# Deep Learning pour la classification de morceaux de musique

Romain Hennequin

ENSEA, 2026

Deezer Research

# Outline

# Introduction

# Music Information Retrieval (MIR)

Quelques tâches :

- Détection de tempo/pulsation
- Détection de tonalité
- Détection de hauteur (multiple)
- Détection d'accords
- Identification de l'artiste.
- Reconnaissance des instruments
- Classification en genre
- Classification en ambiance/émotion (moods)
- Détection de reprise
- ...

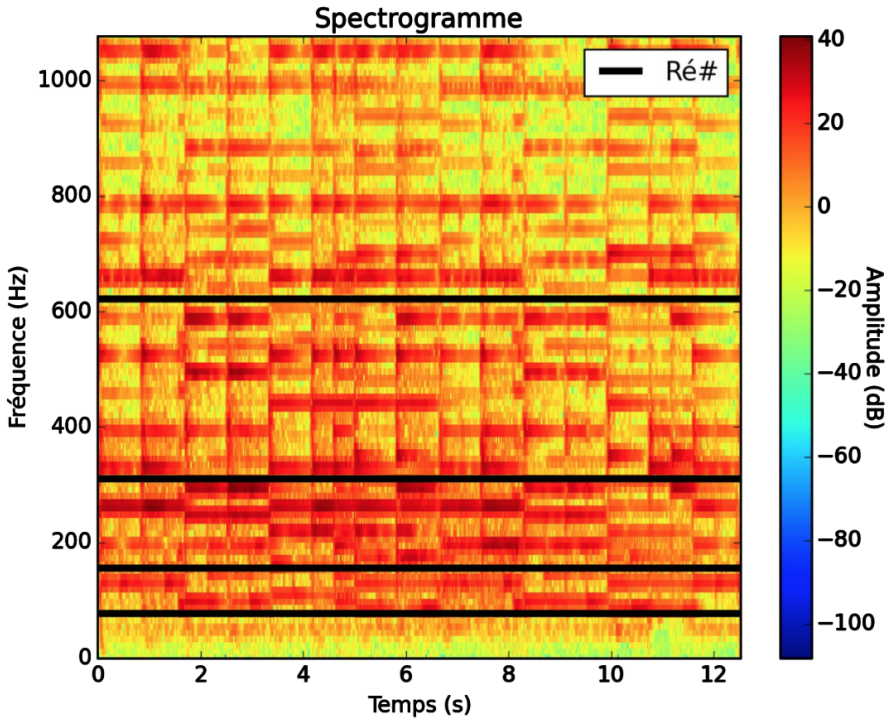
# MIR "classique"

Façon "*classique*" d'aborder le MIR avant l'arrivée du deep learning :

- Définir "à la main" des descripteurs adaptés à la tâche.
- Utiliser un classifieur "classique" (SVM, Forêt aléatoire, KNN, GMM...).

# Détection d'accords : descripteurs

- Accord = présence de plusieurs classes de notes simultanées.
- Comment détecter la présence d'une classe de notes ?



# Détection d'accords : descripteurs

Descripteur : chromagramme.

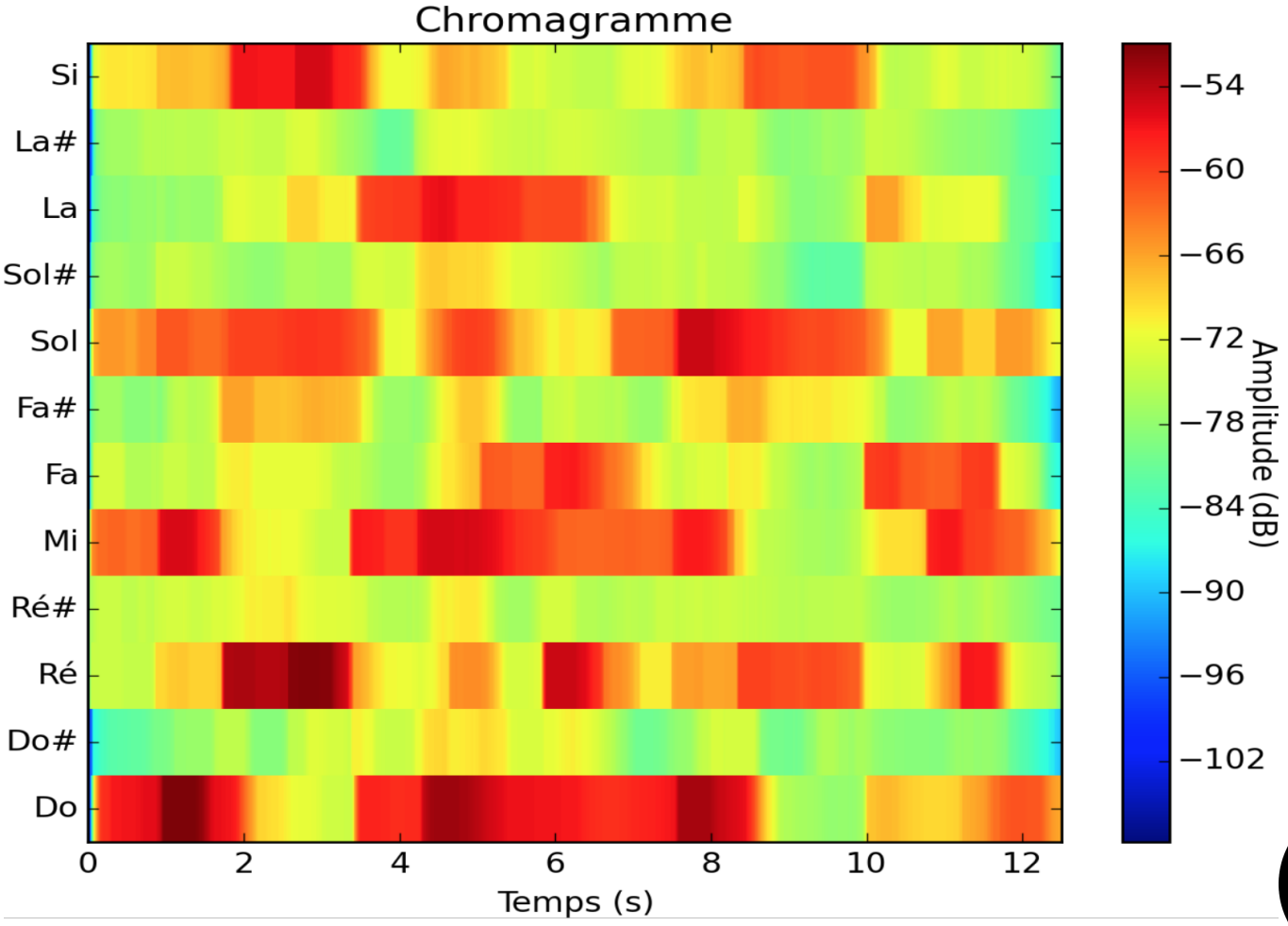
$$C(\mathcal{N}, t) = \sum_{k=0}^{K-1} |S(2^k f_0(\mathcal{N}), t)|$$

où :

- $\mathcal{N} \in \{do, do\#, re, re\#, mi, fa, fa\#, sol, sol\#, la, la\#, si\}$
- $f_0(\mathcal{N})$ est la fréquence fondamentale la plus basse considérée pour la classe de note $\mathcal{N}$
- $S(f, t)$ est une TFCT du signal considéré à la fréquence $f$ et à l'instant $t$.

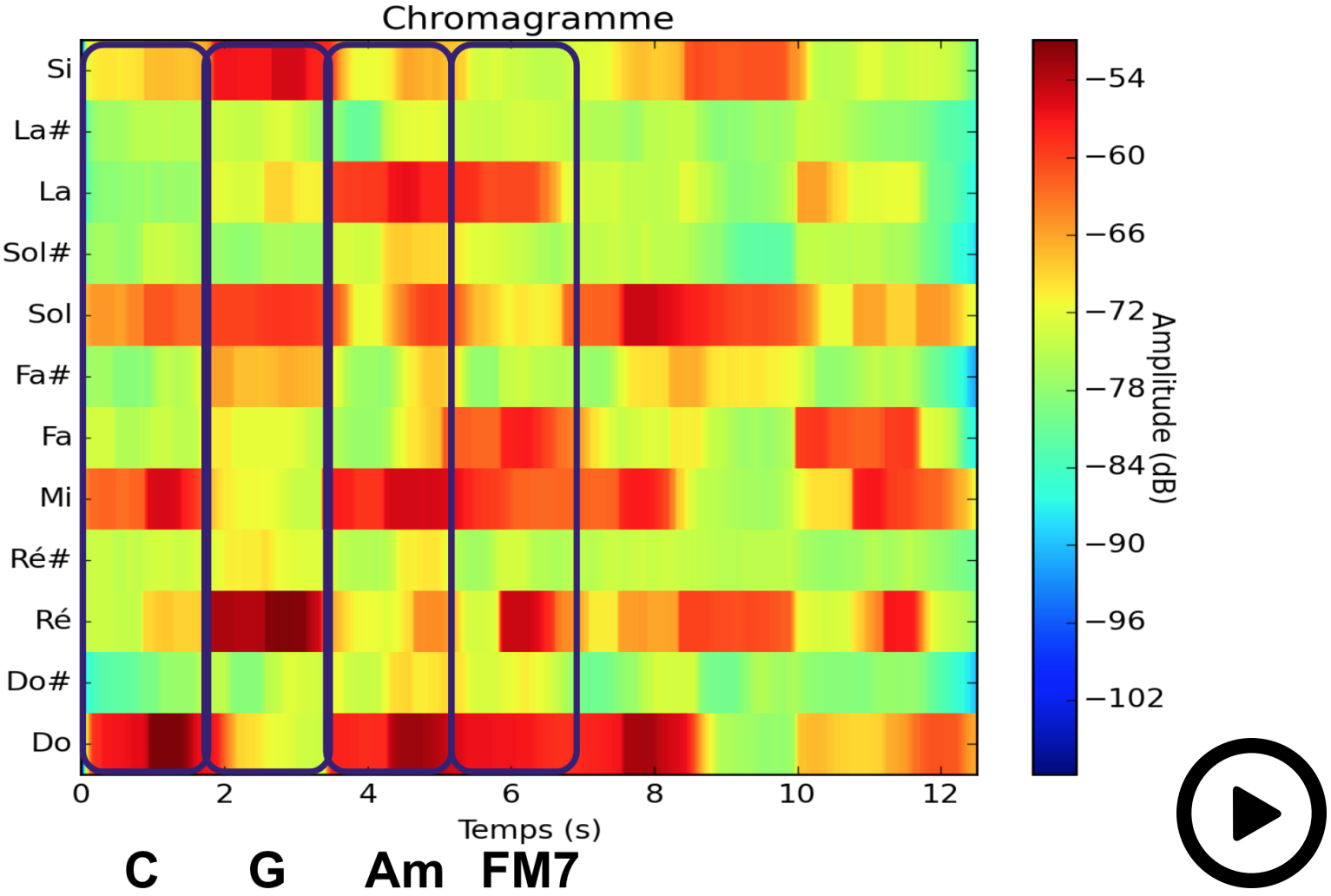
# Détection d'accords : descripteurs

Descripteur chromagramme :



# Détection d'accords : descripteurs

Descripteur chromagramme :



# Détection d'accords : classifieur

"classifieur" temporel :

Modèles de Markov cachés (HMM) pour une détection par trame à partir des chromas.

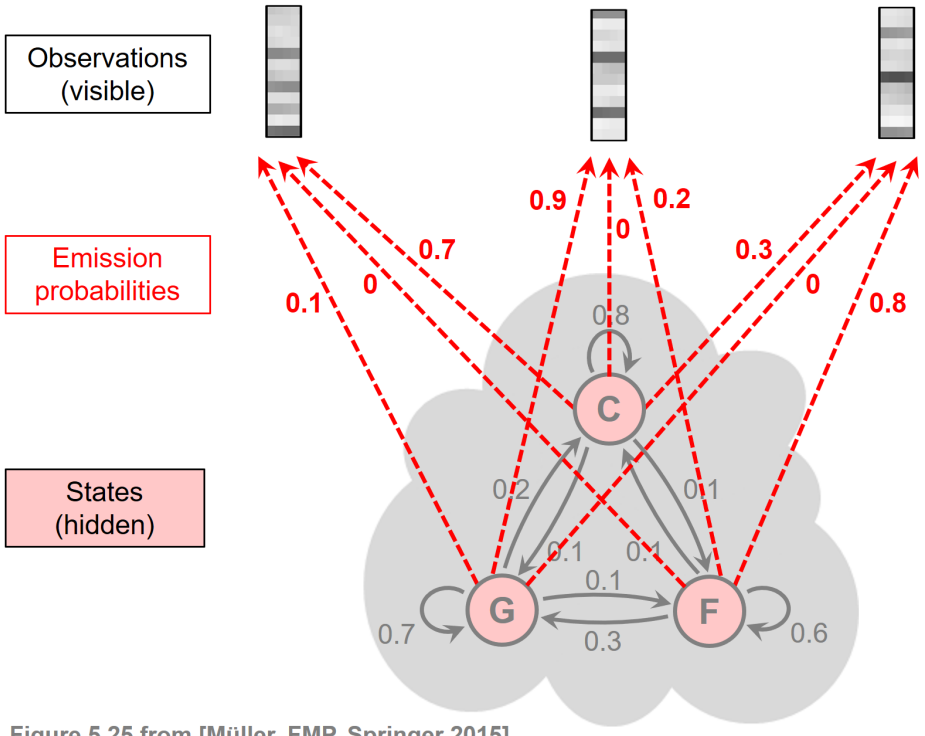


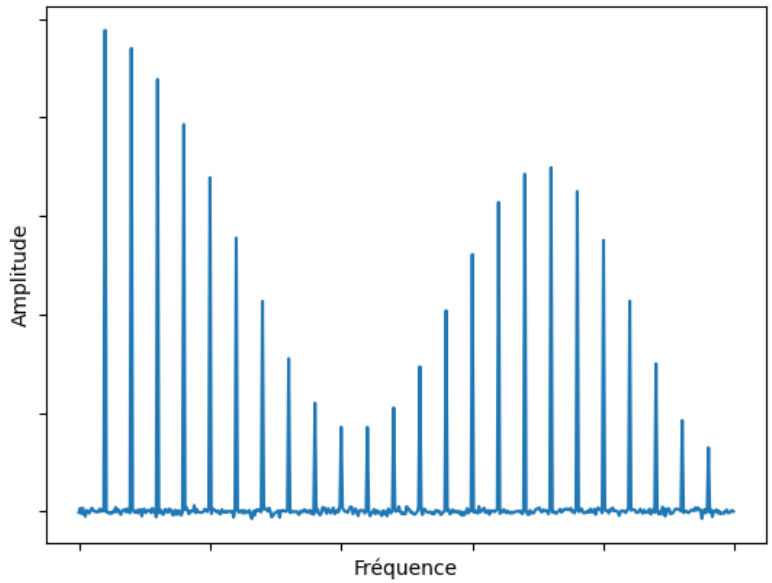

Figure 5.25 from [Müller, FMP, Springer 2015]

# Détection d'instruments : descripteurs

- Instrument principalement défini par son timbre.
- Une caractéristique du timbre est l'enveloppe spectrale et ses variations.

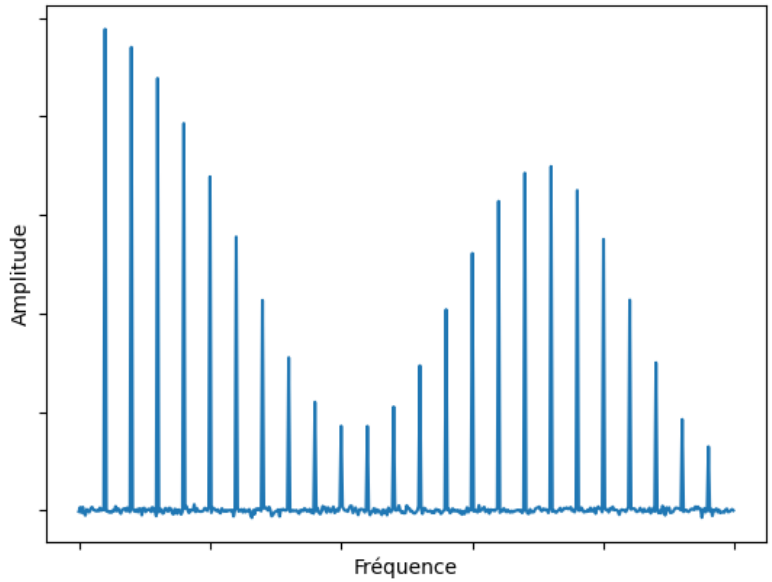
# Détection d'instruments : descripteurs

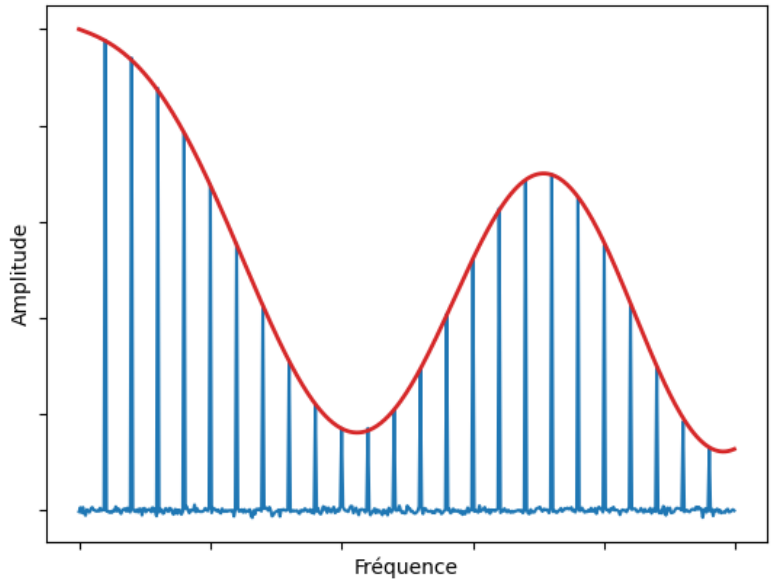
- Instrument principalement défini par son timbre.
- Une caractéristique du timbre est l'enveloppe spectrale et ses variations.



# Détection d'instruments : descripteurs

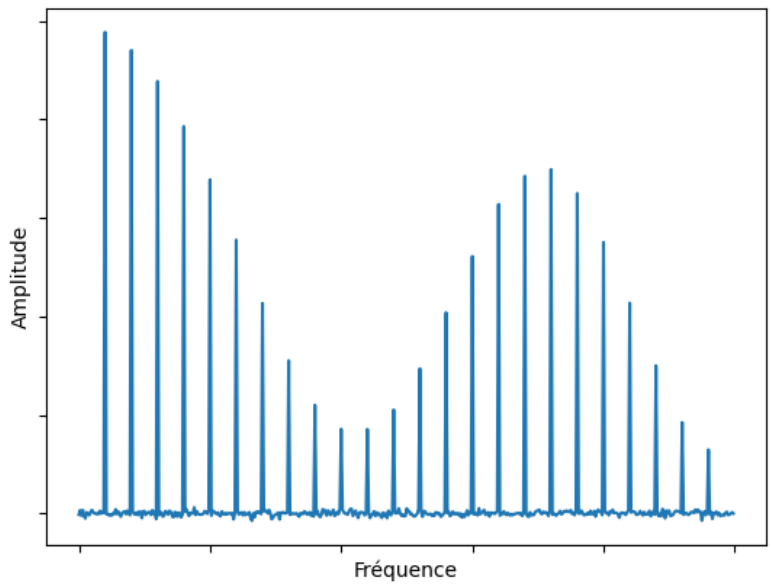
- Instrument principalement défini par son timbre.
- Une caractéristique du timbre est l'enveloppe spectrale et ses variations.

