Apprentissage robuste

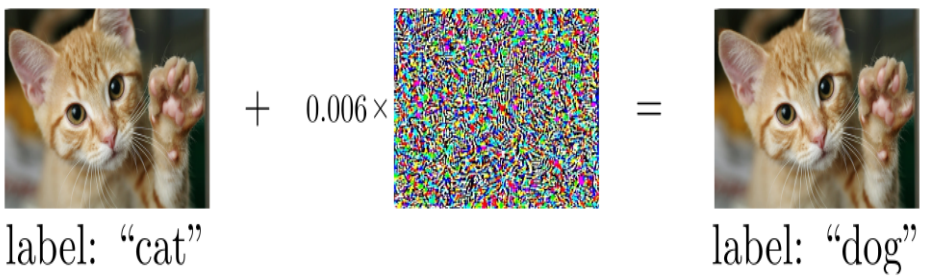
J-M. Martinez, S. Diarrassouba, M. Poyer

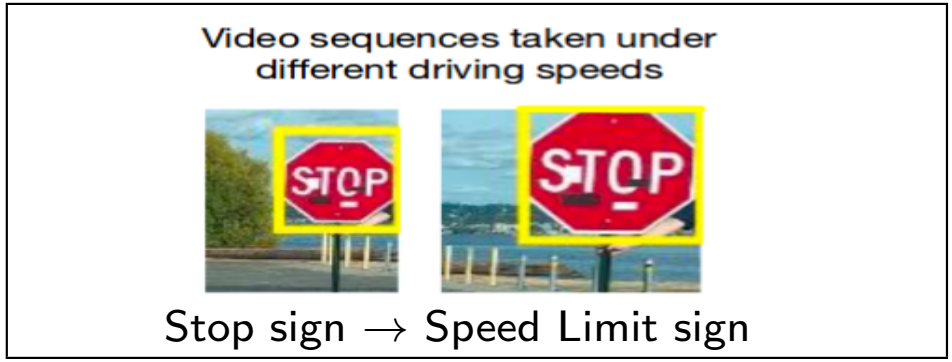
CEA-Saclay, DES, DM2S, Université Paris Saclay, F-91191 Gif-sur-Yvette, France

Ecole IN2P3 d'informatique, septembre 2020

Contexte

- Travaux (Ch. Szegedy, 2014) ont montré qu'on peut très facilement *tromper* un Deep Learning
  - vulnérabilité $\simeq$ sensibilité aux perturbations des données (images)
  - liée à la grande dimension du modèle définissant les frontières de décision
- Exemple : sur une image une perturbation imperceptible (pour l'oeil humain) fausse complètement l'interprétation du réseau de neurones





  - perturbations *calculées* par un *adversaire* pour *tromper* le classifieur
- Nécessité de robustifier (adversial robustness) l'apprentissage machine
  - applications : systèmes/voitures autonomes, contrôle d'accès, ré-identification, cybersécurité, ...
- Nombreux travaux en cours pour un apprentissage résistant aux attaques adverses
  - I. Goodfellow, N. Papernot, A. Madry, A. Kurakin, Z. Kolter

Classification supervisée

- Données $\mathcal{D} = \{(x_i, y_i)_{i=1,\ldots,n}\}, x_i \in \mathcal{X} \subset \mathbb{R}^d, y_i \in \{0,1\}^p$, codage 1 parmi p : $||y_i||_1 = 1$
- Classifieur $f_\theta : \mathcal{X} \to [0,1]^p$ paramétrisé par les coefficients θ (poids et biais)
- Apprentissage par minimisation de la moyenne empirique d'une *loss function* $\mathcal{L}(f_\theta(x), y)$

$$\hat{\theta} = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f_\theta(x_i), y_i)$$

