НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

DeepFool: a simple and accurate method to fool deep neural networks

Арсения Шихова

Национальный исследовательский университет «Высшая школа экономики»

13 декабря 2018г.

- Нейронная сеть обучается на конечном наборе изображений
- Злоумышленник знает всё: входные данные, архитектуру, параметры
- Цель: немного исказить картинку из трейна так, чтобы получить другой класс

Мотивация

- Беспилотные автомобили
- Всевозможные роботы и дроны
- Распознавание речи (голосовых команд)
- И так далее

Обозначения

- Нейронная сеть — это функция $f(x) = y; f : \mathbb{R}^n \to \mathbb{R}^c$, c — кол-во классов
- $\hat{k}(x) = \underset{1 \leqslant i \leqslant c}{\arg\max}\, y_i$ — класс
- Цель — $r : \hat{k}(x + r) \neq \hat{k}(x)$ — состязательный пример (adversarial example)

Обозначения

- Нейронная сеть — это функция $f(x) = y; f : \mathbb{R}^n \to \mathbb{R}^c$, c — кол-во классов
- $\hat{k}(x) = \arg\max_{1 \leqslant i \leqslant c} y_i$ — класс
- Цель — $r : \hat{k}(x + r) \neq \hat{k}(x)$ — состязательный пример (adversarial example)
- $\triangle(x, \hat{k}) = \min_r ||r||_2 : \hat{k}(x + r) \neq \hat{k}(x)$ — устойчивость (robustness) $\hat{k}$ в точке x
- $\rho_{adv}(\hat{k}) = \mathbb{E}_x \frac{\triangle(x, \hat{k})}{||x||_2}$ — устойчивость классификатора $\hat{k}$

DeepFool для бинарных классификаторов, линейный случай

- Пусть $\hat{k}(x) = sign(f(x))$, где $f(x) = w^T x + b$
- Обозначим $\mathcal{F} = \{x : f(x) = 0\}$ — гиперплоскость
- Тогда $\triangle(x_0, \hat{k})$ — расстояние между x_0 и $\mathcal{F}$
- $r(x_0) = \underset{r:\hat{k}(x_0+r)\neq\hat{k}(x_0)}{\arg\min} ||r||_2 = -\frac{f(x_0)}{||w||_2} w$
- $x_0 + r(x_0) = \frac{-b}{||w||_2} w$

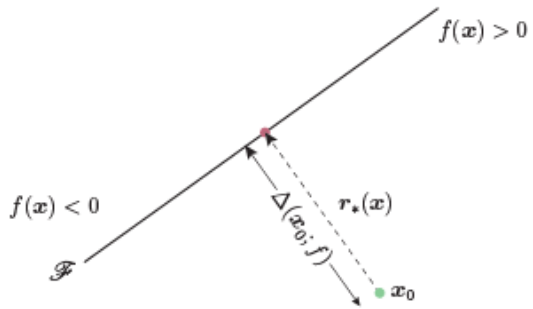


Figure 2: Adversarial examples for a linear binary classifier.

DeepFool для бинарных классификаторов, общий случай

- Пусть $\hat{k}(x) = sign(f(x))$
- Итеративно повышаем устойчивость $\triangle(x_0, \hat{k})$
- На i-м шаге $x_i = x_{i-1} + r_i$ с минимальным $||r_i||_2$, то есть $f(x_i) + \bigtriangledown f(x_i)^T r_i = 0$
- $r = \sum_i r_i$
- Ровно на границе быть плохо, в конце домножаем r на $1 + \eta$

Бинарный DeepFool

Алгоритм 1: Binary DeepFool

Вход: $x \in \mathbb{R}^n$, $\hat{k}(x) = sign(f(x))$
Выход: r: $\hat{k}(x) \neq \hat{k}(x + r)$
Function *DeepFool*(x, f)*:*