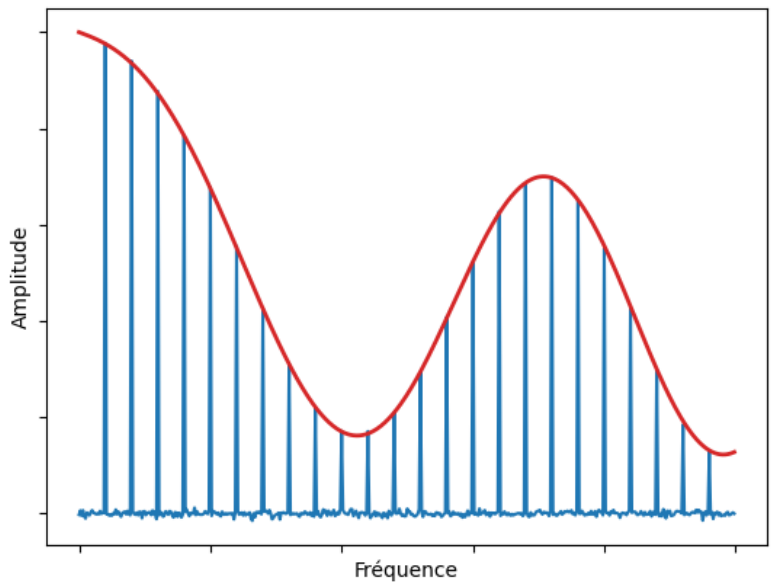


# Détection d'instruments : descripteurs

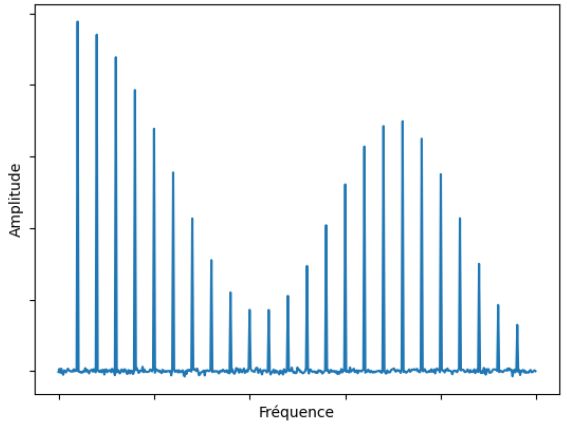
- Instrument principalement défini par son timbre.
- Une caractéristique du timbre est l'enveloppe spectrale et ses variations.



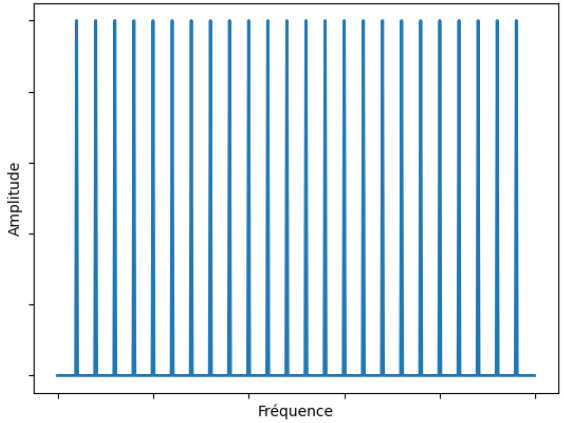




$\Longrightarrow$ Mel-Frequency Cepstral Coefficients (MFCC) + $\Delta$MFCC

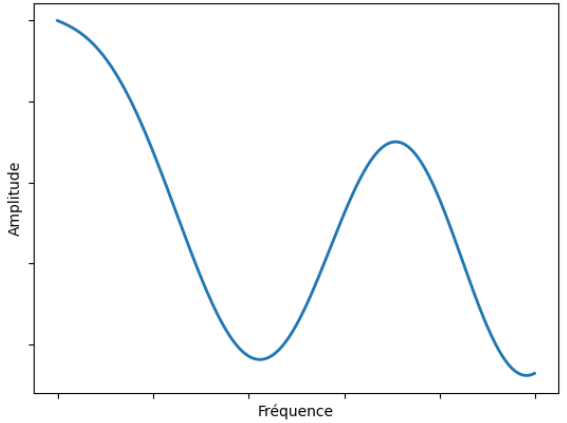
# Détection d'instruments : descripteurs




=




×



# Détection d'instruments : descripteurs

$$S = H \times E$$

où :

- $S$ : spectre de la note
- $H$ : peigne harmonique
- $E$ : enveloppe spectrale

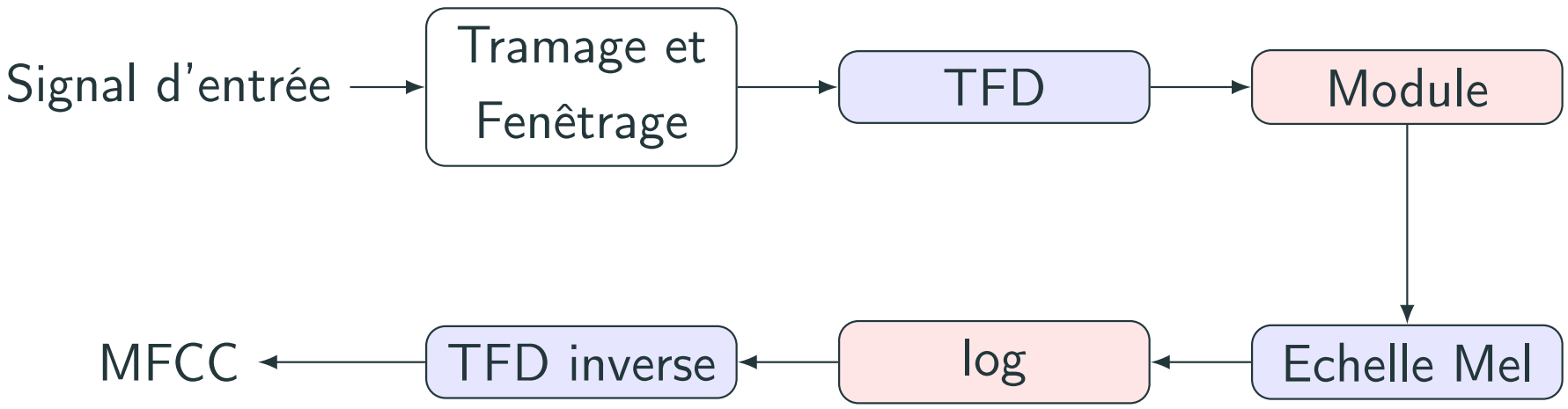
Donc :

$$\log(S) = \log(H) + \log(E)$$

$$\mathcal{TF}\left(log(S)\right) = \underbrace{\mathcal{TF}\left(log(H)\right)}_{\text{hautes fréquences}} + \underbrace{\mathcal{TF}\left(log(E)\right)}_{\text{basses fréquences}}$$

# Détection d'instruments : descripteurs

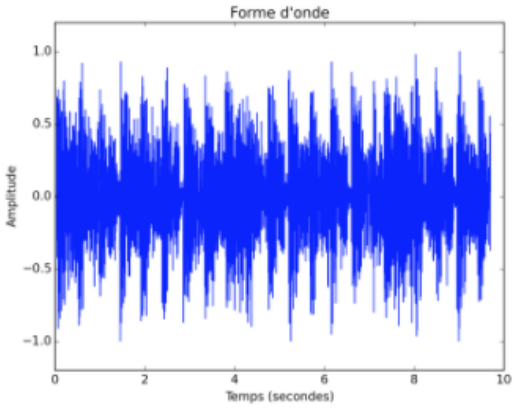
MFCC :



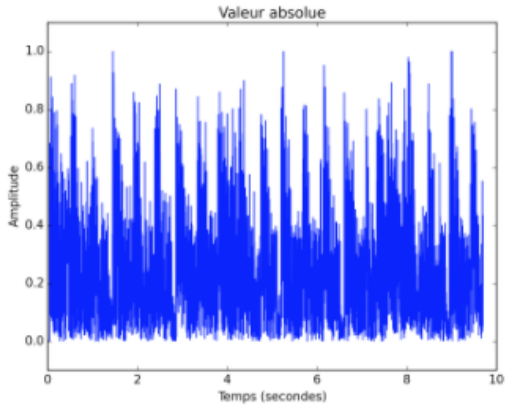
# Détection d'instruments : classifieur

- Machines à vecteur de support (SVM)
- Modèle de mélange gaussien (GMM)
- K plus proches voisins (KNN)
- Réseau de neurones
- ...

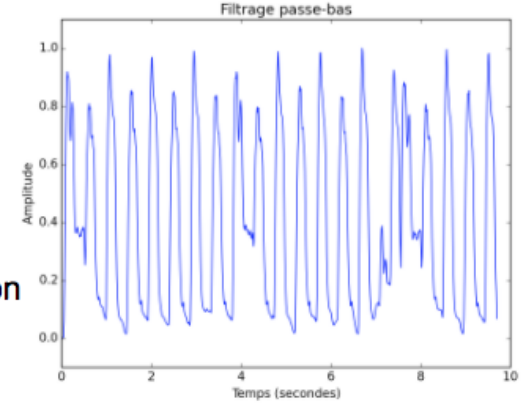
# Détection de pulsation : descripteurs



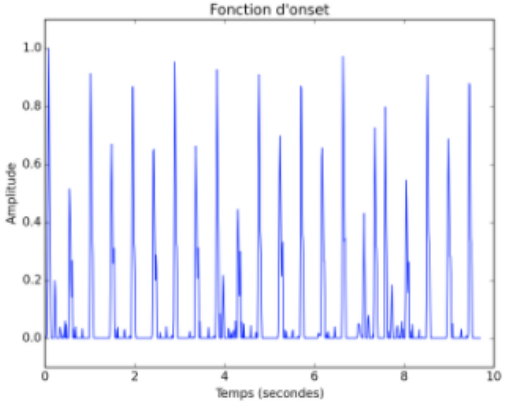

Valeur absolue




Filtrage passe-bas



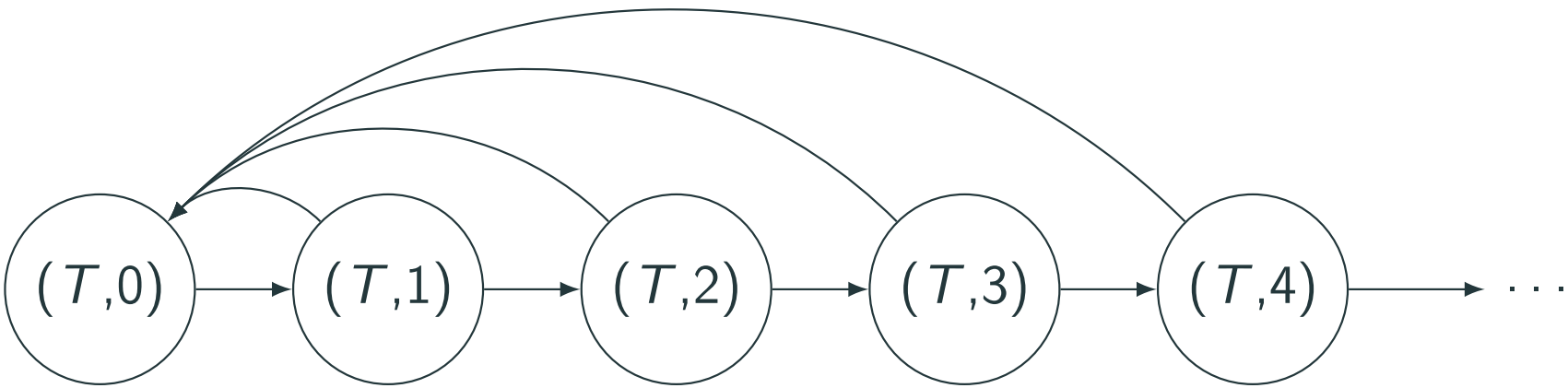

Dérivée + rectification demi-onde



# Détection de pulsation : classifieur

Lissage par modèles de Markov cachés (HMM).

Etat du HMM : couple (Tempo, temps depuis la dernière pulsation).



# Problèmes

Limitations de l'approche *classique* :

- Besoin de travail humain pour définir les descripteurs.
- Les descripteurs ne sont généralement pas optimaux.
- Les deux étapes sont séparées (pas d'optimisation jointe des paramètres des descripteurs et des paramètres du classifieur).

# Problèmes

Questions :

- Est-il possible de s'affranchir d'un design manuel des descripteurs ?
- Est-il possible d'inclure directement le calcul de descripteurs optimaux dans la phase d'apprentissage ?
- Est-il possible d'apprendre simultanément ces descripteurs et les paramètres du classifieur ?

# Réseaux de neurones artificiels

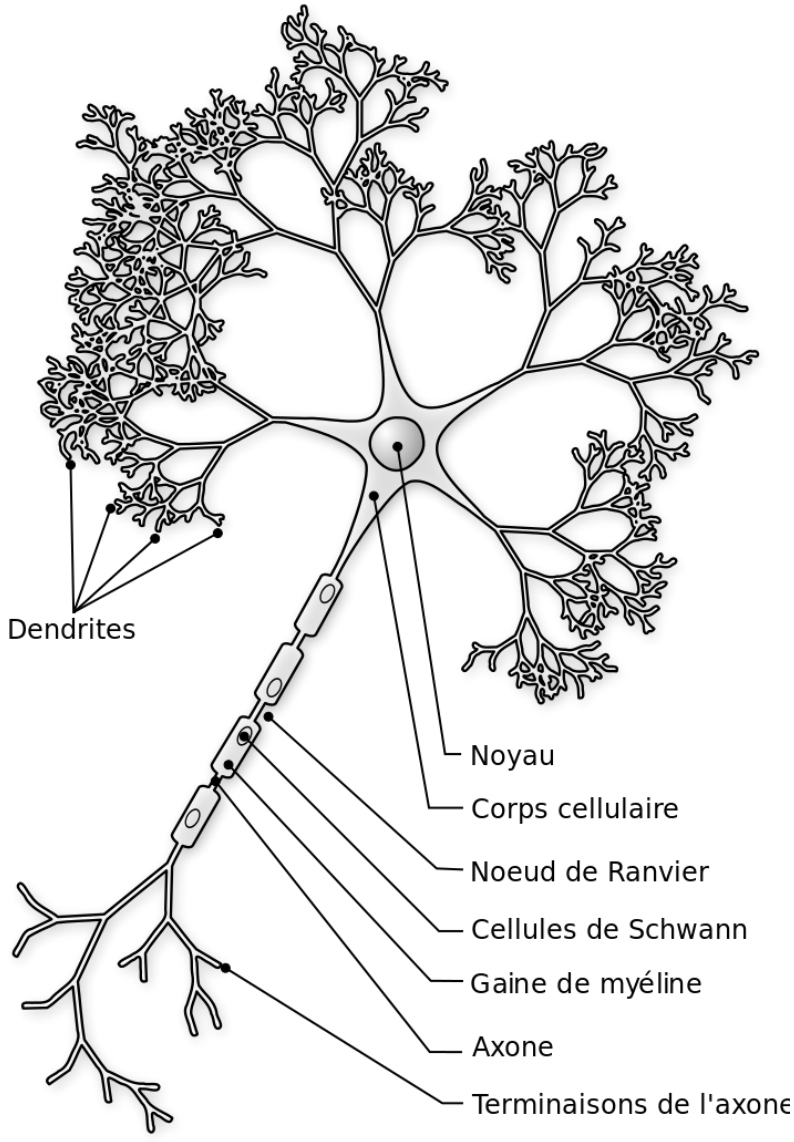
# Description

Ensemble de modèles d'apprentissage très vaguement inspirés du fonctionnement du cerveau humain.

- Dans le cerveau, l'information est traitée par un réseau complexe de neurones inter-connectés
- Les neurones des différentes régions du cerveau sont spécialisés dans des traitements spécifiques

# Neurone biologique

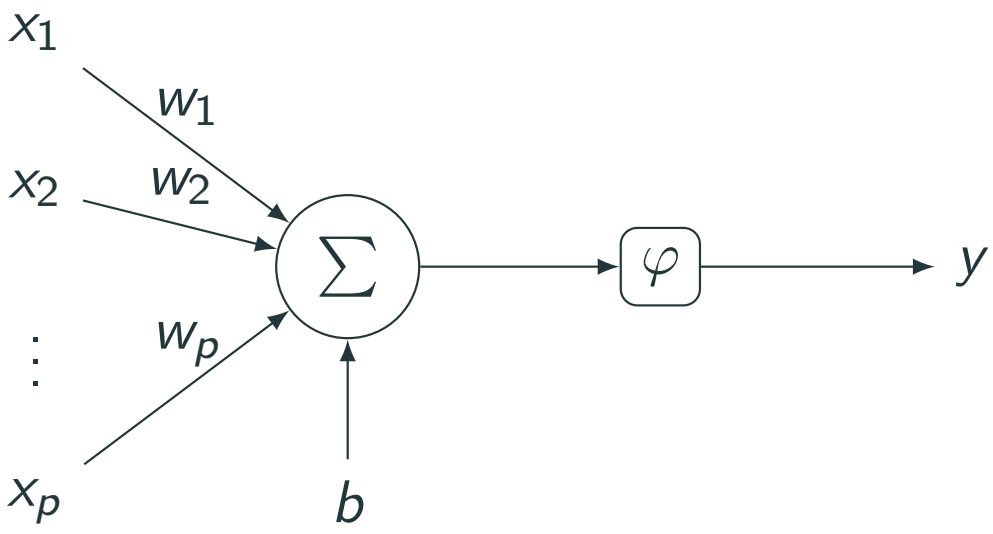
- Sommation des potentiels gradués des synapses le long des dendrites au niveau du cône d'émergence.
- Génération d'un potentiel d'action se transmettant le long de l'axone.
- Potentiel transmis aux dendrites d'autres neurones via des synapses au niveau des terminaisons de l'axone.
- La morphologie et le nombre des dendrites peuvent varier.




Source : Wikipedia

# Neurone artificiel

Modèle de neurone artificiel : $y = \varphi(\sum_{i=1}^{p} w_i x_i + b)$




$x_i$ : entrées du neurone.

$w_i$ : poids (variables) associés à chaque entrée

$b$ : biais ($-b$ : seuil d'activation)

$\varphi$ : fonction d'activation

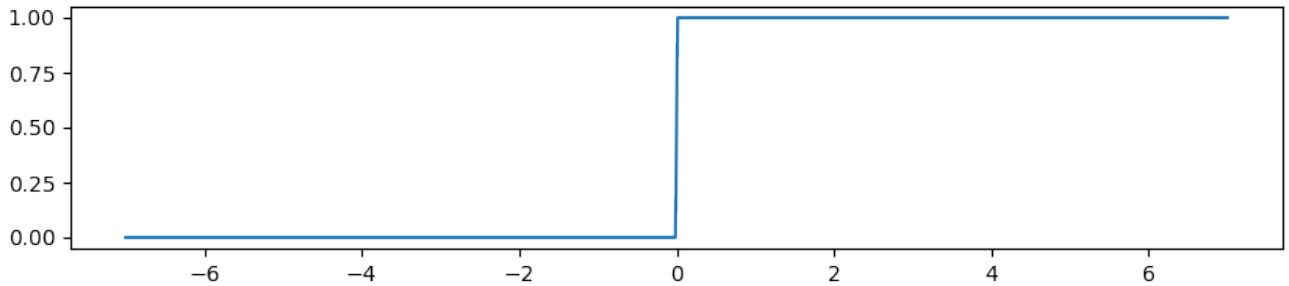
$y$ : sortie du neurone

# Fonction d'activation

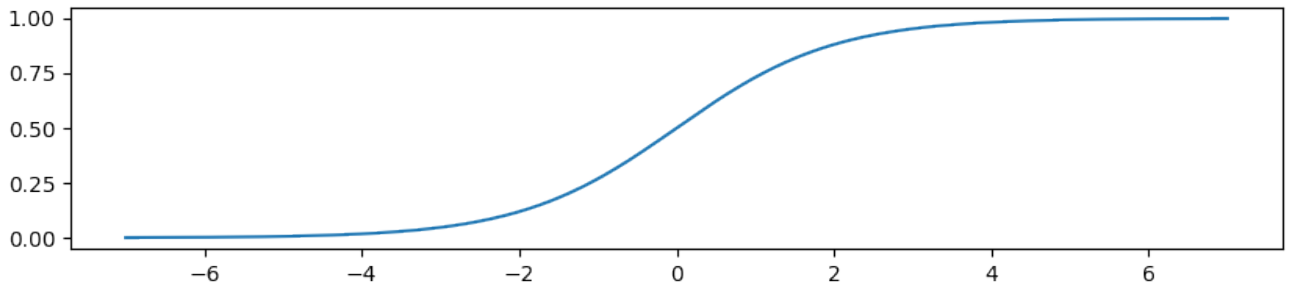
$\varphi$ : fonction d'activation.

