Réseaux de neurones

Julien JACQUES

17/10/2018

Définition d'un neurone

Un neurone est un *modèle*, fonction à p variables, qui relie p *entrées* $x^1, \ldots, x^p$ à une *sortie* y :

$$y = g\left(\alpha_0 + \sum_{j=1}^{p} \alpha_j x^j\right)$$

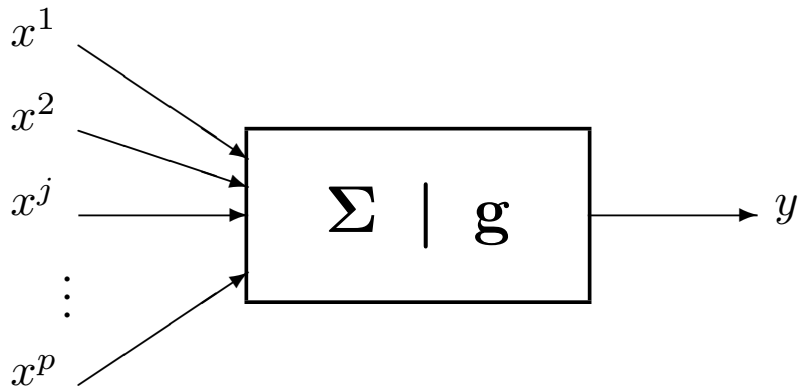


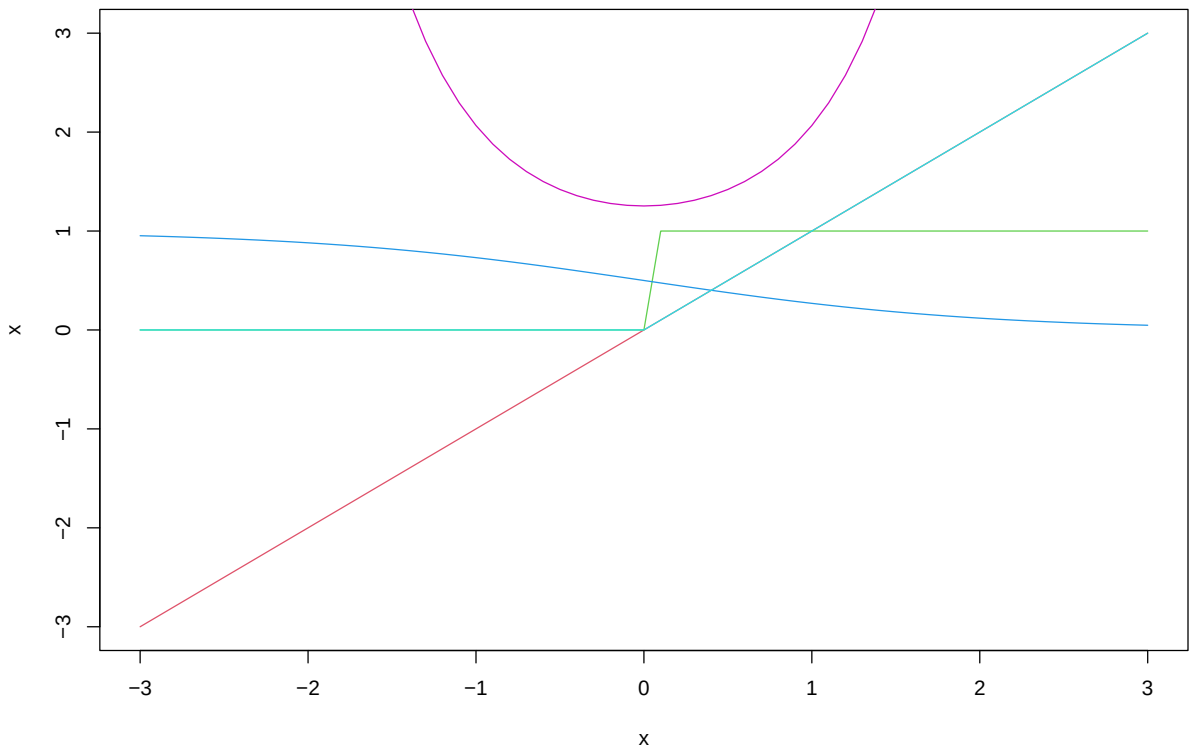
Figure 1: Représentation d'un neurone

- Σ : combinaison linéaire des entrées
- g : fonction d'activation

Fonction d'activation

La fonction d'activation g caractérise le neurone :

- $g(x) = x$: identité
- $g(x) = 1_{x>0}$: seuil
- $g(x) = 1/(1 + e^{-x})$: sigmoïde (logistique, softmax)
- $g(x) = max(0, x)$: ReLU
- $g(x) = \sqrt{\pi/2}\exp(-x^2/2)$: radiale
- ...



