ENHANCING ADVERSARIAL ATTACKS: THE SIMILAR TARGET METHOD

Shuo Zhang[1], *Ziruo Wang*[1], *Zikai Zhou*[1], *Huanran Chen*[1*]

School of Computer Science, Beijing Institute of Technology
{huanranchen, shuozhangbit, ziruowang, zikaizhou}@bit.edu.cn

ABSTRACT

Deep neural networks are vulnerable to adversarial examples, posing a threat to the models' applications and raising security concerns. An intriguing property of adversarial examples is their strong transferability. Several methods have been proposed to enhance transferability, including ensemble attacks which have demonstrated their efficacy. However, prior approaches simply average logits, probabilities, or losses for model ensembling, lacking a comprehensive analysis of how and why model ensembling significantly improves transferability. In this paper, we propose a similar targeted attack method named Similar Target (ST). By promoting cosine similarity between the gradients of each model, our method regularizes the optimization direction to simultaneously attack all surrogate models. This strategy has been proven to enhance generalization ability. Experimental results on ImageNet validate the effectiveness of our approach in improving adversarial transferability. Our method outperforms state-of-the-art attackers on 18 discriminative classifiers and adversarially trained models.

***Index Terms*—** Adversarial examples, Black-box attack, Transfer attack.

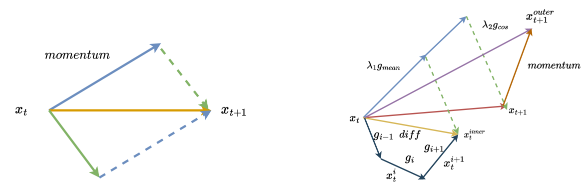
1. INTRODUCTION

Deep learning has achieved remarkable progress in recent years. However, it has been observed that these models are vulnerable to adversarial attacks, where imperceptible perturbations are added to original images, leading to misclassification [1,2]. Moreover, adversarial examples exhibit transferability [3,4], implying that attackers can craft adversarial examples on their own models and use them to attack deployed models. These methods are known as transfer attacks, and they requires no prior knowledge of the deployed models, which poses a threat to the application of deep learning models and social security, even for state-of-the-art large vision language models like llama and GPT-4 [5,6].

Researchers have made every effort to study the transferability to further enhance the security and comprehend the transferability. Dong et al. [4] proposed the Momentum Iterative (MI) method which introduces the momentum to prevent the adversarial examples from falling into the undesirable local optima. Dong et al. [7] proposed Translation-Invariant (TI) method by optimizing a perturbation over an ensemble of translated images. Xie et al. [8] proposed the Diverse Inputs (DI) method by applying random transformations to the input images at each iteration. Wang et al. [9] proposed the variance tuning (VMI) to improve the transferability by reducing the variance of the gradient and tuning the current gradient with the neighbors.

Additionally, ensemble methods have been proven to be effective and compatible with previous techniques [4], but there is a dearth of research that solely averages logits, probabilities, or losses for ensemble creation. Xiong et al. [10] introduces the SVRG optimizer into ensemble attack to reduce the variance of the gradients during optimization. Chen et al. [11] are more effective. They encourage the cosine similarity between gradients of the ensemble models, which is the upper bound of the distance between local optimums, thus, it provably enhances the generalization ability, i.e., transferability. However, the algorithms in Chen et al. [11] are limited. First, their algorithms are ineffective because their methods have to promote the cosine similarity between gradients and loss with a fixed ratio $1 : \frac{\beta}{2}$. Besides, the interplay between loss and cosine similarity, which corresponds to the optimization and regularization, remains inadequately explored.

*Corresponding author



(a) MI (b) MI-ST

Fig. 1. Illustration of our method.

To this end, in our work, we propose MI-ST algorithm in order to avoid the constraint weights and explore the relationship between optimization and regularization. We derive an algorithm that could directly encourage the cosine similarity between gradients, and combine them with previous state-of-the-art strategies. We also did lots of ablation studies to analyze the relationship between the trade-off of optimization and regularization. Experimental results on ImageNet validate the effectiveness of our approach in improving adversarial transferability. On average, our method outperforms state-of-the-art attackers on 18 discriminative classifiers and adversarially trained models.

