Learning
Deep or not Deep…
De MENACE à AlphaGo

JM Alliot

Le Deep Learning en Version Simple

Qu'est-ce que c'est ?

Croisement entre deux univers

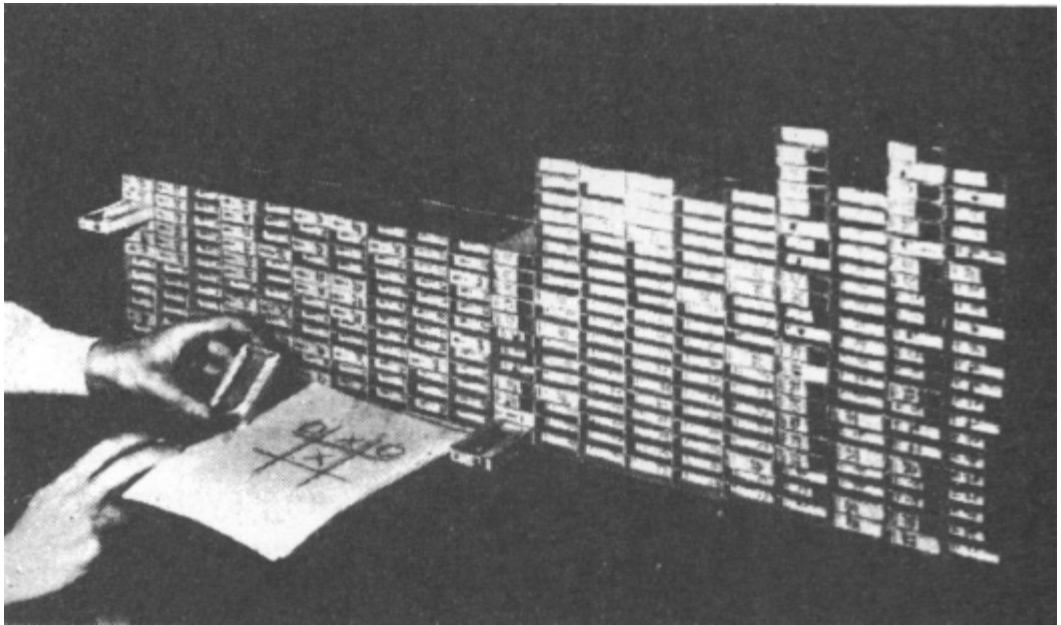
- L'apprentissage
  - Par renforcement: on est capable d'évaluer le résultat et de l'utiliser pour améliorer le programme
  - Supervisé: une base de données d'apprentissage est disponible cas par cas
  - Non supervisé: algorithmes de classification (k-means,…),…
- Les neurosciences

L'apprentissage par renforcement

- *Turing:*
  - « Instead of trying to produce a program to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain. Presumably the child brain is something like a notebook as one buys it from the stationers. Rather little mechanism, and lots of blank sheets."
  - "The use of punishments and rewards can at best be a part of the teaching process. Roughly speaking, if the teacher has no other means of communicating to the pupil, the amount of information which can reach him does not exceed the total number of rewards and punishments applied. By the time a child has learnt to repeat "Casabianca" he would probably feel very sore indeed, if the text could only be discovered by a "Twenty Questions" technique, every "NO" taking the form of a blow."

L'apprentissage par renforcement

- Arthur Samuel (1959): « Some Studies in Machine Learning Using the Game of Checkers »
  - Memoization
  - Ajustement automatique de paramètres par régression.
- Donald Michie (1962): MENACE (*Matchbox Educable Noughts And Crosses Engine)*

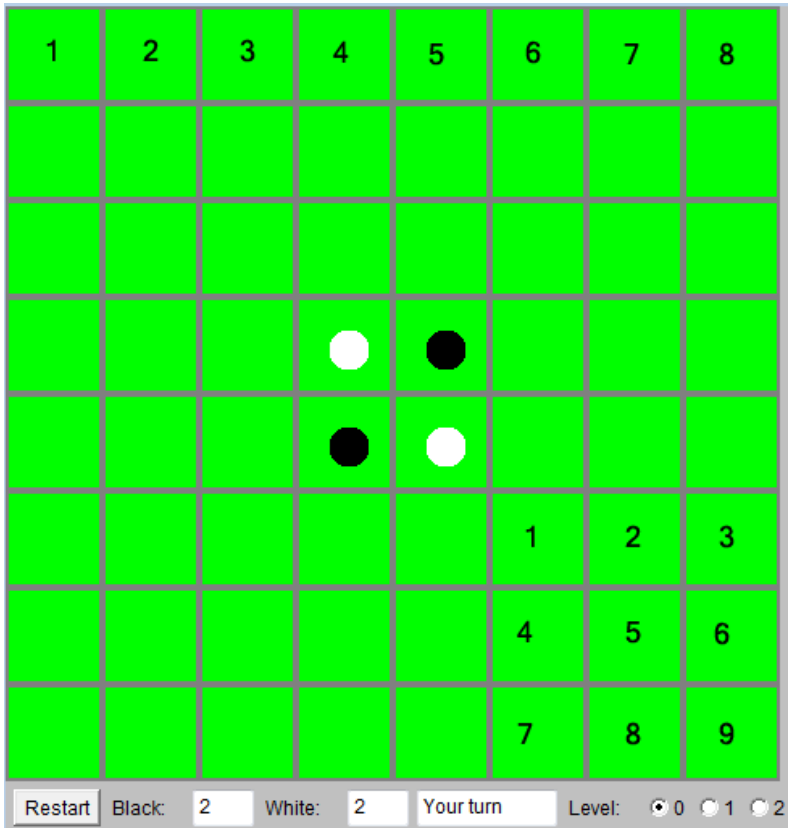


L'apprentissage par renforcement

- *MENACE permet de comprendre de façon « visuelle » de nombreux problèmes que l'on rencontre dans l'apprentissage:*
  - *La représentation de la connaissance (front end processing)*
  - *Le nombre élevé de paramètres*
  - *La façon d'initialiser le système*
  - *La façon de récompenser / punir: comment identifier le coup « responsable » en cas de victoire ou de défaite*
  - *La façon de constituer la base d'apprentissage*
  - *La façon de présenter la base d'apprentissage*
  - *Le nombre élevé d'exemples nécessaires*

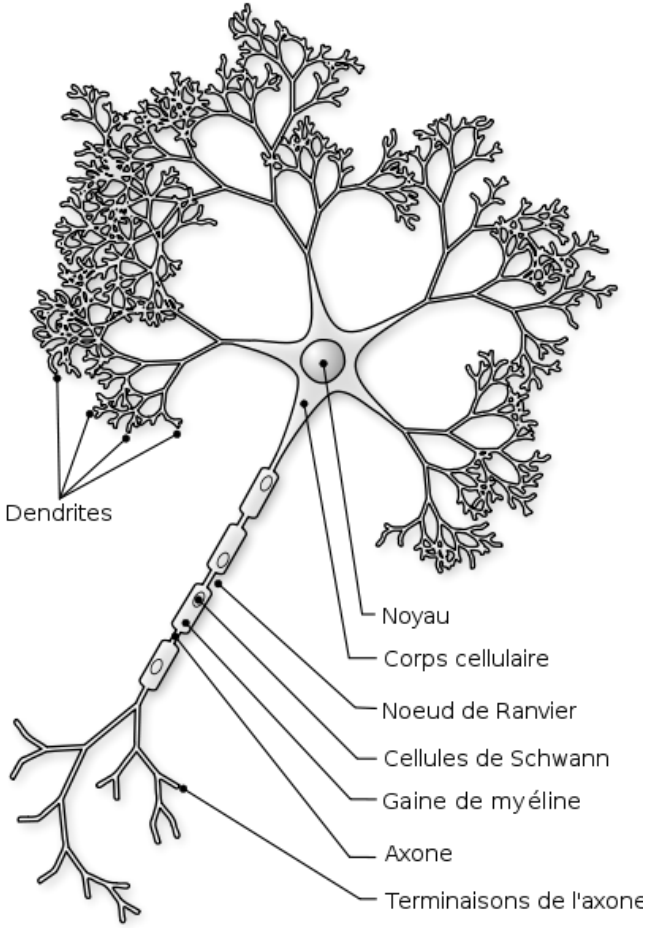
L'apprentissage par renforcement

- La méthode, aussi simpliste qu'elle paraisse, fonctionne même dans des cas beaucoup plus complexes
- OTAGE, programme d'Othello que j'ai écrit dans les années 90 fonctionne exactement sur le même principe, a figuré dans les 10 premiers mondiaux



