apprentissage profond - deep learning

Structure des réseaux de neurones

partie 1 : Multi layer perceptron

Adrien CHAN-HON-TONG
ONERA

Préambule

Objectif pédagogique

- À quoi servent ces systèmes appris ?
- Sont-ils indispensables ?
- Peuvent ils être surs ?
- Comment les valide-t-on ?
- Comment ces systèmes sont-ils conçus ?

Préambule

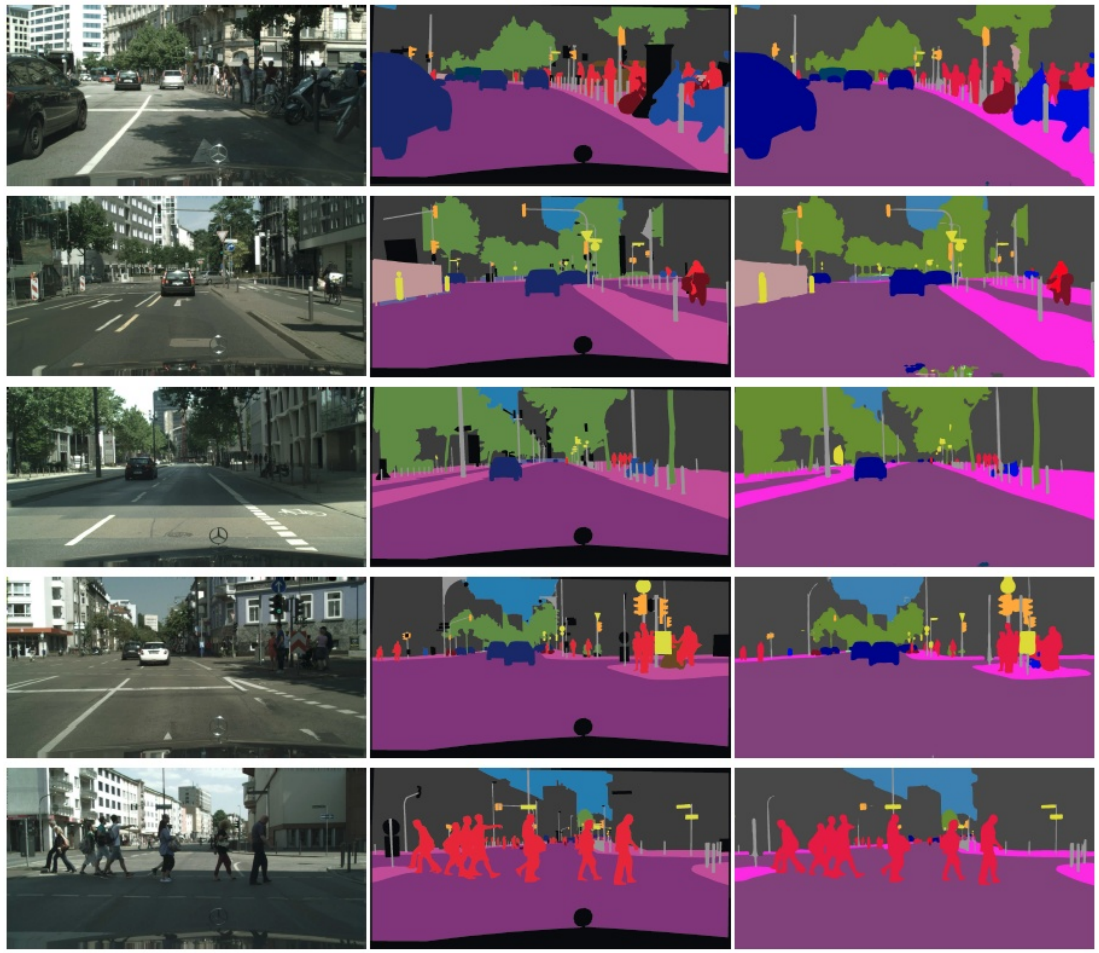
Objectif pédagogique

- À quoi servent ces systèmes appris ?
- Sont-ils indispensables ?
- Peuvent ils être surs ?
- Comment les valide-t-on ?
- Comment ces systèmes sont-ils conçus ?

Seule la dernière question (la moins importante) change avec le deep learning ! Cependant, la rupture de performance (ces systèmes entendent conduire votre voitures) devrait inviter à se focaliser sur les premières...