2. RELATED WORK

2.1. Adversarial attacks

Denote the original images as $\boldsymbol{x}^{real}$ and the adversarial examples as $\boldsymbol{x}^{adv}$. Let $\mathcal{F}$ represents the set of all image classifiers and $\mathcal{F}_t \subset \mathcal{F}$ represents the set of surrogate models. Meanwhile, we use $f(\cdot)$ to denote the classifiers and the L to denote the corresponding loss function (e.g. cross-entropy loss). Crafting adversarial examples could be formalized as an optimization problem:

$$\arg\max_{\boldsymbol{x}^{adv}} \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} L(f(\boldsymbol{x}^{adv}), \boldsymbol{y}),\ s.t.\ ||\boldsymbol{x}^{adv} - \boldsymbol{x}^{real}||_\infty \leq \epsilon. \quad (1)$$

This objective function means we need to find the adversarial sample which can make the maximize loss function over all target models. However, in real scenarios, attackers usually could not access the deployed model in $\mathcal{F}$. An alternative solution is to craft adversarial examples on surrogate models $\mathcal{F}_t$, and transfer them to target models, a.k.a. transfer attacks. That is,

$$\arg\max_{\boldsymbol{x}^{adv}} \frac{1}{|\mathcal{F}_t|} \sum_{f \in \mathcal{F}_t} L(f(\boldsymbol{x}^{adv}), \boldsymbol{y}),\ s.t.\ ||\boldsymbol{x}^{adv} - \boldsymbol{x}^{real}||_\infty \leq \epsilon. \tag{2}$$

2.2. Transfer attacks

Due to its effectiveness and simplicity, transfer attacks have garnered significant attention. Several methods have been devised to enhance transferability, and we classify them into three categories.

Gradient-based Methods: Drawing an analogy to the optimization and regularization in neural network training, Dong et al. [4] proposed the Momentum Iterative (MI) method and Lin et al. [12] proposed the Nesterov Iterative (NI) method, which introduce the momentum and the Nesterov accelerated gradient to prevent the adversarial examples from falling into the undesired local optima. Wang et al. [9] proposed the variance tuning (VMI) to reduce the variance of the gradient and tuning the current gradient with the gradient variance in the neighborhood of the previous data point. We will show that our method could act as a plug-and-play regularization term and be incorporated with these methods in Sec. 3.3.

Input Transformations: Analogous to the data augmentation, Xie et al. [8] proposed the Diverse Inputs (DI) method by applying random transformations to the input images at each iteration. Dong et al. [7] proposed Translation-Invariant (TI) method by optimizing a perturbation over an ensemble of translated images. Lin et al. [12] also propose to average the gradient of scaled copies of the input images. These methods add some preprocessing operation before feeding the input data into the neural network, thus it is orthogonal to our work.

Ensemble Attacks: Huang et al. [13] draw an analogy between the number of classifiers used in crafting adversarial examples and the size of training sets in neural network training, argue that increasing the surrogate models could reduce the generalization error upper bound. Xiong et al. [10] introduce the SVRG optimizer into ensemble attack to reduce the variance of the gradients during optimization. Chen et al. [11] proposed the common weakness of the ensemble models by showing that both the flatness of loss landscape and the distance between the local optimums are strongly correlated with the transferability, and the cosine similarity between gradients are upper bound of the latter term. However, the algorithms in Chen et al. [11] are limited. First, their algorithms are ineffective because their methods has to promote the cosine similarity between gradients and loss with a fixed ratio $1 : \frac{\beta}{2}$. Besides, the interplay between loss and cosine similarity, which correspond to the optimization and regularization, remains inadequately explored.