- On note $\ell(x_i)$ la classe de x_i (composante de y_i égale à 1). Exemple *bien appris* par $f_{\hat{\theta}}$ si

$$\arg\max f_{\hat{\theta}}(x_i) = \ell(x_i)$$

- Remarque, simplification lorsque la *loss function* est l'*entropie croisée*

$$\mathcal{L}(f_{\hat{\theta}}(x_i), y_i) = -\log f_{\hat{\theta}, \ell(x_i)}(x_i), (\texttt{composante } \ell(x_i) \texttt{ de } f_{\hat{\theta}}(x_i))$$

- Objectif de l'*attaquant* : modifier un exemple par une *petite* perturbation δ telle que :

$$\begin{aligned} \texttt{Attaque non ciblée} &: \arg\max f_{\hat{\theta}}(x_i + \delta) \neq \ell(x_i) \\ \texttt{Attaque ciblée} &: \arg\max f_{\hat{\theta}}(x_i + \delta) = \ell_{cible} \neq \ell(x_i) \end{aligned}$$

  - pour un exemple *mal appris*, l'attaque non ciblée est inutile (solution $\delta = 0$)
  - perturbation de l'attaque ciblée $\geq$ perturbation de l'attaque non ciblée

Objectif de l'attaquant

- Contrôle de la valeur de la perturbation $\rightarrow$ choix d'une norme $||\delta||_{p=1,2,\infty}$ et calcul par

$$\delta = \arg\min ||\delta||_p \ \texttt{sous} \ \begin{cases} \arg\max f_{\hat{\theta}}(x_i + \delta) \neq \ell(x_i) & \texttt{non ciblée} \\ \arg\max f_{\hat{\theta}}(x_i + \delta) = \ell_{cible} \neq \ell(x_i) & \texttt{ciblée} \end{cases}$$

- Problème difficile $\rightarrow$ plusieurs *heuristiques* d'attaques proposées
- On se limite aux méthodes qui supposent connu le modèle $f_{\hat{\theta}}(x)$
- La perturbation δ est calculée itérativement par une ou plusieurs *remontées ou descentes de gradient* de la *loss function*, gradient calculé par rapport aux entrées x_i
- Attaque non ciblée $\rightarrow$ maximiser $\mathcal{L}(f_{\hat{\theta}}(x_i + \delta), y_i)$ pour ne plus associer x_i à la classe $\ell(x_i)$
- Attaque ciblée $\rightarrow$ minimiser $\mathcal{L}(f_{\hat{\theta}}(x_i + \delta), y_{cible})$ pour associer x_i à la classe $\ell_{cible} \neq \ell(x_i)$

`Attaque non ciblée`	`remontée de gradient`	$+\eta \nabla_x \mathcal{L}(f_{\hat{\theta}}(x_i), y_i)$
`Attaque ciblée`	`descente de gradient`	$-\eta \nabla_x \mathcal{L}(f_{\hat{\theta}}(x), y_{cible})$

Quelques méthodes d'attaques non ciblées

- Méthode *Fast Gradient Sign* (Goodfellow, 2014)

$$\delta = \epsilon \times \text{sign}(\nabla_x \mathcal{L}(f_{\hat{\theta}}(x_i), y_i))$$

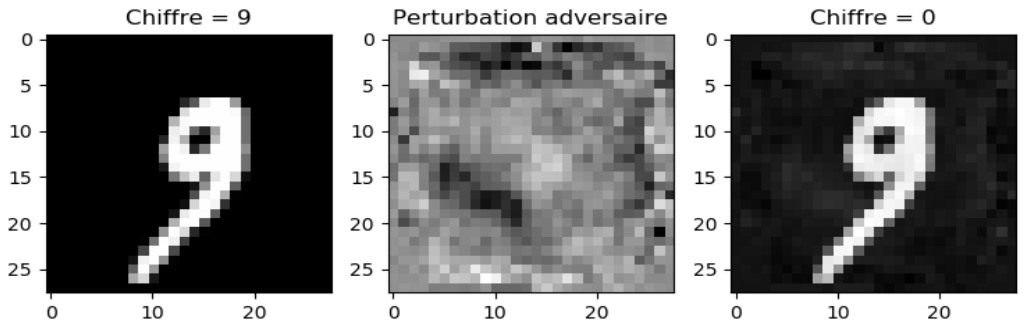
- Méthode *Iterative Gradient Sign* (Kurakin, 2017), *Project Gradient Descent* (Madry, 2019)

