Improving Black-box Adversarial Attacks with a Transfer-based Prior

Shuyu Cheng*, Yinpeng Dong*, Tianyu Pang, Hang Su, Jun Zhu
Dept. of Comp. Sci. and Tech., Tsinghua University
{chengsy18, dyp17, pty17}@ mails.tsinghua.edu.cn,
{suhangss, dcszj}@mail.tsinghua.edu.cn
Background

- An adversarial example should be **visually indistinguishable** from the corresponding normal one, but yet are **misclassified** by the target model.

- **Adversarial attacks** find such examples.
Adversarial Attacks

- Goal: Given classifier $C(x)$ and input-label pair $(x, y)$, find an adversarial example $x^{\text{adv}}$ such that
  \[ C(x^{\text{adv}}) \neq y, \text{ s.t. } \|x^{\text{adv}} - x\|_p \leq \epsilon. \]

- $x^{\text{adv}}$ can be generated by solving
  \[ x^{\text{adv}} = \arg \max_{x'} f(x', y) \quad \text{subject to } x' : \|x' - x\|_p \leq \epsilon \]

- $f$ is a loss function that we need to maximize in attacks. In untargeted attacks, it can be:
  - Cross entropy loss of the original label $y$
  - C&W loss $\max_{i \neq y} Z(x)_i - Z(x)_y$, $Z(x)$ is the logit

- 0-surface is the decision boundary
White-box Attacks

- Projected gradient ascent (PGD)
  \[ x_{t+1}^{\text{adv}} = \Pi_{B_p(x, \epsilon)}(x_t^{\text{adv}} + \eta \cdot g_t) \]

  - \( \Pi \) is the projection operation
  - \( B_p(x, \epsilon) \) is the \( \ell_p \) ball centered at \( x \) with radius \( \epsilon \)
  - \( g_t \) is the normalized gradient under the \( \ell_p \) norm

  - \( p = 2 \): \( g_t = \frac{\nabla_x f(x_t^{\text{adv}}, y)}{\|\nabla_x f(x_t^{\text{adv}}, y)\|_2} \)
  - \( p = \infty \): \( g_t = \text{sign}(\nabla_x f(x_t^{\text{adv}}, y)) \)

- Key: We need to know \( \nabla_x f(x_t^{\text{adv}}, y) \)
  - In the following part, we omit the dependency w.r.t. \( y \), write the objective as \( f(x) \) and write the gradient as \( \nabla f(x) \).
Black-box Attacks

- Transfer-based
  - Generate adversarial examples against white-box models, and leverage transferability for attacks
  - Require no knowledge of the target model, no queries
  - Need white-box models (datasets), assumes similarity

- Query-based
  - Get some information from the target model directly, through queries
    - Score-based
    - Decision-based
  - Goal: Improve success rate (e.g., success rate under 10000 queries) and save queries
Score-based Attacks

- Query loss function $f(x)$ given $x$

- We need to maximize $f(x)$ until attack succeeds.

- Gradient-based method: **Estimate** $\nabla f(x)$ by queries, and apply first-order optimization methods.
Random Gradient-Free (RGF) Method

- \( \hat{g} = \frac{1}{q} \sum_{i=1}^{q} \hat{g}_i \), where \( \hat{g}_i = \frac{f(x+\sigma u_i)-f(x)}{\sigma} \cdot u_i \)

- \( \{u_i\}_{i=1}^{q} \) are i.i.d. r.v. sampled from a distribution on \( \mathbb{R}^D \).

- In ordinary RGF method, \( u_i \) is sampled uniformly from the \( D \)-dimensional Euclidean hypersphere.

- \( \hat{g}_i \approx u_i u_i^T \nabla f(x) \) \(^{[1]}\)

- Pros: Unbiased

- Cons: High variance

- How to improve: Incorporating informative priors

- Evaluation metric / Loss function: Something like MSE?

---

1. Assume \( f \) is differentiable and \( \sigma \to 0 \).
Gradient estimation framework

- Suppose we want to choose a best estimator in the set $G$ of all possible gradient estimators, so we want to design a loss function for a gradient estimator.

