Dynamic Generative Targeted Attacks with Pattern Injection

Weiwei Feng*, Nanqing Xu†*, Tianzhu Zhang1,2†, Yongdong Zhang1
1 University of Science and Technology of China, 2 Deep Space Exploration Lab
fengww@mail.ustc.edu.cn, xng@mail.ustc.edu.cn, {tzzhang, zhyd73}@ustc.edu.cn

Abstract

Adversarial attacks can evaluate model robustness and have been of great concern in recent years. Among various attacks, targeted attacks aim at misleading victim models to output adversary-desired predictions, which are more challenging and threatening than untargeted ones. Existing targeted attacks can be roughly divided into instance-specific and instance-agnostic attacks. Instance-specific attacks craft adversarial examples via iterative gradient updating on the specific instance. In contrast, instance-agnostic attacks learn a universal perturbation or a generative model on the global dataset to perform attacks. However, they rely too much on the classification boundary of substitute models, ignoring the realistic distribution of the target class, which may result in limited targeted attack performance. And there is no attempt to simultaneously combine the information of the specific instance and the global dataset. To deal with these limitations, we first conduct an analysis via a causal graph and propose to craft transferable targeted adversarial examples by injecting target patterns. Based on this analysis, we introduce a generative attack model composed of a cross-attention guided convolution module and a pattern injection module. Concretely, the former adopts a dynamic convolution kernel and a static convolution kernel for the specific instance and the global dataset, respectively, which can inherit the advantages of both instance-specific and instance-agnostic attacks. And the pattern injection module utilizes a pattern prototype to encode target patterns, which can guide the generation of targeted adversarial examples. Besides, we also provide rigorous theoretical analysis to guarantee the effectiveness of our method. Extensive experiments demonstrate that our method shows superior performance than 10 existing adversarial attacks against 13 models.

1. Introduction

With the encouraging progress of deep neural networks (DNNs) in various fields [6, 19, 16, 44, 42], recent studies

† Corresponding Author
MIM [9], DIM [53] and CD-AP [40], have made an impressive performance of boosting the transferability for untargeted attacks, which mislead the model to make an incorrect classification without specifying a target class. However, targeted attacks are more challenging compared with untargeted attacks, making the model output the adversary-desired target class. It is claimed in recent works [14, 55] that transferable targeted attacks are more worthy of study because attackers can directly control the unknown model to output the desired prediction, which can expose huge threats to data privacy and security. Therefore, it is of great significance to develop transferable targeted attacks.

Existing methods of transferable targeted attacks can be roughly categorized into instance-specific [9, 14, 13, 54, 50, 34, 31] and instance-agnostic [52, 35, 27, 40, 39] attacks. Specifically, almost all instance-specific attacks craft adversarial examples via iterative gradient updating, where attackers can only take advantage of the specific input instance, the white-box model and the target class label. Instance-specific attacks rely on optimizing the classification score of the adversary-desired class label to perturb the specific instance, which ignore the global data distribution. As a result, they inevitably lead to adversarial examples over-fitting the white-box model and result in modest transferability of targeted attacks. On the other hand, via learning a universal perturbation [37] or a generative attack model [52, 55], instance-agnostic attacks optimize adversarial perturbations on the global data distribution rather than the specific instance. To a certain extent, they can alleviate the problem of data-specific over-fitting and lead to more transferable targeted adversarial examples. However, taking the generative attack methods as an example, they suffer from two limitations. (1) Most generative attacks [52, 40, 35] still rely on the target label and the classification boundary information of white-box models rather than the realistic data distribution of the target class. Consequently, it is claimed that most generative attacks still have the possibility of over-fitting the white-box model, which may result in limited performance of transferable targeted attacks. (2) Another limitation is that existing generative attacks [39, 55, 27] apply the same network weights to every input instance in the test dataset. Nevertheless, it is considered that the shared network weights cannot stimulate the best attack performance of generative models [39, 55, 55]. Thus these aforementioned limitations have become the bottleneck of developing transferable targeted attacks, to a certain extent.