Le modèle physiologique du neurone

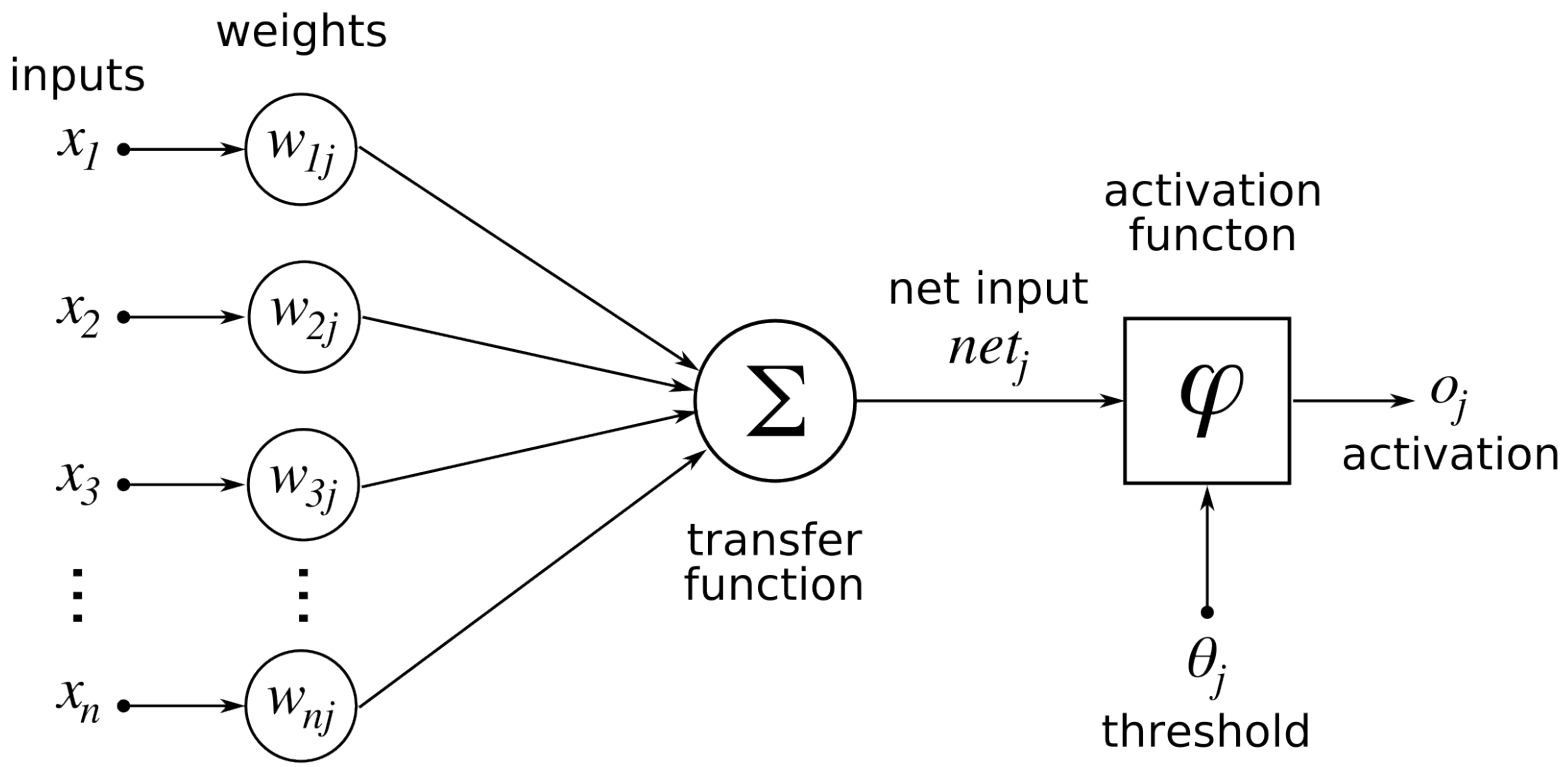
- Les dendrites véhiculent l'information vers le neurone
- Les synapses sont les zones de liaison entre deux neurones (entre l'axone de l'émetteur et la dendrite du récepteur).
- Les synapses peuvent être de nature électriques ou chimiques.
- Les synapses peuvent être excitatrices ou inhibitrices.
- La réponse du neurone est de la forme « tout ou rien » relativement à la valeur de ses entrées par rapport à un seuil.