Cas particulier : modèle linéaire

Un unique neurone avec une fonction d'activation $g(x) = x$ est un *modèle linéaire* classique :

$$y = \alpha_0 + \sum_{j=1}^{p} \alpha_j x^j$$

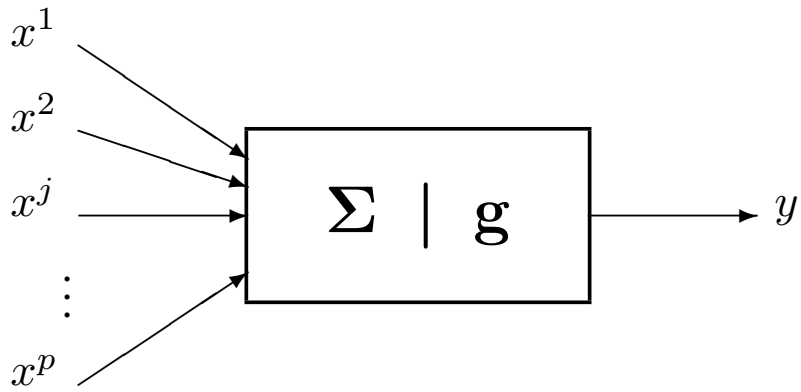


Figure 2: Représentation d'un neurone

Différents types de réseaux de neurones

Un réseaux de neurones est l'association de plusieurs neurones, en un graphe plus ou moins complexe, caractérisé par :

- son architecture (en couches...)
- sa complexité (nombre de neurones, présence de boucles)
- les fonctions d'activation
- l'objectif : apprentissage supervisé ou non...

Le perceptron multicouche

- Un perceptron multicouche est composé de **couches**
- couche : ensemble de neurones sans connexion entre eux
- Il dispose d'une couche d'entrée, une couche de sortie, et d'**une ou plusieurs couches cachées**
- Les neurones sont tous connectés en entrée à chacun des neurones de la couche précédente et en sortie à chacun des neurones de la couche suivante

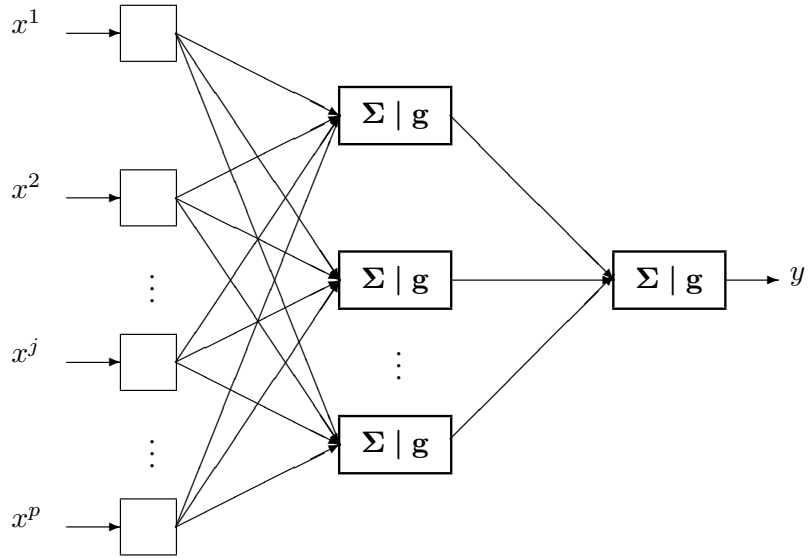


Figure 3: Perceptron multicouche avec une couche cachée

Choix des fonctions d'activation

Neurone(s) de sortie :

- en régression (y quantitative) :
  - couche de sortie $= 1$ neurone avec g identité
- en classification binaire ($y \in 0, 1$) :
  - couche de sortie $= 1$ neurone avec g sigmoïde
- en classification à m classes ($y \in c_1, \ldots, c_m$) :
  - couche de sortie $=$ m neurones avec g sigmoïde

Neurone(s) de la couche cachée :

- la fonction d'activation la plus généralement utilisée est la sigmoïde.