$x_0 = x$
$i = 0$
while $sign(f(x_i)) = sign(f(x_0))$ **do**
$r_i = -\frac{f(x_i)}{||\bigtriangledown f(x_i)||_2^2} \bigtriangledown f(x_i)$
$x_{i+1} = x_i + r_i$
$i = i + 1$
end
return $r = \sum_i r_i$

end

DeepFool
для многоклассовых классификаторов, линейный случай

- Используем подход **one-vs-all**
- $f : \mathbb{R}^n \to \mathbb{R}^c$
- $\hat{k}(x) = \underset{1 \leqslant k \leqslant c}{\arg\max} f_k(x)$
- $f(x) = W^T x + b$

DeepFool
для многоклассовых классификаторов, линейный случай

- Используем подход **one-vs-all**
- $f : \mathbb{R}^n \to \mathbb{R}^c$
- $\hat{k}(x) = \arg\max\limits_{1 \leqslant k \leqslant c} f_k(x)$
- $f(x) = W^T x + b$
- Цель: $\arg\min\limits_{r: \exists k \neq \hat{k}(x_0)\ w_k^T(x_0+r)+b_k \geqslant w_{\hat{k}(x_0)}^T(x_0+r)+b_{\hat{k}(x_0)}} ||r||_2^2$
- Геометрическое объяснение: ищем расстояние между границей полиэдра P и точкой x_0
- $P = \bigcap\limits_{k=1}^{c} \{x : f_{\hat{k}(x_0)}(x) \geqslant f_k(x)\}$
- Номер ближайшей к x_0 грани P есть $\hat{l}(x_0) = \arg\min\limits_{k \neq \hat{k}(x_0)} \frac{|f_k(x_0) - f_{\hat{k}(x_0)}(x_0)|}{||w_k - w_{\hat{k}(x_0)}||_2}$
- $r(x_0) = \frac{|f_{\hat{l}(x_0)}(x_0) - f_{\hat{k}(x_0)}(x_0)|}{||w_{\hat{l}(x_0)} - w_{\hat{k}(x_0)}||_2^2}(w_{\hat{l}(x_0)} - w_{\hat{k}(x_0)})$ — длина проекции x_0 на ближайшую грань P

DeepFool
для многоклассовых классификаторов, общий случай

- $f : \mathbb{R}^n \to \mathbb{R}^c$
- $\hat{k}(x) = \arg\max_{1 \leqslant k \leqslant c} f_k(x)$
- $P = \bigcap_{k=1}^{c} \{x : f_{\hat{k}(x_0)}(x) \geqslant f_k(x)\}$
- На каждой итерации минимизируем расстояние до
  $P_i = \bigcap_{k=1}^{c} \{x : f_k(x_i) - f_{\hat{k}(x_0)}(x_i) + \bigtriangledown f_k(x_i)^T x - \bigtriangledown f_{\hat{k}(x_0)}(x_i)^T x \leqslant 0\}$

Многоклассовый DeepFool

Алгоритм 2: Multiclass DeepFool

Вход: $x \in \mathbb{R}^n$, $\hat{k}(x) = sign(f(x))$
Выход: r: $\hat{k}(x) \neq \hat{k}(x + r)$
Function *DeepFool*(x, f)*:*

$x_0 = x$
$i = 0$
while $\hat{k}(x_i) \neq \hat{k}(x_0)$ **do**
  for $k \neq \hat{k}(x_0)$ **do**
    $w'_k = \bigtriangledown f_k(x_i) - \bigtriangledown f_{\hat{k}(x_0)}(x_i)$
    $f'_k = f_k(x_i) - f_{\hat{k}(x_0)}(x_i)$
  end
  $\hat{l} = \arg\min_{k \neq \hat{k}(x_0)} \frac{|f'_k|}{||w'_k||_2}$
  $r_i = \frac{|f'_{\hat{l}}|}{||w'_{\hat{l}}||_2^2} w'_{\hat{l}}$
  $x_{i+1} = x_i + r_i$
  $i = i + 1$
end
return $r = \sum_i r_i$

end

Fast gradient sign

- Другой метод поиска примеров
- $r = \varepsilon \cdot sign(J(x, y, \theta))$, где θ — веса модели, J — стоимость обучения
- Подбираем ε так, чтобы исказить хотя бы 90% обучающей выборки

