Introduction à l'intelligence artificielle et à l'apprentissage automatique

Eric Gaussier

LIG - MIAI
Université Grenoble Alpes
Eric.Gaussier@imag.fr

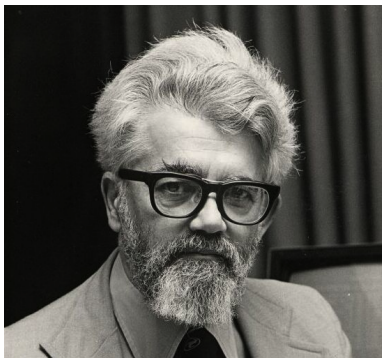
Table des matières

Des définitions qui ont varié au cours du temps (1)

1. Les débuts de l'informatique : Test de Turing – *Computing Machinery and Intelligence (1950)*
ELIZA (J. Weizenbaum - 1966) - PARRY (K. Cosby - 1972, paranoïd AI)

2. Conférence Darthmouth (1956) *J. McCarthy, M. Minsky (org.)*
Adoption du terme IA (Machine Intelligence, D. Michie)

"Getting computers do things human do easily (seeing, talking, driving, manipulating objects, planning everyday lives)"
From Artificial Intelligence, Y. Wilks, 2019



Des définitions qui ont varié au cours du temps (2)

3. *Construction de programmes informatiques qui s'adonnent à des tâches qui sont, pour l'instant, accomplies de façon plus satisfaisante par des êtres humains car elles demandent des processus mentaux de haut niveau tels que l'apprentissage perceptuel, l'organisation de la mémoire et le raisonnement critique*, M. Minsky.

Des définitions qui ont varié au cours du temps (3)

4. Plus récemment (High-Level Expert Group in AI)

Artificial intelligence (AI) refers to systems designed by humans that, given a complex goal, act in the physical or digital worldby perceiving their environment, interpreting the collected structured or unstructured data, reasoning on the knowledge derived from this data and deciding the best action(s) to take (according to pre-defined parameters) to achieve the given goal.

Domaines

1. Domaines cœur
   - Raisonnement (programmation logique, programmation par contraintes), aide à la décision, représentation des connaissances, web sémantique
   - Apprentissage automatique, fouille de données, science des données
   - Robotique et perception, vision, traitement automatique des langues et de la parole, planification
   - Intelligence collective (systèmes multi-agents)
   - ...
2. Tout domaine applicatif
   - IA et société, IA et santé, IA et environnement, IA et Industrie 4.0, ...

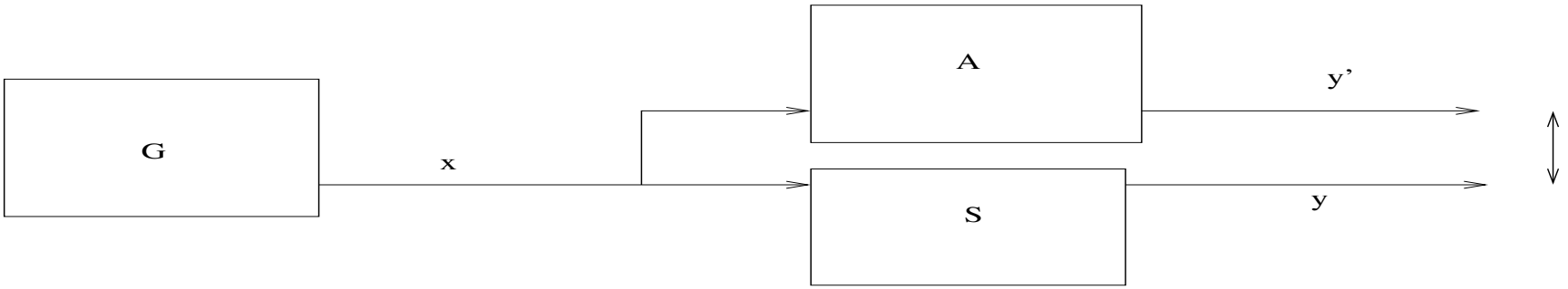
Table des matières

Qu'est-ce que l'apprentissage ?

