Frank-Wolfe for White Box Adversarial Attacks

Department of Mathematics "Tullio Levi-Civita"
Master's Degree in Data Science

Eleonora Brasola
Alberto Cocco
Greta Farnea

Università degli Studi di Padova

Adversarial example: element drawn from data distribution that is perturbed with some noise

- **Misclassification**: distorted image is not properly recognized by the DNN
- **Transferability**: different DNNs misclassify in the same way

daisy : 85.09% Confidence
distortion = 0.0000

→ adding a small noise, the adversarial image fools the DNN →

bee : 71.64% Confidence
distortion = 0.0015

Figures obtained from *demo.py*

Different types of attacks

We can decide if we want control on the output target of the adversarial image

- **Untargeted** attacks: just interested in the misclassification
- **Targeted** attacks: want a specific class as output

According to the information we can retrieve from the DNNs, we can have:

- **White box** attacks: access to all information, also gradients
- **Black box** attacks: access only to input and output
  $\rightarrow$ techniques for gradient estimation

FGSM

Notation:

- x_{ori} : original image
- $\ell(\cdot)$: loss function

One of the simplest adversarial method and one of first that has been implemented:

- One-step gradient-based method
- Maximum distortion ϵ

$$
\begin{aligned}
x &= x_{\text{ori}} + \epsilon \text{sign}(\nabla_x \ell(x_{\text{ori}})) \qquad &\text{(untargeted)};\\
x &= x_{\text{ori}} - \epsilon \text{sign}(\nabla_x \ell(x_{\text{ori}})) \qquad &\text{(targeted)}.
\end{aligned}
$$

Explaining and Harnessing Adversarial Examples, Goodfellow et al.

PGM

- Projection-based iterative approach
- Slow method

Algorithm 1 PGM

for $k = 1 \dots$ **do**
  Set $\bar{x}_k = \rho_C(x_k + s_k \nabla f(x_k))$ ▷ if untargeted, $s_k > 0$
  Set $\bar{x}_k = \rho_C(x_k - s_k \nabla f(x_k))$ ▷ if targeted, $s_k > 0$
  If $\bar{x}_k$ satisfies some specific condition, then STOP
  Set $x_{k+1} = x_k + \gamma_k(\bar{x}_k - x_k)$ ▷ with $\gamma_k \in (0, 1]$
end for

MI-FGSM

- Iterative version of *FGSM*, adding a momentum term
- High distortion values

Algorithm 2 MI-FGSM

Fix $g_0 = 0$ and x_0^*
for $t = 0$ to $T - 1$ **do**
    Input x_t and obtain the gradient $\nabla_x f(x_t)$
    $g_{t+1} = \beta \cdot g_t + \frac{\nabla_x f(x_t)}{\|\nabla_x f(x_t)\|_1}$
    $x_{t+1} = x_t + \gamma \cdot \mathrm{sign}(g_{t+1})$ ▷ if untargeted
    $x_{t+1} = x_t - \gamma \cdot \mathrm{sign}(g_{t+1})$ ▷ if targeted
end for

Boosting Adversarial Attacks with Momentum, Dong et al.

FW-white

Università degli Studi di Padova

- Projection-free method with momentum term
- Good trade-off between success and distortion

Algorithm 3 FW-White

1: Set $x_0 = x_{\text{ori}}$, $m_{-1} = -\nabla_x f(x_0)$ if untargeted attack, $m_{-1} = \nabla_x f(x_0)$ if targeted attack
2: **for** $t = 0$ to $T-1$ **do**
3: $\quad m_t = \beta \cdot m_{t-1} - (1-\beta) \cdot \nabla f(x_t)$ ▷ if untargeted
4: $\quad m_t = \beta \cdot m_{t-1} + (1-\beta) \cdot \nabla f(x_t)$ ▷ if targeted
5: $\quad v_t = \text{argmin}_{x \in C} \langle x, m_t \rangle = -\epsilon \cdot \text{sign}(m_t) + x_{\text{ori}}$
6: $\quad d_t = v_t - x_t$
7: $\quad x_{t+1} = x_t + \gamma d_t$
8: **end for**

A Frank-Wolfe Framework for Efficient and Effective Adversarial Attacks, Chen et al.

Demo.py

Now we hand over the word to Alberto so that we can see a demo of our project.