3. METHODOLOGY

To this end, by an insightful and complicate mathematical derivation, we propose an new algorithm called MI-ST, which could calculate the exact derivative of the cosine similarity between gradients, enable us to tradeoff between the gradient of original loss (optimization) and the gradient of cosine similarity (regularization). This section is organized as follows: We illustrate our algorithm according to Algorithm 1 and Fig. 1 in Sec. 3.1. We also provide our insightful mathematical derivation in Sec. 3.2. Finally we show demonstrate that how to combine our algorithm with previous state-of-the-art algorithm in Sec. 3.3.

Algorithm 1: ST attacker

Require: natural image $\boldsymbol{x}^{real}$, label $\boldsymbol{y}$, loss function L, surrogate models $\mathcal{F}_t = \{f_i\}_i^n$, perturbation budget ϵ, iterations T, learning rate β, loss weight λ_1, cosine weight λ_2.
for $t = 0 : T-1$ **do**
  Initialize: $\boldsymbol{x}_0 = \boldsymbol{x}^{real}$;
  for $i = 1 : n$ **do**
    Calculate $g_{i-1} = \nabla_{\boldsymbol{x}} L(f_{i-1}(\boldsymbol{x}_t^{i-1}), \boldsymbol{y})$;
    Update adversarial sample by
      $\boldsymbol{x}_t^i = clip_{\boldsymbol{x}^{real},\epsilon}(\boldsymbol{x}_t^{i-1} + \beta \cdot \frac{g_{i-1}}{||g_{i-1}||_2})$
  end
  Calculate $g_{mean} = \frac{1}{|\mathcal{F}_t|} \sum_i \frac{\nabla_{\boldsymbol{x}} L(f_i(\boldsymbol{x}_t), \boldsymbol{y})}{\|\nabla_{\boldsymbol{x}} L(f_i(\boldsymbol{x}_t), \boldsymbol{y})\|_2}$;
  $g_{cos} = \boldsymbol{x}_t^{new} - \boldsymbol{x}_t - \beta \boldsymbol{g}_{mean}$;
  Calculate $g_{whole} = \lambda_1 g_{mean} + \frac{2\lambda_2}{\beta^2} g_{cos}$;
  Update $\boldsymbol{x}_{t+1} = clip_{\boldsymbol{x}^{real},\epsilon}(\boldsymbol{x}_t + g_{whole})$
end
Return: $\boldsymbol{x}_T$

3.1. Our algorithm

As demonstrated in Algorithm 1 and Fig. 1, for a given natural images $\boldsymbol{x}^{real}$, we first iteratively perform gradient update using the normalized gradient get from i-th model:

$$\boldsymbol{x}_t^i = clip_{\boldsymbol{x}^{real},\epsilon}(\boldsymbol{x}_t^{i-1} + \beta \cdot \frac{g_{i-1}}{||g_{i-1}||_2}), \tag{3}$$

where $g_i = \nabla_{\boldsymbol{x}} L(f_i(\boldsymbol{x}), \boldsymbol{y})$, $clip_{\boldsymbol{x}^{real},\epsilon}$ operation is to clip the model into the ϵ neighborhood of the natural image $\boldsymbol{x}^{real}$ to control the perturbation budget. After this iterative update, we calculate the gradient of the original loss and our regularization term:

$$\begin{aligned} g_{mean} &= \frac{1}{|\mathcal{F}_t|} \sum_i^n \frac{\nabla_{\boldsymbol{x}} L(f_i(\boldsymbol{x}_t), \boldsymbol{y})}{\|\nabla_{\boldsymbol{x}} L(f_i(\boldsymbol{x}_t), \boldsymbol{y})\|_2}, \\ g_{cos} &= \boldsymbol{x}_t^{new} - \boldsymbol{x}_t - \beta g_{mean}. \end{aligned} \tag{4}$$

Finally we calculate the update and trade-off the optimization and regularization by λ_1 and λ_2:

$$\begin{aligned} g_{whole} &= \lambda_1 g_{mean} + \frac{2\lambda_2}{\beta^2} g_{cos}, \\ \boldsymbol{x}_{t+1} &= clip_{\boldsymbol{x}^{real},\epsilon}(\boldsymbol{x}_t + g_{whole}). \end{aligned} \tag{5}$$

Following [7, 10–12], we update the adversarial example by the gradient that integrated with momentum to stablize the update direction and avoid undesirable local optimum, as shown in Fig. 1(b), where the final direction is the vector sum of g_{whole} and momentum. We repeat the above process util convergence or exceeding the iteration time limit.