Apprentissage

Le perceptron multicouche est un modèle :

$$y = f(x^1, \ldots, x^p; \boldsymbol{\alpha})$$

où le paramètre $\boldsymbol{\alpha}$ regroupe les coefficients $\boldsymbol{\alpha}_{kj} = (\alpha_{kj}^0, \ldots, \alpha_{kj}^p)$ de chaque neurone $k \in 1, \ldots, q$ et pour chaque couche j.

A partir d'un échantillon d'apprentissage $(y_i, x_i^1, \ldots, x_i^p)_{1 \leq i \leq n}$, ces paramètres sont estimés en **minimisant la perte quadratique** :

$$\mathcal{Q}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \Big(y_i - f(x_i^1, \ldots, x_i^p; \boldsymbol{\alpha}) \Big)^2$$

Cette optimisation est faite numériquement...

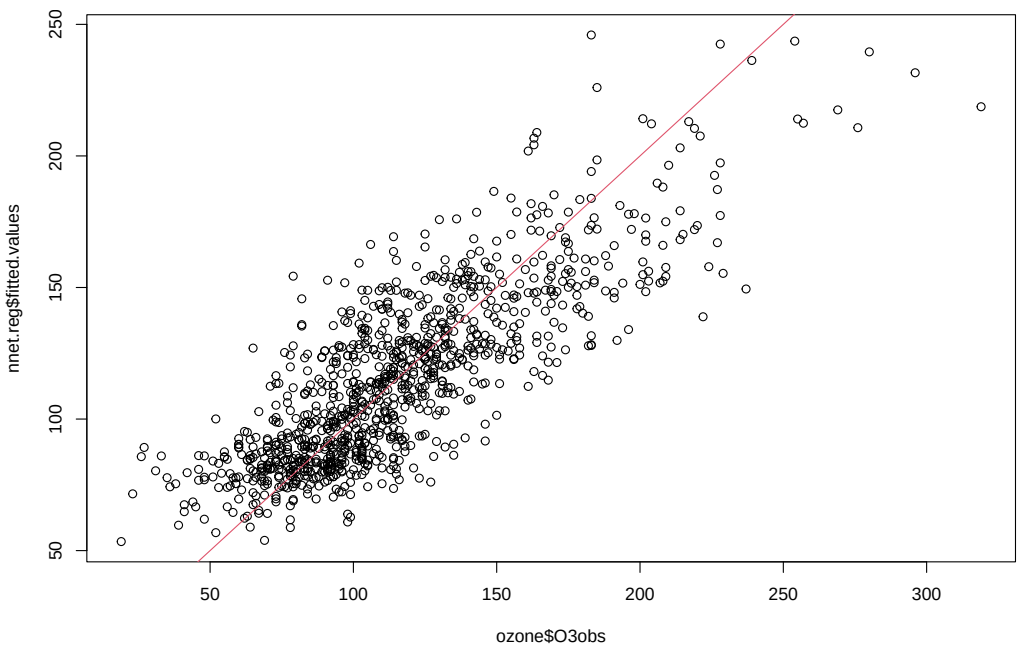
Exemple d'application en régression

On utilise le package **nnet**

```
library(nnet)
```

Chargeons les **data ozone** et essayons de **prévoir la concentration d'ozone** à l'aide d'un **réseau à 1 couche cachée et 5 neurones** :

```
ozone=read.table('ozone.dat',header=T)
nnet.reg=nnet(O3obs~.,data=ozone,size=5,decay=1,linout=TRUE,
              maxit=500,trace=F)
plot(ozone$O3obs,nnet.reg$fitted.values);abline(a=0,b=1,col=2)
```