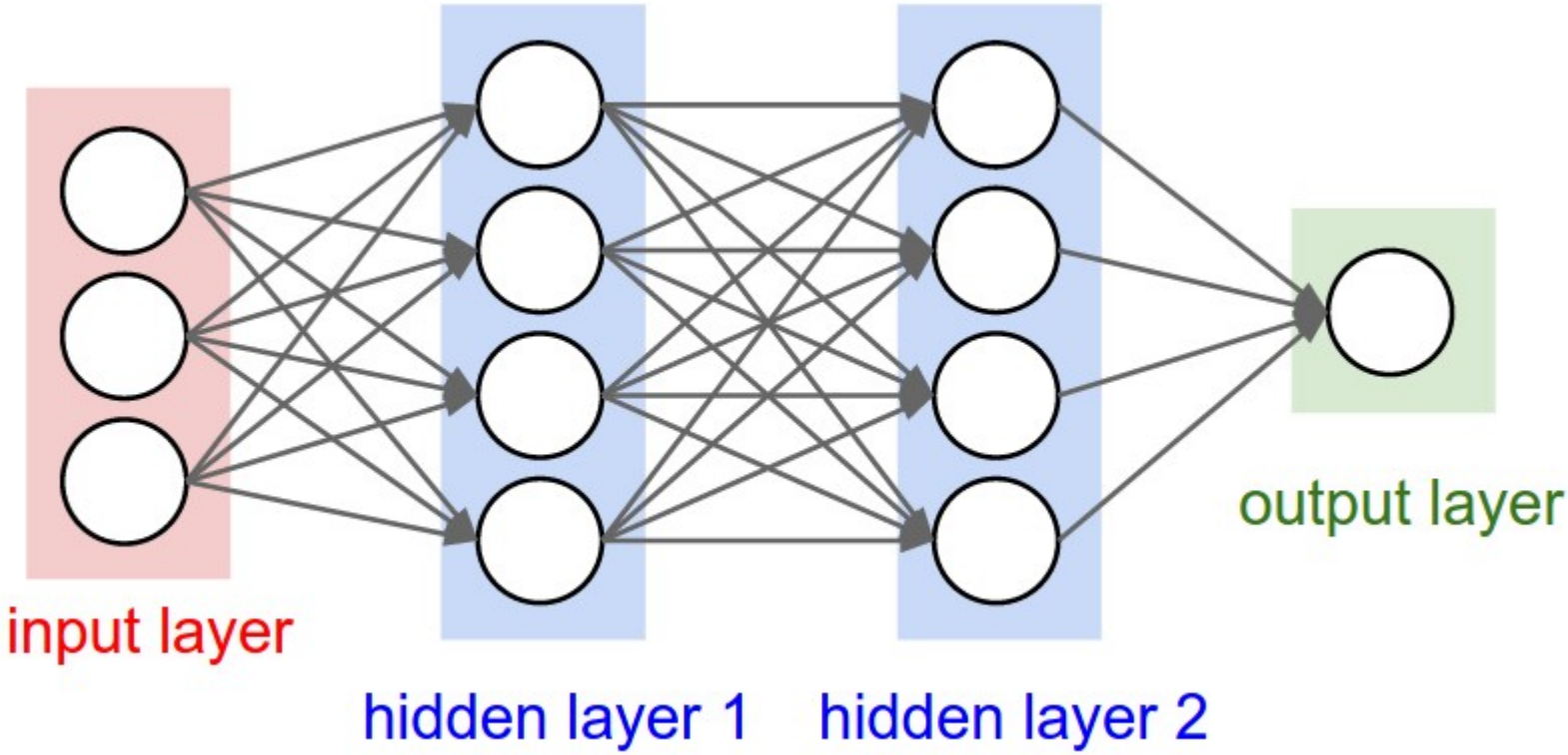
Les réseaux de neurones

- Warren McCulloch and Walter Pitts: *A Logical Calculus of Ideas Immanent in Nervous Activity (1942):*
  - "Because of the "all-or-none" character of nervous activity, neural events and the relations among them can be treated by means of propositional logic. It is found that the behavior of every net can be described in these terms, with the addition of more complicated logical means or nets containing circles; and that for any logical expression satisfying certain conditions, one can find a net behaving in the fashion it describes."
- D. O. Hebb : *The Organization of Behavior (1949):*
  - « Let us assume that the persistence or repetition of a reverberatory activity (or "trace") tends to induce lasting cellular changes that add to its stability.[…] When an axon of cell *A* is near enough to excite a cell *B* and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that *A*'s efficiency, as one of the cells firing *B*, is increased."
- F. Rosenblatt *: The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain (1958)*
- P. Werbos : *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences (1974)*
  - Backpropagation Algorithm

Les réseaux de neurones

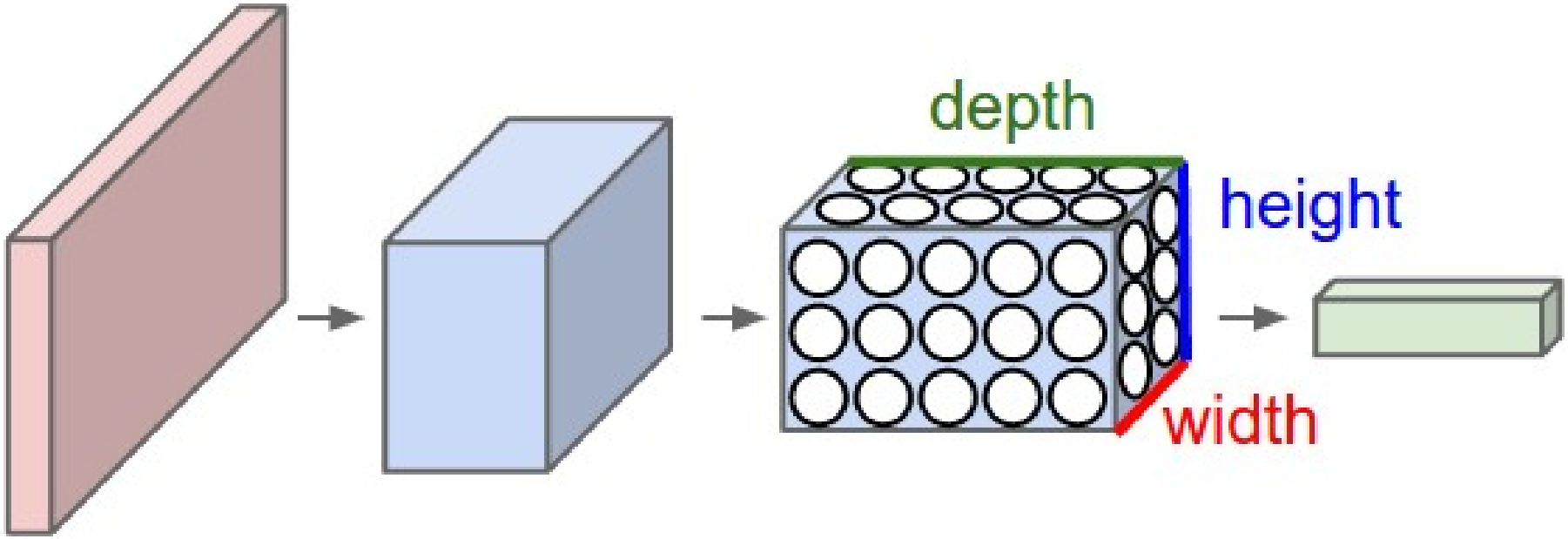


FeedForward Networks

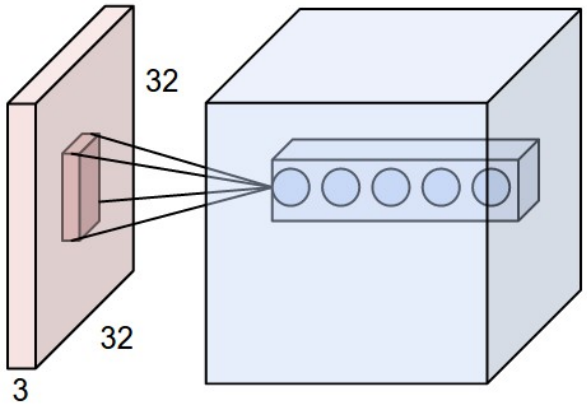


- Apprentissage supervisé: backprop
- Question: comment organiser les couches intermédiaires?

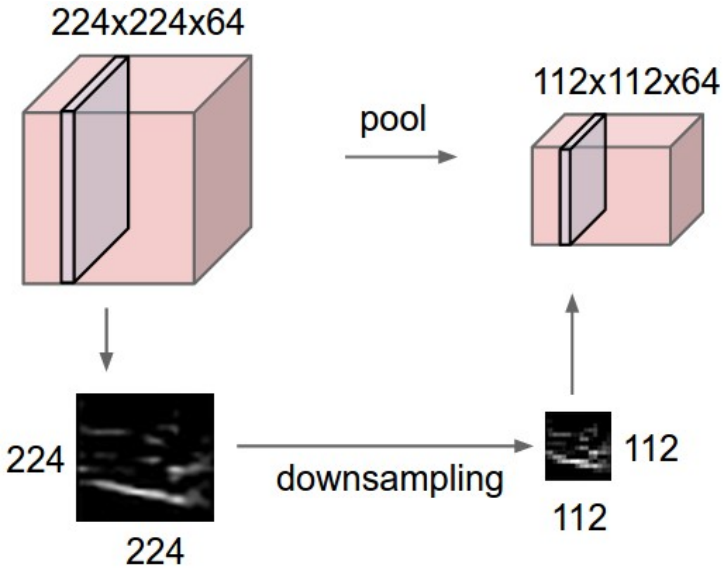
Convolutional Neural Networks (CNN)