To address the aforementioned limitations, in this paper we construct a causal graph to formulate the prediction process of classifiers, and analyze the origin of adversarial examples. Based on this analysis, we propose to generate targeted adversarial examples via injecting the specific pattern or style of the target class. To this end, we introduce a generative attack model, which can not only inject pattern or style information of the target class to improve transferable targeted attacks, but also learn specialized convolutional kernels for each input instance. Specifically, we designed a cross-attention guided convolution module and a pattern injection module in the proposed generative attack model. (1) The cross-attention guided convolution module consists of a static convolutional kernel and a dynamic convolutional kernel that is computed according to the input instance. Consequently, this static and dynamic mixup module can not only encode the global information of the dataset, but also learn specialized convolutional kernels for each input instance. This paradigm makes our generative model inherit the advantages of both instance-specific and instance-agnostic attacks. (2) The pattern injection module is designed to model the pattern or style information of the target class and guide the generation of targeted adversarial examples. Concretely, we propose a pattern prototype to learn a global pattern representation over images from the target class, and use the prototype to guide the generation of more transferable targeted adversarial examples. And the generated adversarial images of our method are presented in Figure 1. It is observed that our generated perturbations (as shown in Figure 1(a)) pose strong semantic patterns or styles of the target class and show an underlying dependency on the input instance. In contrast, the perturbations (as presented in Figure 1(b)) generated by MIM [9] perform like random noises. Finally, to further understand our method, we provide rigorous theoretical analysis to guarantee the effectiveness of our method, as shown in Section 3.4, where we derive a concise conclusion based on the problem of Gaussian binary classification.

In summary, the main contributions of this paper are three-fold: (1) We propose a dynamic generative model to craft transferable targeted adversarial examples, which can not only inject pattern or style information of the target class to improve transferable targeted attacks, but also learn specialized convolutional kernels for each input instance. Besides, our method inherits the advantages of both instance-specific and instance-agnostic attacks, and to the best of our knowledge, we are the first to bridge them. (2) We state that injecting the specific pattern or style of the target class can improve the transferability of targeted adversarial examples, and we provide a comprehensive theoretical analysis to verify the rationality of this statement. (3) Extensive experimental results demonstrate that our method significantly boosts the transferability of targeted adversarial examples over 10 existing methods against 13 models.

2. Related Work

Instance-specific Attacks. As the pioneering work [48] exposes the vulnerability of neural networks, many recent methods [54, 17, 3, 11, 29] utilize gradient-based optimiza-
tion to generate input-dependent adversarial examples. To further boost the transferability of the adversarial example, several works have been proposed by various strategies to optimize the gradient update process. MIM [9] introduces a momentum term during the iterative gradient updating. DIM [53] improves the transferability via integrating diverse input patterns for optimizing, and TIM [10] performs a convolution operation on the gradient that is applicable to any gradient-based attack methods. Besides, some recent works [5, 18, 20] propose to ensemble multiple pre-trained substitute models to craft more transferable adversarial examples. However, these methods pose modest transferability under the targeted attacks setting, because they rely too much on the target label and the classification boundary information of white-box models. Meanwhile, among these methods, they all face the problem of data-specific overfitting, because of ignoring the global data distribution. To overcome these limitations, in this paper, we propose a dynamic generative model to craft more transferable targeted adversarial examples, which can not only encode the global information of the target class to improve transferable targeted attacks, but also learn specialized convolutional kernels for each input instance.

### 3. Method

In this section, we present the overall scheme of our proposed method, including some preliminaries, motivations, network architectures and theoretical analyses.

#### 3.1. Preliminaries and Motivation

We denote the clean image as $x \in \mathcal{X}$, and the attacked model parameterized by $\theta$ is denoted as $f : \mathcal{X} \rightarrow \mathcal{Y}$. The goal of target attacks is to generate an adversarial perturbation $\delta$ that can make the model misclassify the input $x_{\text{adv}} = x + \delta$ to the predefined class $y_i$, which can be formulated as $f(x_{\text{adv}}) = y_i \neq f(x)$. To gain insights into the origin of targeted adversarial examples, we construct a casual graph $\mathcal{G}$ to formulate the inference of deep learning models, as shown in Figure 2. For the input images $x$, we propose to group the whole causes of $x$ into two categories, content-related cause $C$ and content-independent cause $S$ that can be dubbed as style or pattern cause. This indicates that $C \rightarrow x \leftarrow S$, and $C \perp S$. According to human visual intuition, only the content variable $C$ is relevant to the prediction class $y$. Consequently, we can use the Law of total probability to expand the prediction $P_0(y \mid x)$ as:

$$P_0(y \mid x) = \sum_{s \in S} P_0(s \mid x) P_0(y \mid x, s). \quad (1)$$

But several recent works [56] and Equation (1) imply that deep learning models can learn not only the dependencies between the content $C$ and the label $y$, but also the statistical correlation between the style $S$ and the label $y$ (i.e., $P_0(y \mid x, s)$). This phenomenon indicates that deep learning models can capture some detailed texture patterns and style features to achieve good performance, and the output prediction $P_0(y \mid x)$ will change with the statistical correlation between $y$ and $S$ (i.e., $P_0(y \mid x, s)$). And this property certainly offers attackers an opportunity to generate adversarial examples for misleading the model. Therefore, the inference of targeted adversarial examples can be formulated as:

$$P_0(y_i \mid x_{\text{adv}}) = \sum_{s \in S} P_0(s \mid x_{\text{adv}}) P_0(y_i \mid x_{\text{adv}}, s). \quad (2)$$

Inspired by Figure 2 and the aforementioned Equation (2), we first propose to exploit the statistical correlation between $y_i$ and $S$ (i.e., $P_0(y_i \mid x_{\text{adv}}, s)$, via injecting the specific

---

1. The statistical correlation between $y_i$ and $S$ (i.e., $P_0(y_i \mid x_{\text{adv}}, s)$) reflects that the model learns some non-robust features. According to the analysis of [24] and [55], the transferability of adversarial examples may arise from non-robust features, such as texture patterns or styles.
style or pattern of images from the given target class \(y_t\) to generate targeted transferable adversarial examples.

To this end, we propose a generative model to craft targeted adversarial examples by injecting the pattern of the target class. Besides, we design a cross-attention guided dynamic convolution module in the generator, which makes our generative model inherit the advantages of both instance-specific and instance-agnostic attacks. Therefore, the formulation of our method can be denoted as

\[
x_{adv} = \text{clip}\left\{ \text{Proj}\left( W \ast G_\theta(x, p_t), -\varepsilon, \varepsilon \right) + x \right\},
\]

(3)

where \(\varepsilon\) is the perturbation budget, \(W\) is a smoothing operator with fixed weights, \(p_t\) represents the semantic pattern or style of the target class, and \(\text{Proj}\) is a projection operation.

### 3.2. Network Architecture

Specifically, our generative model is illustrated in Figure 3, which is composed of two modules: ① a cross-attention-guided dynamic convolution module (corresponding to “Dy_Conv” in the ResBlock of Figure 3), ② a pattern injection module. And the detailed pipeline of our approach is presented in Algorithm 1. In the next section, we mainly describe the two designed modules in detail.

**Cross-attention guided dynamic convolution module.** Considering that not only the global dataset needs to be modeled, but also specialized convolution kernels need to be learned for each input, we design a cross-attention guided convolution module. To achieve this goal, as shown in Figure 4, we propose a static and a dynamic convolution kernel parameterized with \(W\) and \(\Delta W\), respectively.\(^2\) Thus the designed module is formulated as:

\[
X_l = \text{conv}(X_{l-1}; W + \Delta W),
\]

(4)

where \(X_l\) is the output of layer \(l\). To get dynamic convolution kernels, we design a series of learnable kernels \((N \times C_{in} \times C_{out}\times k^2)\)^3 and attention queries \(q \in \mathbb{R}^{N \times C_{in}}\), which indicate the attention weight of each kernel. For the layer \(l\), we compute attention weights via performing cross-attention between attention queries \(q\) and the input \(X_{l-1} \in \mathbb{R}^{C_{in} \times H \times W}\), which can be indicated as:

\[
\text{att} = \text{softmax}\left( \frac{qK^\top}{\sqrt{d_k}} \right)V,
\]

(5)

\(^2\)Note that we ignore bias terms for the sake of brevity

\(^3\)\(N\) indicates the number of learnable kernels, \(k\) represents the kernel size, \(C_{in}\) and \(C_{out}\) are the number of input and output channels.

---

**Algorithm 1: Dynamic Generative Targeted Attack**

**Input**: A white-box model \(f\), training data \(\mathcal{X}\), target samples \(\mathcal{X}_t\), perturbation budget \(\varepsilon\), training epoch \(T\).