- Apprentissage non supervisé
- Apprentissage supervisé (faiblement, semi-supervisé)
- Apprentissage par renforcement

Focus aujourd'hui sur l'apprentissage supervisé (apprentissage non supervisé plus tard)

L'apprentissage supervisé (1)



- *entrée* x, *sortie* y - $y = f^*(x)$, mais la fonction (processus/algorithme) f^* sur laquelle s'appuie S n'est pas connue
- On observe une série d'entrées et de sorties associées (ensemble d'apprentissage)
- À partir de ces observations, l'apprenant A cherche, parmi une famille de fonctions donnée, une fonction qui permet de passer des entrées aux sorties

L'apprentissage supervisé (2)

Données : ensemble d'apprentissage

- $\mathcal{D} = ((x^{(1)}, y^{(1)}), \cdots, (x^{(n)}, y^{(n)}))$
- x vecteur de réels - $x \in \mathbb{R}^p$
- $y \in \mathcal{Y}$ - catégorisation binaire : $\mathcal{Y} = \{0, 1\}$; régression linéaire simple : $\mathcal{Y} \subseteq \mathbb{R}$

Modèle/algorithme d'apprentissage

- Famille de fonctions $\mathcal{F}$ - exemple : ensemble des fonctions linéaires (régression linéaire)
- Mesure de l'erreur entre sortie réelle (y) et sortie prédite $y' = f(x)$, $f \in \mathcal{F}$

Objectif Sélectionner la fonction $f \in \mathcal{F}$ la plus appropriée (qui minimise les erreurs)

Mesure de la qualité d'une fonction apprise

On mesure la qualité d'une fonction à partir des erreurs qu'elle comment sur les sorties. Cette erreur est fondée sur une fonction de coût

Fonction de coût (*loss function*)

$$L : \mathcal{Y}x\mathcal{Y} \rightarrow \mathbb{R}^+, \text{telle que } L(y, y') > 0 \text{ pour } y \neq y'$$

Illustration

- Coût $0-1$:

$$L(y, y') = \begin{cases} 0 & \text{si } y = y', \\ 1 & \text{sinon} \end{cases}$$

- Coût quadratique :

$$L(y, y') = (y - y')^2$$

Sélection de $f \in \mathcal{F}$

On recherche la fonction qui minimise les erreurs

1. *Idéal* - Minimisation du risque fonctionnel :

$$\arg\min_{f \in \mathcal{F}} \underbrace{\int_x \int_y P(x,y) L(y, f(x)) dx dy}_{R(f) = \mathbb{E}_{P(x,y)}[L(y, f(x))]}$$

2. *Réaliste* - Minimisation du risque empirique :

$$\arg\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}))}_{\text{Remp}(f;\mathcal{D})} = \arg\min_{f \in \mathcal{F}} \text{Remp}(f; \mathcal{D})$$

Sélection de $f \in \mathcal{F}$

On recherche la fonction qui minimise les erreurs

1. *Idéal* - Minimisation du risque fonctionnel :