Fonctions d'activation courantes :

- Heaviside $\forall x \in \mathbb{R},\ H(x) = \begin{cases} 0 & \text{si} \quad x < 0 \\ 1 & \text{si} \quad x \geq 0. \end{cases}$



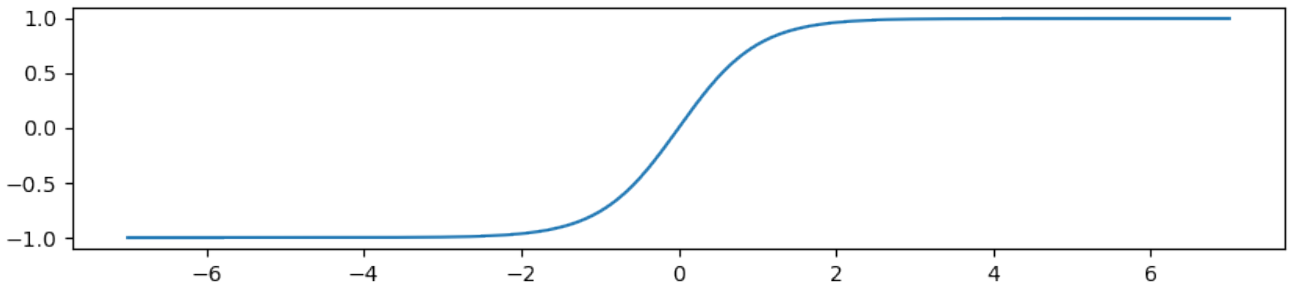

- fonction sigmoïde : $\sigma(x) = \frac{1}{1+e^{-x}}$



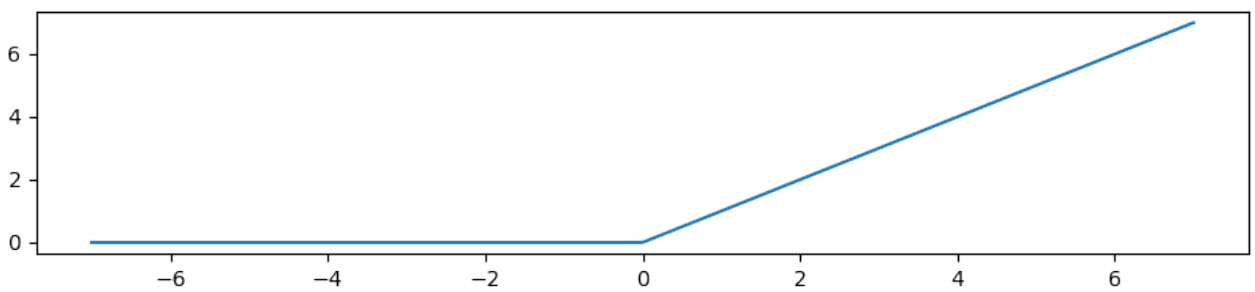
# Fonction d'activation

Fonctions d'activation courantes :

- *tanh*

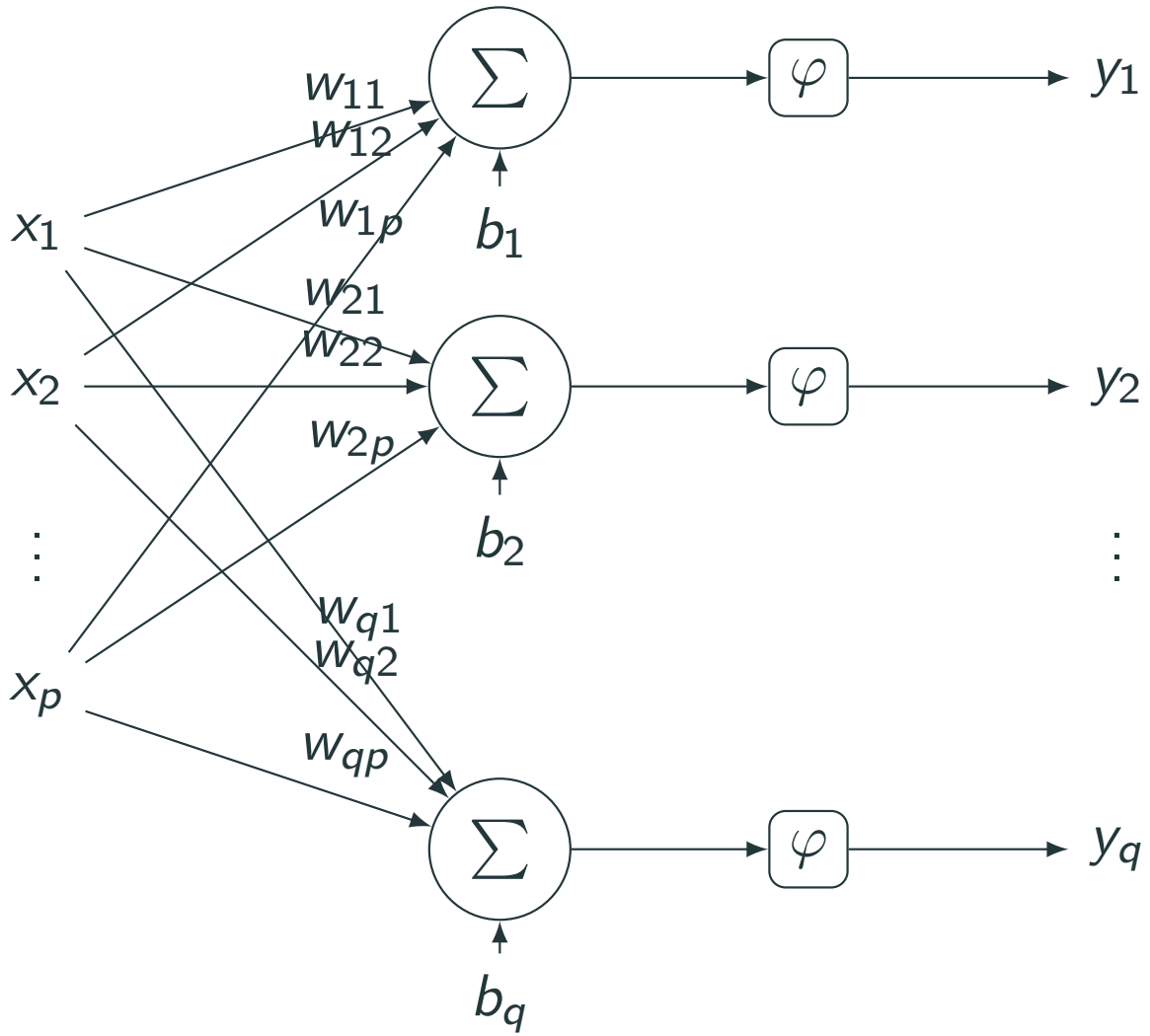



- Rectified Linear Unit (ReLU) : $ReLU(x) = \begin{cases} x \text{ si } x > 0 \\ 0 \text{ sinon} \end{cases}$



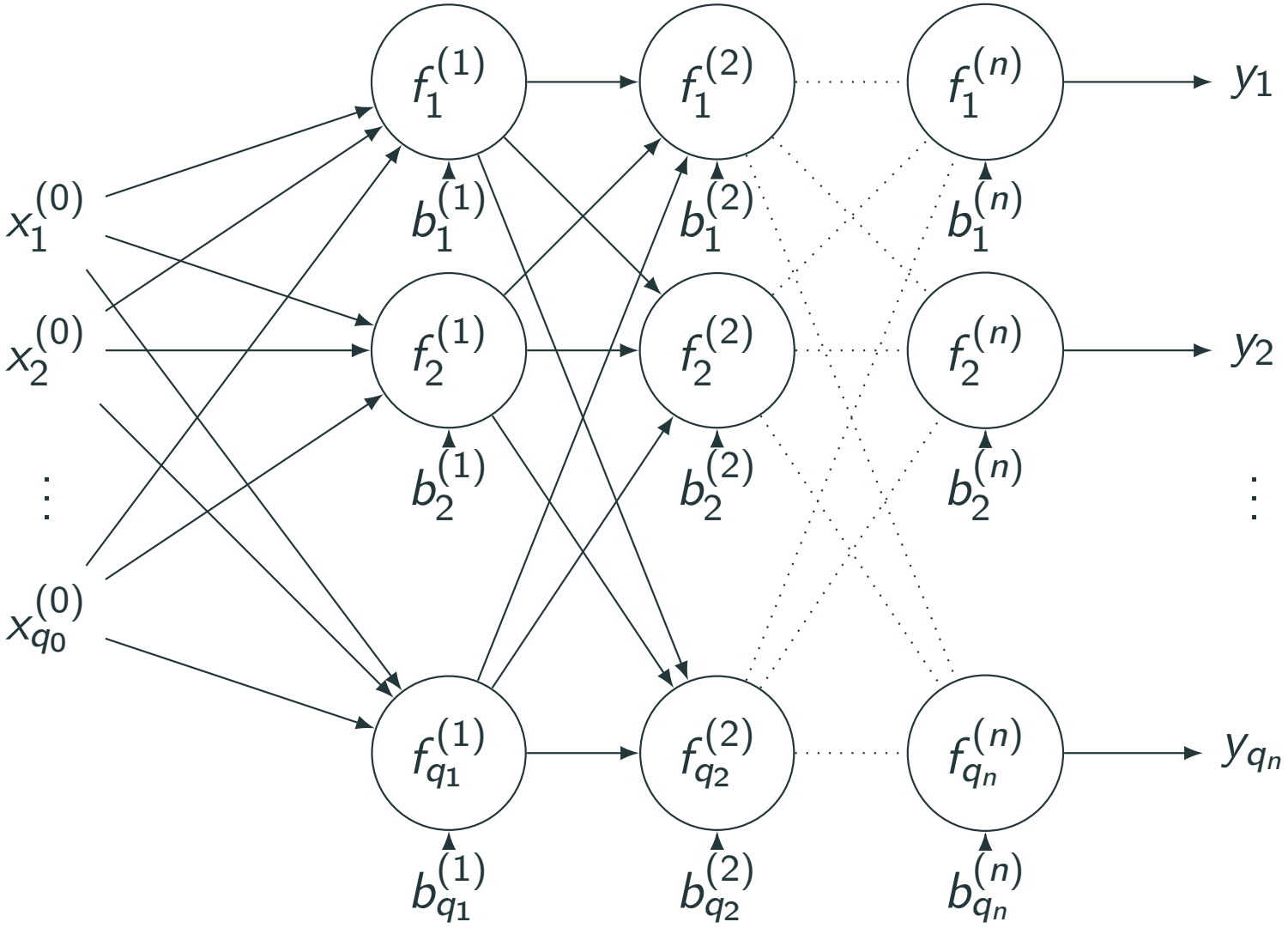
# Couche de neurones

Couche de neurones : $y_j = \varphi(\sum_{i=1}^{p} w_{ji} x_i + b_j)$, $\mathbf{y} = \varphi(\mathbf{W}\mathbf{x} + \mathbf{b})$.



# Description

Réseau de neurones : empilement de couches.



# Description

Réseau de neurones : empilement de couches.

- Entrée : $\mathbf{x}^{(0)}$
- Couches de neurones :
  $\mathbf{x}^{(k+1)} = f^{(k)}(\mathbf{x}^{(k)}) = \varphi_k(\mathbf{W}^{(k)}\mathbf{x}^{(k)} + \mathbf{b}^{(k)})$ avec
  $\mathbf{W}^{(k)} \in \mathbb{R}^{q_{k+1} \times q_k}$ et $\mathbf{b}^{(k)} \in \mathbb{R}^{q_{k+1}}$
- $\mathbf{y} = \mathbf{x}^{(k)} = f^{(n)}(\mathbf{x}^{(k-1)}) = f^{(n)} \circ f^{(n-1)} \circ \ldots \circ f^{(1)}(\mathbf{x}^{(0)})$

Remarque : les biais $\mathbf{b}^{(k)}$ peuvent être intégrés dans les $\mathbf{W}^{(k)}$ en ajoutant une dimension aux vecteurs d'entrée de la couche toujours égale à 1 : $x_{q_k+1} = 1$.

$$\mathbf{W}^{(k)}\mathbf{x}^{(k)} + \mathbf{b}^{(k)} = \mathbf{W_b}^{(k)}\mathbf{x_b}^{(k)}$$

# Théorème d'approximation universelle

- Un réseau de neurones à une seule couche cachée peut approximer toute fonction continue sur des compactes de $\mathbb{R}^n$ avec des hypothèses faibles sur la fonction d'activation (fonction croissante, continue, bornée et non constante).
- Un réseau de neurones simple peut représenter une large famille de fonctions (avec les paramètres appropriés).

# Théorème d'approximation universelle

Cependant :

- Ne donne pas de résultats sur le nombre de paramètres nécessaires.
- Ne présage pas de la possibilité d'estimer ces paramètres.

En pratique, une bonne approximation des fonctions est généralement obtenue plus facilement avec une architecture profonde (plusieurs couches cachées) qu'avec une architecture large (nombreux neurones dans une couche).

# Optimisation

On dispose d'une base de données d'apprentissage annotée :
$\mathcal{B} = \{(\mathbf{x}_1, \mathbf{t}_1), (\mathbf{x}_2, \mathbf{t}_2), \ldots, (\mathbf{x}_N, \mathbf{t}_N)\}$

Fonction de coût :

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{n=1}^{N} d(\mathbf{t}_n | \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_n))$$

$d$ caractérise le coût entre la cible $\mathbf{t}_n$ (vérité terrain) et la sortie du réseau $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_n)$ pour une entrée donnée $\mathbf{x}_n$.

$\boldsymbol{\theta}$ représente l'ensemble des paramètres du réseau $\mathbf{W}^{(\mathbf{k})}$, $\mathbf{b}^{(\mathbf{k})}$ pour $k \in \{1, \ldots, n\}$

On cherche $\boldsymbol{\theta}_b$ qui minimise $\mathcal{L}$ :

$$\boldsymbol{\theta}_b = \text{argmin}_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$$

# Optimisation

Fonctions de coût classiques :

- Erreur quadratique moyenne : $d(\mathbf{t}, \mathbf{y}) = \|\mathbf{t} - \mathbf{y}\|_2^2$
- Similarité cosinus : $d(\mathbf{t}, \mathbf{y}) = 1 - s(\mathbf{t}, \mathbf{y}) = 1 - \frac{<\mathbf{t}, \mathbf{y}>}{\|\mathbf{t}\|_2 \|\mathbf{y}\|_2}$
- Hinge loss : $d(\mathbf{t}, \mathbf{y}) = \sum_i \max(0, 1 - t_i y_i)$
- Entropie croisée : $d(\mathbf{t}, \mathbf{y}) = \sum_i t_i \log(y_i) + (1 - t_i) \log(1 - y_i)$
- ...

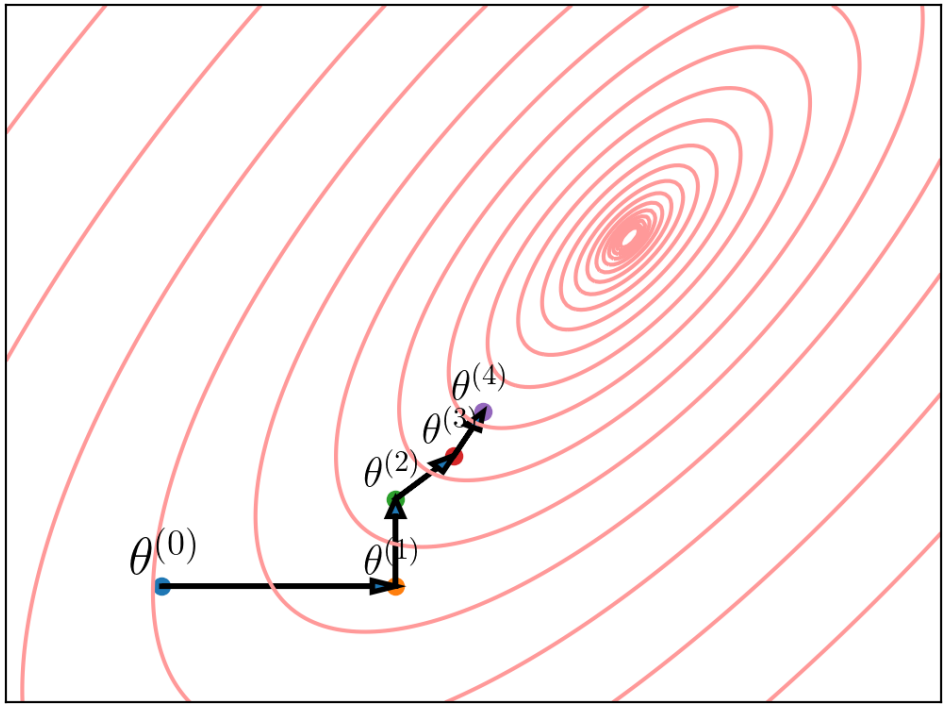
# Optimisation

Comment estimer $\boldsymbol{\theta}_b$ ?

Descente de gradient :

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla \mathcal{L}(\boldsymbol{\theta})$$

$\eta$ est le taux d'apprentissage.



# Optimisation

En pratique, $N$ grand ($\sim 10^6$).

$\mathcal{L}(\boldsymbol{\theta}) = \sum_{n=1}^{N} d(\mathbf{t}_n | \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_n))$ a beaucoup de termes !

$\Longrightarrow$ le calcul du gradient ne peut se faire sur toute la base d'apprentissage

Alternative : descente de gradient stochastique. Le gradient est approximé par le gradient d'un seul terme de la fonction de coût correspondant à un seul exemple d'entraînement :

$$\mathcal{L}_n(\boldsymbol{\theta}) = d(\mathbf{t}_n | \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x_n}))$$

Mise à jour de $\boldsymbol{\theta}$ :

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla \mathcal{L}_n(\boldsymbol{\theta})$$

Bonnes propriétés de convergence vers un minimum local.

# Optimisation

Variantes :

- Descente de gradient mini-batch : au lieu de calculer le gradient sur un seul exemple d'entraînement, on le calcule sur un petit nombre d'exemples. Accélère les calculs (parallèlisation).
- Gradient avec moment, l'estimation du gradient est une moyenne pondéré de l'estimation courante et de l'estimation précédente. Permet d'accélerer la convergence. Permet éventuellement de sortir de minima locaux. Animation.
- Taux d'apprentissage adaptatif : ADAM, Adadelta, Adagrad, RMSprop...

Exemple

# Optimisation

La descente de gradient et ses variantes ne permet pas d'estimer le minimum global de $\mathcal{L}$ :

On obtient des minima locaux qui dans de nombreux cas peuvent fournir des résultats satisfaisants.

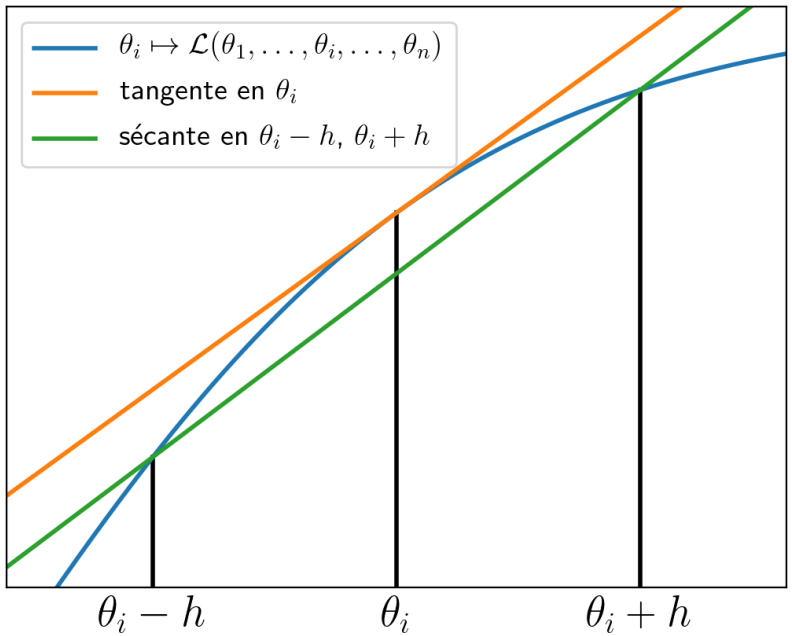
# Optimisation

Calcul du gradient :

- Différences finies
- Algorithme de rétropropagation du gradient

# Différences finies

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_i} = \frac{\mathcal{L}(\theta_1, ..., \theta_i + h, ..., \theta_n) - \mathcal{L}(\theta_1, ..., \theta_i - h, ..., \theta_n)}{2h} + O(h^2)$$




Fournie une approximation du gradient. Besoin de fixer $h$. En pratique, souvent utilisé uniquement pour vérifier une implémentation d'une autre méthode.

# Rétropropagation du gradient

Calcul exact : rétropropagation du gradient.