3.2. The mathematical derivation

In this section, we provide our derivation of proposed Algorithm 1.

Method	AlexNet	VGG16	GoogleNet	InceptionV3	ResNet152	DenseNet121	SqueezeNet	ShuffleNetV2	MobileNetV3	EfficientNetB0	MNasNet	RegNetX400MF	ConvNeXt	ViT-B/16	Swin-S	MaxViT
FGSM	76.4	68.9	54.4	54.5	54.5	57.4	85.0	81.2	58.9	50.8	64.1	57.1	39.8	33.8	34.0	31.3
BIM	54.9	86.1	76.6	64.9	96.0	93.0	80.4	65.3	55.6	80.2	80.8	81.1	68.6	35.0	48.2	49.7
MI	73.2	91.9	89.1	84.6	96.6	95.8	89.4	79.9	71.8	90.1	88.8	89.3	81.6	59.2	66.0	66.1
DI-MI	78.9	92.9	92.0	89.0	93.8	93.8	92.9	85.7	78.6	91.5	91.5	91.2	85.4	66.8	74.2	73.2
TI-MI	78.0	82.5	77.8	75.7	87.8	88.0	85.8	78.2	74.5	76.8	75.5	82.4	56.2	56.9	40.9	32.7
VMI	83.3	94.8	94.2	91.1	97.1	96.6	94.2	89.9	87.3	94.6	94.1	95.3	92.4	81.8	84.2	83.5
MI-SVRG	82.5	96.4	95.7	92.6	99.0	99.1	96.1	90.3	80.6	96.7	94.2	95.4	88.2	65.8	73.4	71.1
MI-SAM	81.0	95.6	94.4	89.2	97.9	98.0	94.1	87.9	80.7	95.2	94.3	93.9	90.1	68.9	75.1	75.6
MI-CSE	93.6	99.6	98.8	97.3	99.9	99.9	99.1	97.2	94.6	98.8	99.1	98.9	96.2	89.6	88.6	85.8
MI-CWA	94.6	99.5	99.0	97.2	99.8	99.8	99.3	97.3	95.7	98.9	98.7	99.4	95.4	89.6	87.6	85.9
MI-ST	94.2	**99.7**	**99.1**	97.1	**100.0**	**100.0**	99.0	96.9	95.2	**99.3**	98.9	98.9	**96.5**	**90.6**	88.5	**86.9**

Table 1. **Black-box attack success rate(%,↑) on NIPS2017 dataset**. Our method performs well on 16 normally trained models with various architectures.

Method	FGSMAT	EnsAT	FastAT	PGDAT	PGDAT	PGDAT	PGDAT†	PGDAT†
Backbone	InceptionV3	IncResV2	ResNet50	ResNet50	ResNet18	WRN50-2	XCiT-M	XCiT-L
FGSM	53.9	32.5	45.6	36.3	46.8	27.7	23.0	19.8
BIM	43.4	28.5	41.6	30.9	41.0	20.9	16.4	15.7
MI	55.9	42.5	45.7	37.4	45.7	27.8	22.8	19.8
DI-MI	61.8	52.9	47.1	38.0	47.7	31.3	25.4	21.7
TI-MI	66.1	58.5	49.3	43.9	50.7	37.0	29.4	26.9
VMI	72.3	66.4	51.4	47.1	48.9	36.2	33.4	30.8
MI-SVRG	66.8	46.8	51.0	43.9	48.5	33.0	30.2	26.7
MI-SAM	64.5	47.9	50.6	43.9	48.0	33.4	31.8	26.9
MI-CSE	89.6	78.2	75.0	73.5	68.4	64.4	77.5	71.0
MI-CWA	89.6	79.1	74.6	73.6	69.5	64.8	77.8	71.7
MI-ST	**90.0**	**81.4**	**75.8**	**75.1**	**69.7**	**65.1**	**78.4**	**72.2**

Table 2. **Black-box attack success rate(%,↑)**. Our method leads the performance on 8 adversarially trained models available on RobustBench. Note that PGDAT† is a variant of PGDAT tuned by bag of tricks. It turns out that our method improve the transferability of the adversarial examples.