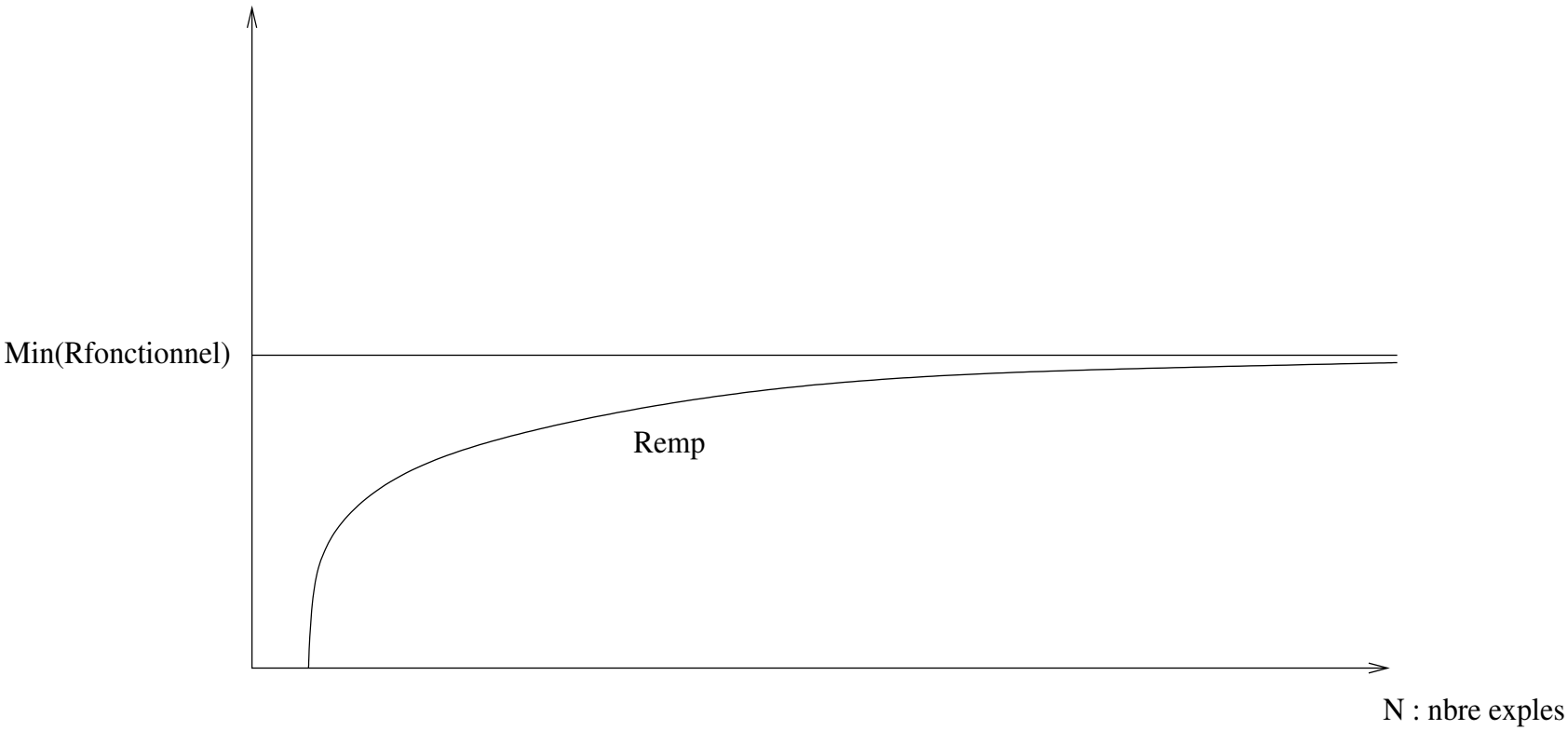
$$\arg\min_{f \in \mathcal{F}} \underbrace{\int_x \int_y P(x,y) L(y, f(x)) dx dy}_{R(f) = \mathbb{E}_{P(x,y)}[L(y, f(x))]}$$

2. *Réaliste* - Minimisation du risque empirique :

$$\arg\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}))}_{\text{Remp}(f;\mathcal{D})} = \arg\min_{f \in \mathcal{F}} \text{Remp}(f; \mathcal{D})$$

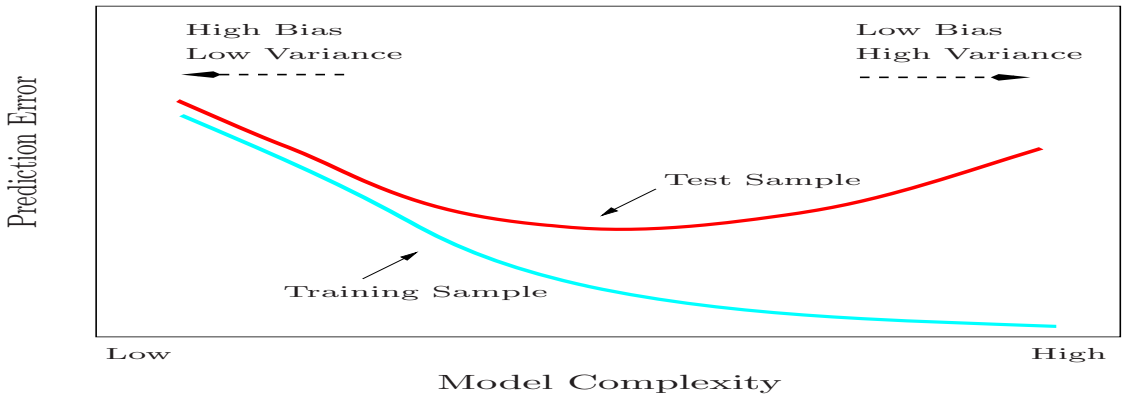
Justification intuitive de la minimisation du risque empirique