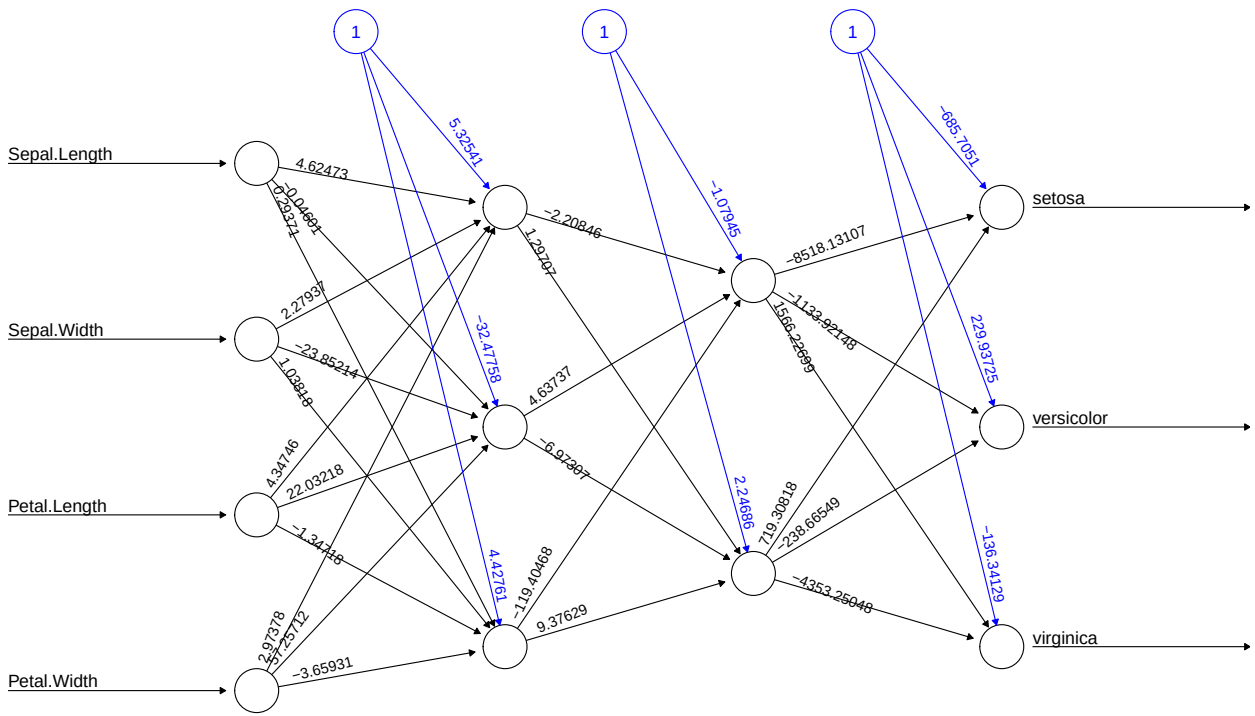
Exemple d'application en classification

On utilise le package **neuralnet**

```
library(neuralnet)
```

Perceptron à 2 couches (3 et 2 neuronnes) sur données iris :

```
nn <- neuralnet(Species~.,iris,linear.output=FALSE,
                hidden=c(3,2),act.fct='logistic')
plot(nn)
```



Deep Learning

Un **théorème d'approximation universelle** montre qu'un perceptron à une couche cachée (avec un nombre de neurones grand mais fini) est suffisant pour résoudre tous les problèmes classiques d'apprentissage.