**Output**: Generative attack model \(G_\theta\).

1. for \(t \leftarrow 0\) to \(T\) do
2. Sample mini-batches \(x \sim \mathcal{X}\) and \(x_t \sim \mathcal{X}_t\).
3. Get the augmented mini-batches \(x'\) from \(x\).
4. Feed \(x\), \(x'\) and \(x_t\) into the generator to get adversarial examples \(x_{adv}\) and \(x'_{adv}\), where \(x_t\) is used for extracting and injecting target patterns \(p_t\).
5. Update \(p_t\) (\(\text{running}\) based on \(p_t\)) via EMA updating.
6. Forward pass \(x_{adv}\), \(x'_{adv}\) and \(x_t\) through \(f\).
7. Compute losses \(L_{\text{attack}}\), and backward for updating \(G_\theta\).
8. return Converged generator \(G_\theta\).
where \( att \in R^{N \times C_{in}} \), \( Q = qW_q \), \( K = X_{t-1}^T W_k \) and \( V = X_{t-1}^T W_v \). Then we can get the attention weight \( \alpha = [\alpha_1, \alpha_2, \cdots, \alpha_N] \in R^{1 \times N} \) by conducting the average pooling on \( att \), and each element in \( \alpha \) indicates the weight of corresponding kernels. Therefore, the dynamic convolution kernels are implemented as \( \Delta W = \alpha_1 \ast W_1 + \alpha_2 \ast W_2 + \cdots + \alpha_N \ast W_N \). Benefitting from this dynamic and static mixup convolution operation, our proposed generative attack model can inherit the advantages of both instance-specific and instance-agnostic attacks.

**Pattern injection module.** First, to encode target patterns and styles during the generation of adversarial examples, we design a pattern prototype \( p_t = \{\gamma_t, \beta_t\} \), which is extracted from the features of target samples, as shown in Figure 3. Following [25], we introduce \( p_t^{\text{running}} \) to model patterns of the global samples for the target class. Note that we update \( p_t^{\text{running}} \) during each training iteration, which can be indicated as \( p_t^{\text{running}} = \lambda p_t + (1 - \lambda)p_t^{\text{running}} \). Then we propose an adaptive class normalization (corresponding to “AdaCN” in Figure 3) to inject \( p_t \) into the generation of target adversarial examples. Similar to [23], the AdaCN module exploits the learned \( p_t = \{\gamma_t, \beta_t\} \) to perform an affine operation for injecting target patterns or styles:

\[
\text{AdaCN}(X) = \gamma_t \left( \frac{X - \mu(X)}{\sigma(X)} \right) + \beta_t,
\]

where \( p_t = \{\gamma_t, \beta_t\} \) is the target pattern prototype. Note that Equation (6) represents pattern injection during training, and it is necessary to replace \( p_t = \{\gamma_t, \beta_t\} \) with \( p_t^{\text{running}} = \{\gamma_t^{\text{running}}, \beta_t^{\text{running}}\} \) during inference.

**3.3. Objective Function**

To lead the victim model to misclassify adversarial examples as the target class, it is necessary to make the output distribution of adversarial examples \( f(x_{adv}) \) and target samples \( f(x_t) \) consistent. Thus we define the loss function as follows:

\[
\mathcal{L} = D_{KL}(f(x_{adv})||f(x_t)) + D_{KL}(f(x_t)||f(x_{adv}))
\]

\[
\mathcal{L}_{\text{aug}} = D_{KL}(f(x_{adv})||f(x_t)) + D_{KL}(f(x_t)||f(x_{adv}))
\]

Besides, similar to [39], we also introduce a local similarity loss. For a batch of generated adversarial examples \( \{x_{adv}^{i}\}_{i=1}^{n} \) and target samples \( \{x_t^{i}\}_{i=1}^{n} \), the similarity matrix can be computed as \( S_{i,j} = \frac{f(x_{adv}^{i}) \cdot f(x_t^{j})}{\|f(x_{adv}^{i})\| \cdot \|f(x_t^{j})\|} \) and \( S_{i,j} = \|f(x_t^{i}) - f(x_t^{j})\| \). Hence, we can get the local similarity loss as:

\[
\mathcal{L}_{\text{sim}} = \sum_{i,j} S_{i,j} \log \frac{S_{i,j}}{S_{i,j}^{\text{target}}} + \sum_{i,j} S_{i,j} \log \frac{S_{i,j}}{S_{i,j}^{\text{source}}},
\]