$$x^{(t+1)} = \texttt{clip}_{x,\epsilon}[x^{(t)} + \alpha \times \text{sign}(\nabla_x \mathcal{L}(f_{\hat{\theta}}(x^{(t)}), y_i))], \quad x^{(0)} = x_i$$

- Notation $\texttt{clip}_{x,\epsilon}(z)$ est la fonction qui impose $||z - x||_\infty \leq \epsilon$
- On doit également assurer que les *patterns* modifiés $x^{(t)} \in \mathcal{X}$
- Remarque : pour les attaques ciblées il suffit de changer le signe du gradient et de remplacer y_i par la cible y_{cible}.

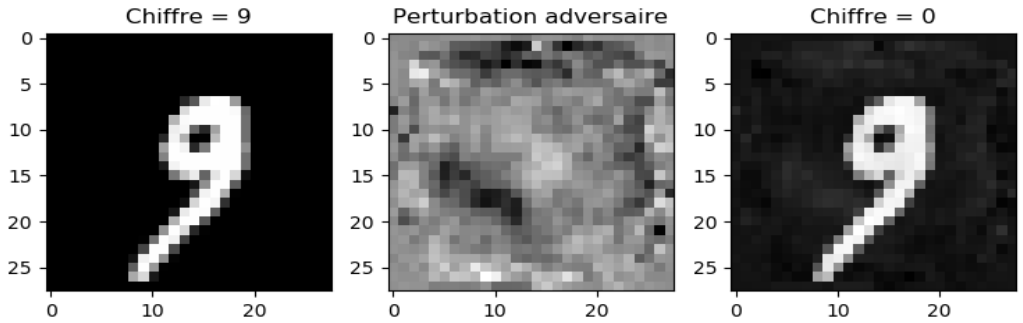
Illustration d'une attaque ciblée

- Base MNIST : chiffres représentés par des images 28×28 pixels $\in \{0, 1, \ldots, 255\}/255$
- Un réseau FC, *Full Connected*
- Attaquant modifie l'image d'un 9 afin que le réseau la confonde avec un 0
- Algorithme *Iterative Gradient*



Root Mean Square Perturbation 0.039

- Algorithme *Project Gradient Descent* (méthode L-BFGS proposée par Ch. Szegedy, 2014)

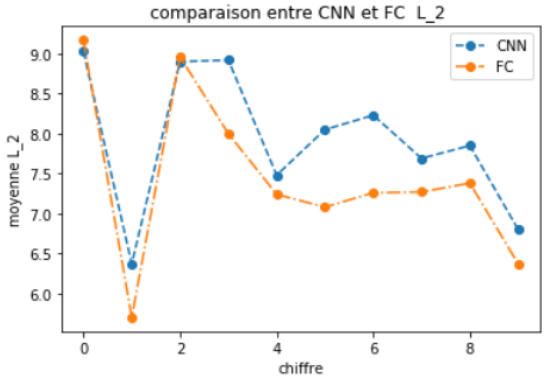


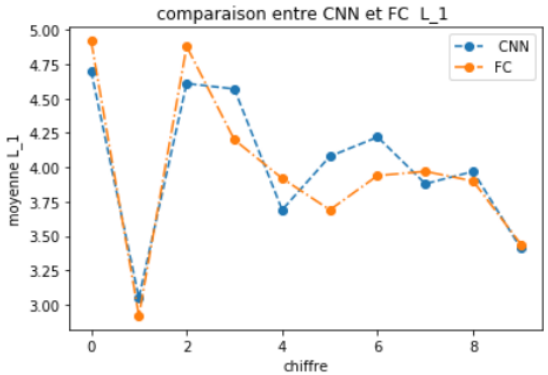
Root Mean Square Perturbation 0.041

- Des résultats similaires avec des architectures CNN (Convolutional Neural Network)