Pour $f \in \mathcal{F}$ fixée, le risque empirique tend vers le risque fonctionnel lorsque le nombre d'exemples d'apprentissage augmente



Mais en pratique ...

... lorsque le nombre d'exemples est limité :

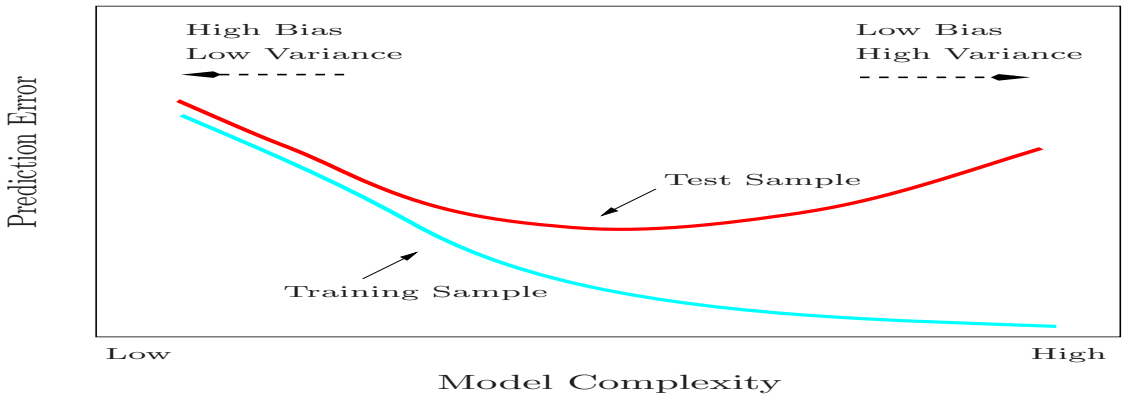


Solution : $\arg\min_{f \in \mathcal{F}} \text{Remp}(f) + \lambda\Omega(f)$
$\Omega(f)$ représente une mesure de la complexité de f

Image tirée de Elements of statistical learning. Hastie, Tibshirani, Friedman. Springer

Mais en pratique ...

... lorsque le nombre d'exemples est limité :



Solution : $\arg\min_{f \in \mathcal{F}} \text{Remp}(f) + \lambda\Omega(f)$
$\Omega(f)$ représente une mesure de la complexité de f

Image tirée de Elements of statistical learning. Hastie, Tibshirani, Friedman. Springer

Régularisation : complexité, connaissances et contraintes

$$\arg\min_{f \in \mathcal{F}} \text{Remp}(f) + \overbrace{\underbrace{\lambda}_{\text{paramètre de régularisation}} \Omega(f)}^{\text{régularisation}}$$

La régularisation permet :

- De rendre compte de la complexité de la fonction choisie
- D'intégrer des connaissances et des contraintes

Modèles d'apprentissage (1)

Un modèle (algorithme) d'apprentissage :

- A accès à un ensemble de fonctions $\mathcal{F}$ (par ex. perceptrons multi-couches)
- Sélectionne la fonction la plus approprié à partir (1) de l'ens. d'apprentissage et (2) de la fonction objectif définie par le concepteur
- Réalise cette sélection suivant des méthodes d'optimisation propres (par ex. descente de gradient stochastique (*stochastic gradient descent (SGD)*)

Modèles d'apprentissage (2)

Le concepteur définit :

- La fonction de coût qui l'intéresse (peut dépendre du modèle - *attention à la dérivabilité*)
- La fonction de régularisation Ω qui lui semble la plus appropriée (régularisation L_1, L_2, ... ; là aussi peut dépendre du modèle)