Сравнение методов

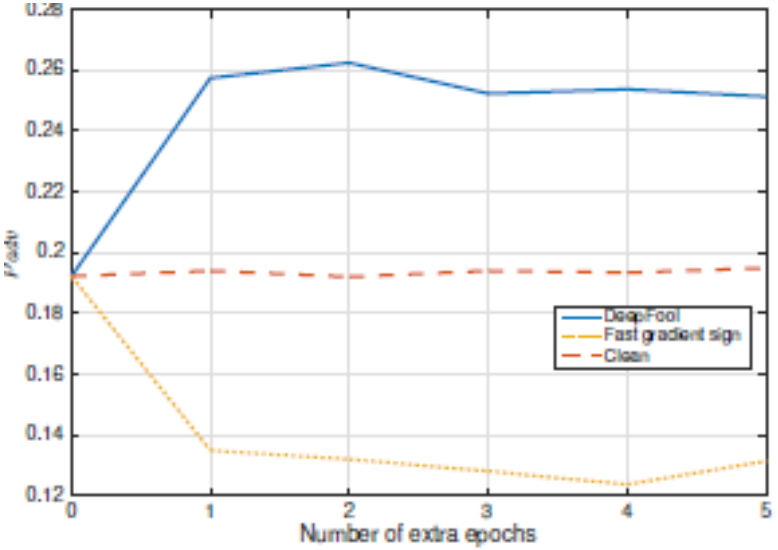
Classifier	Test error	$\hat{\rho}_{\text{adv}}$ [DeepFool]	time	$\hat{\rho}_{\text{adv}}$ [4]	time	$\hat{\rho}_{\text{adv}}$ [18]	time
LeNet (MNIST)	1%	2.0×10^{-1}	110 ms	1.0	20 ms	2.5×10^{-1}	> 4 s
FC500-150-10 (MNIST)	1.7%	1.1×10^{-1}	50 ms	3.9×10^{-1}	10 ms	1.2×10^{-1}	> 2 s
NIN (CIFAR-10)	11.5%	2.3×10^{-2}	1100 ms	1.2×10^{-1}	180 ms	2.4×10^{-2}	>50 s
LeNet (CIFAR-10)	22.6%	3.0×10^{-2}	220 ms	1.3×10^{-1}	50 ms	3.9×10^{-2}	>7 s
CaffeNet (ILSVRC2012)	42.6%	2.7×10^{-3}	510 ms*	3.5×10^{-2}	50 ms*	-	-
GoogLeNet (ILSVRC2012)	31.3%	1.9×10^{-3}	800 ms*	4.7×10^{-2}	80 ms*	-	-

Table 1: The adversarial robustness of different classifiers on different datasets. The time required to compute one sample for each method is given in the time columns. The times are computed on a Mid-2015 MacBook Pro without CUDA support. The asterisk marks determines the values computed using a GTX 750 Ti GPU.

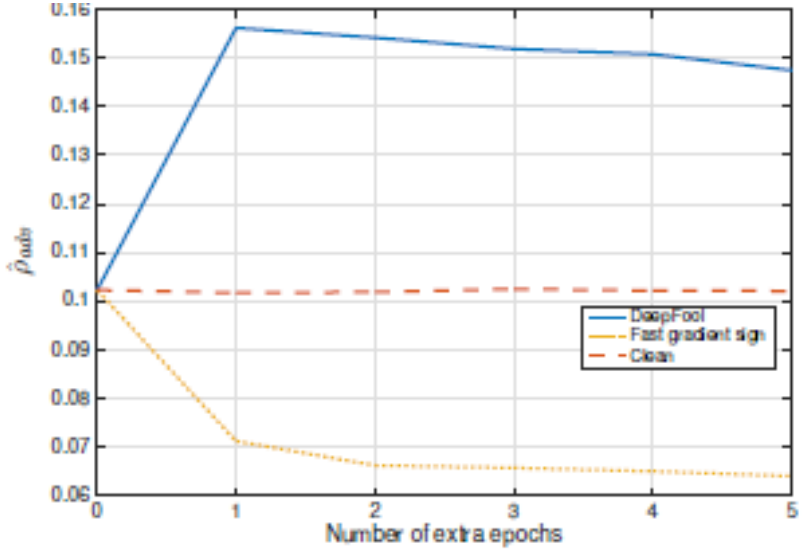
здесь $\hat{\rho}_{adv}(f) = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \frac{||\hat{r}(x)||_2}{||x||_2}$