Couches

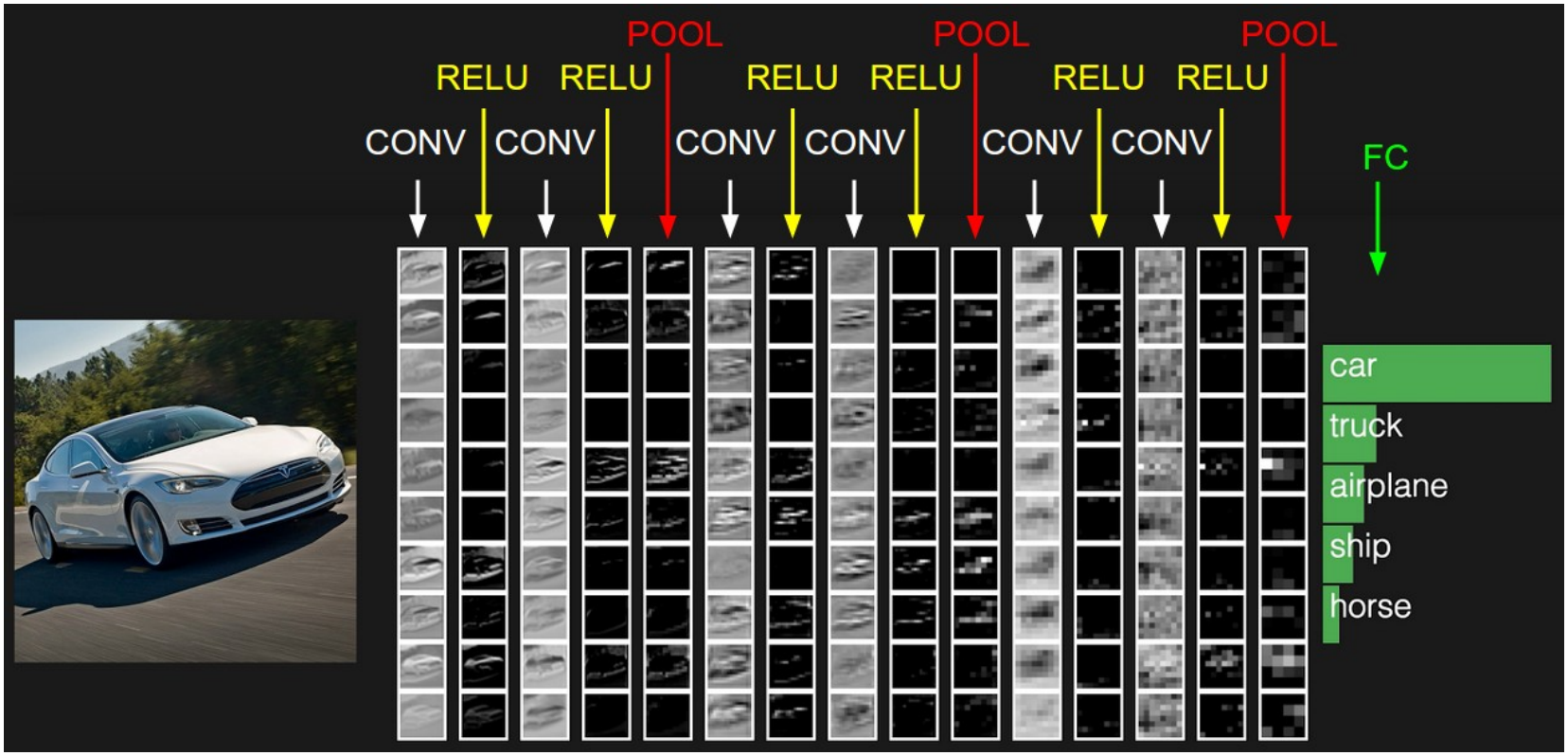


CONVolution layer



POOLing layer

CNN: apprentissage supervisé pour la classification d'images



CNN et apprentissage supervisé

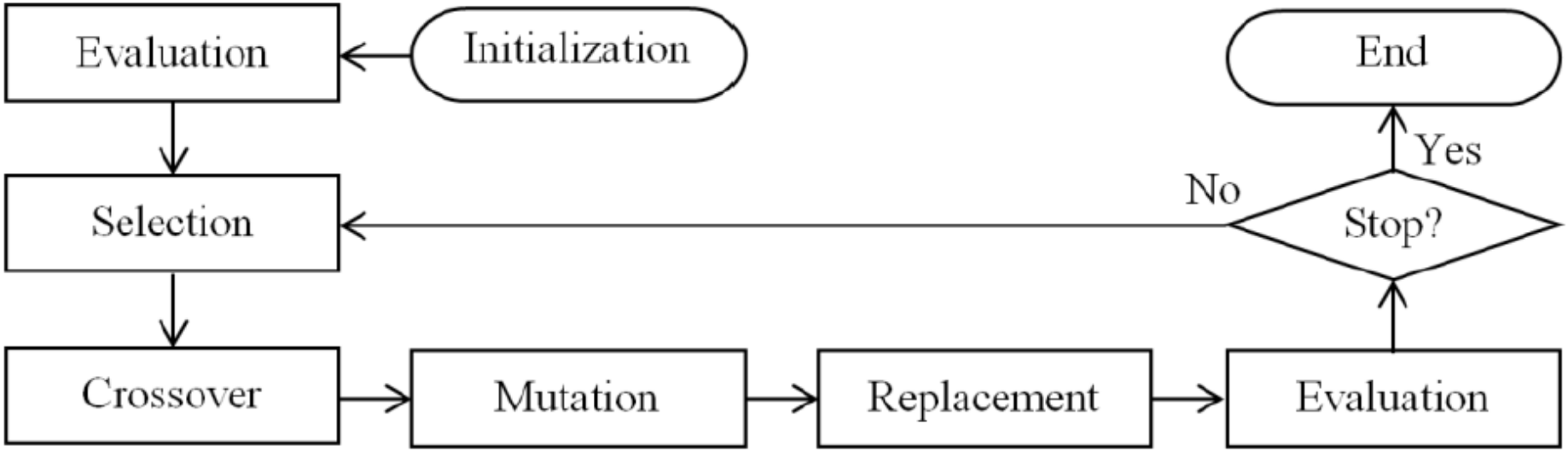
- La construction du réseau est un art
  - 1998: LeNet par LeCun/Bottou/Bagio : CONV layer
  - Large Scale Visual Recognition Challenger
    - 2012: AlexNet : multiple CONV layers
    - 2014: GoogLeNet: Inception modules
      - Doing each convolution in parallel and concatenating the resulting feature
    - 2015: Residual Networks (ResNet)
      - Skip Layer
      - Batch normalization
      - No Fully Connected Layers
    - etc…….
- L'apprentissage supervisé est un domaine actif déjà largement fouillé

Réseaux de neurones et renforcement

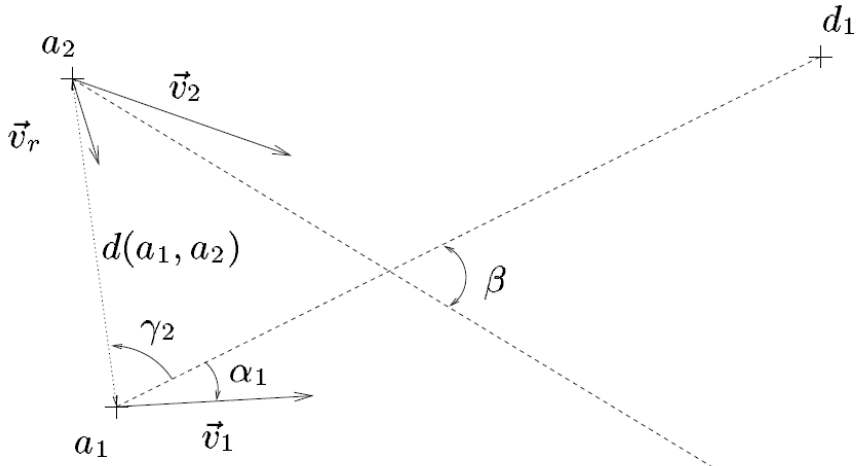
Comment faire lorsque l'on ne dispose pas de base d'exemples mais que l'on sait évaluer la qualité du résultat?

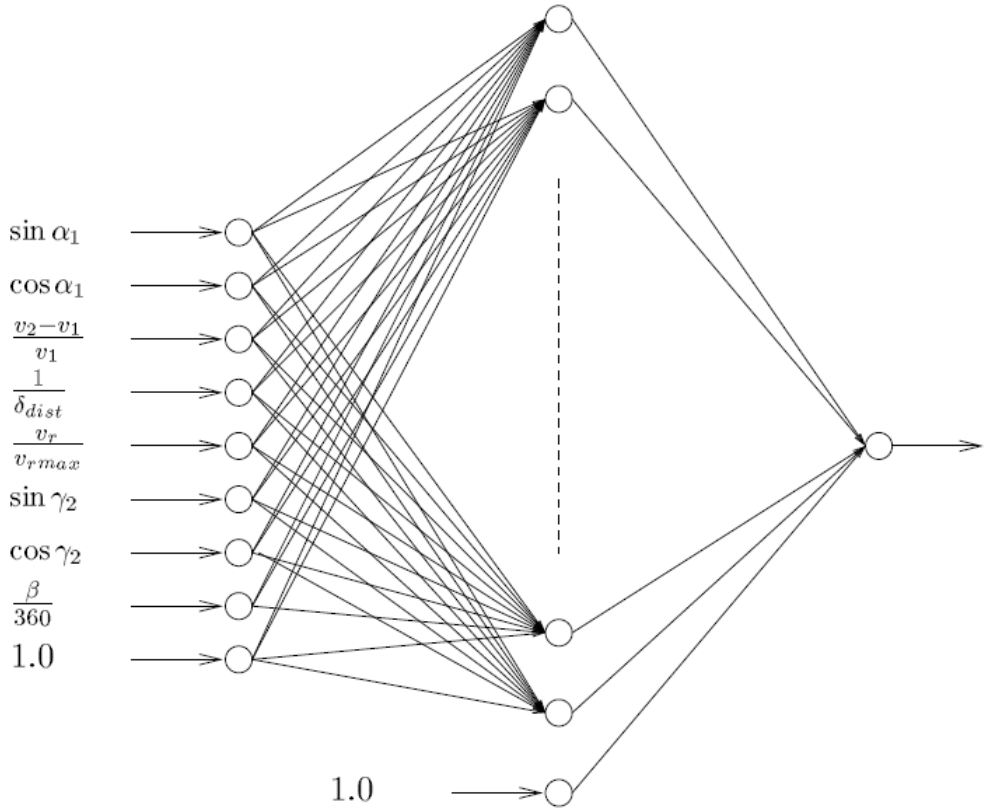
- Utiliser des techniques de type génétique pour sélectionner et faire évoluer les réseaux de neurones (AG+NN) ou les fonctions d'évaluation (GP)
- Sutton (1988): « Learning to predict by the methods of temporal differences"
  - Algorithme TD-lambda (exemple: TD-gammon)
- Watkins (1989): « Learning from Delayed Rewards »
  - Algorithme de « Q-learning »
  - Algorithme SARSA
  - Deep Q learning
  - Double Q learning
  - etc….