Effets de la régularisation

- Réseau FC, *Full Connected* versus CNN *Convolutional Neural Network*
  - FC : $\{784 - 300 - 300 - 10\}$
  - CNN : $\{(28 \times 28) - 32(3 \times 3) - \texttt{Max pooling}(2,2) - 64(3 \times 3) - 128 - 10\}$
  - régularisation : Drop Out(0.2), *Weighth Decay* $(1.e^{-6}, 1.e^{-8})$
- Modification des images de la base d'apprentissage jusqu'aux succès de l'attaquant





Les normes L_1, L_2 sur les distorsions sont moyennées sur la totalité des pixels de l'image

- Analyse des apprentissages :
  - la fragilité *semble* dépendre de la représentation du *pattern* et pas de l'architecture du réseau : les chiffres "1" et "9" sont les plus *fragiles*, "0" et "2" plus *robustes* pour le FC et pour le CNN
  - le CNN est plus robuste
  - la régularisation améliore la *robustesse* mais le gain est faible

$\Rightarrow$ Besoin en méthodes d'apprentissage plus robuste

Algorithme d'apprentissage robuste

- Objectif : entraîner un réseau de neurones de façon à le rendre robuste face à d'éventuelles attaques adverses sur les entrées
- Méthode *adversial training* (Goodfellow 2015, Madry 2017) : modification de la *loss function* en considérant le pire cas amené par une perturbation δ dans un *petit* domaine S

$$\mathcal{L}(f_\theta(x), y) \rightarrow \max_{\delta \in S} \mathcal{L}(f_\theta(x+\delta), y)$$

- Choix du domaine des perturbations $S = \{\delta \in \mathbb{R}^d : ||\delta||_p < \epsilon\}$
- Algorithme d'apprentissage : recherche du minimum par rapport aux paramètres du réseau de neurones et du maximum par rapport aux perturbations (recherche d'un *point-selle*)

$$\min_\theta \frac{1}{n} \sum_{i=1}^{n} \max_{||\delta_i||_p \leq \epsilon} \mathcal{L}(f_\theta(x_i+\delta_i), y_i)$$

- Forme *probabiliste*

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}}[\max_{||\delta||_p \leq \epsilon} \mathcal{L}(f_\theta(x+\delta), y)]$$

Apprentissage robuste au cas simple de la régression logistique

- Modèle régression logistique $\sigma(x) = (1 + e^{-(w^t x + b)})^{-1}$
- Fonction *soft plus* $\zeta(z) = \log(1 + e^z)$ comme *loss function* pour le codage $y \in \{-1, +1\}$:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[\zeta(-y(w^t x + b))]$$

- Par la méthode *Fast Gradient Sign* la perturbation δ de l'attaque adverse est :

$$\delta = \epsilon \times \text{sign}\nabla_x \zeta(-y(w^t x + b)) = -y\epsilon \times \text{sign}(w)$$

- L'apprentissage robuste consiste alors à minimiser la *loss function* en remplaçant x par $x + \delta$

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[\zeta(-y(w^t x + b - \epsilon||w||_1))]$$

- On obtient la pénalité $\epsilon||w||_2^2$ par *Fast Gradient*, c.à.d. pour $\delta = \epsilon \times \nabla_x \zeta(-y(w^t x + b))$
- Méthode similaire au *weight decay* := *loss function* $+ \lambda||w||_p^p$ mais différente car :
  - la pénalité est soustraite au potentiel synaptique et pas ajoutée à la *loss function*
  - l'apprentissage robuste est moins pénalisant que le *weight decay* car il apparait comme une régularisation plus locale dépendant de la donnée d'entrée (Goodfellow, 2015)