Un choix crucial : la représentation des exemples

Modèles d'apprentissage (2)

Le concepteur définit :

- La fonction de coût qui l'intéresse (peut dépendre du modèle - *attention à la dérivabilité*)
- La fonction de régularisation Ω qui lui semble la plus appropriée (régularisation L_1, L_2, ... ; là aussi peut dépendre du modèle)

Un choix crucial : la représentation des exemples

Feature engineering vs representation learning

1. Avant apprentissage profond : effort mis dans la recherche de la meilleure représentation

2. Apprentissage profond : effort mis dans l'architecture qui permet d'apprendre une représentation adaptée (nécessité néanmoins d'une première représentation)

Quelle famille de fonctions ?

Soit R^* le risque fonctionnel minimal sur toutes les familles de fonctions possibles. Soit $R_{\mathcal{F}}(f_{\text{min}})$ le risque fonctionnel minimal sur la famille de fonctions $\mathcal{F}$ et soit $R_{\mathcal{F}}(f)$ le risque d'une fonction f de $\mathcal{F}$.
On a :

$$R_{\mathcal{F}}(f) - R^* = \underbrace{(R_{\mathcal{F}}(f) - R_{\mathcal{F}}(f_{\text{min}}))}_{\text{erreur d'estimation}} + \underbrace{(R_{\mathcal{F}}(f_{\text{min}}) - R^*)}_{\text{erreur d'approximation}}$$

Remarque (attention, c'est une tendance)

- *Plus la famille est simple, plus l'erreur d'estimation est faible et plus l'erreur d'approximation est grande*
- *Inversement, plus la famille est complexe et plus l'erreur d'estimation est grande et plus l'erreur d'approximation est faible*

Quelle famille de fonctions ?

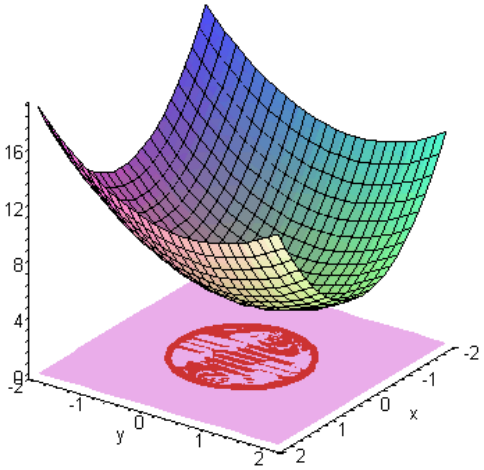
Soit R^* le risque fonctionnel minimal sur toutes les familles de fonctions possibles. Soit $R_{\mathcal{F}}(f_{\text{min}})$ le risque fonctionnel minimal sur la famille de fonctions $\mathcal{F}$ et soit $R_{\mathcal{F}}(f)$ le risque d'une fonction f de $\mathcal{F}$.
On a :

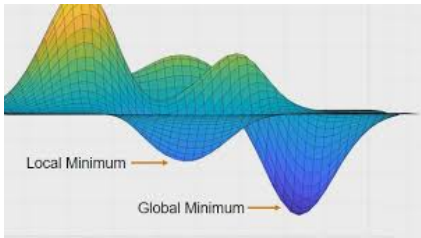
$$R_{\mathcal{F}}(f) - R^* = \underbrace{(R_{\mathcal{F}}(f) - R_{\mathcal{F}}(f_{\text{min}}))}_{\text{erreur d'estimation}} + \underbrace{(R_{\mathcal{F}}(f_{\text{min}}) - R^*)}_{\text{erreur d'approximation}}$$

Remarque (attention, c'est une tendance)

- *Plus la famille est simple, plus l'erreur d'estimation est faible et plus l'erreur d'approximation est grande*
- *Inversement, plus la famille est complexe et plus l'erreur d'estimation est grande et plus l'erreur d'approximation est faible*