Fonction de coût : $\mathcal{L} = \frac{1}{2}\sum_i (t_i - y_i)^2$

Entrée du réseau : $x_i^{(0)}$

Sortie de la couche $k$ : $x_i^{(k)} = \varphi^{(k)}(h_i^{(k)}) = \varphi^{(k)}(\sum_j w_{ij}^{(k)} x_j^{(k-1)})$

Sortie du réseau : $y_i = x_i^{(n)}$

On s'intéresse à $\frac{\partial \mathcal{L}}{\partial w_{ij}^{(k)}}$. On a :

$$\frac{\partial \mathcal{L}}{\partial w_{ij}^{(k)}} = \frac{\partial \mathcal{L}}{\partial x_i^{(k)}} \frac{\partial x_i^{(k)}}{\partial h_i^{(k)}} \frac{\partial h_i^{(k)}}{\partial w_{ij}^{(k)}} = \frac{\partial \mathcal{L}}{\partial x_i^{(k)}} (\varphi^{(k)})'(h_i^{(k)}) x_j^{(k-1)}$$

# Rétropropagation du gradient

Il faut donc calculer $\frac{\partial \mathcal{L}}{\partial x_i^{(k)}}$.

Pour $k = n$ :

$$\frac{\partial \mathcal{L}}{\partial x_i^{(n)}} = y_i - t_i$$

Pour $k < n$ dérivée totale :

$$\frac{\partial \mathcal{L}}{\partial x_i^{(k)}} = \sum_l \frac{\partial \mathcal{L}}{\partial x_l^{(k+1)}} \frac{\partial x_l^{(k+1)}}{\partial x_i^{(k)}}$$

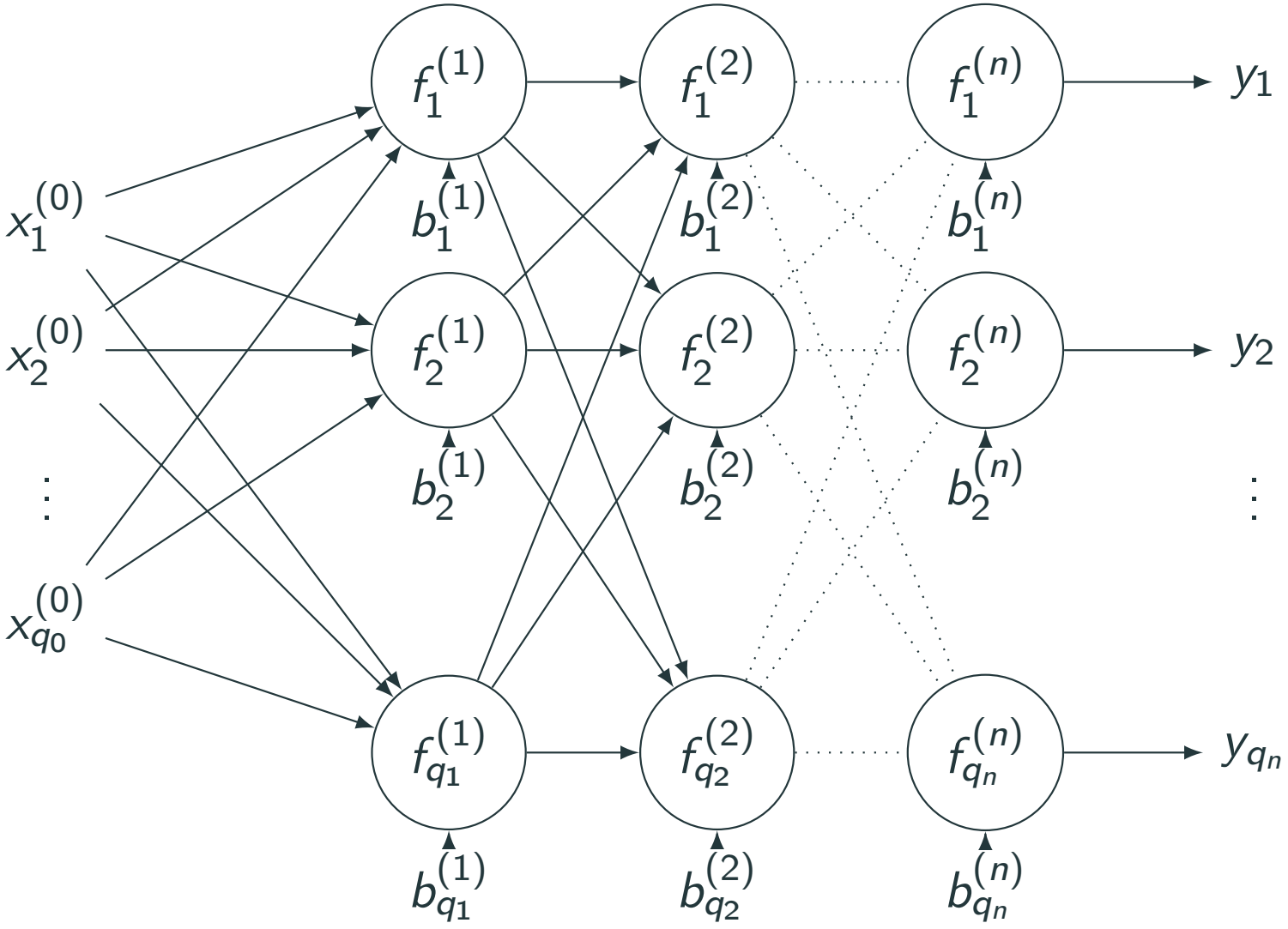
avec :

$$\frac{\partial x_l^{(k+1)}}{\partial x_i^{(k)}} = (\varphi^{(k+1)})'(h_i^{(k+1)}) w_{li}^{(k+1)} \equiv \delta_{il}^{(k+1)}$$

On a donc une relation de récurrence qui lie $\frac{\partial \mathcal{L}}{\partial x_i^{(k)}}$ aux $\frac{\partial \mathcal{L}}{\partial x_l^{(k+1)}}$ : on peut rétropropager le gradient du haut (la sortie) vers le bas (l'entrée) du réseau.

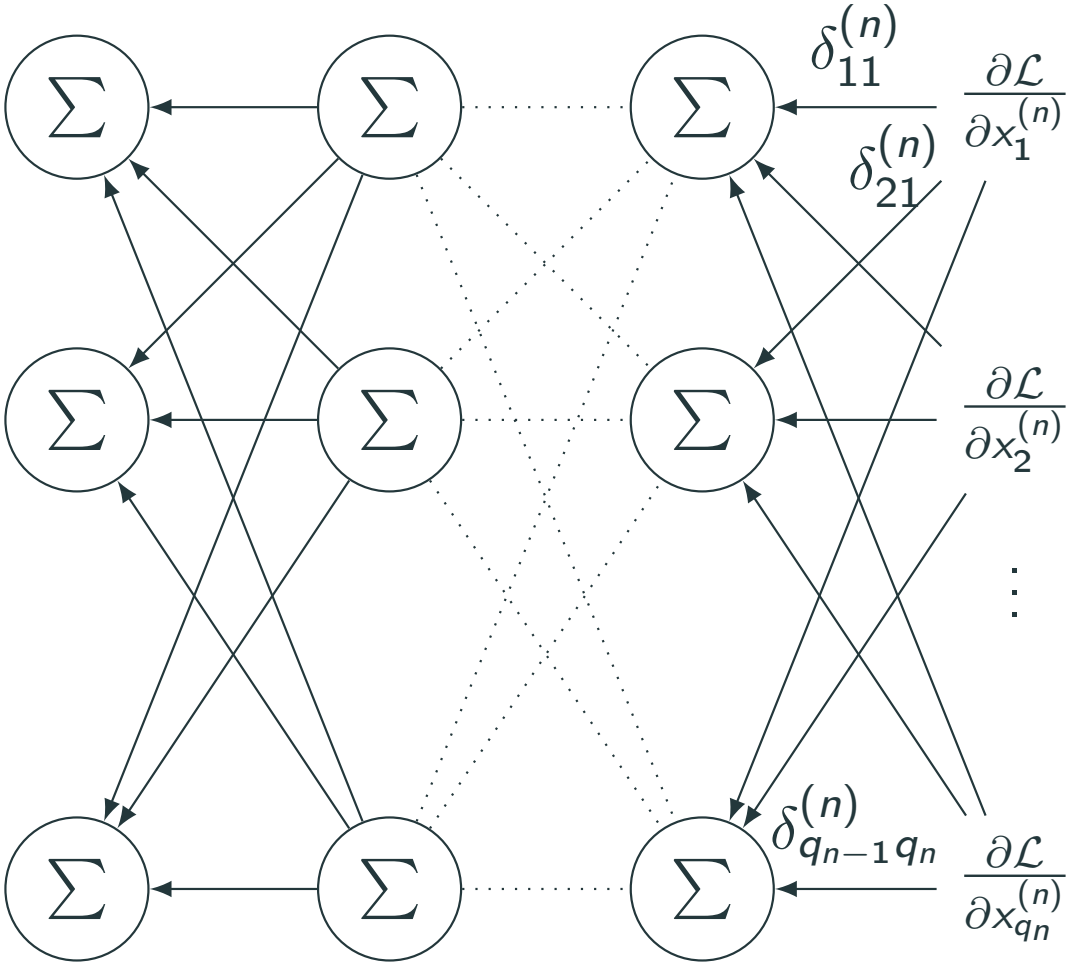
# Rétropropagation du gradient

Propagation avant :



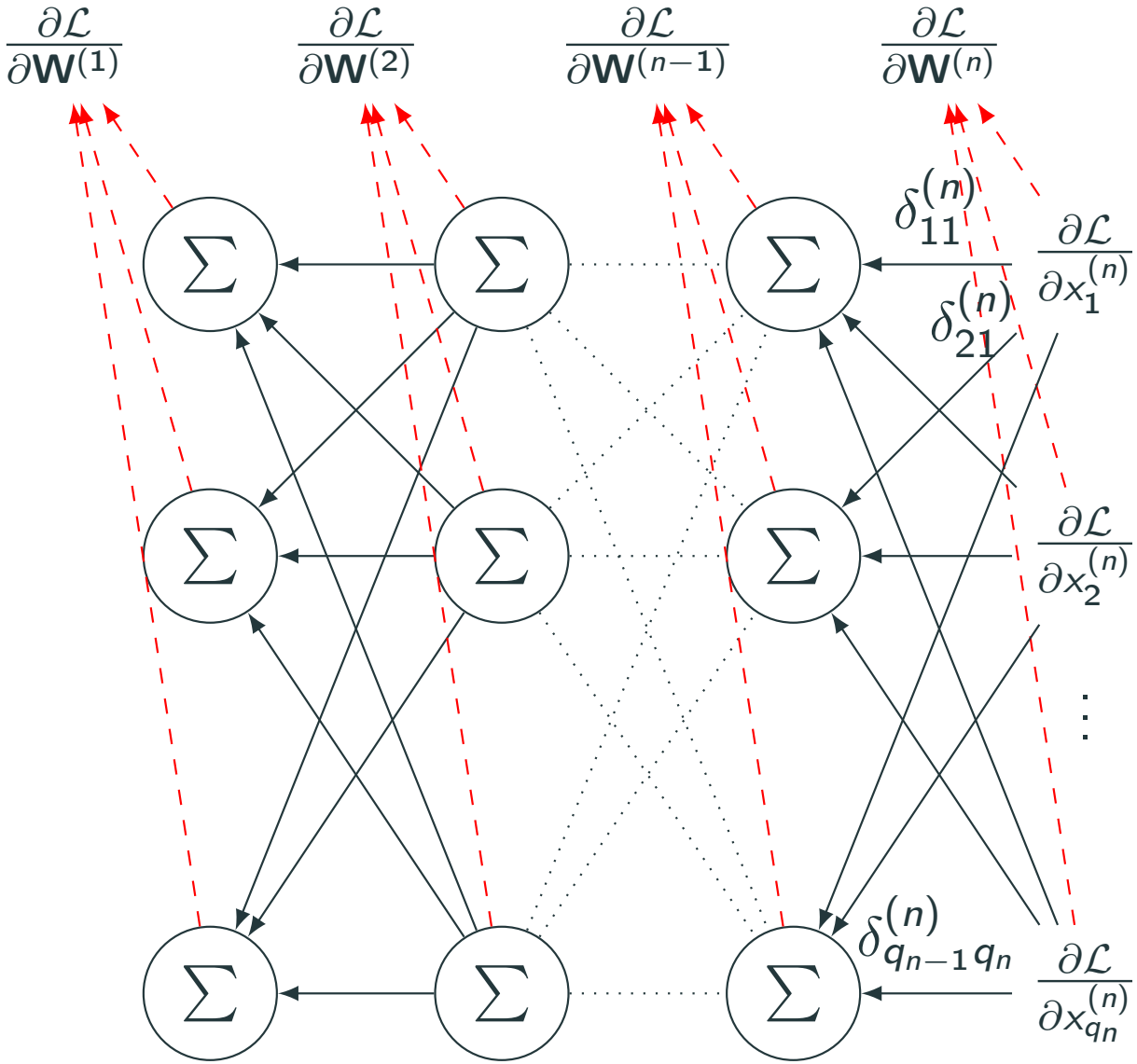
# Rétropropagation du gradient

Rétropropagation :



# Rétropropagation du gradient

Rétropropagation :



# Descente de gradient mini-batch

On initialise $\boldsymbol{\theta}$ à une valeur aléatoire $\boldsymbol{\theta}_0$. Puis on itère :

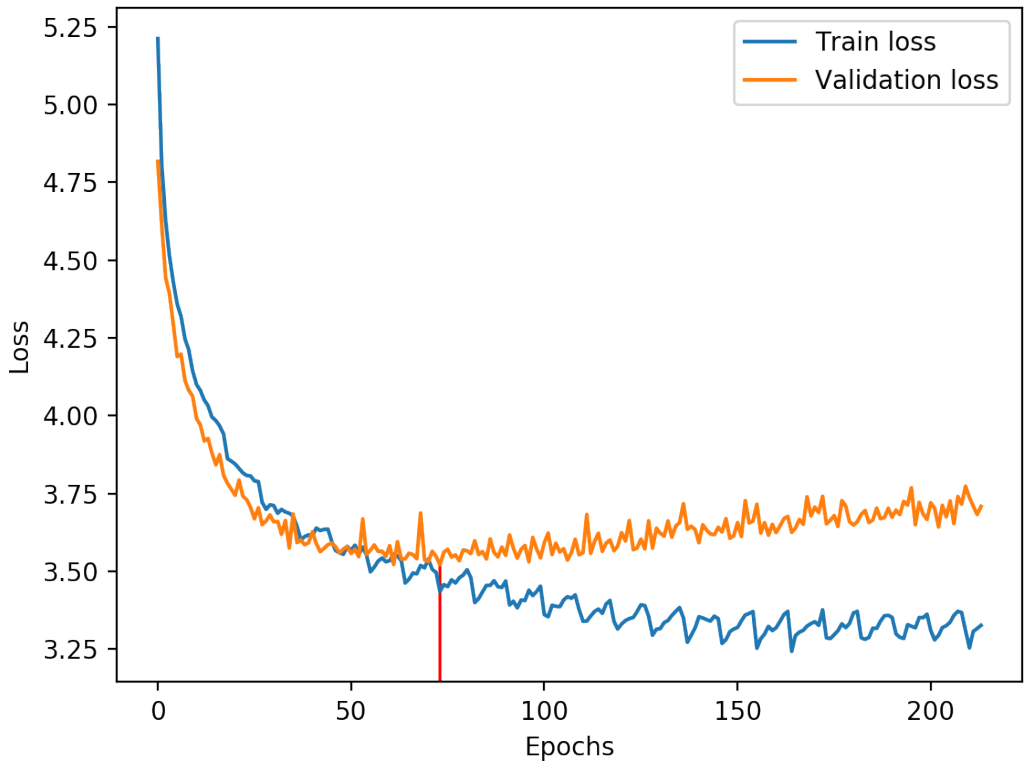
- On tire au hasard un batch de données de taille $K$
  $B_n = \{(\mathbf{x}_{a_1}, \mathbf{t}_{a_1}), (\mathbf{x}_{a_2}, \mathbf{t}_{a_2}), ..., (\mathbf{x}_{a_K}, \mathbf{t}_{a_K})\}$
- On calcule le gradient de la fonction de coût restreinte à ce batch (en utilisant la méthode de rétropropagation du gradient) :
$$\mathcal{L}_B(\boldsymbol{\theta}) = \sum_{k=1}^{K} d(\mathbf{t}_{a_k} | \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{a_k}))$$
- On met à jour $\boldsymbol{\theta}$ :
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_B$$

Quand est-ce qu'on s'arrête ?

# Early stopping

L'erreur sur la base de données d'apprentissage n'est pas révélatrice de la capacité à généraliser sur de nouvelles données.

On utilise l'état du réseau pour lequel la généralisation est la meilleure (coût de validation le plus faible).



# Classification

- Classification multiclasse multilabel : sortie cible $\mathbf{t} \in \{0,1\}^{N_c}$
- Classification multiclasse monolabel : sortie cible $C \in \{1,\ldots,N_c\}$.
  En pratique, encodage one-hot : $\mathbf{t} \in \{0,1\}^{N_c}$ avec $[\mathbf{t}]_C = 1$ et $[\mathbf{t}]_k = 0$ pour $k \neq C$

# Classification monolabel

sortie cible $C \in \{1, \ldots, N_c\}$.

La fonction $f$ estimée par le réseau de neurones est considérée comme une estimation de la probabilité d'observer la classe considérée :

$$[\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})]_k = P(c = k|\mathbf{x}, \boldsymbol{\theta})$$

On doit donc avoir $\sum_k [\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})]_k = 1$ et $0 \leq [\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})]_k \leq 1$.

On choisit généralement la fonction softmax comme fonction d'activation de la dernière couche cachée :

$$\mathbf{softmax}(\mathbf{z})_j = \frac{\mathrm{e}^{z_j}}{\sum_{k=1}^{K} \mathrm{e}^{z_k}}$$

(Remarque : agit sur l'ensemble du vecteur d'entrée et non sur une valeur individuelle).

On cherche alors à maximiser la log-vraisemblance sur l'ensemble d'apprentissage annoté $\mathcal{S} = \{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \ldots (\mathbf{x}_N, c_N)\}$.

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{n=1}^{N} \log(P(c = c_n | \mathbf{x}_n, \boldsymbol{\theta}))$$

# Classification monolabel

Fonction de coût : entropie croisée catégorielle.

Encodage *one-hot* :

$$[\mathbf{t}]_k = \begin{cases} 1 \text{ si } c = k \\ 0 \text{ sinon} \end{cases}$$

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) &= \sum_n \log(P([\mathbf{t}_n]_{c_n} | \mathbf{x}_n, \boldsymbol{\theta})) \\ &= \sum_n \sum_k [\mathbf{t}_n]_k \log(P([\mathbf{t}_n]_k | \mathbf{x}_n, \boldsymbol{\theta})) \end{aligned}$$

# Classification monolabel

Décision :

Une fois le réseau entraîné (paramètre optimal $\theta_b$), on peut obtenir pour tout entrée $\mathbf{x}$, une probabilité d'observer la classe $k$ :

$$P(c = k|\mathbf{x}, \boldsymbol{\theta}_b) = [\mathbf{f}_{\boldsymbol{\theta}_b}(\mathbf{x})]_k$$

La classe $C$ prédite est alors celle avec la plus grande probabilité :

$$C = \text{argmax}_k P(c = k|\mathbf{x}, \boldsymbol{\theta}_b)$$

# Classification multilabel

Sortie cible $\mathbf{t} \in \{0,1\}^{N_c}$

- La fonction $f$ estimée par le réseau de neurones est généralement considérée comme une estimation de la probabilité de présence de la classe considérée :

$$[\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})]_k = P(t_k = 1|\mathbf{x}, \boldsymbol{\theta})$$

- Dans le cas multilabel, les probabilités pour les différentes classes ne sont pas directement liées : seule la contrainte $0 \leq [\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})]_k \leq 1$ doit être vérifiée (pour tout $k$).

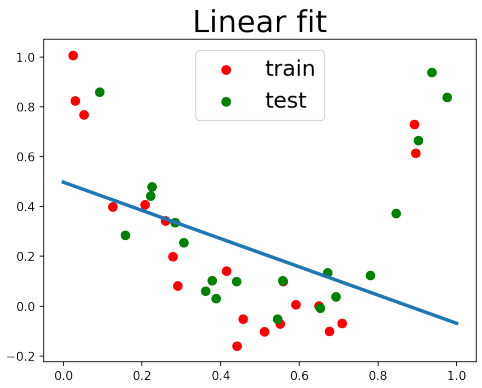
# Classification multilabel

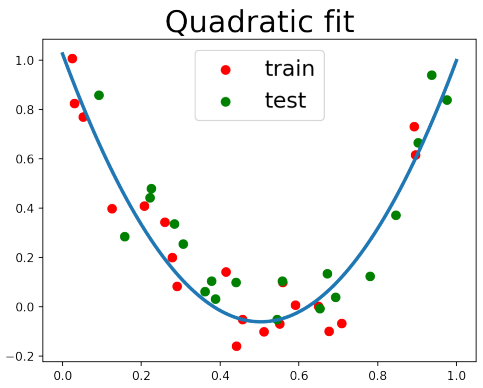
- On utilise généralement la fonction sigmoïde comme fonction d'activation de la dernière couche : $f(x) = \frac{1}{1+\mathrm{e}^{-x}}$
- Fonction de coût : somme des log-vraisemblance pour chaque classe $\equiv$ entropie croisée binaire.