Theorem 1 *When $\beta \to 0$, Updating by our Algorithm 1 is equivalent to optimizing:*

$$\max_{\boldsymbol{x}} \lambda_1 \frac{1}{|\mathcal{F}_t|} \sum_{f \in \mathcal{F}_t} L(f_i(\boldsymbol{x}), \boldsymbol{y}) + \lambda_2 \sum_{i,j}^{i<j<|\mathcal{F}_t|} \frac{g_i^T g_j}{\|g_i\| \|g_j\|} \tag{6}$$

Where $g_i = \nabla_{\boldsymbol{x}} L(f_i(\boldsymbol{x}), \boldsymbol{y})$, λ_1 and λ_2 is the trade-off hyperparameter setted in our algorithm.

Proof 1 *Denote g_i' as the gradient at i^{th} iteration in the inner loop, we can represent g_i' by g_i using Taylor expansion:*

$$\begin{aligned}
\boldsymbol{g}_i' &= \boldsymbol{g}_i + \boldsymbol{H}_i(\boldsymbol{x}_t^i - \boldsymbol{x}_t^0) \\
&= \boldsymbol{g}_i + \beta \boldsymbol{H}_i \sum_{j=1}^{i-1} \frac{\boldsymbol{g}_j'}{\|\boldsymbol{g}_j'\|_2} \\
&= \boldsymbol{g}_i + \beta \boldsymbol{H}_i \sum_{j=1}^{i-1} \frac{\boldsymbol{g}_j + o(\beta)}{\|\boldsymbol{g}_j + o(\beta)\|_2} \\
&= \boldsymbol{g}_i + \beta \boldsymbol{H}_i \sum_{j=1}^{i-1} \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2} + O(\beta^2).
\end{aligned}$$

Therefore, the update over the entire inner loop is :

$$\begin{aligned}
\boldsymbol{x}_t^n - \boldsymbol{x}_t^0 &= \beta \sum_{i=1}^{n} \frac{\boldsymbol{g}_i'}{\|\boldsymbol{g}_i'\|_2} \\
&= \beta \sum_{i=1}^{n} \frac{\boldsymbol{g}_i + \beta \boldsymbol{H}_i \sum_{j=1}^{i-1} \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2} + O(\beta^2)}{||\boldsymbol{g}_i + O(\beta)||_2} \\
&\approx \beta \sum_{i=1}^{n} \frac{\boldsymbol{g}_i + \beta \boldsymbol{H}_i \sum_{j=1}^{i-1} \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2} + O(\beta^2)}{||\boldsymbol{g}_i||_2}
\end{aligned}$$

Since:

$$\begin{aligned}
&\mathbb{E}[\frac{\partial}{\partial \boldsymbol{x}} \frac{\boldsymbol{g}_i \boldsymbol{g}_j}{\|\boldsymbol{g}_i\|_2 \|\boldsymbol{g}_j\|_2}] \\
=&\mathbb{E}[\frac{\boldsymbol{H}_i}{\|\boldsymbol{g}_i\|_2} \left(\boldsymbol{I} - \frac{\boldsymbol{g}_i \boldsymbol{g}_i^\top}{\|\boldsymbol{g}_i\|_2}\right) \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2} + \frac{\boldsymbol{H}_j}{\|\boldsymbol{g}_j\|_2} \left(\boldsymbol{I} - \frac{\boldsymbol{g}_j \boldsymbol{g}_j^\top}{\|\boldsymbol{g}_j\|_2}\right) \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2}] \\
\approx&\mathbb{E}[\frac{\boldsymbol{H}_i}{\|\boldsymbol{g}_i\|_2} \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2} + \frac{\boldsymbol{H}_j}{\|\boldsymbol{g}_j\|_2} \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2}] \\
=&2\mathbb{E}\left[\frac{\boldsymbol{H}_i}{\|\boldsymbol{g}_i\|_2} \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2}\right].
\end{aligned}$$