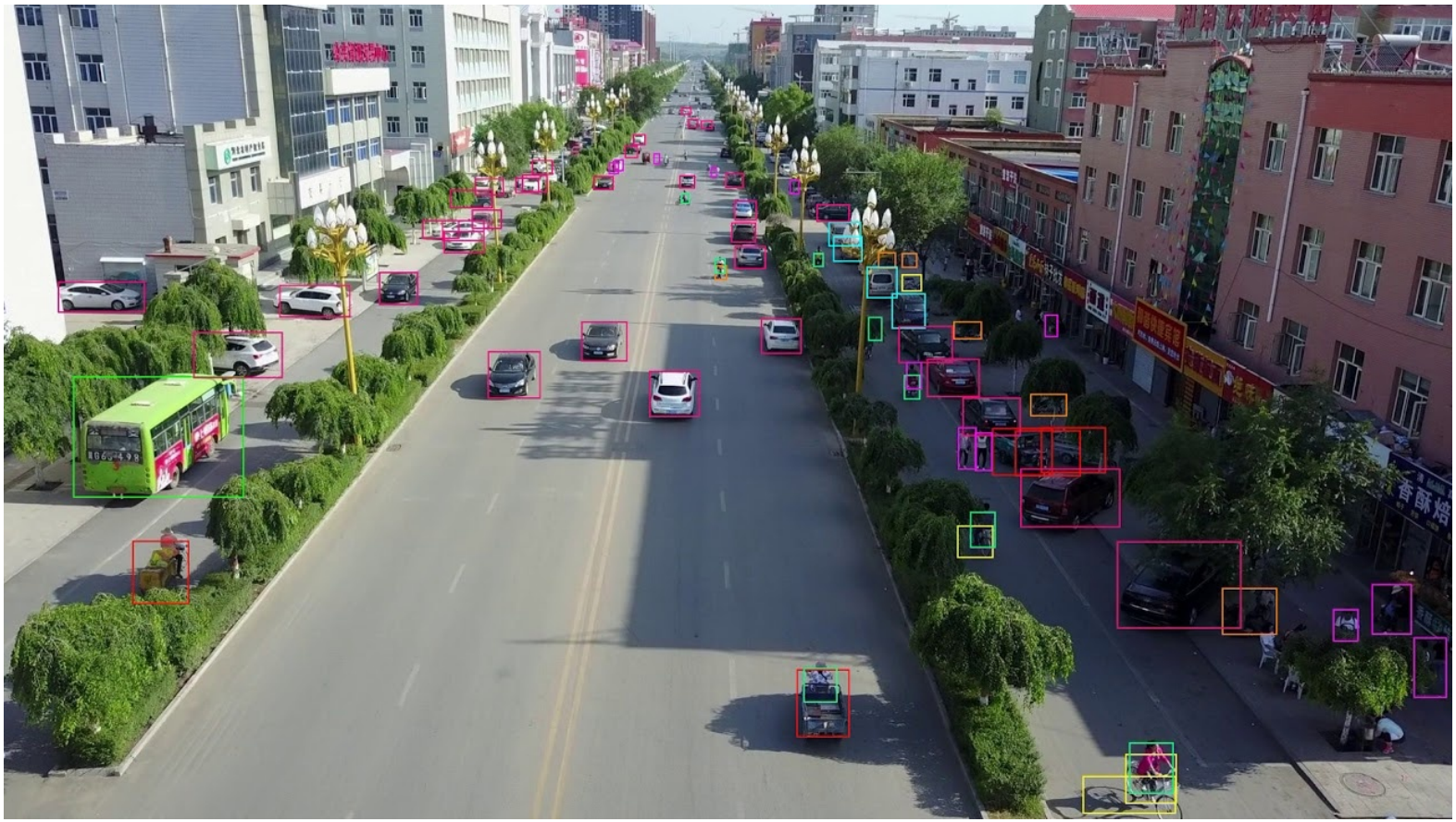
Introduction

Shock and Awe



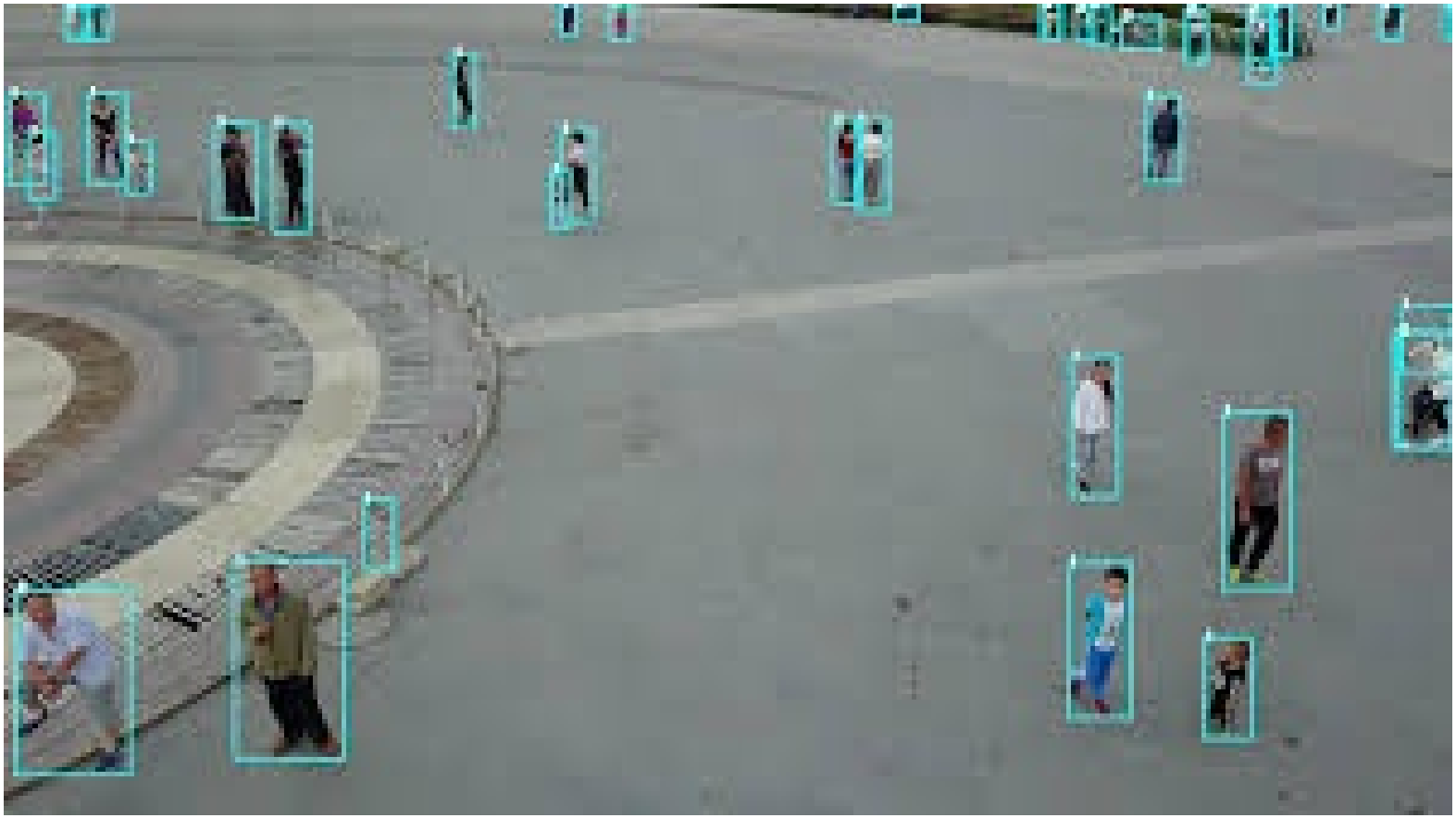
Introduction

Shock and Awe



Introduction

Shock and Awe



Introduction

Shock and Awe

Monteriez vous dans un avion ou taxi autonome sans pilote ?