Compromis estimation-approximation

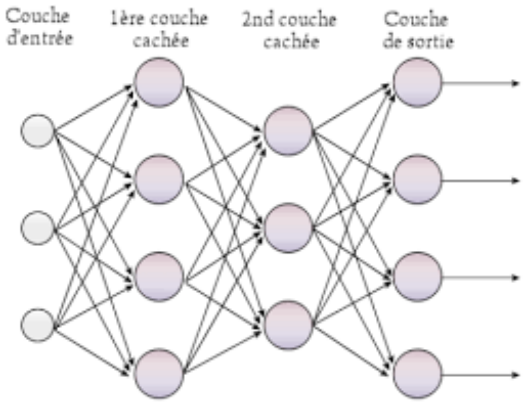




Apprentissage profond : perceptron multicouche (1)

Terminologie : *Multilayer Perceptron (MLP), Feedforward Neural Network (FFNN), fully connected network*

- $\mathbf{y} \in \mathbb{R}^4, \mathbf{x} \in \mathbb{R}^3$
- $\mathbf{y} = f(\mathbf{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x})))$
- Profondeur du réseau, dimensionalité des couches



Apprentissage profond : perceptron multicouche (2)

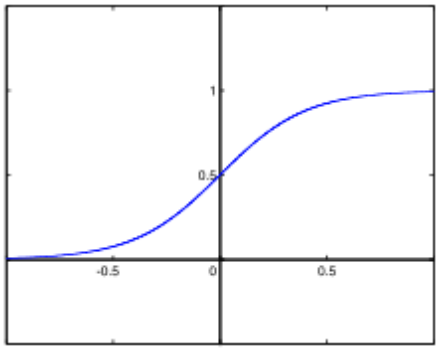
Quelles fonctions f^i intermédiaires ?

Soit $\mathbf{h^{i-1}}$ l'entré de f^i ($\mathbf{h^0} = \mathbf{x}$) :

$$f^i(\mathbf{h^{i-1}}) = \sigma(\mathbf{W}^i\mathbf{h^{i-1}} + \mathbf{b}^i)$$

avec $\mathbf{h^{i-1}} \in \mathbb{R}^{p_i}$, $\mathbf{W}^i \in \mathbb{R}^{p_{i+1} \times p_i}$, $\mathbf{b}^i \in \mathbb{R}^{p_{i+1}}$

La fonction σ est une fonction *en général non linéaire* d'activation (sigmoïde, tanh, RELU, identité)



Apprentissage profond : perceptron multicouche (3)

- Un MLP est un approximateur universel
- Famille de fonctions riche : bonne approximation mais estimation plus délicate a priori
- Nombre de paramètres important (exemple ci-dessus)
- Nécessité d'un grand nombre d'exemples d'apprentissage
- Régularisation (norme L_1, L_2, ...)
- Compréhension incomplète (qualité des minima locaux)

Table des matières

Comment évaluer un algorithme d'apprentissage ?

Découpage apprentissage/test

- Veiller à avoir suffisamment d'exemples en apprentissage et en test ($\geq$ 100 pour le test)
- On partage souvent un ens. de données à disposition suivant une règle 80-20 ou 70-30 (appr.-test)
- Il est important d'avoir la même distribution entre les données d'apprentissage et de test : partage aléatoire
- Des contraintes (par ex. temporelle) peuvent toutefois être prises en compte
- Apprendre modèle sur appr. et évaluer sur test - données test non utilisées pour l'apprentissage

Comment évaluer un algorithme d'apprentissage ?

Découpage apprentissage/validation/test

- Ens. de validation pour ajuster les hyperparamètres des modèles (degré d'un polynôme, nbre neurones sur couche cachée, ...)
- Partage aléatoire appr.-valid.-test suivant une règle 64-16-20 ou 49-21-30
- Pour chaque valeur des hyperparamètres (par ex. degré = 1, 2 ou 3), apprendre modèle sur appr. et évaluer sur valid.
- Sélectionner le meilleur hyperparamètre et réapprendre modèle sur appr.+valid.
- Évaluer sur test - données test non utilisées pour l'apprentissage et la validation

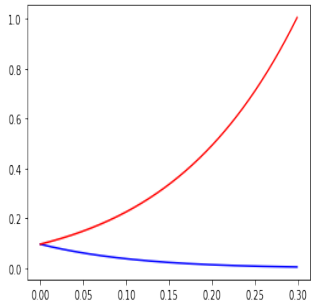
Comment évaluer un algorithme d'apprentissage ?