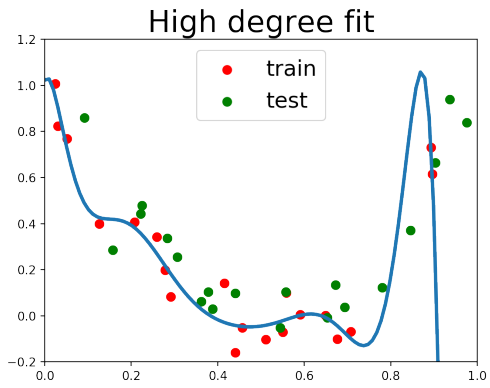
$$\mathcal{L}(\boldsymbol{\theta}) = \sum_n \sum_k [\mathbf{t}_n]_k \log(P([\mathbf{t}_n]_k | x_n, \boldsymbol{\theta})) + (1 - [\mathbf{t}_n]_k) \log(P(1 - [\mathbf{t}_n]_k | x_n, \boldsymbol{\theta}))$$

# Choix de complexité du modèle

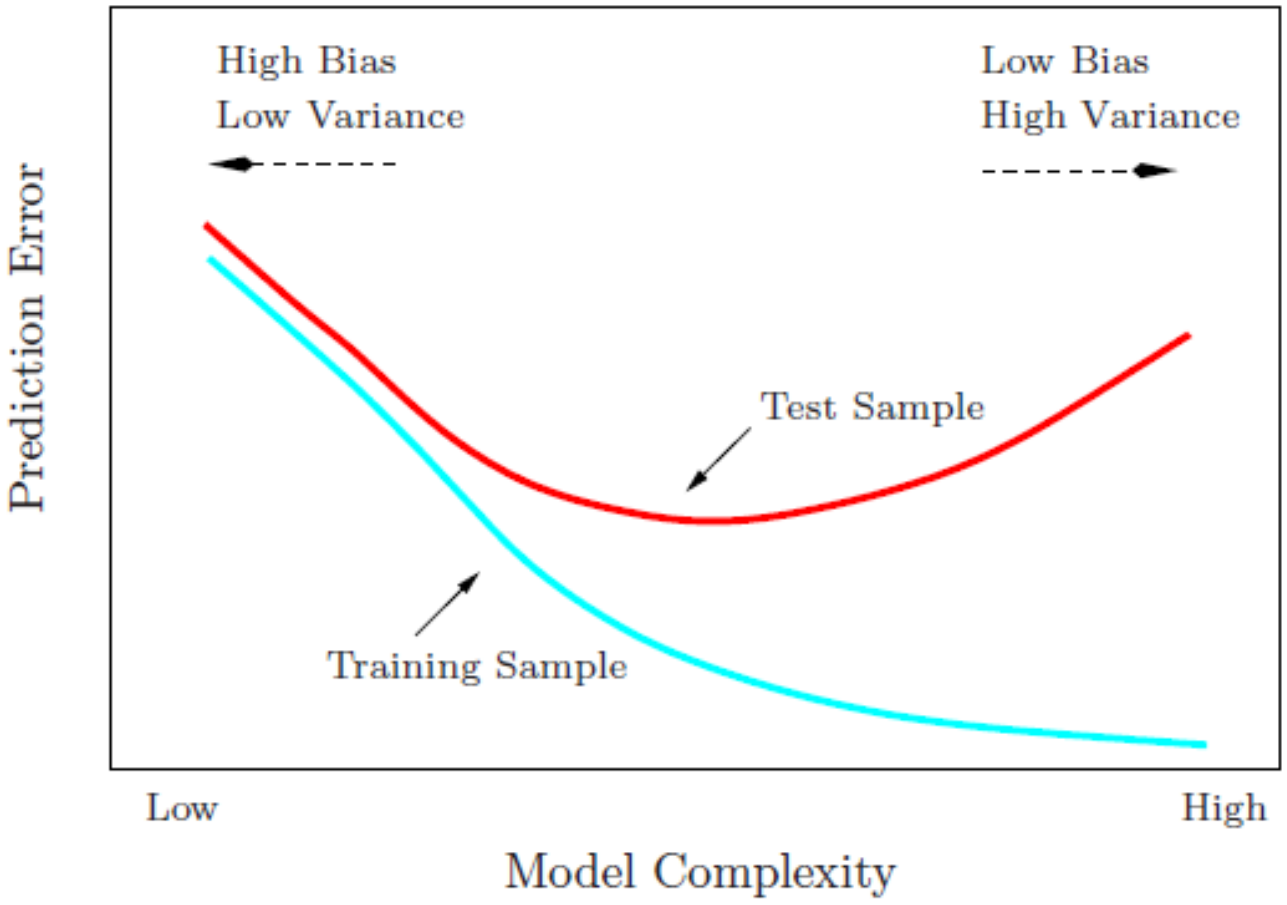
L'erreur sur la base de données d'apprentissage n'est pas révélatrice de la capacité à généraliser sur de nouvelles données.



# Choix de complexité du modèle




Source : The Elements of statistical learning.

# Sur-apprentissage

Contrôle du sur-apprentissage :

- Partitionnement de la base de données en base d'entraînement, de validation (choix de modèle) et de test (estimation de la performance du système).
- Validation croisée

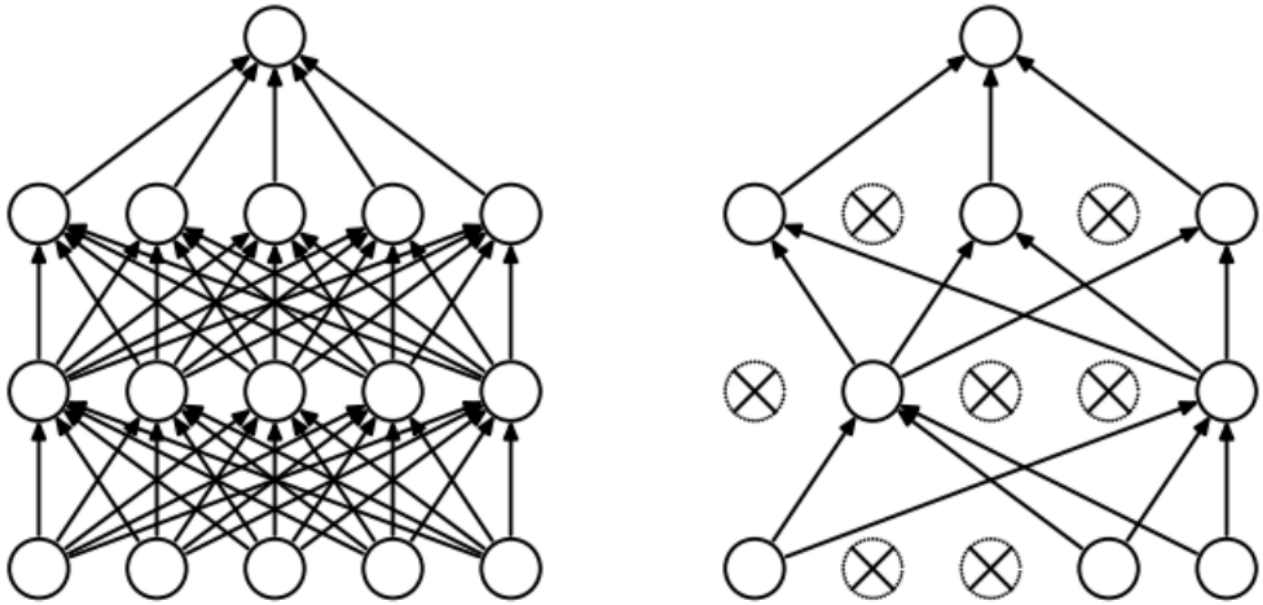
# Sur-apprentissage

Réduction du sur-apprentissage :

- Réduire le nombre de paramètres (nombres de couches, nombre de paramètres par couche).
- Régularisation, Dropout.
- Augmenter la taille de la BDD d'entraînement.
- Early stopping

# Dropout

A l'entraînement, on désactive aléatoirement certains neurones avec une probabilité $p$. Permet de combiner (et de rendre indépendant) l'apprentissage de nombreux sous-réseaux de manière efficace.



(a) Standard Neural Net (b) After applying dropout.

Source : Srivastava, Nitish, et al. "Dropout : a simple way to prevent neural networks from overfitting", JMLR 2014

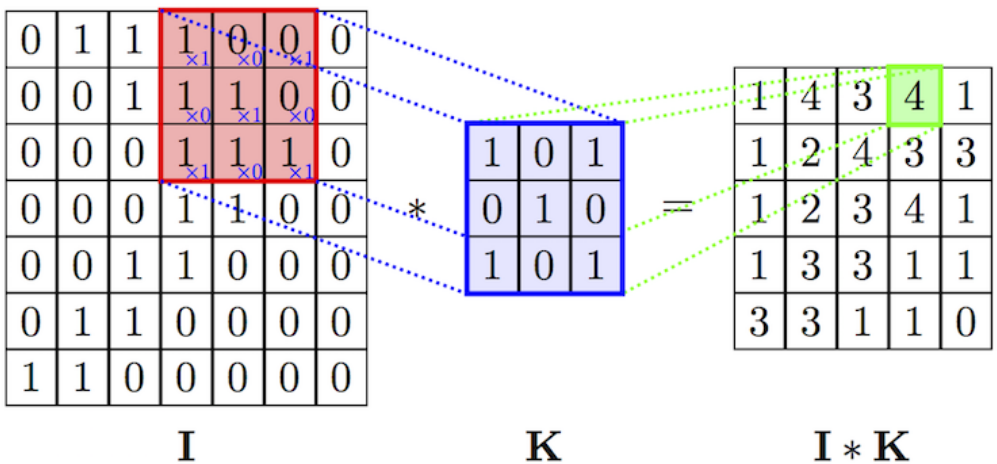
# Architectures de réseaux de neurones

# Couches convolutives

- Poids : tenseur d'ordre 3, $\mathbf{W} \in \mathbb{R}^{K \times F \times T}$. $K$ noyaux $w_{k,f,t}$.
- Entrée : $x_{c,f,t}$

  $c$ canaux, $f$, $t$ dimensions spatiales (images), ou fréquence/temps (audio).
- Sortie :

$$y_{k,f,t} = \sum_{c=0}^{C-1} \sum_{p=0}^{F-1} \sum_{q=0}^{T-1} w_{k,p,q} x_{c,f-p,t-q}$$




Source : Deep learning for complete beginners

# Couches convolutives

Une couche convolutive est une couche complètement connectée avec des contraintes particulières :

- De nombreux poids sont nuls : connectivité locale.
- Poids partagés entre les connexions

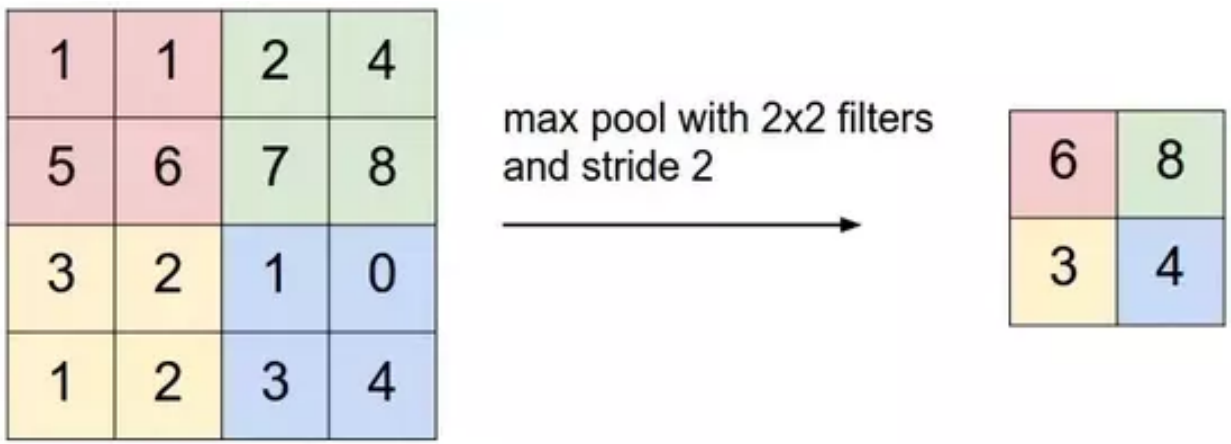
Intérêt :

- Réduit le nombre de paramètres
- Connectivité locale : encode des relations de proximité.

# Max Pooling

Principe : sous-échantillonage.

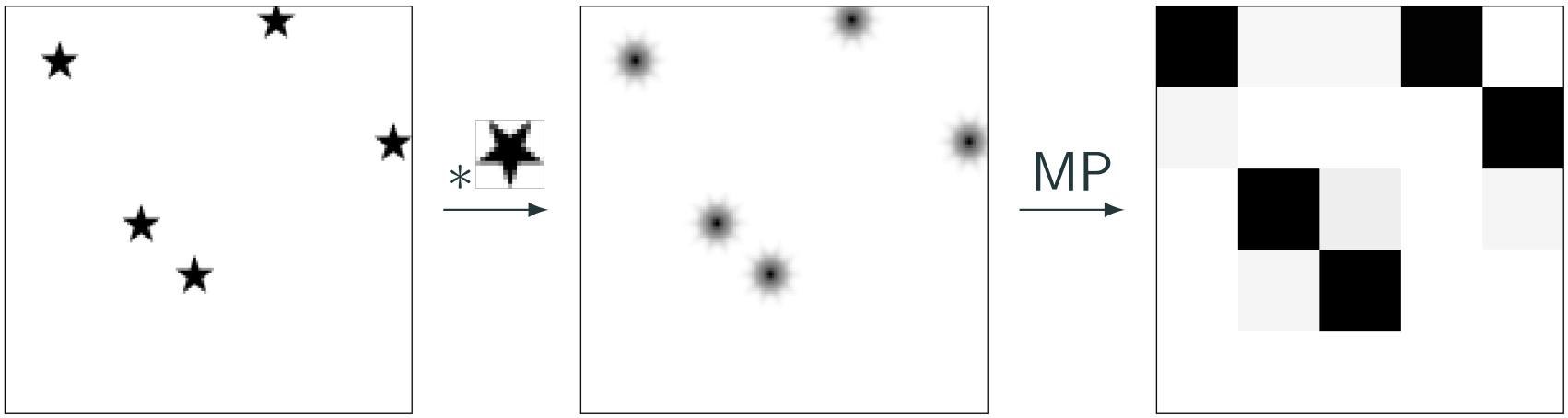
- $MP(y_{k,f,t}) = \max_{p \in [\![\lfloor f \rfloor_{\delta_f}, \lfloor f \rfloor_{\delta_f} + \delta_f [\![, q \in [\![\lfloor t \rfloor_{\delta_t}, \lfloor t \rfloor_{\delta_t} + \delta_t [\![} x_{k,p,q}$ où $\lfloor f \rfloor_{\delta_f} = \lfloor \frac{f}{\delta_f} \rfloor \delta_f$
- Le max peut-être remplacé par une autre fonction de pooling : moyenne, médiane, minimum, variance...




Source : http ://cs231n.github.io/convolutional-networks/

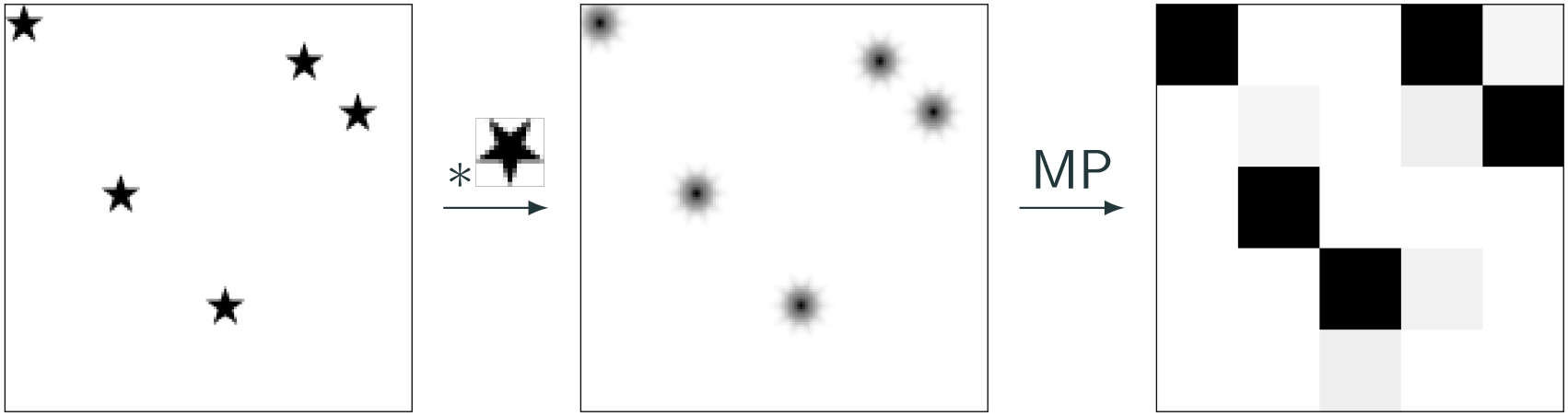
# Max Pooling

Convolution + Max pooling $\Longrightarrow$ Invariance par translation locale :

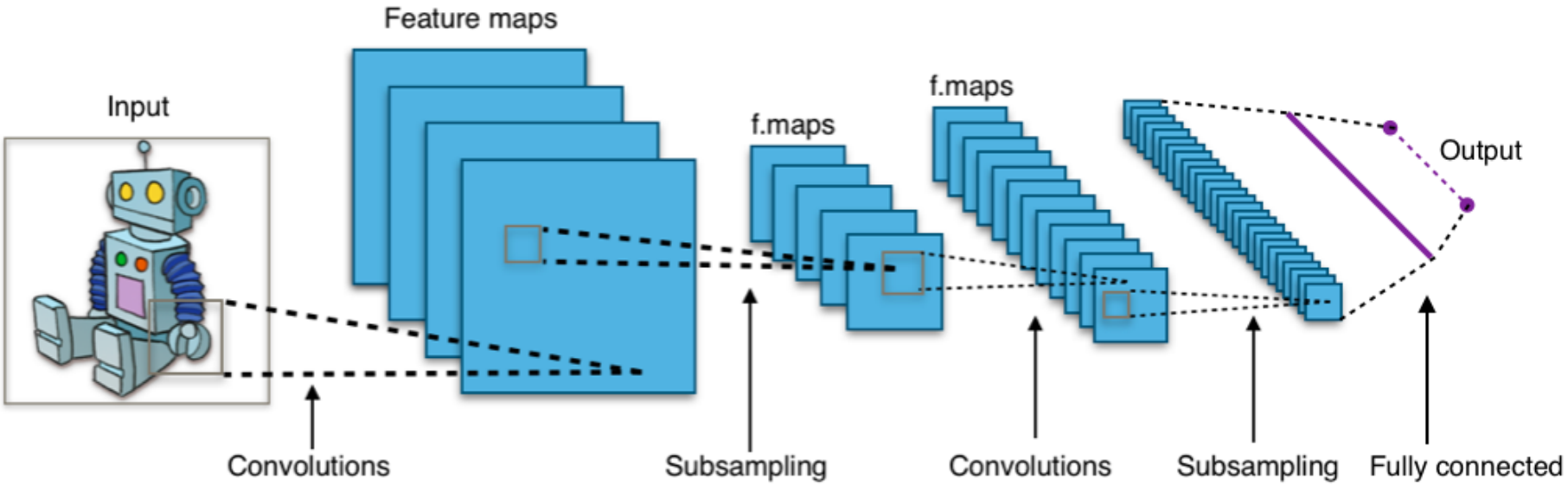


# Max Pooling

Convolution + Max pooling $\Longrightarrow$ Invariance par translation locale :