- Our loss function for $\hat{g}$:

\[
L(\hat{g}) = \min_{b \geq 0} \mathbb{E} \| \nabla f(x) - b \hat{g} \|_2^2
\]

- Minimized mean square error w.r.t. the scale coefficient $b$
  - Usually the normalized gradient is used, hence the norm does not matter
Application to the RGF estimator

- For example, when $\hat{g}$ is an RGF estimator with $u_i$ i.i.d. sampled from any distribution on the hypersphere:

- **Theorem 1.** Suppose $\|u_i\|_2 = 1$ in the RGF method. If $f$ is differentiable at $x$, the loss of the RGF estimator $\hat{g}$ is

$$
\lim_{\sigma \to 0} L(\hat{g}) = 
\frac{\|\nabla f(x)\|_2^2 - \left( (\nabla f(x)^\top C \nabla f(x)) \right)^2}{\left( 1 - \frac{1}{q} \right) \nabla f(x)^\top C^2 \nabla f(x) + \frac{1}{q} \nabla f(x)^\top C \nabla f(x)}
$$

where $C = \mathbb{E}[u_i u_i^\top]$. 

Prior-guided RGF (P-RGF) method

- For the ordinary RGF estimator, $\mathbf{C} = \frac{\mathbf{I}}{D}$. Any better one?
- Suppose we know $\mathbf{v}$, the normalized ($\|\mathbf{v}\|_2 = 1$) transfer gradient of a surrogated model. Then we can design $\mathbf{C}$ as
  
  $$
  \mathbf{C} = \lambda \mathbf{v}\mathbf{v}^\top + \frac{1 - \lambda}{D - 1} (\mathbf{I} - \mathbf{v}\mathbf{v}^\top)
  $$

  which can be implemented by $u_i = \sqrt{\lambda} \cdot \mathbf{v} + \sqrt{1 - \lambda} \cdot (\mathbf{I} - \mathbf{v}\mathbf{v}^\top)\xi_i$, where $\xi_i$ is sampled uniformly from the unit hypersphere.
Prior-guided RGF (P-RGF) method

- $\lambda = \frac{1}{D} \approx 0$ is ordinary RGF estimator: unbiased, high variance
- $\lambda = 1$ corresponds to $u_i = v$, i.e. directly using the transfer gradient without queries: highly biased, no variance
- We need to find the optimal $\lambda$.

\[
\mathbb{E}[u_i u_i^\top] = \lambda v v^\top + \frac{1 - \lambda}{D - 1} (I - vv^\top)
\]
Solving for the optimal $\lambda$

- Let $\alpha = \nu^\top \nabla f(x)$ where $\nabla f(x)$ is the $l_2$ normalization of the true gradient $\nabla f(x)$.
  - $\alpha$ denotes the usefulness of the prior $\nu$

- By our gradient estimation framework, the optimal $\lambda$ is solved by minimizing $L(\hat{g})$.

$$
\lambda^* = \begin{cases} 
0 & \text{if } \alpha^2 \leq \frac{1}{D + 2q - 2} \\
\frac{(1 - \alpha^2)(\alpha^2(D + 2q - 2) - 1)}{2\alpha^2Dq - \alpha^4D(D + 2q - 2) - 1} & \text{if } \frac{1}{D + 2q - 2} < \alpha^2 < \frac{2q - 1}{D + 2q - 2} \\
1 & \text{if } \alpha^2 \geq \frac{2q - 1}{D + 2q - 2}
\end{cases}
$$
Estimating $\alpha$

- The ground truth value of $\alpha = \nabla f(x)^T v$ is not accessible, which needs an estimation\[1\].

- Note that $\alpha = \frac{\nabla f(x)^T v}{||\nabla f(x)||_2}$. The numerator is easy to estimate by finite difference. Hence the key problem is to estimate $||\nabla f(x)||_2$.

  - Finite difference: $\nabla f(x)^T v \approx \frac{f(x+\sigma v) - f(x)}{\sigma}$.

- Good thing: A scalar is much easier to estimate than a vector!

---

1. $\bar{x}$ denotes the $\ell_2$ normalization of $x$ in this work.
Norm estimation: The framework

- Suppose by $S$ queries, we can get $\nabla f(x)^T w_1, ..., \nabla f(x)^T w_S$ by finite difference, and $\|w_i\| = 1$.

- If we have a $S$-variable function $g$ such that
  $$g(ax_1, ax_2, ..., ax_n) = a^r g(x_1, x_2, ..., x_n)$$

- Then
  $$g(w_1^T \nabla f(x), ..., w_S^T \nabla f(x)) = \|\nabla f(x)\|_2^r \cdot g(w_1^T \nabla f(x), ..., w_S^T \nabla f(x))$$

- Hence

Each $w_s^T \nabla f(x)$ can be estimated by finite difference!

$$\mathbb{E} \left[ g \left( w_1^T \nabla f(x), ..., w_S^T \nabla f(x) \right) \right]$$

is an unbiased estimator of $\|\nabla f(x)\|_2^r$.