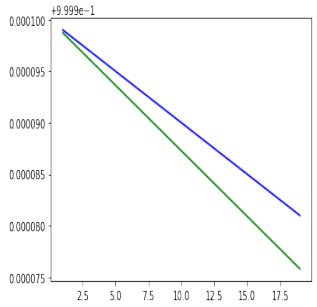
Validation croisée

- Partage (aléatoire et équilibré) des données en k groupes $\{g_1, \cdots, g_k\}$ (*k-fold cross-validation*) - $k = 3, 5, 10$
- Construction de k jeux appr.-valid.-test
  - Jeu 1 : appr.=$\{g_1, \cdots, g_{k-2}\}$; valid.=g_{k-1} ; test = g_k
  - Jeu 2 : appr.=$\{g_2, \cdots, g_{k-1}\}$; valid.=g_k ; test = g_1
  - ...
- Apprentissage, validation et évaluation sur chaque jeu (voir ci-dessus)
- Calcul de la moyenne et de la variance de la mesure d'évaluation sur tous les jeux
- Avantage : moy. et variance ; utilisation de toutes les données en apprentissage et en test

Quelques remarques sur l'évaluation

Attention aux effets d'échelle





Attention aux différences non significatives

- Un système B qui améliore un système A de 0,008 point (par exemple $A = 0,783$ et $B = 0,791$) est-il vraiment meilleur ?
- Utilisation de test de significativité statistique

Table des matières

L'annotation : un processus plus ou moins complexe

La disponibilité de données annotées (ou facilement annotables) dépend de la tâche considérée.

- Traduction automatique
- Systèmes de question-réponse
- Pertinence des pages web pour une requête
- Objets dans des images, actions dans des vidéos

L'annotation : un processus plus ou moins complexe

La disponibilité de données annotées (ou facilement annotables) dépend de la tâche considérée.

- Traduction automatique
- Systèmes de question-réponse
- Pertinence des pages web pour une requête
- Objets dans des images, actions dans des vidéos

Exemple : annotation pour les moteurs de recherche

- Une source importante d'information : les clics des utilisateurs
  - Utiliser les clics pour inférer des préférences entre documents (paires de préférence)
  - Compléter éventuellement par le temps passé sur le résumé d'un document (*eye-tracking*)
- Que peut-on déduire des clics ?

Exploiter les clics (1)

Les clicks ne fournissent pas des jugements de pertinence absolus, mais relatifs. Soit un ordre $(d_1, d_2, d_3, \cdots)$ et C l'ensemble des documents cliqués. Les stratégies suivantes peuvent être utilisées pour construire un ordre de pertinence entre documents :

1. Si $d_i \in C$ et $d_j \notin C$, $d_i \succ_{pert-q} d_j$
2. Si d_i est le dernier doc cliqué, $\forall j < i$, $d_j \notin C$, $d_i \succ_{pert-q} d_j$
3. $\forall i \geq 2, d_i \in C, d_{i-1} \notin C, d_i \succ_{pert-q} d_{i-1}$
4. $\forall i, d_i \in C, d_{i+1} \notin C, d_i \succ_{pert-q} d_{i+1}$

Exploiter les clics (2)

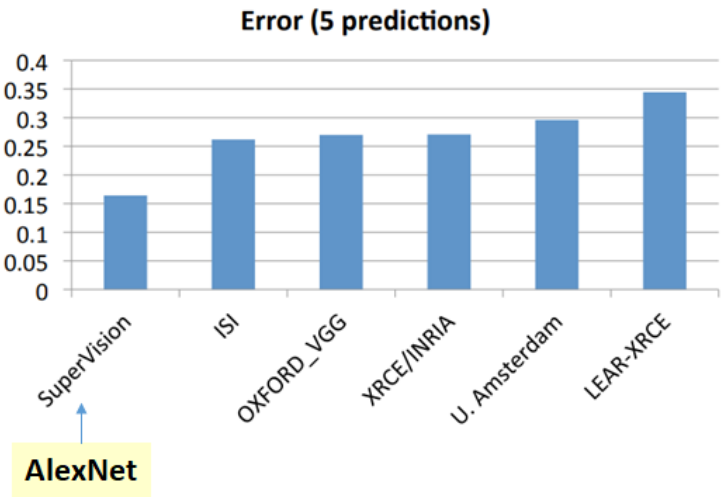
- Ces différentes stratégies permettent d'inférer un ordre partiel entre documents
- La collecte de ces données fournit un ensemble d'apprentissage très large, sur lequel on peut déployer les techniques d'apprentissage par ordonnancement
- La RI sur le web est en partie caractérisée par une course aux données :
  - Indexer le maximum de pages
  - Récupérer le maximum de données de clics