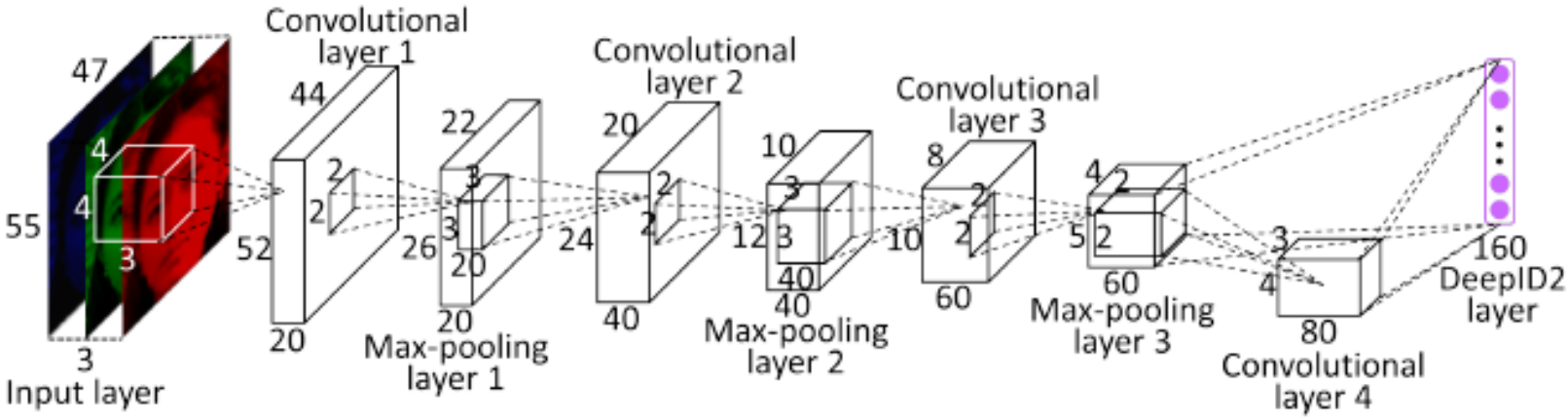
En pratique, le grand nombre de paramètres nécessaires induit des problèmes d'estimation et de stabilité. Ce résultat est donc essentiellement théorique.

Les **réseaux de neurones profonds** partent du principe qu'il est plus efficace de mutliplier le nombre de couches plutôt que le nombre de neurones.

Ainsi, plusieurs réseaux de neurones profonds ont été développés récemment en **empilant** un nombre généralement important de **couches de neurones aux propriétés spécifiques**.

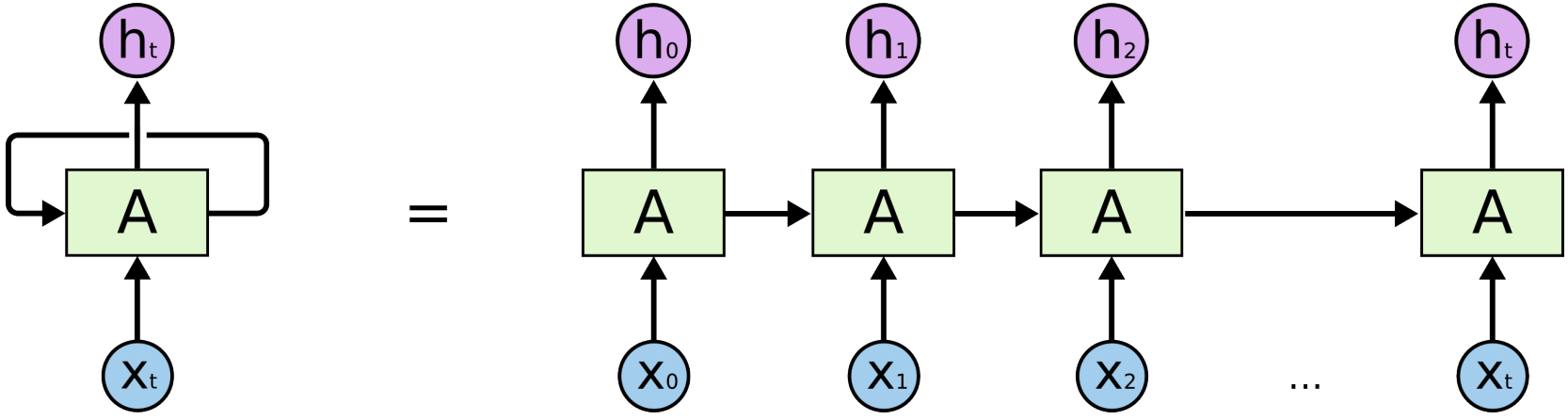
Trois principaux types de réseaux profonds (1/3)