The expectation can be computed when each $w_s$ is uniformly distributed on the sphere!
Norm estimation

Here, we choose

\[ g(z_1, z_2, \ldots, z_S) = \frac{1}{S} \sum_{s=1}^{S} z_s^2 \]

when \( r = 2 \).

Then the estimator of \( \|\nabla f(x)\|_2 \) is

\[ \|\nabla f(x)\|_2 \approx \sqrt{\frac{D}{S} \sum_{s=1}^{S} (w_s^T \nabla f(x))^2} \]

where \( \{w_s\}_{s=1}^{S} \) is i.i.d. uniformly sampled from the unit hypersphere.
Summary of the P-RGF method

**Algorithm 1** Prior-guided random gradient-free (P-RGF) method

**Input:** The black-box model $f$; input $x$ and label $y$; the normalized transfer gradient $v$; sampling variance $\sigma$; number of queries $q$; input dimension $D$.

**Output:** Estimate of the gradient $\nabla f(x)$.

1: Estimate the cosine similarity $\alpha = v^T \nabla f(x)$ (detailed in Sec. 3.3);
2: Calculate $\lambda^*$ according to Eq. (12) given $\alpha$, $q$, and $D$;
3: if $\lambda^* = 1$ then
4: return $v$;
5: end if
6: $\hat{g} \leftarrow 0$;
7: for $i = 1$ to $q$ do
8: Sample $\xi_i$ from the uniform distribution on the $D$-dimensional unit hypersphere;
9: $u_i = \sqrt{\lambda^*} \cdot v + \sqrt{1 - \lambda^*} \cdot (I - vv^T) \xi_i$;
10: $\hat{g} \leftarrow \hat{g} + \frac{f(x + \sigma u_i, y) - f(x, y)}{\sigma} \cdot u_i$;
11: end for
12: return $\nabla f(x) \leftarrow \frac{1}{q} \hat{g}$. 
Incorporating data-dependent prior

- Restrict the adversarial perturbations to lie in a $d$-dimensional linear subspace spanned by $\{v_1, v_2, \ldots, v_d\}$

- For example, for $4 \times 4 \times 1$ images, $D = 16$, $d = 4$, we choose the subspace to be “in lower resolution”:

\[ v_1 = \quad v_2 = \]
\[ v_3 = \quad v_4 = \]
Incorporating data-dependent prior

- To perform the RGF method incorporating data-dependent prior, we need to set

\[ C = \frac{1}{d} \sum_{i=1}^{d} v_i v_i^T \]

- To further incorporate the transfer-based prior, we can set

\[ C = \lambda vv^T + \frac{1 - \lambda}{d} \sum_{i=1}^{d} v_i v_i^T \]

- Similarly we can obtain the optimal \( \lambda \).
Performance of gradient estimation

- Average cosine similarity between the gradient estimate and the true gradient:

- which shows the effectiveness of the derived optimal $\lambda$ (i.e., $\lambda^*$) for gradient estimation compared with any fixed $\lambda \in [0,1]$
Performance of gradient estimation

- Cosine similarity (averaged over all images) between the gradient estimate and the true gradient w.r.t. attack iterations:

- The transfer gradient is more useful at the beginning and less useful later
  - Showing the advantage of using adaptive $\lambda^*$
Gradient averaging

- Alternative method of biased sampling
  - Also integrate the transfer-based prior into the query-based algorithm
    \[ \hat{g} = (1 - \mu)v + \mu\hat{g}^U \]
  - \( \hat{g}^U \) is the normalized ordinary RGF estimator with \( C = \frac{1}{d} \)
  - The optimal coefficient \( \mu^* \) can be derived by the gradient estimation framework too.
Results of black-box attacks on normal models