Algorithme d'apprentissage robuste basé sur des attaques PGD

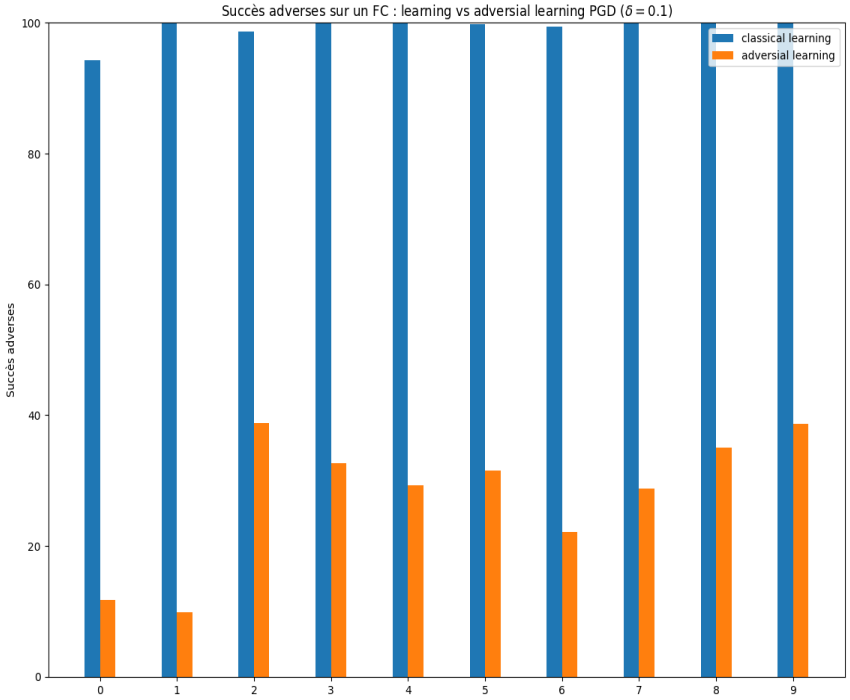
- PGD Projection Gradient Descent (A. Madry, 2019), équivalent à un FGS itératif (Fast Gradient Sign)