where \( S_{i,j}^{\text{target}} = \exp(S_{i,j}) \) and \( S_{i,j}^{\text{source}} = \exp(S_{i,j}^{t}) \). Finally, the total objective function can be formulated as:

\[
\mathcal{L}_{\text{attack}} = \mathcal{L} + \mathcal{L}_{\text{aug}} + \mathcal{L}_{\text{sim}}.
\]
Taking advantage of the method of Lagrange multipliers, we can easily get the optimal solution as follows:

\[
\delta^* = \frac{1}{2} \sum_{i=1}^{n} \mu_i,
\]

\[
\delta = \frac{1}{2\lambda} \left[ A^{-1} \left( A(x_s - \mu_s) + \mu_t - x_s \right) \right]. \tag{13}
\]

Going a step further, we rewrite the solution of \(\delta\) into a more concise formula as:

\[
\delta = C_1 \left[ \begin{array}{c}
\frac{\sigma_{t1}}{\sigma_{s1}} \\
\frac{\sigma_{t2}}{\sigma_{s2}} \\
\vdots \\
\frac{\sigma_{tn}}{\sigma_{sn}}
\end{array} \right] (x_s - \mu_s) + \mu_t - C_2 x_s, \tag{14}
\]

where \(C_1 = \frac{1}{2\lambda} A^{-1}\) and \(C_2 = \frac{1}{2\lambda} \Sigma^{-1} \Sigma\). In fact, note that the item of \(\mu_t\) represents target pattern or style injection, which is consistent with the previous works [23, 26]. Therefore, the formulation of Equation (14) shows a close underlying correlation between the optimal targeted adversarial perturbation and the embedding of target pattern or style, which also theoretically guarantees the effectiveness of our proposed generative model for targeted attacks.

4. Experiments

In this section, we conduct extensive experiments to demonstrate the effectiveness of our method for targeted attacks under various settings. Please feel free to get more experimental results in our Supplementary Material.

4.1. Experimental Setup


Victim Models. We consider 13 attacked models in our experiments. The normally trained models include Inception-v3 (Inc-v3) [47], Inception-v4 (Inc-v4) [45], DenseNet-121 [22], GoogleNet [46], ResNet-152 (Res-152) [19], and VGG16 (Vgg-16) [44] and Inception-ResNet (Inc-Res-v2) [45]. Besides, we also consider several models with robust training mechanisms, including adv-Inception-v3 (Adv-Inc-v3) [17], ens-adv-Inception-ResNet-v2 (Ens-Adv-IncRes-v2) [18] and ResNet-50 trained with various robust training tricks [21, 15].