We can get :

$$\begin{aligned}
\mathbb{E}[\boldsymbol{x}_t^n - \boldsymbol{x}_t^0] &= \mathbb{E}[\beta \sum_{i=1}^{n} \frac{\boldsymbol{g}_i + \beta \boldsymbol{H}_i \sum_{j=1}^{i-1} \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2} + O(\beta^2)}{||\boldsymbol{g}_i||_2}] \\
&= \beta \mathbb{E}[\sum_{i=1}^{n} \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2}] + \beta^2 \mathbb{E}[\sum_{j=1}^{i-1} \frac{\boldsymbol{H}_i}{\|\boldsymbol{g}_i\|_2} \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|_2}] + O(\beta^3) \\
&\approx \beta \mathbb{E}[\sum_{i=1}^{n} \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2}] + \frac{\beta^2}{2} \mathbb{E}[\sum_{i,j}^{i<j} \frac{\partial \frac{\boldsymbol{g}_i \boldsymbol{g}_j}{\|\boldsymbol{g}_i\|_2 \|\boldsymbol{g}_j\|_2}}{\partial \boldsymbol{x}}] + O(\beta^3).
\end{aligned} \tag{7}$$

Hence, our final update g_{whole} is:

$$\begin{aligned}
g_{whole} &= \lambda_1 g_{mean} + \frac{2\lambda_2}{\beta^2} g_{cos} \\
&= \lambda_1 \mathbb{E}[\sum_{i=1}^{n} \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2}] + \frac{2\lambda_2}{\beta^2}[\boldsymbol{x}_t^{new} - \boldsymbol{x}_t - \beta g_{mean}] \\
&= \lambda_1 \mathbb{E}[\sum_{i=1}^{n} \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2}] + \frac{2\lambda_2}{\beta^2}[\frac{\beta^2}{2} \mathbb{E}[\sum_{i,j}^{i<j} \frac{\partial \frac{\boldsymbol{g}_i \boldsymbol{g}_j}{\|\boldsymbol{g}_i\|_2 \|\boldsymbol{g}_j\|_2}}{\partial \boldsymbol{x}}] + O(\beta^3)] \\
&= \lambda_1 \mathbb{E}[\sum_{i=1}^{n} \frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|_2}] + \lambda_2 \mathbb{E}[\sum_{i,j}^{i<j} \frac{\partial \frac{\boldsymbol{g}_i \boldsymbol{g}_j}{\|\boldsymbol{g}_i\|_2 \|\boldsymbol{g}_j\|_2}}{\partial \boldsymbol{x}}] + 2\lambda_2 O(\beta)
\end{aligned} \tag{8}$$

Hence, update by our Algorithm 1 is equivalent to minimizing:

$$\max_{\boldsymbol{x}} \lambda_1 \frac{1}{|\mathcal{F}_t|} \sum_{f \in \mathcal{F}_t} L(f_i(\boldsymbol{x}), \boldsymbol{y}) + \lambda_2 \sum_{i,j}^{i<j<|\mathcal{F}_t|} \frac{g_i^T g_j}{\|g_i\| \|g_j\|} \tag{9}$$

We get the result.

Remark 2 *As shown in Eq. (9), comparing with [11], our method has two advantages. First, we could tradeoff easily between the optimization (original loss) and regularization (the cosine similarity between gradient) by tuning λ_1 and λ_2. We do lots of exploration of this tradeoff in Sec. 4.3. Besides, the error term in our algorithm is $O(\beta)$ rather than $O(\beta^3)$. Since β is usually set to the value that larger than one, our method has much less approximation error.*