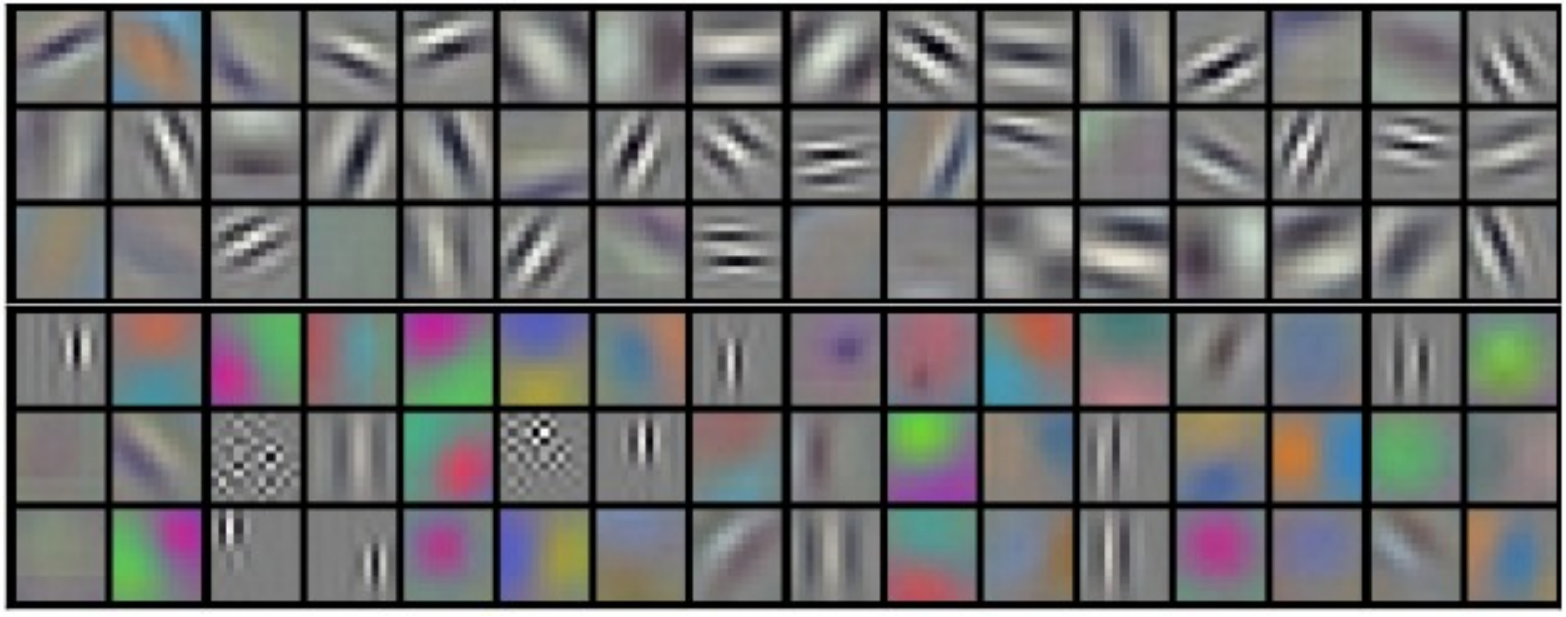
# Réseau convolutif




Source : Wikipedia

# Réseau convolutif

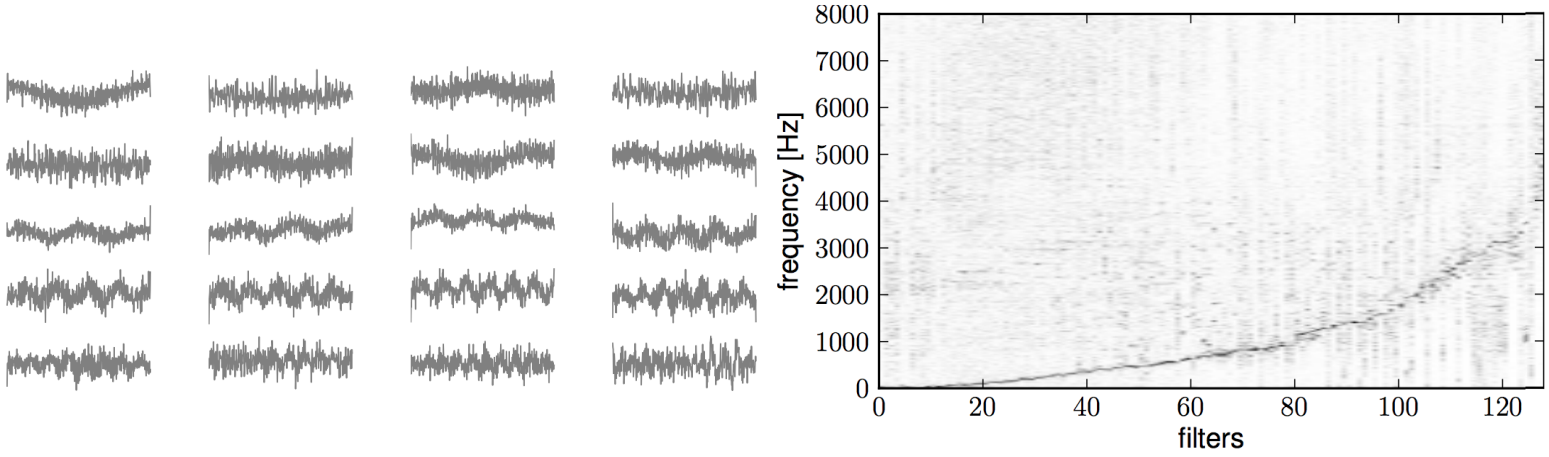
Filtres première couche (entrée : images).



Source : Krizhevsky et al. *ImageNet Classification with Deep Convolutional Neural Networks*

# Réseau convolutif

Filtres première couche (entrée : forme d'onde audio).




Source : Dieleman et al. *End-to-end learning for music audio*

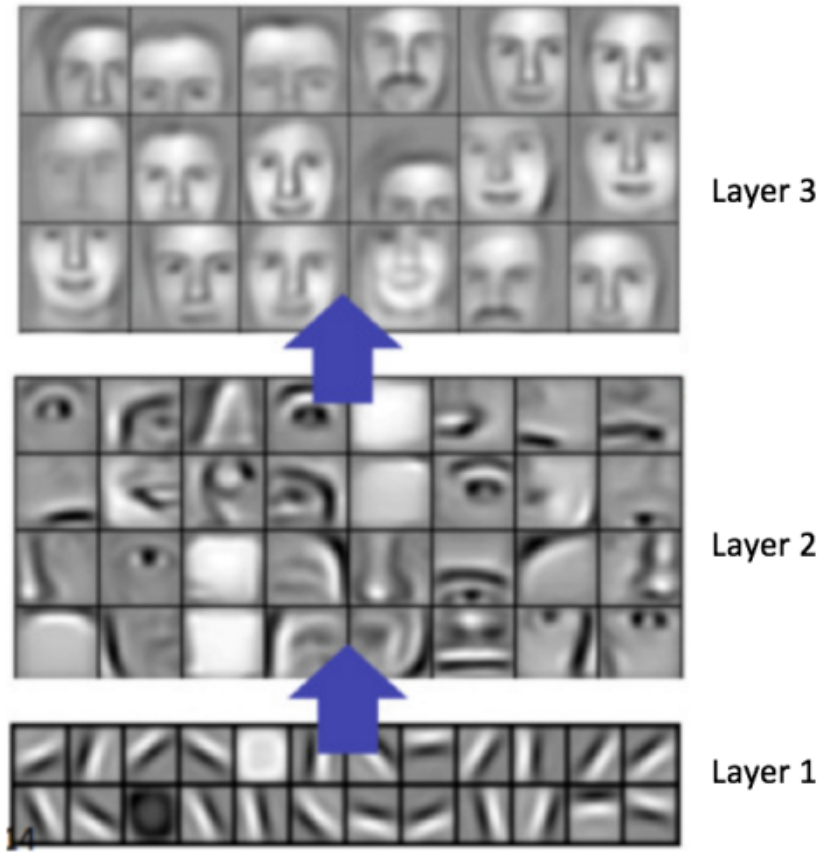
En pratique : on travaille généralement avec une représentation type spectrogramme en entrée.

# Réseau convolutif

Intuition : la combinaison des filtres permet d'apprendre des objets de plus en plus haut niveau.

- Couche 1 : s'active sur des contours simples
- Couche 2 : s'active sur des parties d'objet (oeil, nez, bouche...)
- Couche 3 : s'active sur des objets complets (visages)

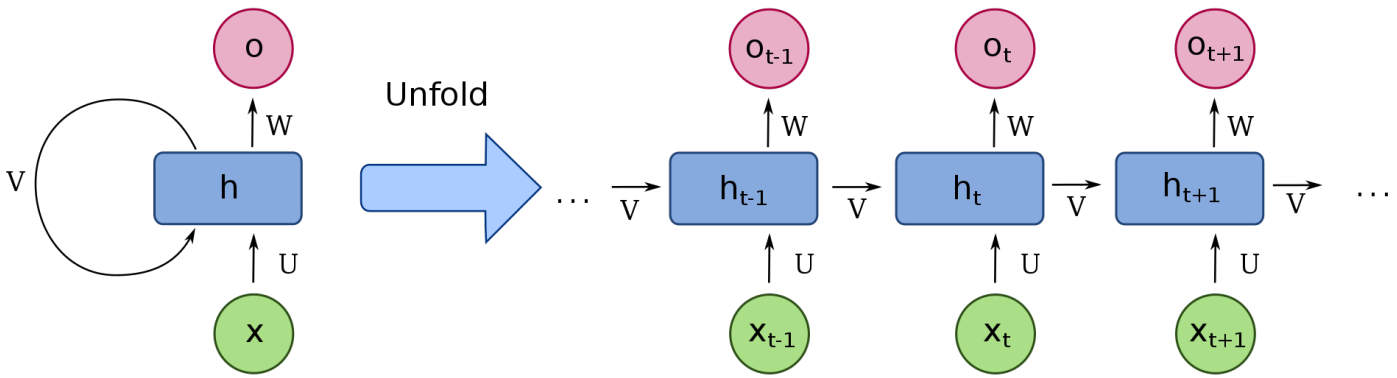
En pratique, un peu plus complexe : pas facile d'intérpreter le comportement du réseau.




Source : Honglak Lee et al. *Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations*

# Réseaux récurrents

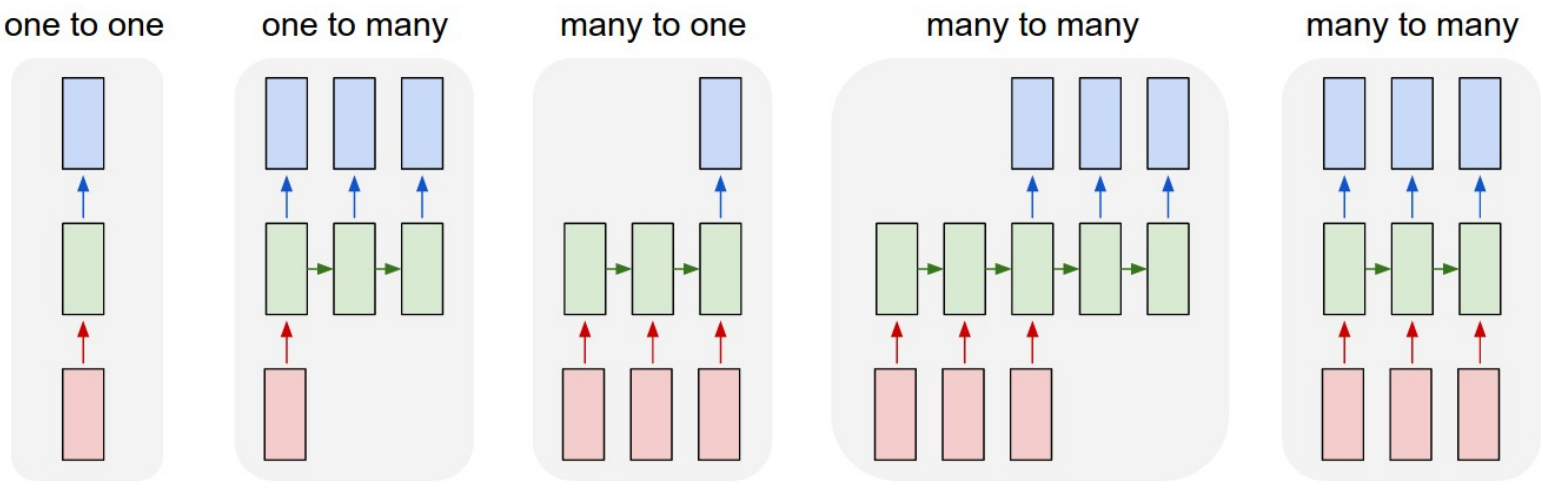
- Traitement de séquences
- Réseau répété à chaque étape temporel avec des connexions récurrentes entre chaque étape.




Source : Wikipedia

# Réseaux récurrents

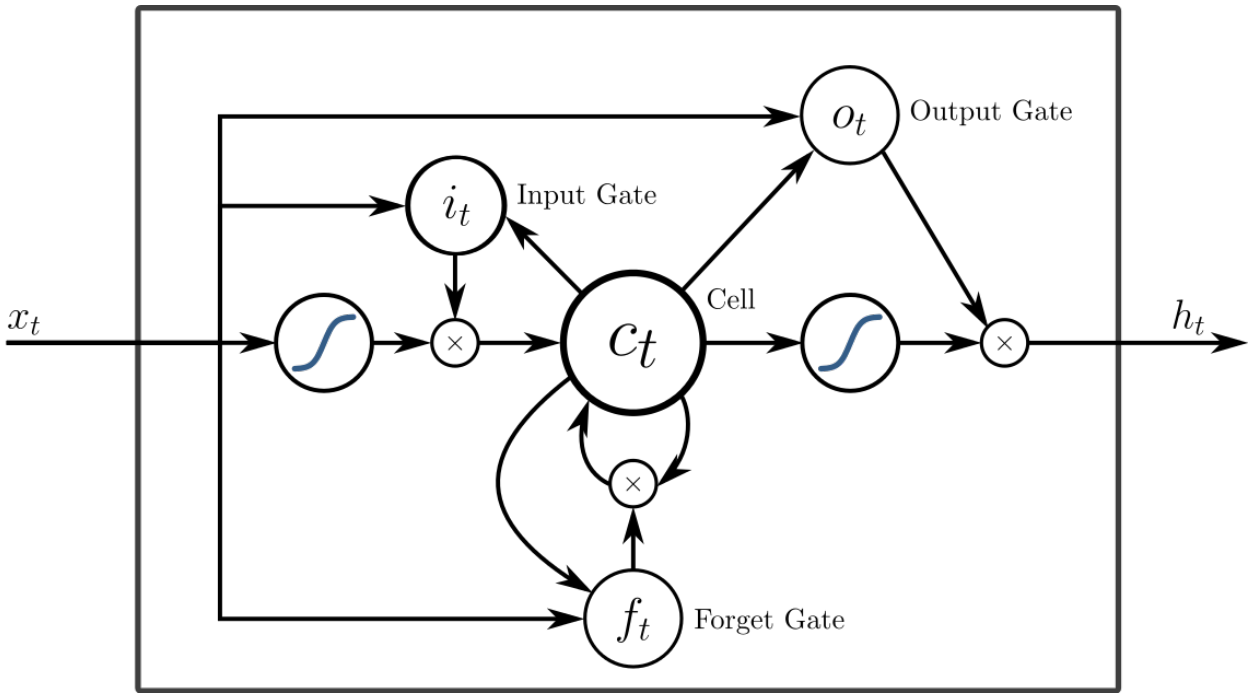
Permet de nombreuses combinaisons entrée/sortie.




Source : Andrej Karpathy

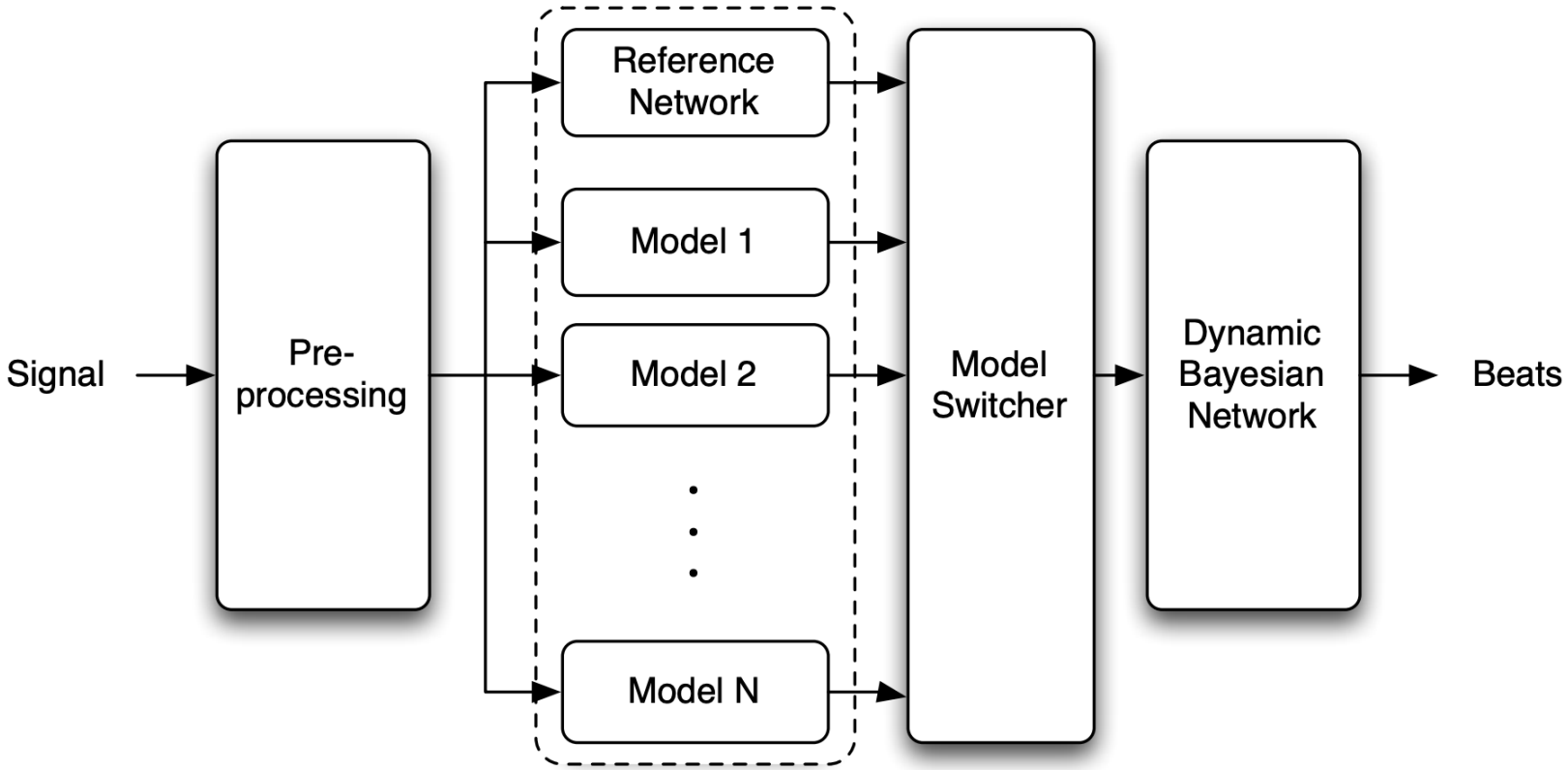
Problème : réseau très profond $\Longrightarrow$ *vanishing gradient*.

Solution : bloc à mémoire LSTM



# Réseaux récurrents : applications

Beat tracking : le réseau apprend une fonction d'onset.