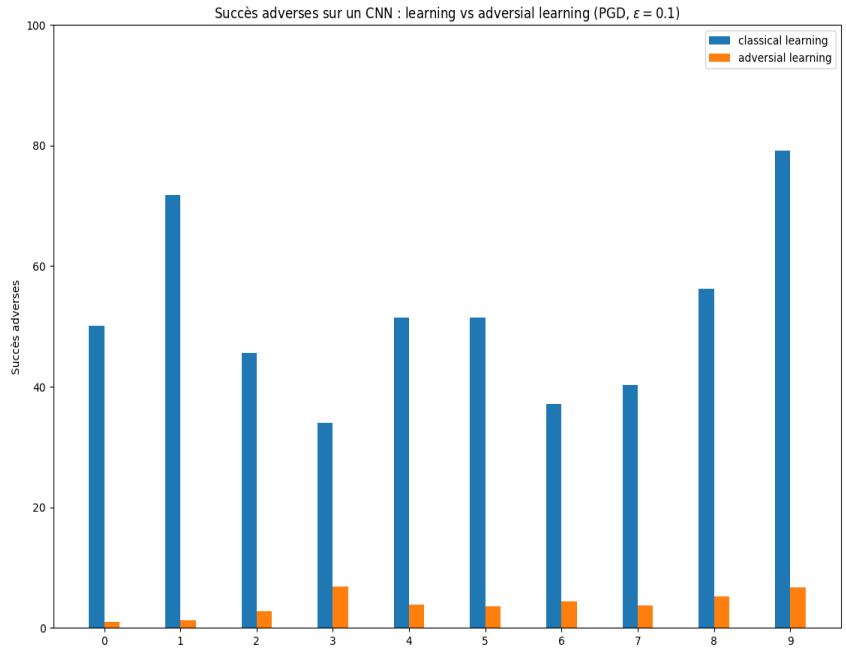
Algorithm 1 Algorithme PGD sur un Batch

1: Hyperparamètres : ϵ variation maximale par pixel, N nombre d'itérations du gradient de l'attaquant
2: Modification de chaque exemple (x, y) du Batch. On suppose les entrées dans $[0, 1]$
3: **for** $x \in$ Batch **do**
4: $\quad x_{adv} = x$
5: $\quad x_{min} = \max(x - \epsilon, 0), x_{max} = \min(x + \epsilon, 1)$
6: $\quad$ **for** k=1,N **do**
7: $\quad\quad x_{adv} = x_{adv} + \frac{\epsilon}{N}\texttt{signe}[\nabla_{x_{adv}}\mathcal{L}(f_\theta(x_{adv}), y)]$
8: $\quad\quad x_{adv} = \max(x_{min}, x_{adv}), x = \min(x_{max}, x_{adv})$ pour que $x_{adv} \in [0, 1] \cap [x + \epsilon, x + \epsilon]$
9: $\quad$ **end for**
10: **end for**
11: Apprentissage par descente du gradient *moyenné* sur le Batch $\mathbb{E}_{\sim(x_{adv}, y)}[\nabla_\theta \mathcal{L}(f_\theta(x_{adv}), y)]$

Résultats de l'algorithme PGD sur la base MNIST

- Réseaux entraînés par apprentissage *classique* puis réentraînés par apprentissage robuste (attaque PGD non ciblée, $\epsilon = 01, N = 4$)
- Architectures FC et CNN non régularisées
- Précision sur la base pratiquement pas modifiée (97,6% pour le FC et 99.8% pour le CNN)





- CNN plus robuste que le FC (on s'y attendait de par son architecture), et pour les 2 architectures le chiffre 9 est plus facile à *attaquer*

- L'apprentissage robuste par perturbation des entrées est facile à mettre en oeuvre (simple calcul d'un gradient), mais il augmente les temps d'apprentissage et rajoute des hyper paramètres (plage des variations, nombre d'itérations)
- D'autre part l'apprentissage se fait sur des données perturbées $\rightarrow$ la précision sur la base de test peut diminuer $\rightarrow$ compromis entre robustesse et précision
- Dans le domaine de la *Data Science* un fort besoin en apprentissage robuste (sécurité, éthique, ...)
- Constat aujourd'hui : problème difficile (Z. Kolter, Workshop ICLR 2020) : *we still haven't solved the underlaying problem*
- D'autres approches de robustesse par introduction de bruit (R. Pinot CEA/LIST) ou de nature plus géométrique (Kolter, Mirman, *certified/provable defenses*)
- Perspectives : la R&D sur l'interprétabilité des réseaux de neurones comme support à l'amélioration de leur robustesse

Quelques références


Kurakin A., Goodfellow I., and S. Bengio, *Adversial machine learning at scale*, arXiv :1611.01236V2 (2017).


Madry A., A. Makelova, L. Schmidt, D. Tsipras, and Vladu A., *Towards deep learning models resistant to adversial attacks*, arXiv :1706.06083v4 (2019).


Szegedy Ch., Zaremba W., Sutskever I., Bruna J., Erhan D., Goddfellow I., and R. Fergus, *Intriguing properties of neural networks*, arXiv :1312.6199v4 (2014).


Tramèr F., Papernot N, Goodfellow I., Boneh D., and P. McDaniel, *The space of transferable adversarial examples*, arXiv :1704.03453v2 (2017).


Goodfellow I., Schlens J., and Szegedy Ch., *Explaining and harnessing adversial examples*, arXiv :1412.6572v3 (2015).


Kolter Z. and al., *Adversial robustness*, "`https://www.youtube.com/watch?v=fiqcI1d9AnQ`", Workshop ICLR (2020).