Algorithmes génétiques

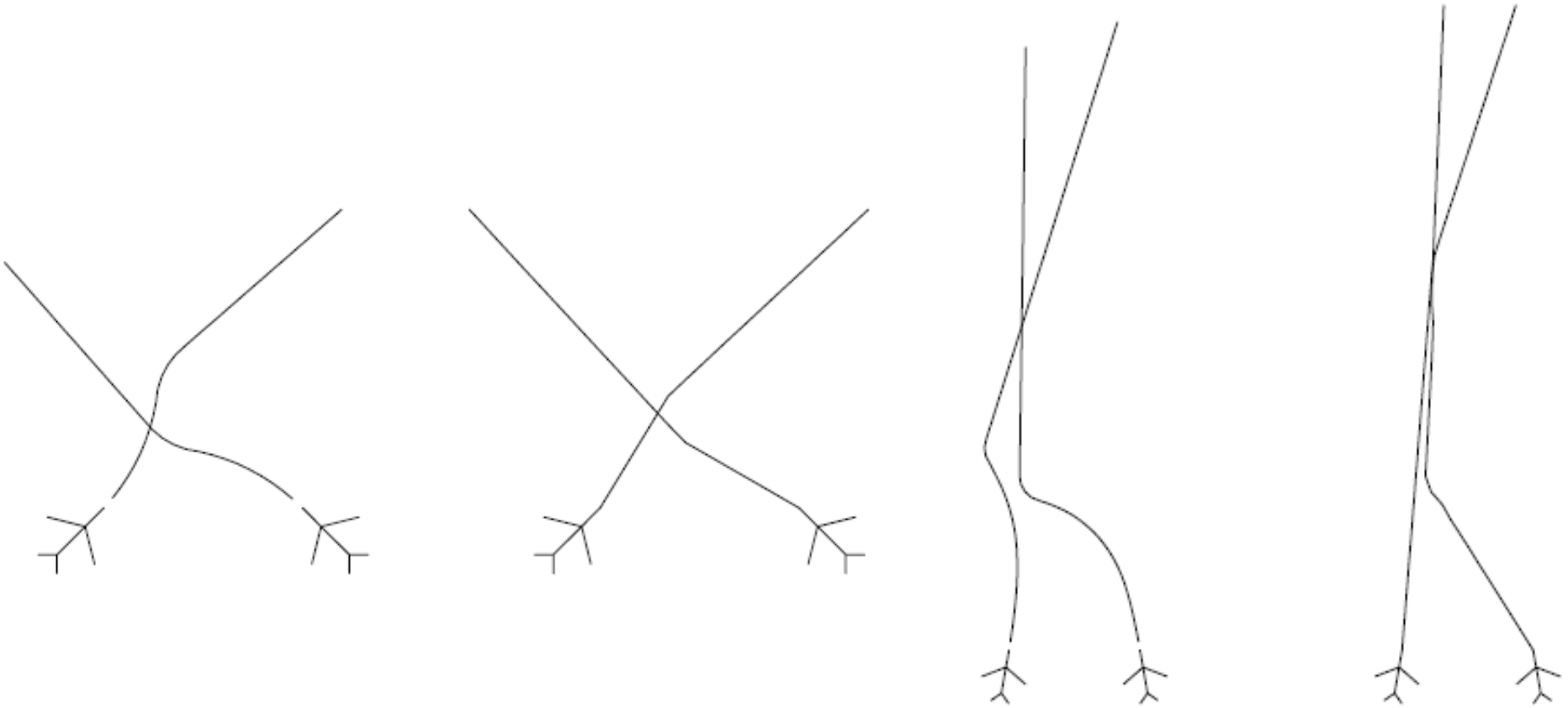


Résolution de conflits (Alliot-Durand-Médioni 1996)





Résolution de conflits (RN+AG)



Learning from delayed rewards

Famille de problèmes qui ont tous en commun la même difficulté:

- Le problème des bandits manchots
- Les algorithmes de type « Monte Carlo Tree Search »
- L'apprentissage d'un RN en fonction du résultat

Les bandits manchots

- On dispose de n « bandits manchots », quelle est la meilleure stratégie d'exploration pour maximiser le gain? (*exploitation-exploration dilemna*)
  - L'exploration permet de mieux identifier les « meilleurs » bandits manchots, mais fait chuter le gain
  - Se focaliser rapidement sur un bandit manchot peut amener à manquer le "bon" et donc à faire également chuter le gain.
- Upper Confidence Bound (UCB) algorithm:
  - Xj: moyenne des résultats pour le bras j
  - Nj: nombre de fois où le bras j a été utilisé
  - N: nombre total de coups joués
  - Prendre le j qui maximise:

$$\mathrm{UCB1} = \overline{X}_j + \sqrt{\frac{2\ln n}{n_j}}$$

Q learning

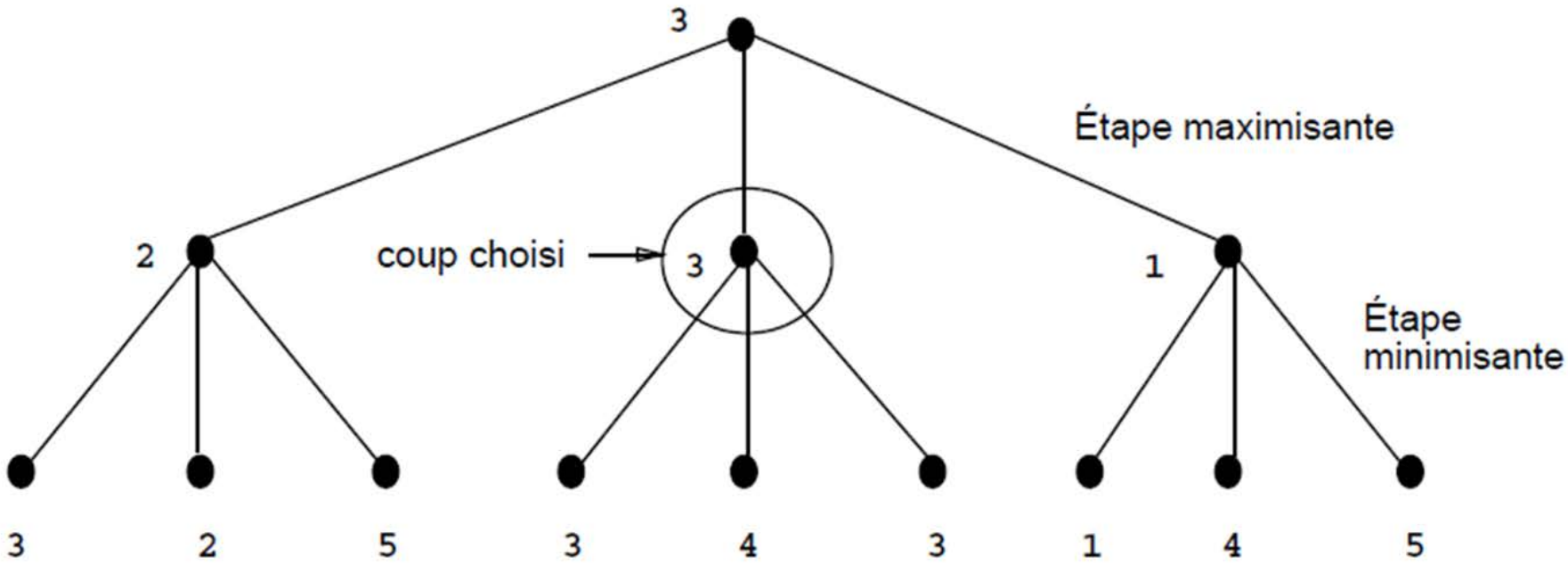
- On dispose d'un ensemble S d'états et un ensemble A d'actions. A chaque instant t, dans l'état s_t, on peut exécuter une ou plusieurs actions a_t qui permettent de passer dans un nouvel état s_{t+1}. Une action amène un gain immédiat r_t, et le but est de maximiser la somme des gains jusqu'à l'état final en partant de l'état initial.
- Problème dont on « voit » la proximité avec le bandit manchot, mais aussi avec le problème de l'apprentissage par renforcement.
- On construit une fonction $Q(s_t,a_t)$ qui va estimer la « valeur » de l'exécution de l'action a_t dans l'état s_t