Source :

S. Bock et al. A Multi-model Approach To Beat Tracking Considering Heterogeneous Music Styles

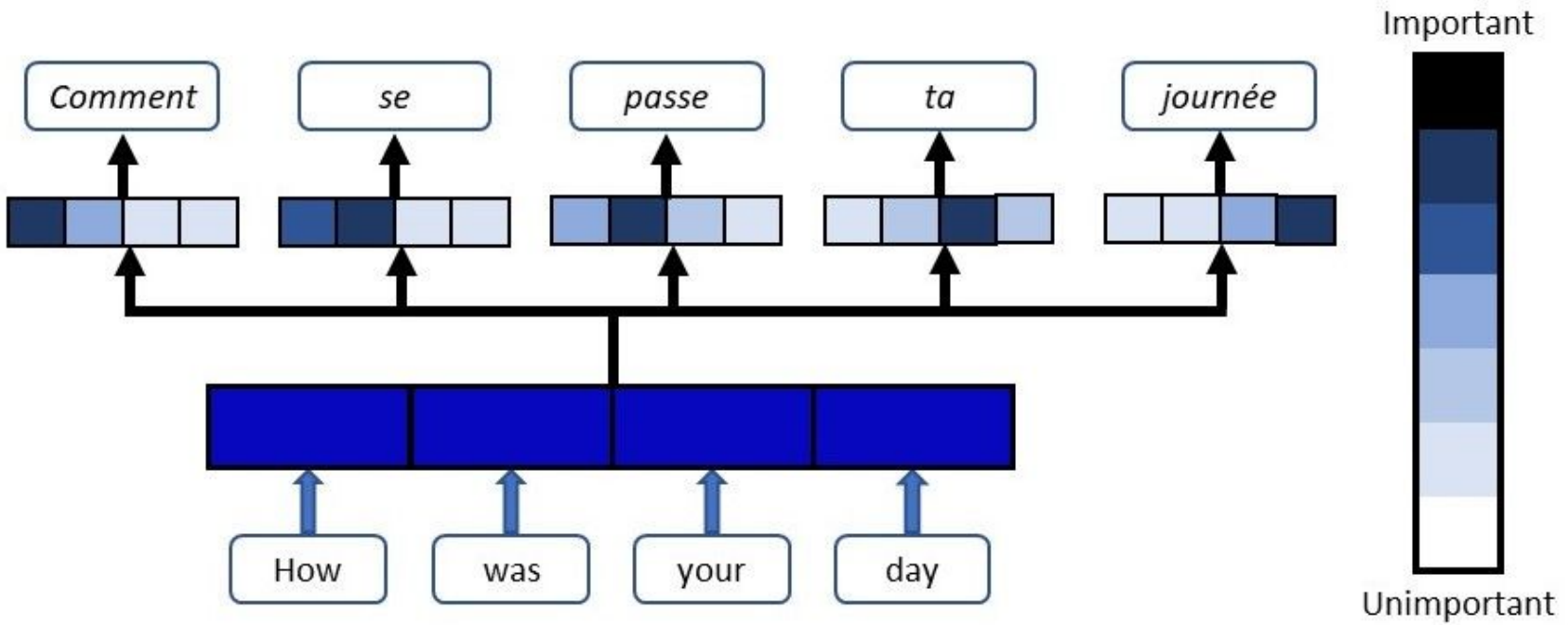
# Transformeurs

Architecture permettant de modèliser des séquences :

- très utilisée en NLP, notamment dans les *Large Language Models* (e.g. *ChatGPT*)
- basée sur les mécanismes d'auto-attention

# Mécanisme d'attention

Calcul d'un poids mesurant l'utilité des entrées pour estimer la sortie. Le modèle fait sa prédiction en se concentrant sur les "entrées importantes".




Source : blog.floydhub.com

# Mécanisme d'attention

- Séquence de vecteurs proxi $z_q^1, \ldots, z_q^T$ et $z_k^1, \ldots, z_k^T$ (dépendant de l'entrée $v^t$ et potentiellement de l'état caché au temps précédent).
- Fonction de similarité $s(z_q^t, z_k^{t'})$.
- Construction de l'attention au temps $t$ :

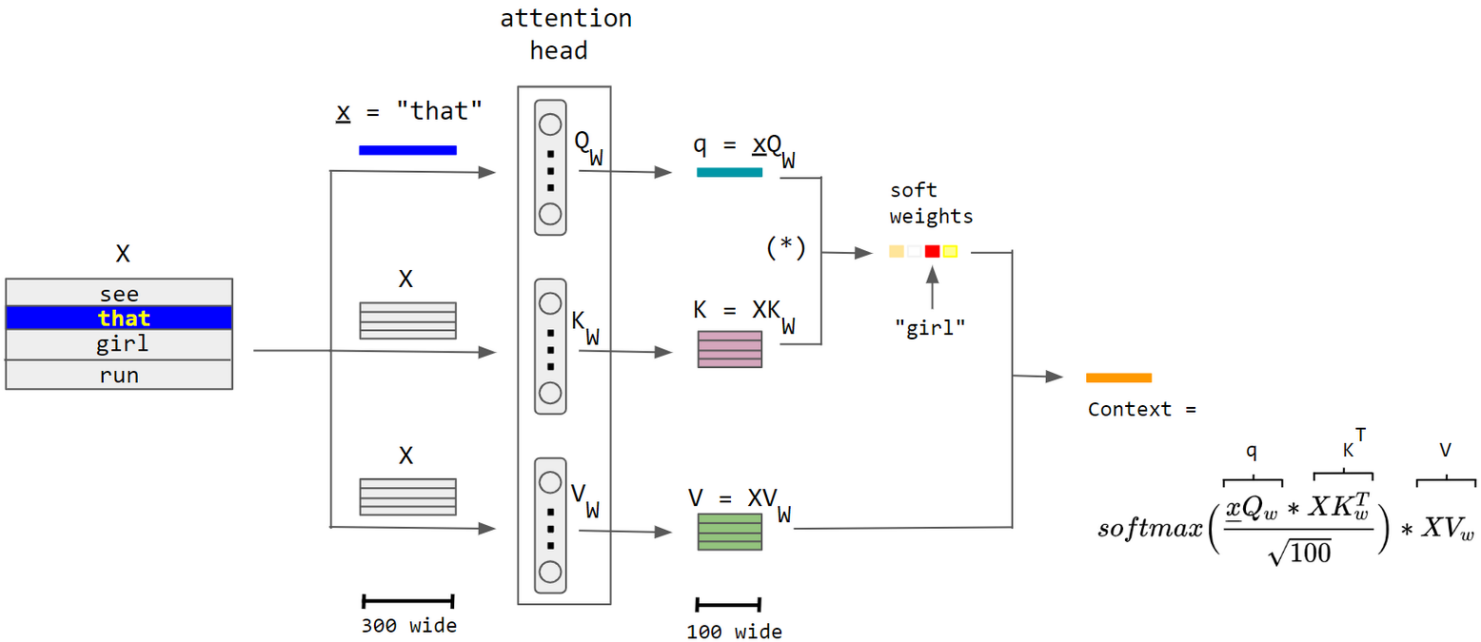
$$\alpha(t) = \text{Softmax}(s(z_q^t, z_k^1), s(z_q^t, z_k^2) \ldots, s(z_q^t, z_k^T))$$

- Sortie du mécanisme d'attention :

$$u^t = \sum_{t=1}^{T} \alpha(t) v^t$$

# Transformeurs

Unité de base du transformeur : *Tête d'auto-attention à produit scalaire*




Source : Wikipedia

Chaque élément de la séquence est comparé avec toute la séquence.

Généralement, plusieurs têtes d'attention encodent des relations différentes entre les éléments de la séquence d'entrée.

# Transformeurs

Permet de s'affranchir des problèmes des RNN :

- permet de modéliser les dépendances à long terme.
- pas de phénomène de *vanishing gradient*.
- meilleure parallélisation que les RNN.

Remarque : perte de la notion d'ordre dans la séquence,
généralement compensée par l'utilisation de *positional embeddings*.

# Machines de calcul et librairies

Architecture convolutive/transformeurs : calcul massivement parallèlisable, opérations simples (multiplication/addition floattante).

Utilisation de processeurs graphiques (GPU) à plusieurs milliers de coeurs.

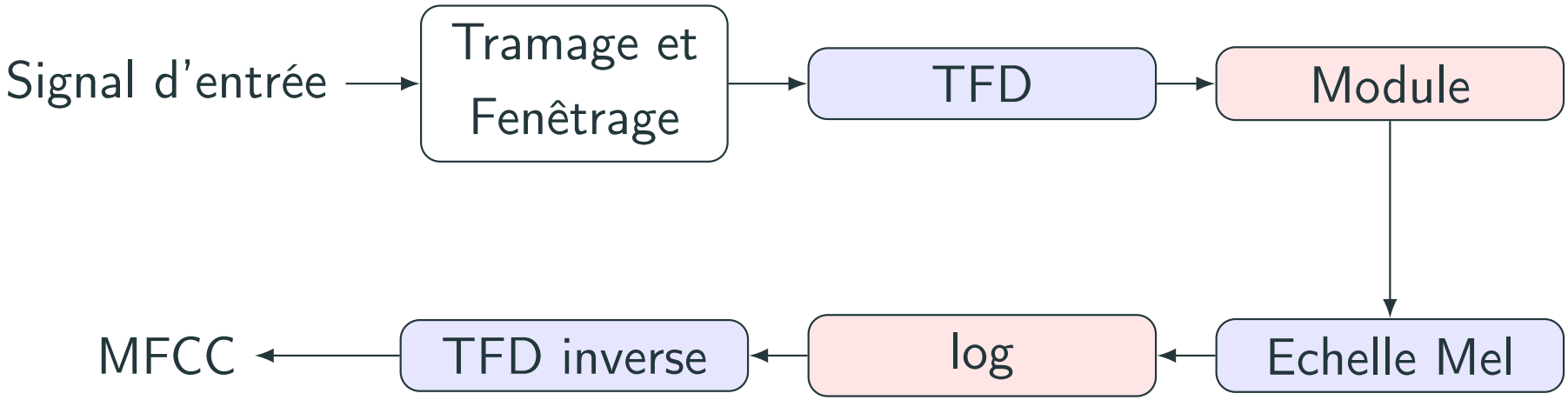
Librairies :

- TensorFlow/Keras (Google)
- Torch/PyTorch (Facebook)
- Caffe
- MxNet
- ...

# Deep Learning pour la classification musicale

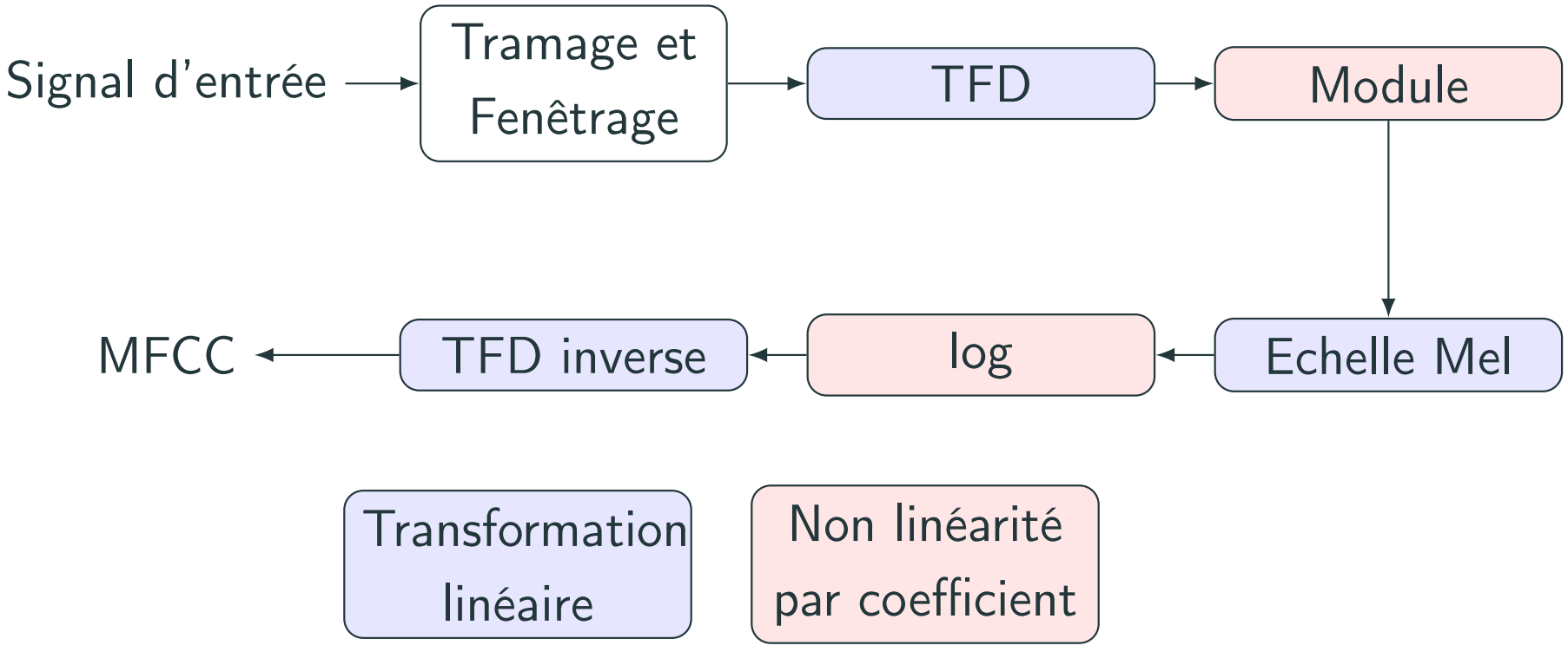
# Analogie descripteurs classiques/architectures profondes.

MFCC :



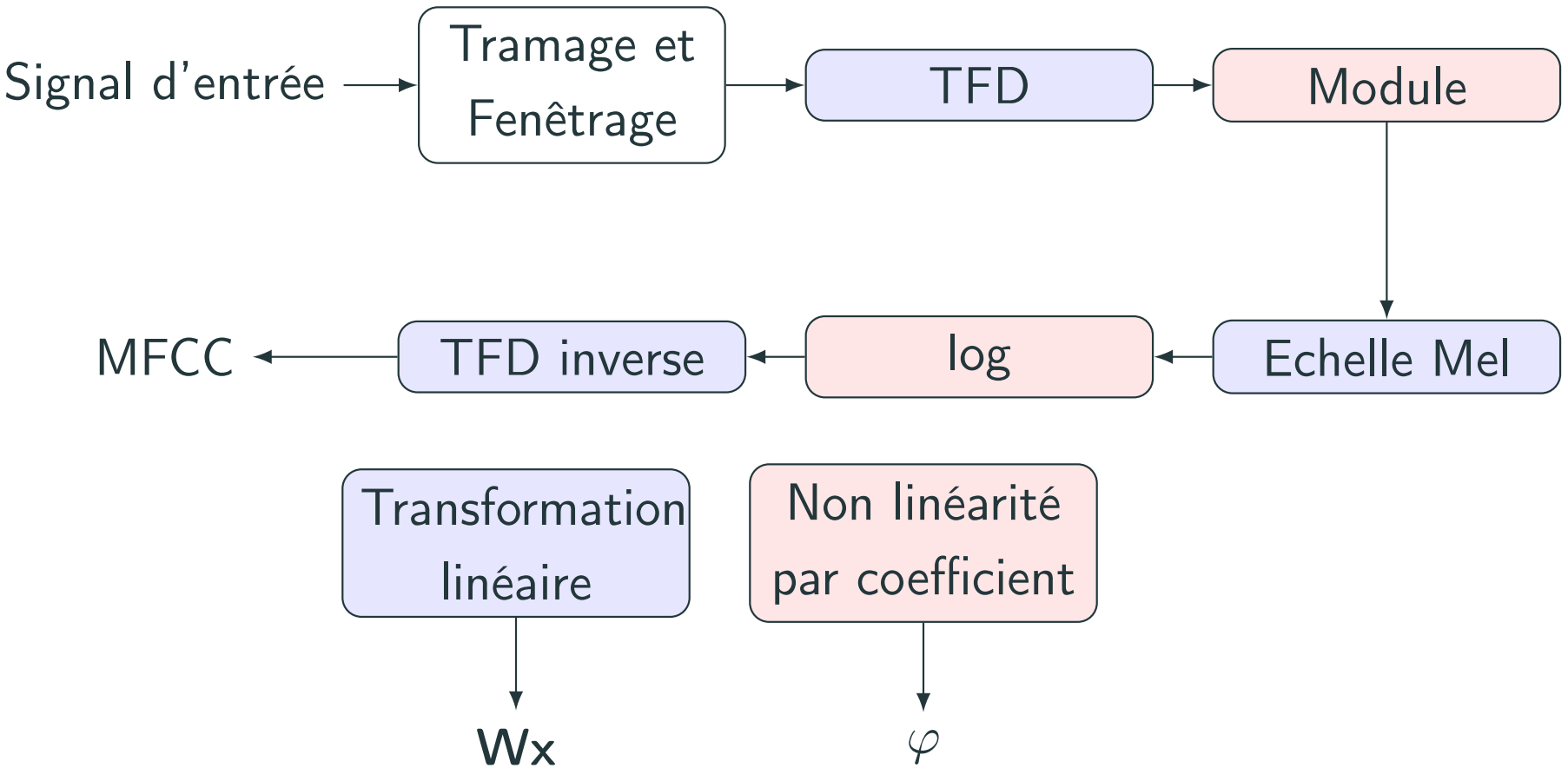
# Analogie descripteurs classiques/architectures profondes.

MFCC :



# Analogie descripteurs classiques/architectures profondes.

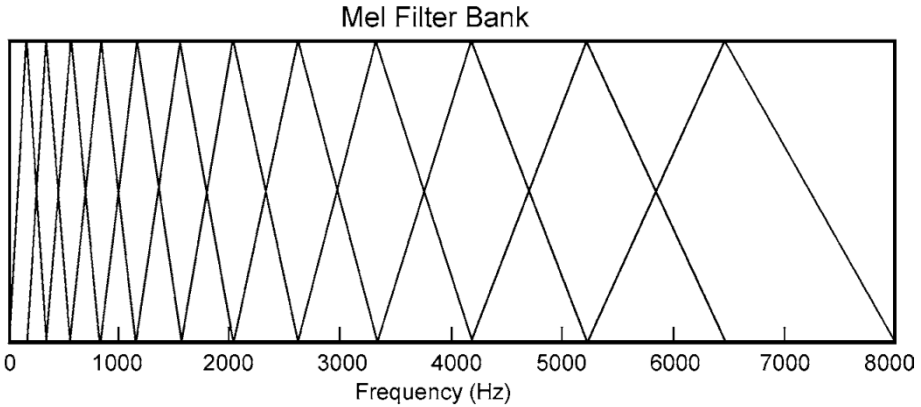
MFCC :



# Analogie descripteurs classiques/architectures profondes.

Le calcul des MFCC est un réseau de neurones ! appliqué trame par trame au signal d'entrée, avec :

- Pour la TFD, $\mathbf{W}$ est une matrice de sinusoïdes complexes.
- Pour la transformation en échelle Mel, $\mathbf{W}$ est une matrice de filtres de Mel.




- Pour la TFD inverse, $\mathbf{W}$ est à nouveau une matrice de sinusoïdes complexes.

# Analogie descripteurs classiques/architectures profondes.

Chromagramme :

$$C(\mathcal{N}, t) = \sum_{k=1}^{K} |S(2^k f_0(\mathcal{N}), t)| = \sum_{f=0}^{F} \mathbb{1}_{2^k f_0(\mathcal{N})}(f) |S(f, t)|$$

- TFD : transformation linéaire
- Module : fonction d'activation non-linéaire
- Somme sur $k$ : transformation linéaire, avec $W$ une matrice à 12 lignes (12 notes), dont la ligne $\mathcal{N}$ a ses coefficients non nuls en les $2^k f_0(\mathcal{N})$

Le calcul des chromas peut aussi être vu comme un réseau de neurones.

# Analogie descripteurs classiques/architectures profondes.

Idem calcul des onsets pour la détection de pulsations :

- Valeur absolue, rectification demi-onde (=ReLU) : fonction non-linéaire scalaire.
- filtrage passe-bas, dérivée : transformation linéaire (convolutive).