Baseline Attacks. To illustrate the effectiveness of our method, we compare it with instance-specific attacks and instance-agnostic attacks. Instance-specific attacks mainly include MIM [9], DIM [53], SIM [32] and TIM [10], while...
Table 2. The attack success rates against robust models on ImageNet NeurIPS validation set. The perturbation budget $\ell_{\infty} \leq 16/255$. Adv-Inc-v3 [29] and Ens-Adv-Inc-Res-v2 [49] are trained with the adversarial training mechanism. Res50_SIN (stylized ImageNet), Res50_SIN_IN (mixture of stylized ImageNet and Nature ImageNet) and Res50_SIN_fine_IN (mixture of stylized ImageNet and Nature ImageNet with finetuning tricks) are trained with auxiliary dataset [15], and Res50_Augmix [21] is trained with the state-of-the-art data augmentation approach. Note that the results are averaged on 8 different target classes.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Res152</td>
<td>MIM</td>
<td>0.19</td>
<td>0.15</td>
<td>0.28</td>
<td>1.38</td>
<td>2.75</td>
<td>0.78</td>
</tr>
<tr>
<td></td>
<td>TI-MIM</td>
<td>0.61</td>
<td>0.73</td>
<td>0.50</td>
<td>2.51</td>
<td>4.75</td>
<td>1.76</td>
</tr>
<tr>
<td></td>
<td>SI-MIM</td>
<td>0.24</td>
<td>0.24</td>
<td>0.39</td>
<td>0.66</td>
<td>0.84</td>
<td>0.36</td>
</tr>
<tr>
<td></td>
<td>DIM</td>
<td>0.63</td>
<td>0.37</td>
<td>0.94</td>
<td>8.50</td>
<td>14.22</td>
<td>3.77</td>
</tr>
<tr>
<td></td>
<td>TI-DIM</td>
<td>0.23</td>
<td>0.30</td>
<td>0.28</td>
<td>0.76</td>
<td>1.49</td>
<td>0.49</td>
</tr>
<tr>
<td></td>
<td>SI-DIM</td>
<td>0.71</td>
<td>0.71</td>
<td>0.75</td>
<td>2.73</td>
<td>3.89</td>
<td>1.37</td>
</tr>
<tr>
<td></td>
<td>CD-AP</td>
<td>3.77</td>
<td>6.48</td>
<td>7.09</td>
<td>63.72</td>
<td>76.79</td>
<td>49.67</td>
</tr>
<tr>
<td></td>
<td>TTP</td>
<td>27.99</td>
<td>26.08</td>
<td>24.61</td>
<td>72.47</td>
<td>74.51</td>
<td>70.96</td>
</tr>
<tr>
<td></td>
<td>GAP</td>
<td>5.72</td>
<td>4.51</td>
<td>7.33</td>
<td>71.04</td>
<td>83.64</td>
<td>52.07</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td><strong>31.10</strong></td>
<td><strong>30.07</strong></td>
<td><strong>27.70</strong></td>
<td><strong>77.13</strong></td>
<td><strong>80.55</strong></td>
<td><strong>76.78</strong></td>
</tr>
<tr>
<td>VGG16</td>
<td>MIM</td>
<td>0.14</td>
<td>0.15</td>
<td>0.16</td>
<td>0.40</td>
<td>0.34</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>TI-MIM</td>
<td>0.26</td>
<td>0.24</td>
<td>0.20</td>
<td>0.45</td>
<td>0.57</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td>SI-MIM</td>
<td>0.28</td>
<td>0.20</td>
<td>0.21</td>
<td>0.49</td>
<td>0.25</td>
<td>0.14</td>
</tr>
<tr>
<td></td>
<td>DIM</td>
<td>0.22</td>
<td>0.16</td>
<td>0.27</td>
<td>0.93</td>
<td>0.99</td>
<td>0.49</td>
</tr>
<tr>
<td></td>
<td>TI-DIM</td>
<td>0.23</td>
<td>0.19</td>
<td>0.21</td>
<td>0.35</td>
<td>0.34</td>
<td>0.21</td>
</tr>
<tr>
<td></td>
<td>SI-DIM</td>
<td>0.50</td>
<td>0.36</td>
<td>0.33</td>
<td>0.80</td>
<td>0.69</td>
<td>0.26</td>
</tr>
<tr>
<td></td>
<td>CD-AP</td>
<td>0.36</td>
<td>0.34</td>
<td>0.35</td>
<td>4.63</td>
<td>10.20</td>
<td>3.60</td>
</tr>
<tr>
<td></td>
<td>TTP</td>
<td>3.75</td>
<td>3.20</td>
<td><strong>2.66</strong></td>
<td>27.80</td>
<td>32.70</td>
<td>16.57</td>
</tr>
<tr>
<td></td>
<td>GAP</td>
<td>0.30</td>
<td>0.52</td>
<td>0.42</td>
<td>4.52</td>
<td>8.92</td>
<td>3.35</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td><strong>4.14</strong></td>
<td><strong>3.22</strong></td>
<td><strong>2.66</strong></td>
<td><strong>30.16</strong></td>
<td><strong>38.10</strong></td>
<td><strong>17.95</strong></td>
</tr>
</tbody>
</table>

Figure 5. Attack success rates of targeted adversarial examples generated by various generative attack methods against different input process defense methods. Figure (a) displays the results against various input smooth methods (including gaussian, medium and average smoothing). Figure (b) shows the results against JPEG compression (the JPEG quality factor varies from 70 to 90). Here, the substitute model is Res-152 and the target model is Vgg-16.

instance-agnostic attacks include CD-AP [40], TTP [39], GAP [41] and C-GSP [55]. In our experiments, the perturbation size $\varepsilon = 16/255$, the decay factor $\mu_0$ is 1 in MIM, the transform probability is 0.7 in DIM, and the kernel size is 15 in TIM. And other hyper-parameters follow the default settings provided in their original works.