$$Q(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$$

- Q(s,a) est normalement une fonction discrète stockée sous forme de table. Si le nombre d'états et/ou d'actions deviennent grands, on peut estimer Q en utilisant un DNN => algorithme Deep Neural Q Learning (DQN, puis DDQN)
- 2013: Deepmind développe un programme basé sur un DQN pour jouer parfaitement à certains jeux d'Atari

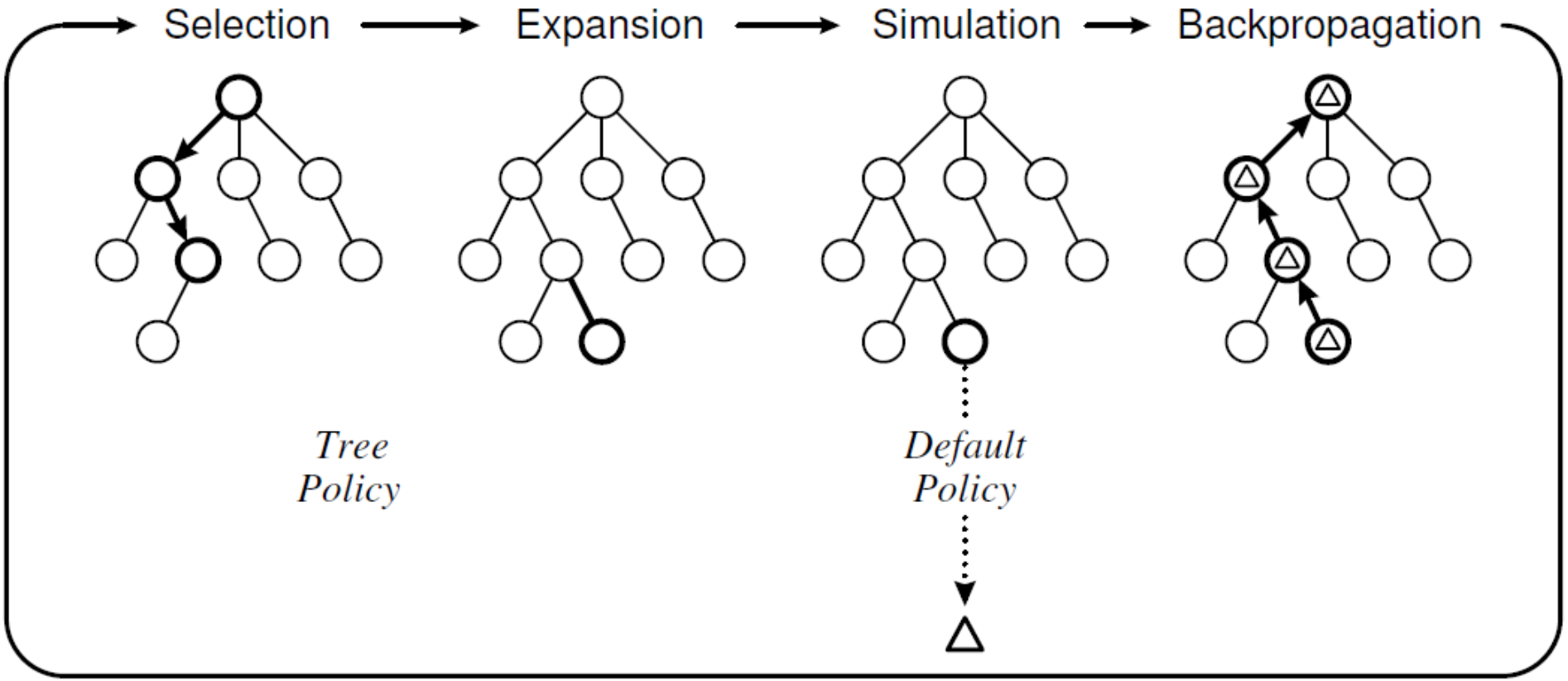
Monte-Carlo Tree Search (MCTS)

- En théorie des jeux, les algorithmes « standards » utilisés sont de la forme minimax (avec toutes leurs variantes) à profondeur fixe.

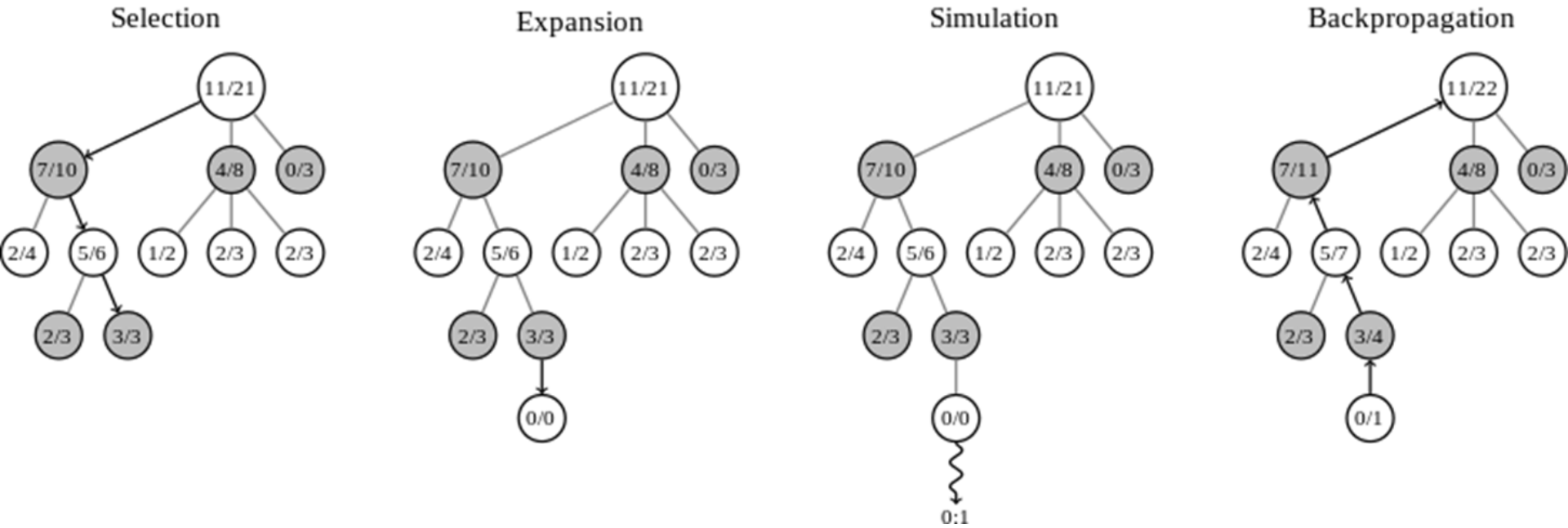


Monte-Carlo Tree Search (MCTS)