Pour le projet ATTOL (Autonomous Taxi, Take-Off and Landing), le 18 décembre dernier, une première mondiale a été accomplie : un décollage entièrement autonome basé vision, sans utilisation de l'ILS, ni du GPS, a été réalisé à plusieurs reprises sur l'aéroport de Toulouse-Blagnac. L'ONERA a contribué au développement et à la mise au point l'algorithme de fusion de données, qui élabore le signal de déviation à l'axe de piste, nécessaire au contrôle de l'avion, à partir des informations visuelles issues de la caméra et inertielles provenant des centrales à bord de l'appareil. Ainsi, contrairement aux systèmes actuels dont les signaux se trouvent hors de l'appareil, ceux-ci se trouvaient embarqués, d'où la qualification d'un *système autonome*.

https ://www.onera.fr/fr/actualites/attol-des-pilotes-dans-lavion-mais-un-systeme-autonome

Introduction

Shock and Awe

On a l'impression qu'on est à l'aube d'une révolution industrielle !

Pourtant, quasiment rien n'est nouveau (les performances)

Le problème de base

classification binaire

- $X = \mathbb{R}^D$ l'espace des points, P une distribution sur X
- y la fonction cible dans {-1,1}
- on approxime y par une fonction f (avec des poids w)
- erreur réelle $e_r = \int_X \mathbf{1}_-(y(x)f(x,w))P(x)\mathrm{d}x$
- $x_1, ..., x_N$ une base d'apprentissage tirée selon P
- $\chi_1, ..., \chi_M$ une base de test tirée selon P
- l'erreur d'apprentissage $e_a = \frac{1}{N}\sum_n \mathbf{1}_-(y(x_n)f(x_n,w))$
- l'erreur de test $e_t = \frac{1}{M}\sum_m \mathbf{1}_-(y(\chi_m)f(\chi_m,w)) \approx e_r$

Le problème de base

schéma général

l'apprentissage c'est trouver f et w tel que e_a soit petit

sachant que ce qui nous intéresse en réalité c'est que e_r soit petit

mais on ne peut pas contraindre directement à ce que e_r soit petit
(car on ne connaît pas y)

donc on prie et on mesure e_t à la fin pour vérifier si ça a marché

Le problème de base

classification binaire

X c'est l'espace des images (vu comme des points), P c'est leur probabilité d'être sur internet, y c'est la fonction qui dit si une image est un chat (+1) ou pas (-1).

On ne la connaît pas mais avec un humain on peut construire une base d'apprentissage qui permettra de construire un modèle f (avec ses poids w) pour approximer y.

La qualité de cette approximation c'est e_r qu'on approxime en mesurant e_t.

Le problème de base

digression

y n'a aucun sens mathématique c'est une fonction anthropique !

Même avec une puissance de calcul infini, vous ne pouvez ni définir ni calculer y, pourtant n'importe quelle humain peut l'évaluer !

- IA
  - des problèmes qu'on pourrait résoudre avec une puissance infinie
  - qu'on essaye de résoudre en pratique avec de l'intelligence
  - pour aider l'humain
- ML
  - des problèmes qu'on ne peut même pas définir
  - qu'on résout en pratique avec des méthodes bêtes et beaucoup de puissance de calcul
  - pour faire du dumping technologique de l'humain

Le problème de base

Overfitting

$$f(x, w) = \begin{cases} si\ \exists n\ /\ x = x_n & y(x_n) \\ sinon & -1 \end{cases}$$

$e_a = 0$ mais $e_t \approx e_r = \frac{1}{2}$ quelque soit N

(en supposant y d'espérance nulle)

Le problème de base

PAC learning

sous certaines hypothèses l'écart entre les 3 erreurs e_a, e_r et e_t est borné (en probabilité)

Oui mais

en pratique ça sert jamais (les bornes sont lâches et probabilistes)

Le problème de base

schéma général

l'apprentissage c'est prendre f réputée pertinente

et optimiser w tel que e_a soit petit

et vérifier via e_t si f est pertinente pour ce problème précis

deep learning

C'est la famille de fonction qui *marche* le mieux, sachant que ça marche pas tout le temps...
C'est à la fois pas grand chose et beaucoup.

Du neurone au réseau

Le neurone

Dans les architectures de réseaux de neurones (profond ou pas), le neurone est un filtre linéaire :

$$\begin{array}{rccc} \textit{neurone}_{\alpha,\beta} : & \mathbb{R}^{\phi} & \rightarrow & \mathbb{R} \\ & u & \rightarrow & \alpha.u + \beta \end{array}$$

$\alpha \in \mathbb{R}^{\phi}$ et $\beta \in \mathbb{R}$ sont les **poids** du neurones.