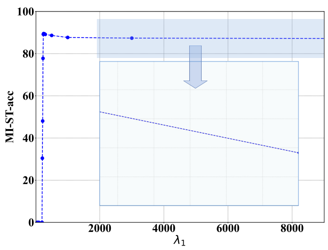


Fig. 2. Loss weight λ_1

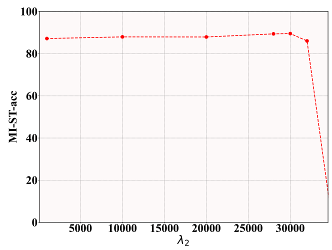


Fig. 3. Cosine weight λ_2

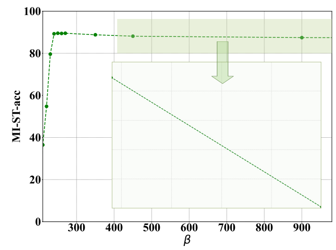


Fig. 4. Inner step size β

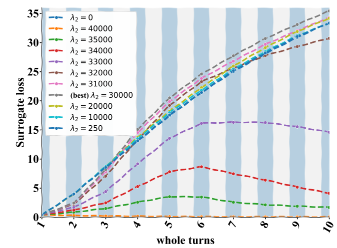


Fig. 5. Gap between λ_1 and λ_2

3.3. Incorporation with previous attackers

As shown in Algorithm 1, our method could be view as first calculating the update $g_{whole} = \lambda_1 g_{loss} + \lambda_2 g_{cos}$ and then perform gradient ascent. Since our method are orthogonal to input transformation methods like DI [8], TI [7], it can be incorporated with them seamlaessly to achieve improved performance. For gradient-based method, like MI [4], NI [12] and VMI [9], we could directly view g_{whole} as current gradients and calculating the corresponding momentums. Hence, our method is easy to be combined with other previous works to further improve the transferability.

4. EXPERIMENT

4.1. Experiment settings

We adopt exactly the same settings as [11]. **Dataset:** We use the NIPS2017 dataset, which is comprised of 1000 images selected from ImageNet. All the images are resized to 224×224. **Surrogate Models**: We choose four normally trained models, ResNet18, ResNet32, ResNet50, and ResNet101 from TorchVision [14] and two adversarially trained models, ResNet50 [15] and XCiT-S12 [16] from [17], which is effective to assess the method's ability to utilize diverse surrogate models. **Black-box Models:** We evaluate the attack success rate on 24 black-box models, including 16 normally trained models with different architectures and 8 adversarially trained models from RobustBench. **Compared Method:** We compare our methods with FGSM [2], BIM [18], MI [4], DI [8], TI [7], VMI [9], SVRG [10], SAM [11], CSE [11], CWA [11]. **Hyper-parameters:** We set the hyper-parameters as: perturbation threshold $\epsilon = 16/255$, total iteration rounds $T = 10$, momentum decay rate $\mu = 1$, learning rate $\beta = 250$, $\alpha = 16/255/5$.

4.2. Adversarial attacks on state-of-the-art models

Attacks on Discriminative Classifiers: As shown in Table 1, MI-ST achieves more than 80% attack success rate over all target models, indicating that even the state-of-the-art classifiers could be easily attacked in black-box setting. Besides, for models resembling any of the surrogate models, MI-ST results in a higher attack success rate. This could be attributed to MI-ST's encouragement of cosine similarity between gradients, effectively enhancing the optimization across all surrogate models simultaneously. For other models, MI-ST exhibits an improvement of at least approximately 20%, highlighting the strong generalization ability of our approach.

Attacks on Secured Models: As shown in Table 2, among all target models, MI-ST demonstrates an improvement of at least 30% over MI. Moreover, MI-ST surpasses the attack success rates of all preceding algorithms. These results demonstrate the strong effectiveness of MI-ST even against the most formidable defenses, underscoring the threat posed to deployed deep learning models by potential attacks.

4.3. Ablation studies

The gradient of the loss, controlled by λ_1, primarily governs the optimization of the loss function. Meanwhile, the gradient of the cosine term, governed by λ_2, play a crucial role in regularization and generalization. Therefore, we ought to carefully tradeoff between these two terms.