Table des matières

Enthousiasmes (1)

- Des résultats inégalés sur certaines tâches (vision 2012, traitement automatique des langues 2016, ...)

Ranking of the best results from each team



Enthousiasmes (2)

- Des approches de bout-en-bout permettant de résoudre des problèmes complexes
  - Bibliothèques (Tensorflow, Pytorch) qui permettent de construire des applications complexes
  - Composition de réseaux *en légo* (descente de gradient avec backpropagation - dérivabilité des fonctions)
  - Question-réponse, compréhension de textes : de l'intégration de services à des réseaux intégrés (code simplifié de manière très significative)
- Accessible à n'importe quelle personne ayant des compétences en programmation !

Limites

- Nombre d'exemples : quelques exemples sont nécessaires à un enfant de 3 ans pour reconnaître un chien par exemple (des dizaines de millions pour un ordinateur)
- Certifiabilité, explicabilité, équité (CEE - *FAT*)

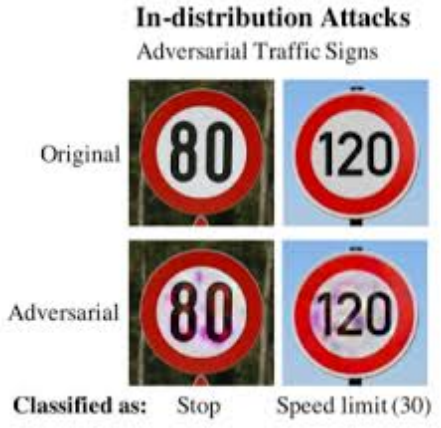


Table des matières

Conclusion

- L'IA : un domaine en effervescence, notamment grâce à l'apprentissage profond et à l'accès *facile* aux technologies
- Un mouvement qui s'inscrit dans le développement numérique (*nil novi sub sole*)
- Régulation juridique et éthique *délicate*
- Quelle en sera la suite ? Il est difficile de prédire ...

... surtout le futur !

Conclusion

- L'IA : un domaine en effervescence, notamment grâce à l'apprentissage profond et à l'accès *facile* aux technologies
- Un mouvement qui s'inscrit dans le développement numérique (*nil novi sub sole*)
- Régulation juridique et éthique *délicate*
- Quelle en sera la suite ? Il est difficile de prédire ...

... surtout le futur !

Quelques Références (1)

- *Artificial Intelligence : Modern Magic or Dangerous Future ?*, Y. Wilks, Icon Books, 2019
- *Le temps des algorithmes*, S. Abiteboul and G. Dowek, Éditions Le Pommier, 2017
- *Le mythe de la singularité*, J.-G. Ganascia, Éditions du Seuil, 2017
- *Deep Learning*, I. Goodfellow,Y. Bengio and A. Courville, MIT Press, 2016
- *Thinking, Fast and Slow*, D. Kahneman, Penguin Books, 2011

Quelques Références (2)

- *An introduction to information retrieval*, C. D. Manning, P. Raghavan , H. Schütze, Cambridge University Press, 2009
- *Elements of statistical learning*, T. Hastie, R. Tibshirani and J. Friedman, Springer, 2008
- *Causality : Models, Reasoning and Inference*, J. Pearl, Cambridge University Press, 2000
- *Artificial Intelligence : A Modern Approach*, S. Russel and P. Norvig, Prentice Hall, 1995 (1st edition)

Les auteurs à éviter

- Hors domaine IA (discours simplificateur voire erroné, scientifiquement non fondé)
- À la recherche d'une audience (promoteurs de leurs ouvrages)