Улучшение модели

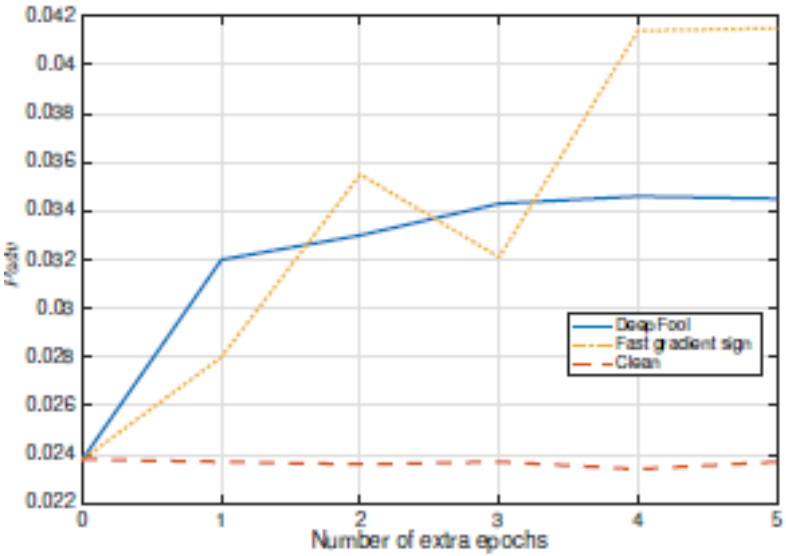
- Добавляем при обучении сгенерированные состязательные примеры
- Увеличиваем время обучения



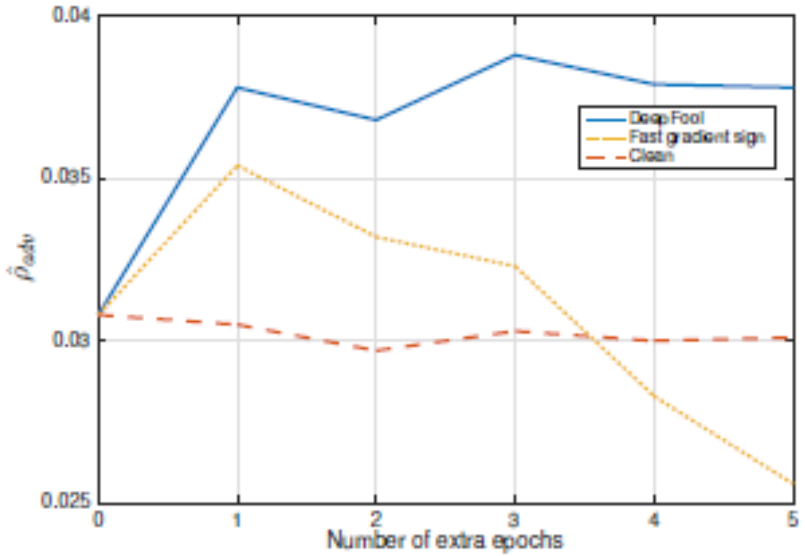
a) Effect of fine-tuning on adversarial examples computed by two different methods for LeNet on MNIST.



(b) Effect of fine-tuning on adversarial examples computed by two different methods for a fully-connected network on MNIST.



c) Effect of fine-tuning on adversarial examples computed by two different methods for NIN on CIFAR-10.



(d) Effect of fine-tuning on adversarial examples computed by two different methods for LeNet on CIFAR-10.

Ссылки

- cv-foundation.org/openaccess/content_cvpr_2016/papers/Moosavi-Dezfooli_DeepFool_A_Simple_CVPR_2016_paper.pdf — оригинал статьи
- arxiv.org/pdf/1412.6572.pdf — другой метод поиска состязательных примеров
- github.com/lts4/deepfool — репозиторий с кодом алгоритма