Attention, le neurone n'est pas nécessairement connecté à x l'entrée - à ce stade, il a une entré de taille ϕ indépendante de D.

Du neurone au réseau

La couche de neurone

Une couche de ψ de neurones est une séquence de ψ neurones prenant la même entrée, et, dont les ψ sorties sont regroupées en 1 vecteur :

$$\begin{array}{rccc} & \mathbb{R}^{\phi} & \rightarrow & \mathbb{R}^{\psi} \\ couche_{A,b} : & u & \rightarrow & \begin{pmatrix} neurone_{A_1,b_1}(u) \\ \dots \\ neurone_{A_\psi,b_\psi}(u) \end{pmatrix} \end{array}$$

$A \in \mathbb{R}^{\psi \times \phi}$ et $b \in \mathbb{R}^{\psi}$ sont les **poids** de chacun des ψ neurones.

La couche de neurone est aussi linéaire : $couche_{A,b}(u) = Au + b$ mais avec des tailles arbitraires en entrée et sorti.

Du neurone au réseau

Le réseau de neurone

Si on empile 2 couches de neurones c'est exactement comme s'il y en avait qu'une :

$$A'(Au + b) + b' = (A'A)u + (A'b + b')$$

Oui mais si on met une non linéarité entre les 2 c'est différents.

Laquelle ? Sur pytorch : il y en a un certain nombre relu, elu, leaky-relu, sigmoide, arctan, hard sigmoid, hard arctan, prelu, relu6, rrelu, celu, selu, gelu, hard shirk, soft shirk, log sigmoid, soft sign, tanh, tanhshirk

globalement avant on utilisait une sigmoide (lisse) et aujourd'hui c'est plutôt relu $relu(u) = [u]_+ = max(u, 0)$ car c'est rapide.

Du neurone au réseau

Le réseau de neurone

Un réseau de neurones entièrement connectées (multi layer perceptron en anglais) de profondeur Q est un empilement de Q couche de neurones - séparé par des activations - la dernière est classiquement un seul neurone :

$$reseau_w : \begin{array}{ccc} \mathbb{R}^D & \rightarrow & \mathbb{R} \\ x & \rightarrow & C_{w_Q}(relu(C_{w_{Q-1}}(...relu(C_{w_1}(x))...))) \end{array}$$

c'est à dire

$$f(x, w) = w_Q \times relu(w_{Q-1} \times relu(...(relu(w_1 \times x))))$$

$w_1, ..., w_Q$, Q matrices dont la seule chose imposée étant que w_1 ait D colonnes, et w_Q 1 ligne (et que les tailles soit cohérentes entre elles)

Apprentissage profond

Quand le réseau a beaucoup de couche, on parle de réseau profond. Ce terme est apparu en 2012 avec un réseau à 10 couches dit Alexnet (*imagenet classification with deep convolutional neural networks*).

Les réseaux de neurones existent depuis 1950 mais la puissance de calcul n'était pas suffisante pour envisager des réseaux de neurones profonds.

ResNet 100 a 100 couches (2016) et il y en a des encores plus gros aujourd'hui...

Bilan

MLP

- $X = \mathbb{R}^D$ l'espace des points, P une distribution sur X
- y la fonction cible dans {-1,1}
- on approxime y par une fonction f (avec des poids w)
- erreur réelle $e_r = \int_X \mathbf{1}_-(y(x)f(x,w))P(x)\mathrm{d}x$
- $x_1, ..., x_N$ une base d'apprentissage tirée selon P
- $\chi_1, ..., \chi_M$ une base de test tirée selon P
- l'erreur d'apprentissage $e_a = \frac{1}{N}\sum_n \mathbf{1}_-(y(x_n)f(x_n,w))$
- l'erreur de test $e_t = \frac{1}{M}\sum_m \mathbf{1}_-(y(\chi_m)f(\chi_m,w)) \approx e_r$
- deep learning = choisir
  $f(x,w) = w_Q \times relu(w_{Q-1} \times relu(...(relu(w_1 \times x))))$ et optimiser w pour avoir e_a faible
- c'est ce qui *marche le mieux aujourd'hui*

Questions ouvertes

- si l'entrée n'est pas $X = \mathbb{R}^D$?
- comment on optimise w ?