- ConvNet : *convolutional neural networks*, pour l'analyse d'image



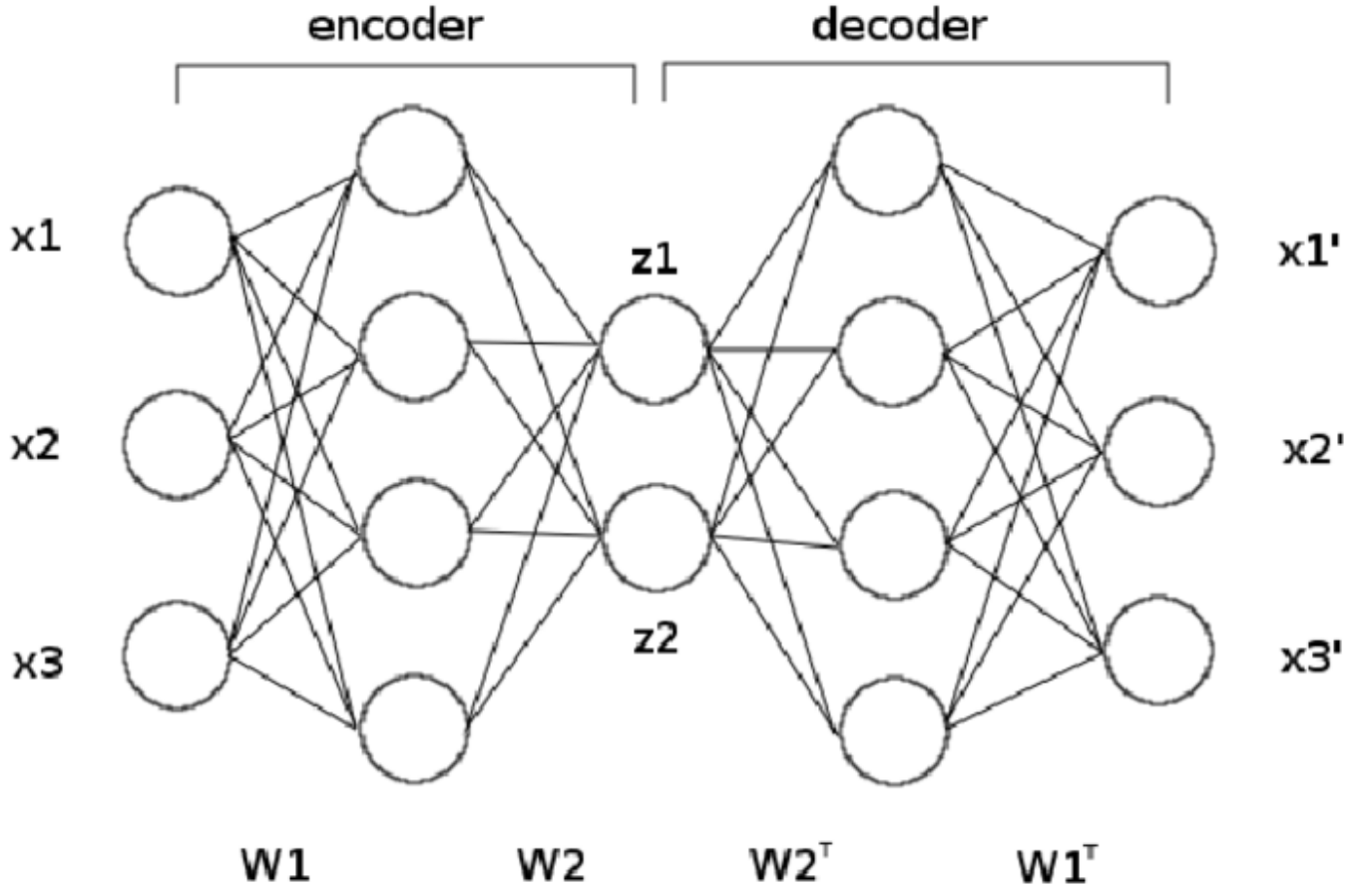
Trois principaux types de réseaux profonds (2/3)

- LSTM : *long-short term memory*, pour le traitement du signal ou analyse du langage naturel



Trois principaux types de réseaux profonds (3/3)

- *autoEncoder decoder* (diabolo), pour l'apprentissage non supervisé (débruitage d'images ou signaux, détection d'anomalies...)