<table>
<thead>
<tr>
<th>Methods</th>
<th>Inception-v3</th>
<th>VGG-16</th>
<th>ResNet-50</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR</td>
<td>AVG. Q</td>
<td>ASR</td>
</tr>
<tr>
<td>NES</td>
<td>95.5%</td>
<td>1718</td>
<td>98.7%</td>
</tr>
<tr>
<td>Bandits_T</td>
<td>92.4%</td>
<td>1560</td>
<td>94.0%</td>
</tr>
<tr>
<td>Bandits_TD</td>
<td>97.2%</td>
<td>874</td>
<td>94.9%</td>
</tr>
<tr>
<td>AutoZoom</td>
<td>85.4%</td>
<td>2443</td>
<td>96.2%</td>
</tr>
<tr>
<td>RGF</td>
<td>97.7%</td>
<td>1309</td>
<td>99.8%</td>
</tr>
<tr>
<td>P-RGF (λ = 0.5)</td>
<td>96.5%</td>
<td>1119</td>
<td>97.3%</td>
</tr>
<tr>
<td>P-RGF (λ*)</td>
<td>98.1%</td>
<td>745</td>
<td>99.8%</td>
</tr>
<tr>
<td>Averaging (μ = 0.5)</td>
<td>96.9%</td>
<td>1140</td>
<td>94.6%</td>
</tr>
<tr>
<td>Averaging (μ*)</td>
<td>97.9%</td>
<td>735</td>
<td>99.8%</td>
</tr>
<tr>
<td>RGF_D</td>
<td>99.1%</td>
<td>910</td>
<td>100.0%</td>
</tr>
<tr>
<td>P-RGF_D (λ = 0.5)</td>
<td>98.2%</td>
<td>1047</td>
<td>99.3%</td>
</tr>
<tr>
<td>P-RGF_D (λ*)</td>
<td>99.1%</td>
<td>649</td>
<td>99.7%</td>
</tr>
<tr>
<td>Averaging_D (μ = 0.5)</td>
<td>99.2%</td>
<td>768</td>
<td>99.9%</td>
</tr>
<tr>
<td>Averaging_D (μ*)</td>
<td>99.2%</td>
<td>644</td>
<td>99.8%</td>
</tr>
</tbody>
</table>

- **ASR**: Attack Success Rate (#queries is under 10,000);
- **AVG. Q**: Average #queries over successful attacks.
- Methods with the subscript “D” refers to the data-dependent version of the P-RGF method.
## Results on defensive models

<table>
<thead>
<tr>
<th>Methods</th>
<th>JPEG Compression</th>
<th></th>
<th>Randomization</th>
<th></th>
<th>Guided Denoiser</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR</td>
<td>AVG. Q</td>
<td>ASR</td>
<td>AVG. Q</td>
<td>ASR</td>
<td>AVG. Q</td>
</tr>
<tr>
<td>NES</td>
<td>47.3%</td>
<td>3114</td>
<td>23.2%</td>
<td>3632</td>
<td>48.0%</td>
<td>3633</td>
</tr>
<tr>
<td>SPSA</td>
<td>40.0%</td>
<td>2744</td>
<td>9.6%</td>
<td>3256</td>
<td>46.0%</td>
<td>3526</td>
</tr>
<tr>
<td>RGF</td>
<td>41.5%</td>
<td>3126</td>
<td>19.5%</td>
<td>3259</td>
<td>50.3%</td>
<td>3569</td>
</tr>
<tr>
<td>P-RGF</td>
<td>61.4%</td>
<td>2419</td>
<td>60.4%</td>
<td>2153</td>
<td>51.4%</td>
<td>2858</td>
</tr>
<tr>
<td>Averaging</td>
<td>69.4%</td>
<td>2134</td>
<td>72.8%</td>
<td>1739</td>
<td>66.6%</td>
<td>2441</td>
</tr>
<tr>
<td>RGF(_D)</td>
<td>70.4%</td>
<td>2828</td>
<td>54.9%</td>
<td>2819</td>
<td>83.7%</td>
<td>2230</td>
</tr>
<tr>
<td>P-RGF(_D)</td>
<td>81.1%</td>
<td>2120</td>
<td>82.3%</td>
<td>1816</td>
<td>89.6%</td>
<td>1784</td>
</tr>
<tr>
<td>Averaging(_D)</td>
<td>80.6%</td>
<td>2087</td>
<td>77.4%</td>
<td>1700</td>
<td>87.2%</td>
<td>1777</td>
</tr>
</tbody>
</table>

- **ASR**: Attack Success Rate (#queries is under 10,000);
- **AVG. Q**: Average #queries over successful attacks.
- Methods with the subscript “D” refers to the data-dependent version of the P-RGF method.
Thanks!