- L'algorithme et le terme MCTS ont été définis par Remi Coulom (un des directeurs de thèse d'Aja Huang et l'auteur de CrazyStone) en 2006 (l'idée remonte à Abramson en 1987). L'algorithme est séparé en 4 phases et étend l'arbre de jeu de façon « inégale ».
- La« tree policy » utilisée est souvent l'algorithme UCB.
- MCTS est généralement moins efficace que l'alpha-béta sauf…



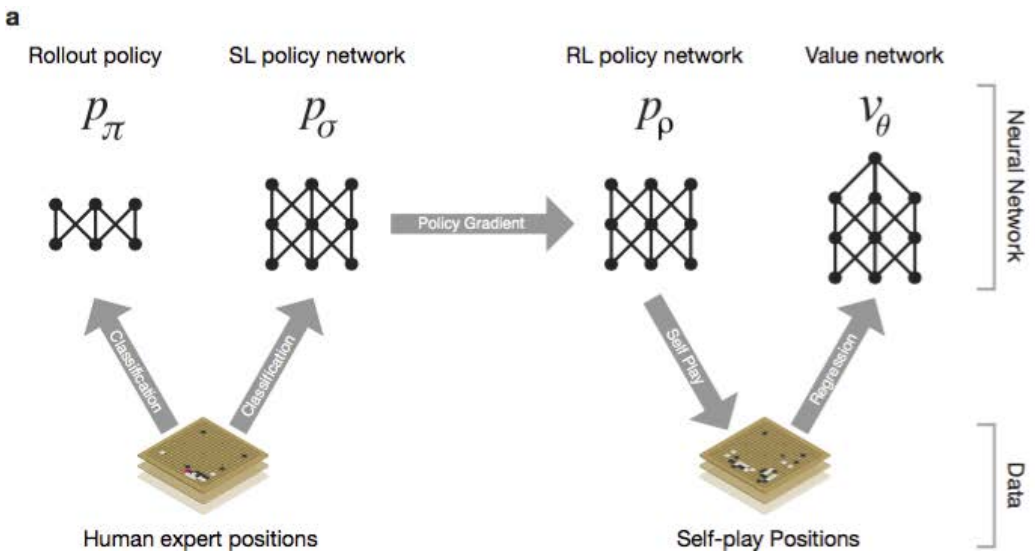
Monte-Carlo Tree Search (MCTS)



DLVS

AlphaGo

- 2 DNN: un « policy network » et un « value network »
- Etape 1: apprentissage supervisé de p_π et p_σ à partir de 30 millions de positions récupérées sur le serveur KGS. La sortie est un vecteur de probabilités évaluant l'ensemble des coups possibles à partir de la position courante.
- Etape 2: Construction de p_ρ à partir de p_σ en faisant jouer le programme contre lui-même.
- Etape 3: Construction de v_θ (en utilisant p_ρ) qui retourne la probabilité de victoire pour une position donnée.
- Etape 4: on utilise un algorithme de type MCTS avec une « tree policy » proche du Q learning



AlphaGo: détail de l'algorithme MCTS

- Une action pour un nœud est choisi suivant l'équation (proche UCB):

$$a_t = \underset{a}{\operatorname{argmax}} \left(Q(s_t, a) + u(s_t, a)\right) \qquad u(s, a) \propto \frac{P(s,a)}{1+N(s,a)}$$

- Tous les *P(s,a)* sont initialisés par le vecteur fourni par le réseau p_σ (et non p_ρ) lors de la première visite du nœud (p_ρ fonctionne moins bien !).
- *N(s,a)* est le nombre de fois où l'action *a* a été exécutée pour le nœud *s*
- Lorsque l'on atteint un nœud terminal s_L sa valeur est évaluée en utilisant la combinaison linéaire du réseau v_θ et d'une simulation terminale utilisant la politique de choix rapide p_π

$$V(s_L) = (1 - \lambda)v_\theta(s_L) + \lambda z_L$$

- Comme dans un MCTS standard, les valeurs *Q(s,a)* impliquant les nœuds *s* traversées sont mises à jour suivant l'équation:

$$N(s, a) = \sum_{i=1}^{n} \mathbf{1}(s, a, i) \qquad Q(s, a) = \frac{1}{N(s, a)} \sum_{i=1}^{n} \mathbf{1}(s, a, i) V(s_L^i)$$

AlphaZero

- AlphaGo contre LeeSedol (9-15 mars 2016): 4-1
- AlphaGoMaster (4 TPUv2):
  - 60-0 dans des matchs rapides en ligne contre les meilleurs humains
  - 5-0 dans « The future of Go Summit » dont 3-0 contre KeJie
- AlphaGoZero est une version d'AlphaGo qui n'a construit sa connaissance qu'à partir de parties jouées contre lui-même (pas d'apprentissage supervisé à partir des positions de KGS). Il est légèrement plus fort qu'AlphaGoMaster.
- AlphaZero est un algorithme générique utilisé à la fois pour le Go, les échecs et pour le shogi qui reprend l'architecture générale d'AlphaGoZero avec quelques modifications. Il utilise 64 TPUv2 pour l'apprentissage et 4 TPUv2 pour jouer.
  - AlphaZero est légèrement plus fort qu'AlphaGoZero: 60-40
  - AlphaZero bat Stockfish (64 threads, 1Gb de hash): 25-72-3

AlphaZero

- Les résultats contre Stockfish sont « discutables »
- AlphaZero est indiscutablement plus fort, mais cela ne signifie pas la fin des algorithmes minimax car:
  - Stockfish est privé de ses bibliothèques d'ouverture et de finale
  - Stockfish n'utilise que 1 Gb de table de Hash
  - Stockfish utilise 64 CPU:
    - On est très loin de la puissance des 4 TPUv2 de Google
    - Stockfish est un programme développé par des « amateurs » qui ne disposent pas de machines massivement parallèles. Le programme n'est pas optimisé pour le parallélisme massif (la recherche sur la parallélisation des algorithmes de type alpha-béta est peu active).