Révolution du Deep Learning

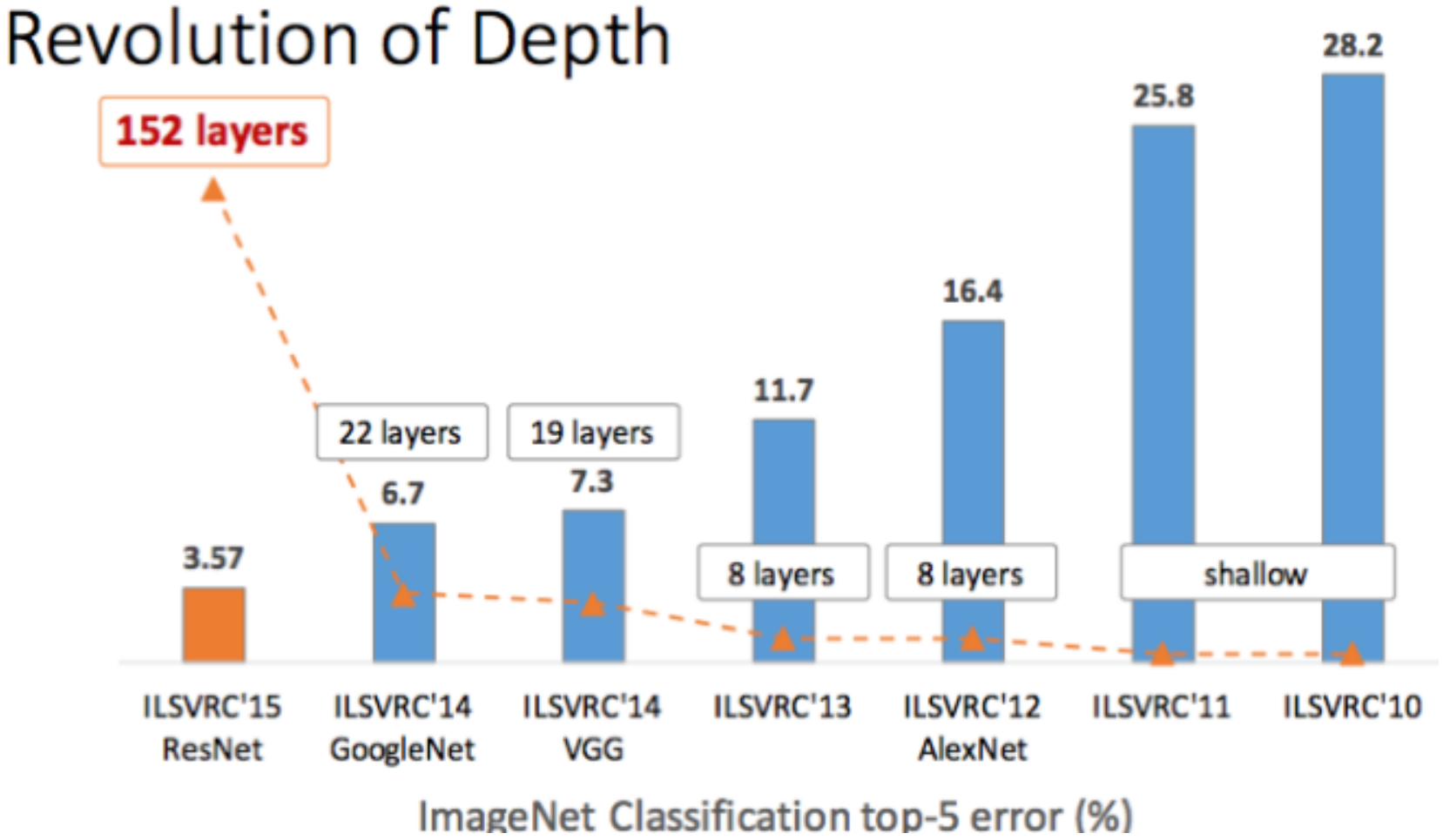


Figure 4: La révolution du deep learning

Révolution du Deep Learning

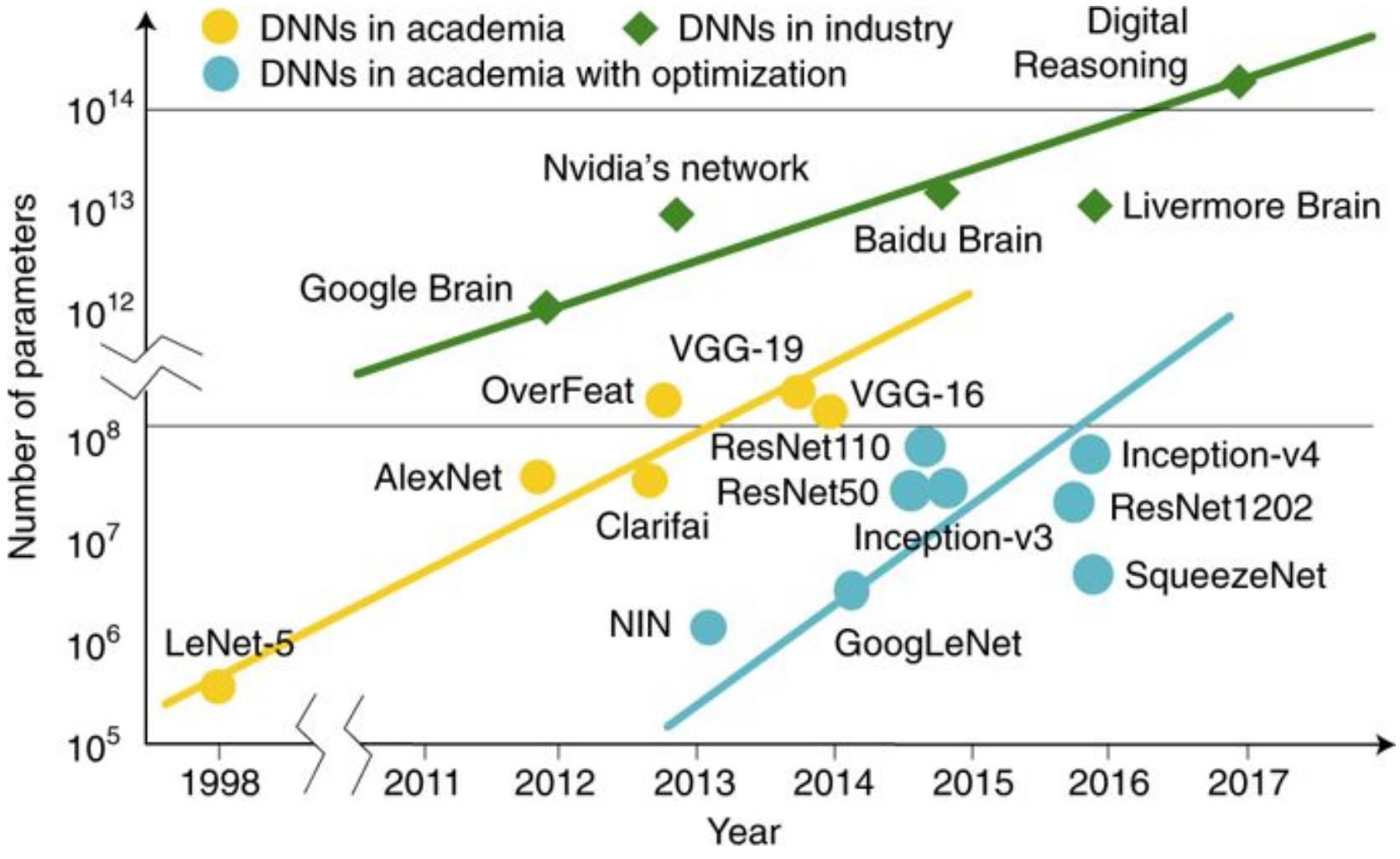


Figure 5: La révolution du deep learning

Deep Learning in practice

- la complexité de ces réseaux entraine l'introduction de **millions de paramètres à estimer**
- il faut donc pour cela un échantillon d'apprentissage gigantesque (des **milliards** de données) et une **puissance de calcul** très importante.
- une utilisation dans un cadre plus conventionel est la suivante :
  - identifier un réseau déjà appris sur des données similaires
  - supprimer la dernière couche du modèle dédiée à la classification
  - apprendre les poids de cette seule dernière couche sur nos données spécifiques