Loss weight λ_1: As shown in Fig. 2, an increase in λ_1 weakens the impact of the regularization, resulting in a slight decline in attack success rate. We also observe that the attack success rate of models that are not similar to the surrogate models drops significantly. This shows the regularization ability of cosine similarity between gradients, especially for models that are not similar to surrogate models. On the other hand, a too small λ_1 will leading to insufficient optimization of loss functions over the surrogate models, thus leading to a decline in the attack success rate.

Cosine weight λ_2: To further understand the impact towards optimization, we visualize the average losses over surrogate models with respect to λ_2 and T in Fig. 5. As shown, an excessively large λ_2 significantly impacts optimization, making it hard to maximize the loss function, resulting in degradation of the attack success rate for both surrogate and target models. Furthermore, as observed in Fig. 3, when λ_2 gradually decreases, the regularization effect gradually diminishes, leading to a gradual reduction in transferability.

Inner step size β: Based on Fig. 4, an increase in β leads to a larger error term in the Taylor expansion, resulting in a slight decrease in the attack success rate. This decline is more notable for defense models, as attacking such models demands more precise gradients. However, excessively reducing β can cause our model to converge into local optima and lead to insufficient optimization, significantly impacting the attack success rate.

5. CONCLUSION

In this paper, we propose MI-ST to boost adversarial attacks by promoting cosine similarity between the gradients of each model. We conduct extensive experiments to validate the effectiveness of the proposed methods and explain why they work in practice. To further improve the transferability of the generated adversarial examples, we did extensive experiments to find the best trade-off between optimization and regularization. Among 24 discriminative classifiers and defensed models, our method outperforms state-of-the-art attackers on 18 of them. The results identify the vulnerability of the current defenses, and raise security issues for the development of more robust deep learning models.

6. REFERENCES

[1] Nicholas Carlini and David Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*, 2017, pp. 39–57.

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

[3] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song, "Delving into transferable adversarial examples and black-box attacks," *arXiv preprint arXiv:1611.02770*, 2016.

[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*, 2018, pp. 9185–9193.

[5] Yinpeng Dong, Huanran Chen, Jiawei Chen, Zhengwei Fang, Xiao Yang, Yichi Zhang, Yu Tian, Hang Su, and Jun Zhu, "How robust is google's bard to adversarial image attacks?," *arXiv preprint arXiv:2309.11751*, 2023.

[6] Zeming Wei, Yifei Wang, and Yisen Wang, "Jailbreak and guard aligned language models with only few in-context demonstrations," *arXiv preprint arXiv:2310.06387*, 2023.

[7] 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*, 2019, pp. 4312–4321.

[8] 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*, 2019, pp. 2730–2739.

[9] 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*, 2021, pp. 1924–1933.

[10] Yifeng Xiong, Jiadong Lin, Min Zhang, John E Hopcroft, and Kun He, "Stochastic variance reduced ensemble adversarial attack for boosting the adversarial transferability," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14983–14992.

[11] Huanran Chen, Yichi Zhang, Yinpeng Dong, and Jun Zhu, "Rethinking model ensemble in transfer-based adversarial attacks," *arXiv preprint arXiv:2303.09105*, 2023.

[12] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks," *arXiv preprint arXiv:1908.06281*, 2019.

[13] Hao Huang, Ziyan Chen, Huanran Chen, Yongtao Wang, and Kevin Zhang, "T-sea: Transfer-based self-ensemble attack on object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 20514–20523.

[14] Sébastien Marcel and Yann Rodriguez, "Torchvision the machine-vision package of torch," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 1485–1488.

[15] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry, "Do adversarially robust imagenet models transfer better?," *Advances in Neural Information Processing Systems*, pp. 3533–3545, 2020.

[16] Edoardo Debenedetti, Vikash Sehwag, and Prateek Mittal, "A light recipe to train robust vision transformers," in *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2023, pp. 225–253.

[17] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein, "Robustbench: a standardized adversarial robustness benchmark," *arXiv preprint arXiv:2010.09670*, 2020.

[18] Jiakai Wang, "Adversarial examples in physical world.," in *IJCAI*, 2021, pp. 4925–4926.