only use NI in combination with DWP to produce perturbations in this experiment to prevent other factors from affecting the result.

Fig. 6 is a symmetric matrix containing 16 (4×4) blocks. The diagonal blocks summarize ten (C_2^5) intra-CNN similarities while the non-diagonal blocks summarize 25 (5×5) inter-CNN similarities in cells. The diagonal cells are all 1.0 since they are all from two identical perturbation vectors. As for the non-diagonal cells, we find the cell values in diagonal blocks (intra-CNN) slightly higher than in non-diagonal blocks (inter-CNN). However, these values are still close to zero, appearing dark red. The results show that whether two auxiliary models come from the same CNN or not, the generated perturbations are always nearly orthogonal. These observations on orthogonality support our claim that auxiliary models obtained via DWP provide more diversity for attacking.

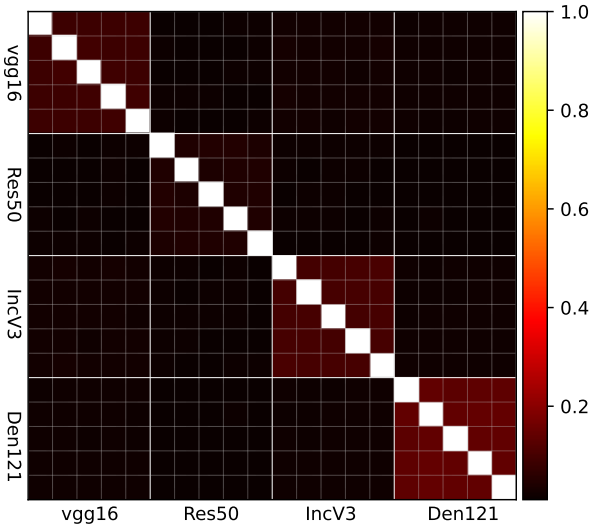
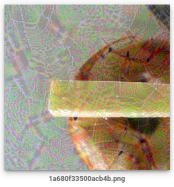


Figure 6. Perturbation cosine similarities between pruned models. Each diagonal block summarizes 10 (C_2^5) intra-CNN similarity cells. Each non-diagonal block summarizes 25 (5×5) inter-CNN similarity cells. The pairwise cosine similarity matrix is symmetric and shows orthogonality between perturbations.

13. Results of DWP on Google Cloud Vision



Insect	91%
Arthropod	90%
Pest	76%
Parasite	72%
Terrestrial Plant	68%
Arachnid	61%

Bagel → Spider

Food	97%
Plum Tomato	87%
Ingredient	87%
Recipe	85%
Natural Foods	84%
Cuisine	82%

Toy Shop → Consomme

Water	97%
Boat	93%
Boats And Boating–Equipment And Supplies	86%
Lake	85%
Outdoor Recreation	85%
Paddle	84%

Mortarboard → Paddle

Bird	96%
Plant	90%
Beak	88%
Twig	83%
Wood	83%
Trunk	80%

Menu → Jay

Performance	59%
Visual Arts	58%
Stage	57%
Tree	57%
Rope	56%
Rock Concert	53%

Dog → Stage

Bird	92%
Phasianidae	88%
Beak	84%
Feather	80%
Chicken	80%
Wild Turkey	79%

Dowitcher → Cock

Plant	91%
Dog Breed	91%
Carnivore	89%
Organism	85%
Terrestrial Plant	84%
Fawn	82%

Butterfly → Dog

Brown	98%
Footwear	98%
Shoe	95%
Outdoor Shoe	87%
Durango Boot	85%
Walking Shoe	84%

Eagle → Geta

Paint	66%
Illustration	65%
Personal Protective Equipment	60%
Measuring Instrument	57%
Helmet	54%
Flesh	50%

Beetle → Weight Machine

Plant	96%
Building	92%
Sky	92%
Fence	86%
Mesh	82%
Wire Fencing	80%

Monastery → Fence

Snails And Slugs	72%
Wood	71%
Snail	67%
Molluscs	62%
Reptile	62%
Grassland	59%

Goose → Conch

Chicken	84%
Feather	83%
Poultry	82%
Fowl	74%
Livestock	73%
Tail	70%

Turtle → Cock

Car	95%
Vehicle	93%
Hood	92%
Motor Vehicle	91%
Automotive Lighting	91%
Automotive Design	84%

Rifle → Taxi

Plant	87%
Mammal	85%
Adaptation	79%
Terrestrial Animal	78%
Grass	74%
Snout	73%

Fox → Squirrel

Bird	77%
Fish	74%
Tail	72%
Underwater	72%
Marine Biology	71%
Electric Blue	69%

Beetle → Cockatoo

Car	95%
Vehicle	94%
Tire	94%
Wheel	91%
Motor Vehicle	88%
Bird	86%

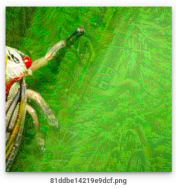
Jeep → Linnet

Event	68%
Grass	68%
Fictional Character	66%
Visual Arts	66%
Mask	66%
Artifact	60%

Otter → Mask

Painting	79%
Marine Biology	75%
Marine Invertebrates	75%
Reef	70%
Sky	70%
Landscape	70%

Dam → Sea Slug



Wheel	67%
Soil	66%
Jungle	66%
Motorcycle	65%
Extreme Sport	64%
Automotive Tire	62%

Leaf Hopper → Bike

Pollinator	92%
Insect	89%
Butterfly	88%
Arthropod	86%
Tints And Shades	76%
Moths And Butterflies	76%

Beer Glass → Butterfly