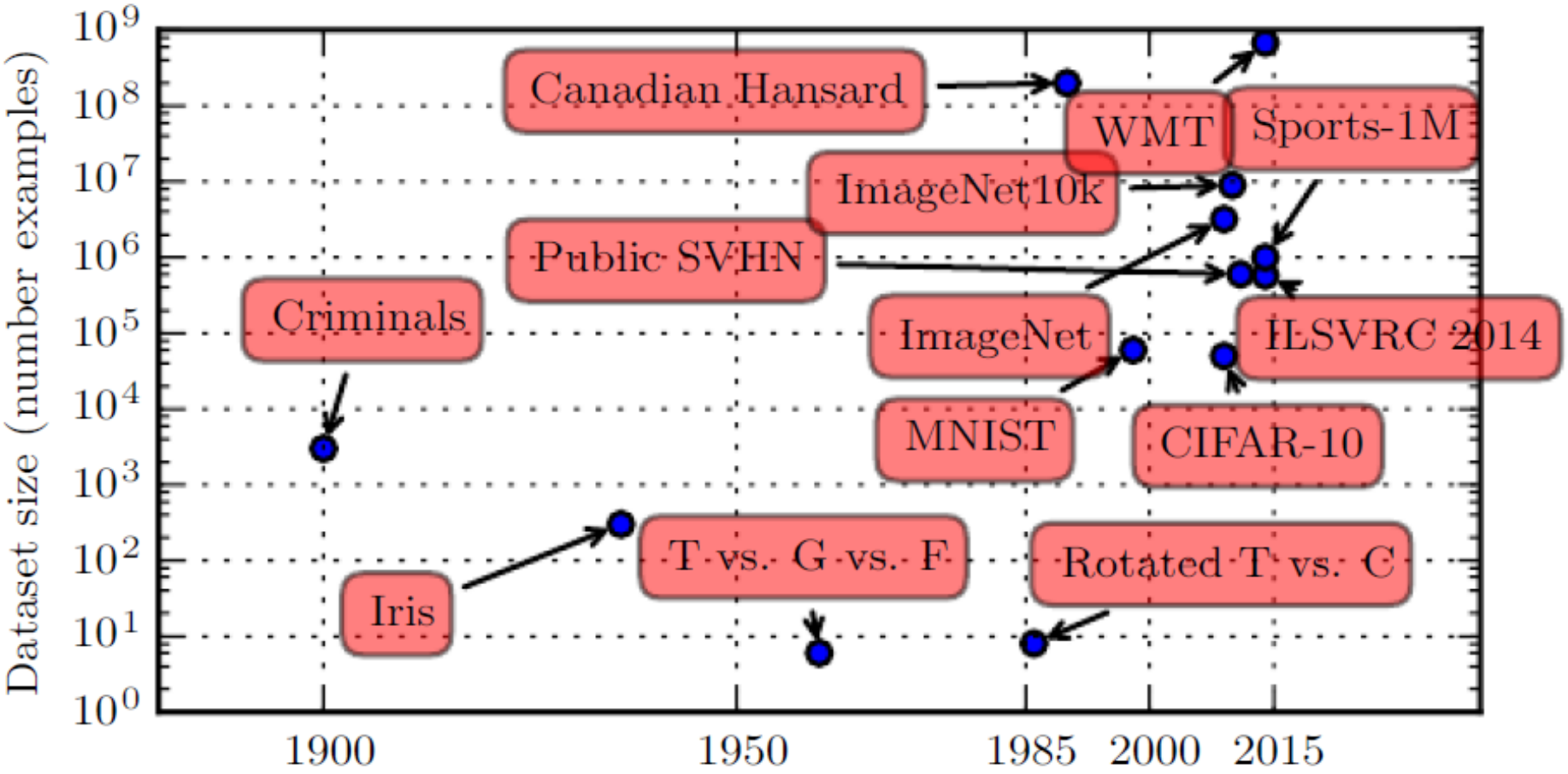
Quelques faits: DNN, CPU, GPU et TPU

- L'explosion des DNN est largement liée à l'arrivée de processeurs SIMD (Single Instruction Multiple Data) massifs
- La progression de vitesse des CPU a considérablement ralenti depuis plusieurs années, alors que la puissance des GPU continue d'augmenter régulièrement :
  - Riva TNT (1998) : 180 Mop/s
  - GTX 1080 (2017) : 8 Tflop/s
- Google a développé pour ses DNN un processeur particulier, le TPU (Tensor Processing Unit) qui est spécialisé dans la multiplication matricielle. La version 2 développe 180 Tflop/s et 64Gb de mémoire pour 200Watts de consommation (Google loue ses TPU en béta-test pour 6.5 dollars de l'heure à quelques chercheurs).
- Nvidia Tesla V100: TPU speed: 120 Tflop/s, 16Gb, 300W, 10000$

Quelques faits: le temps du « « Big Data » »

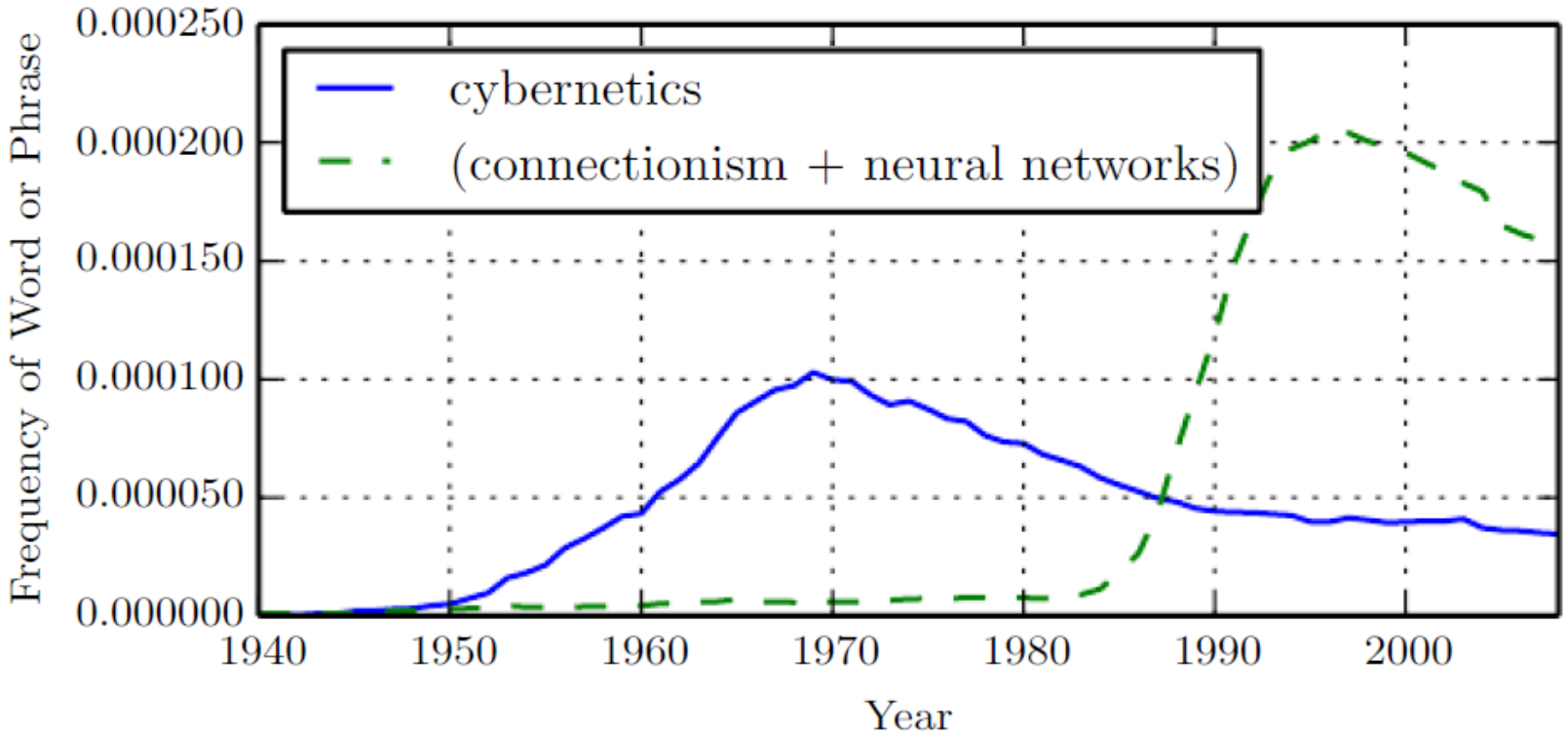
Bagio (2017): for supervised learning :

- Acceptable performance: 5000 examples
- Expert performance: 10 millions examples
- Best performance when examples can be generated at will (games)



N'oublions pas!

- Le Deep Learning et les Deep Neural Networks sont les anciens réseaux de neurones et les anciennes méthodes connexionnistes
- Attention aux effets de mode: l'IA en a suffisamment souffert!!



Some wisdom…

- *Rodney Brooks (1991): Critical paper about the then-mainstream line in AI (cognitive AI and expert systems):*
  - "There is a bandwagon effect in Artificial Intelligence Research, and many lines of research have become goals of pursuit in their own right, with little recall of the reasons for pursuing those lines"
- *Rodney Brooks: Machine Learning Explained (28 Aout 2017):*
  - *https://rodneybrooks.com/forai-machine-learning-explained/*
  - "Vast numbers of new recruits to AI/ML have jumped aboard after recent successes of Machine Learning, and are running with particular versions of it as fast as they can. They have neither any understanding of how their tiny little narrow technical field fits into a bigger picture of intelligent systems, nor do they care. They think that the current little hype niche is all that matters, are blind to its limitations, and are uninterested in deeper questions."
  - "The papers in conferences fall into two categories. One is mathematical results showing that yet another slight variation of a technique is optimal under some carefully constrained definition of optimality. A second type of paper takes a well known learning algorithm, and some new problem area, designs the mapping from the problem to a data representation, and show the results of how well that problem area can be learned."

Wisdom?

Le problème à résoudre est un problème de «hardware»: le cerveau humain, avec son incroyable complexité physiologique, ne peut pas se simuler sur les architectures limitées des calculateurs actuels. (1992)