4.2. Comparison with State-of-the-art Methods

Attack Normally Trained Models. To evaluate the performance of our method, we first perform targeted adversarial attacks to compare the transferability across normally trained models of adversarial examples generated by various methods, including 6 iterative instance-specific attack methods and 5 instance-agnostic attack methods. As shown in Table 1, our dynamic generative attack method reaches the best transferability on 16 out of 18 various black-box cases. Furthermore, the attack success rate of our method against normally trained models is 65.42% on average with the substitute model as Res-152, which outperforms the best of baselines [39] by 5.28% on average. Another interesting discovery is that iterative instance-specific attack methods can reach better performance under the setting of white-box attacks, while generative attack methods show more transferability when adapting the adversarial examples to a black-box model. Among these generative methods, our method can still generate targeted adversarial examples with higher transferability than others.

Attack Models with Robust Training Mechanisms. To comprehensively verify the effectiveness of our method, we compare the transferability of our approach with baselines against several robust models, which are trained with various robust training strategies. As presented in Table 2, although the attack success rate against these robust models is relatively low, our method is still able to outperform baseline methods. Our generative attack method achieves the best performance of targeted transferable adversarial attacks on 11 over 12 cases. And regarding Res-152 as the substitute model, our method outperforms the best of baselines [39] by 4.45% on average. Compared with normally trained models, models with adversarial or robust training strategies are difficult to attack successfully, which indicates they are more robust than normal models.

Attack Models with Input Process Defense. Another widely used and simple defense method is Input Process, which leverages image processing techniques to remove
4.3. Visualization and Ablation Studies

Visualization. To vividly demonstrate the working mechanism of our method, we visualize several adversarial examples and perturbations with different target classes. As shown in Figure 6, there is an underlying dependency between the generated perturbations and input instances, where the perturbations are mainly concentrated on the semantical part of the input images. Besides, for different target classes, our method tends to generate perturbations with different texture patterns, which also verifies the effectiveness of our designed pattern injection module.

Ablation Studies. To look deeper into our proposed method, in this section, we present a series of ablation studies. As shown in Figure 7, we vary the number of target samples and convolution kernels to verify the effectiveness of our method. As presented in Figure 7(a), the performance of our method continues to grow until the number of target samples reaches 1000. Besides, in Figure 7(b), when the number of kernels is greater than 4, our performance reaches a plateau, which means that it is sufficient to reach an impressive attack success rate by learning 4 kernels in each designed dynamic convolution module.

5. Conclusion

In this paper, we first construct a causal graph to expose the origin of targeted adversarial examples, which motivates us to inject target patterns or styles for generating transferable target adversarial examples. Then we introduce a dynamic generative attack model composed of a cross-attention guided convolution module and a pattern injection module. Our generative attack model can not only inject pattern or style information of the target class to improve transferable targeted attacks, but also learn specialized convolutional kernels for each input instance, which inherits the advantages of both instance-specific and instance-agnostic attacks. Moreover, we also provide rigorous theoretical analysis to guarantee the effectiveness of our method, and extensive experiments demonstrate that our method performs better than state-of-the-art targeted attack methods.

6. Acknowledgement

This work was partially supported by the National Nature Science Foundation of China (Grant 62022078, Grant 62021002), and National Defense Basic Scientific Research Program (Grant JCKY2022911B002).
References


[34] Yantao Lu, Yunhan Jia, Jianyu Wang, Xiaolin Hu, Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, Alan Yuille, Sangxia Huang, Yao Zhao, Yuzhe Zhao, Zhonglin Han, Junjiajia Long, Yerkebulan Berdibekov, Takuya Akiba, Seiya Tokui, and Motoki Abe. Adversarial attacks and defenses competition, 2018. 6


[38] Aishan Liu, Jiakai Wang, Xianglong Liu, Bowen Cao, Chongzhi Zhang, and Hang Yu. Bias-based universal adversarial patch attack for automatic check-out, 2020. 1

[39] Yantao Lu, Yunhan Jia, Jianyu Wang, Xiaolin Hu, Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, Alan Yuille, Sangxia Huang, Yao Zhao, Yuzhe Zhao, Zhonglin Han, Junjiajia Long, Yerkebulan Berdibekov, Takuya Akiba, Seiya Tokui, and Motoki Abe. Adversarial attacks and defenses competition, 2018. 6


[56] Yonggang Zhang, Mingming Gong, Tongliang Liu, Gang Niu, Xinmei Tian, Bo Han, Bernhard Schölkopf, and Kun Zhang. Adversarial robustness through the lens of causality. In International Conference on Learning Representations, 2021. 3