# Analogie descripteurs classiques/architectures profondes.

Idée :

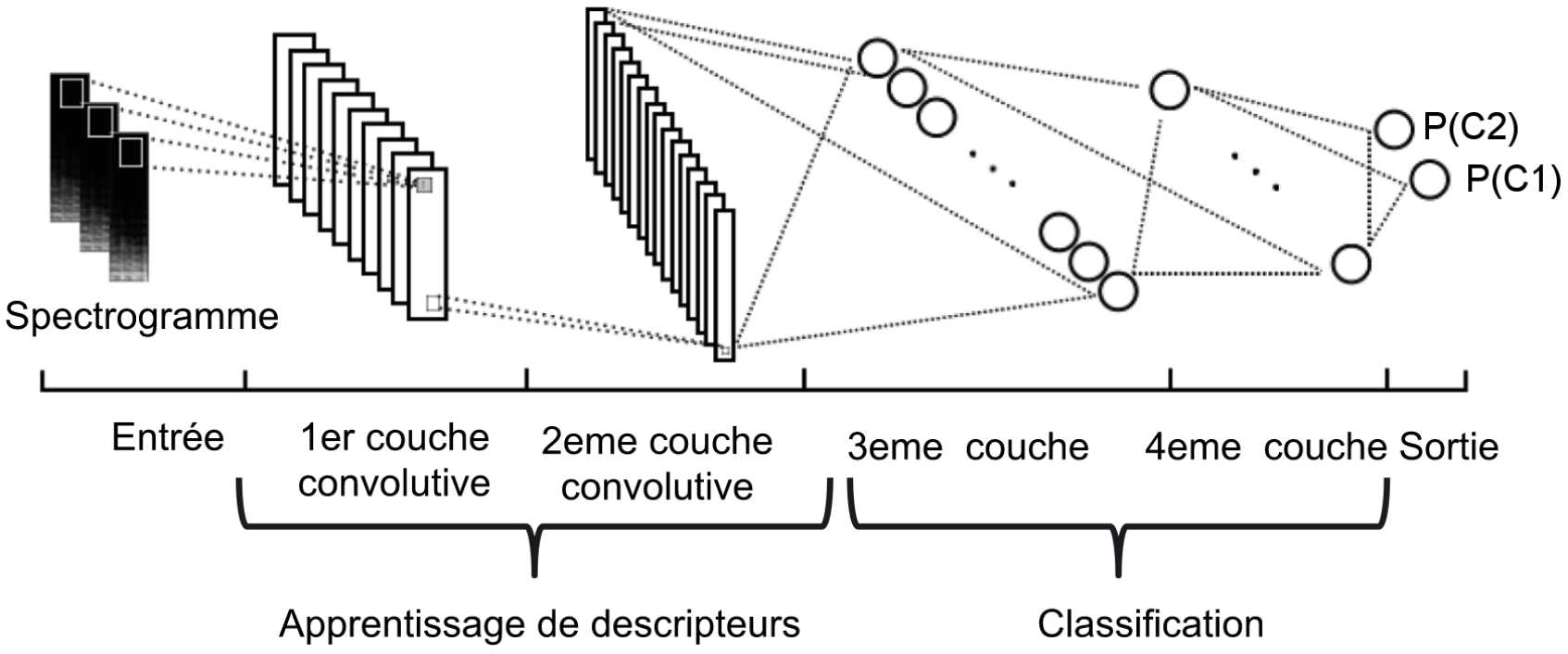
- Remplacer le calcul des descripteurs par un réseau de neurones (par exemple un réseau convolutif).
- Utiliser un réseau de neurones simple comme classifieur.

Intérêt :

- le système peut être optimisé globalement.
- plus besoin de construire des descripteurs à la main.
- descripteurs optimaux (pour peu qu'on arrive à résoudre le problème d'optimisation).

Limitation : nécessite beaucoup de données pour fonctionner.

# Deep learning pour la classification de musique



# Deep learning pour la classification de musique

Sortie (multilabel ou monolabel) :

- Genre
- Mood
- Décennies
- Instruments de musique
- Qualité audio (MP3/non compressé)
- ...

# Genre

Monolabel sur des genres principaux : genre GTzan

blues

classical

country

disco

hiphop

jazz

metal

pop

reggae

rock

# Genre

## Multilabel avec un large ensemble de tags : genre Discogs

aor, abstract, acid, acid jazz, acoustic, african, afro-cuban, afro-cuban jazz, afrobeat, alternative rock, ambient, arena rock, art rock, avant-garde jazz, avantgarde, ballad, baroque, bass music, beat, big band, big beat, black metal, bluegrass, blues rock, bolero, boom bap, bop, bossa nova, bossanova, brass band, breakbeat, breakcore, breaks, brit pop, broken beat, cajun, celtic, chanson, chicago blues, classic rock, classical, comedy, conscious, contemporary, contemporary jazz, contemporary r&b, cool jazz, country, country blues, country rock, crunk, dance-pop, dancehall, dark ambient, darkwave, death metal, deep house, delta blues, disco, dixieland, doom metal, downtempo, drone, drum n bass, dub, dub techno, dubstep, ebm, early, easy listening, electric blues, electro, electro house, emo, ethereal, euro house, eurodance, europop, experimental, field recording, flamenco, folk, folk metal, folk rock, free improvisation, free jazz, freestyle, funk, funk metal, fusion, future jazz, g-funk, gabber, gangsta, garage rock, glam, glitch, goa trance, gospel, goth rock, gothic metal, grime, grindcore, grunge, gypsy jazz, happy hardcore, hard bop, hard house, hard rock, hard trance, hardcore, hardcore hip-hop, hardstyle, heavy metal, hip hop, honky tonk, horrorcore, house, idm, impressionist, indian classical, indie pop, indie rock, industrial, instrumental, italo-disco, italodance, jazz-funk, jazz-rock, jazzy hip-hop, jungle, klezmer, krautrock, latin, latin jazz, leftfield, lo-fi, lounge, mpb, math rock, medieval, melodic death metal, melodic hardcore, metalcore, minimal, mod, modern, modern classical, modern electric blues, musical, musique concrte, neo soul, neo-classical, neo-romantic, neofolk, new age, new wave, noise, nordic, novelty, nu metal, oi, opera, pacific, parody, poetry, pop punk, pop rap, pop rock, post bop, post rock, post-hardcore, post-modern, post-punk, power metal, power pop, prog rock, progressive house, progressive metal, progressive trance, psy-trance, psychedelic rock, psychobilly, punk, ragga hiphop, ragtime, reggae, reggae-pop, religious, renaissance, rhythm & blues, rhythmic noise, rnb/swing, rock & roll, rockabilly, rocksteady, romani, romantic, roots reggae, rumba, salsa, samba, schlager, score, screw, shoegaze, ska, sludge metal, smooth jazz, soft rock, son, soul, soul-jazz, soundtrack, southern rock, space rock, speed metal, spoken word, stoner rock, surf, swing, symphonic rock, synth-pop, tango, tech house, tech trance, techno, theme, thrash, thug rap, trance, tribal, trip hop, viking metal, vocal, blues, brass & military, children's, classical, electronic, folk, world, & country, funk / soul, hip hop, jazz, latin, non-music, pop, reggae, rock, stage & screen

# Genre

Evaluation : Quand de si nombreux tags sont utilisés, on n'est généralement plus très intéressé par une accuracy (beaucoup de tags manquant, erreur d'annotations, vérité terrain floue...).

- Métrique de ranking : Mean Average Precision (aire sour la courbe Recall/Precision).
- AUC : Aire sous la courbe TPR/FPR.

# Mood

## Multilabel :

Happy, Sad, Funny, Humorous, Euphoric, Uplifting, Motivational, Optimistic, Positive, Terrifying, Horror, Scary, Shocking, Frightening, Blue, Poignant, Depressing, Heartbroken, Melancholy, Sad, Hopeful, Nostalgic, Light, Soft, Sentimental, Romantic, Neutral, Agitated, Angry, Passionate, Aggressive, Dramatic, Violent, Intense, Peaceful, Quiet, Introspective, Relaxed, Spiritual, Calm, Dreamy, Sacred, Dignified, Thoughtful, Serious, Solemn, Concerned, Emotional, Noble, Nervous, Supernatural, Evil, Tense, Fearful, Creepy, Anxious, Barren, Spooky, Eerie, Strange, Mysterious, Weird, Cold, Confused, Suspenseful, Disturbing, Jittery, Dark, Restless, Confident, Strong, Heroic, Prestigious, Majestic, Powerful, Energetic, Triumphant, Epic, Exciting, Martial, Exhilarating, Victorious, Adventurous, Determined, Courageous, Relentless, Erotic, Sultry, Sexy, Rollicking, Upbeat, Celebratory, Feel Good, Cheerful, Bright, Bouncy, Boisterous, Excited, Perky, Fun, Joyous, Lighthearted, Playful...

# Mood

Monolabel : MIREX clusters :

Cluster_1 : passionate, rousing, confident, boisterous, rowdy

Cluster_2 : rollicking, cheerful, fun, sweet, amiable/good natured

Cluster_3 : literate, poignant, wistful, bittersweet, autumnal, brooding

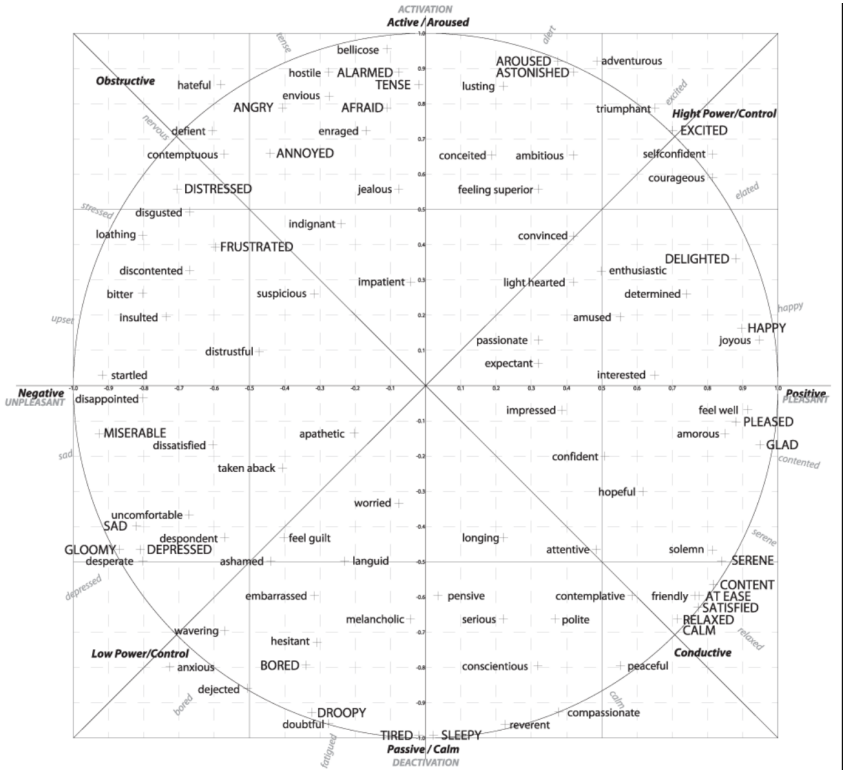
Cluster_4 : humorous, silly, campy, quirky, whimsical, witty, wry

Cluster_5 : aggressive, fiery,tense/anxious, intense, volatile,visceral

# Mood

Russell's Circumplex model.

Valence/Arousal/Dominance




Source : Georgios Paltoglou and Michael Thelwall *Seeing Stars of Valence and Arousal in Blog Posts* 2013.

# Deep learning pour la classification de musique

Entrée :

- Descripteurs classiques (Chroma, MFCC, . . . ).
- Spectrogramme.
- Spectrogramme en échelle Mel (plus compact).
- Transformée à Q-constant (échelle log).
- Forme d'onde : nécessite beaucoup de données.

Pré-traitement courant :

- log : réduit la dynamique des spectrogrammes.
- Centrage et réduction.

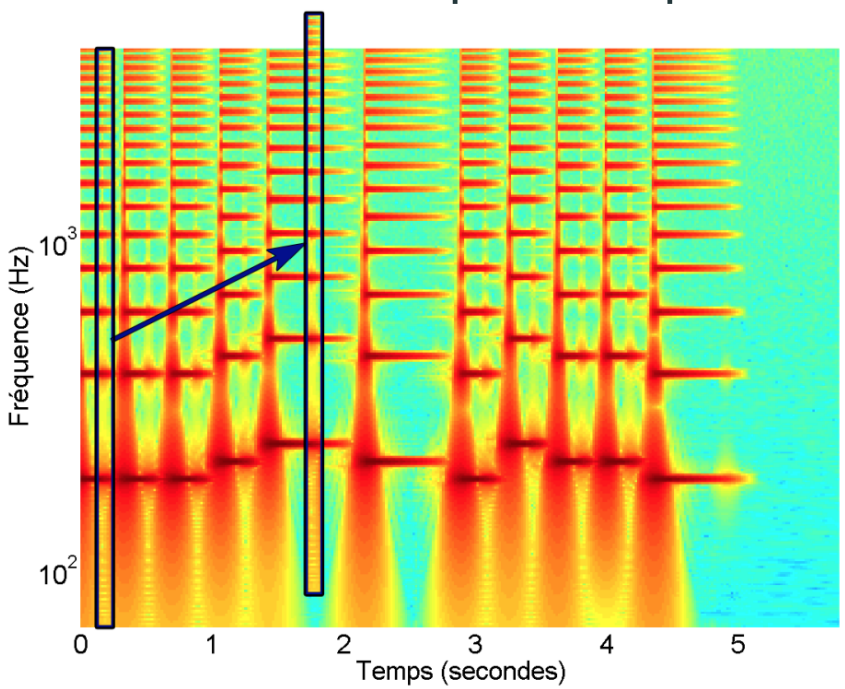
# Deep learning pour la classification de musique

Convolution 2D comme en image ? La dimension temporelle et la dimension fréquentielle n'ont pas la même signification.

Invariance par translation dans le domaine temporel mais pas dans le domaine fréquentiel.

Alternative :

- CQT : invariance partielle par translation



# Deep learning pour la classification de musique

Convolution 2D comme en image ? La dimension temporelle et la dimension fréquentielle n'ont pas la même signification.

Invariance par translation dans le domaine temporel mais pas dans le domaine fréquentiel.

Alternative :

- CQT : invariance partielle par translation
- Convolution 2D locale (par bande de fréquence).
- Convolution 1D.

En pratique pas forcément un problème.

Dimension temporelle variable / sortie fixe :

- entrée de longueur fixe, puis combinaison des sorties a posteriori.
- pooling (max, min, moyenne, variance...) temporelle.
- dernière sortie d'une couche récurrente.
- mécanisme d'attention.

# Manque de données : Augmentation de données

L'apprentissage des réseaux de neurones nécessite de grandes quantités de données annotées. Comment augmenter la taille de la base de données artificiellement ?

Augmentation de données : transformations ne modifiant pas les annotations.

Base de données annotées $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \ldots, (\mathbf{x}_N, c_N)\}$, ensemble de transformation $\{\tau_1, \ldots, \tau_K\}$.

Nouvelle base de données :

$$\{(\mathbf{x}_1, c_1), (\tau_1(\mathbf{x}_1), c_1), \ldots, (\tau_K(\mathbf{x}_1), c_1),$$
$$(\mathbf{x}_2, c_2), (\tau_1(\mathbf{x}_2), c_2), \ldots, (\tau_K(\mathbf{x}_2), c_2),$$
$$\ldots$$
$$(\mathbf{x}_N, c_N), (\tau_1(\mathbf{x}_N), c_N), \ldots, (\tau_K(\mathbf{x}_N), c_N)\}$$

# Manque de données : Augmentation de données

Exemples de transformation :

- Translation temporelle
- Modification de hauteur (pitch-shifting)
- Etirement/contraction temporel (time-stretching)
- Egalisation
- Compression de dynamique
- Dégradation de qualité
- Ajout de bruit
- Réverberation
- ...

# Manque de données : Apprentissage par transfert

L'apprentissage des réseaux de neurones nécessite de grandes quantités de données annotées. Que faire quand les données annotées manquent ?

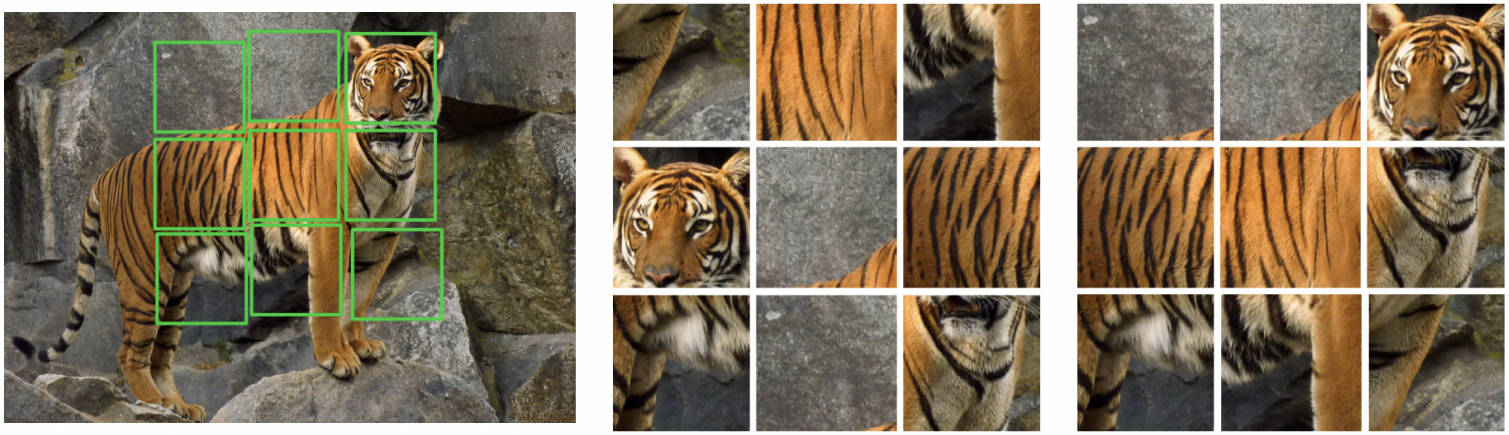
Apprentissage par tranfert : si on dispose d'un réseau entraîné sur une tâche proche, on peut éventuellement transférer la connaissance de cette première tâche à la tâche qui nous intéresse :

- Initialiser les paramètres des couches basses du réseau (couches "descripteurs")
- Fixer ces paramètres, ou les contraindre à ne pas trop s'éloigner de leur initialisation.

# Manque de données : Apprentissage auto-supervisé

Apprentissage auto-supervisé (self-supervised) : tâche pretexte avec les données brutes sans annotations, puis transfert vers la tâche finale.

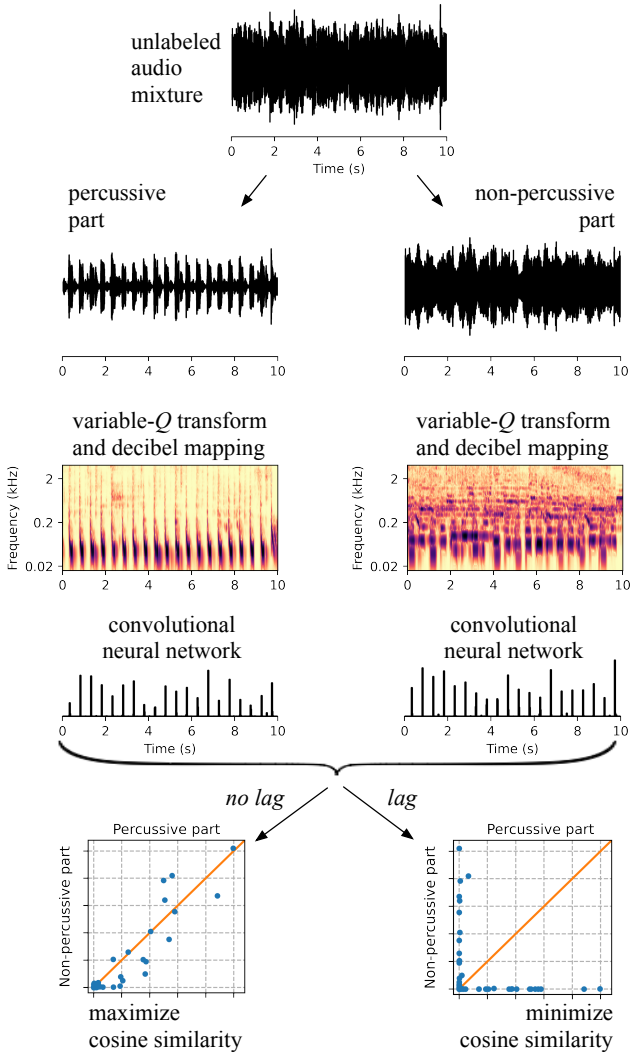
- plus besoin d'annotations !
- la tâche prétexte doit être bien conçue.



(a) (b) (c)

Srce : Noroozi et al. "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles" 2016

# Manque de données : Apprentissage auto-supervisé




Srce : Desblancs et al. "Zero-Note Samba : Self-Supervised Beat Tracking" 2022

# Partionnement entraînement/test

Attention aux variables "latentes" connues pouvant entraîner un sur-apprentissage "caché" :

- Locuteur pour des applications de reconnaissance automatique de la parole.
- Artiste/album/morceau pour de la classification en genre/mood...
- Instrument de musique pour des applications de transcription sur partition.

Exemple : Si des morceaux d'un même artiste apparaîssent dans la base d'entraînement et la base de test, il est possible que le classifieur s'appuie partiellement sur l'identification de l'artiste pour déterminer le genre. Les résultats obtenues peuvent alors être un trop optimiste.

# Apprentissage multimodal

L'audio apporte de l'information mais d'autres sources sont également exploitables :

- Images (couvertures d'albums, photos d'artistes...)
- Texte (paroles, avis/commentaires utilisateur).
- Données d'usage (organisation en playlist, session d'écoutes).

# Apprentissage multimodal : couverture d'albums

# Apprentissage multimodal : couverture d'albums

# Apprentissage multimodal : couverture d'albums

# Apprentissage multimodal : couverture d'albums

# Apprentissage multimodal : avis

*"On parle de quelqu'un qui avant de penser à sois-même pense d'abord à ces fans en leur faisant cadeaux d'album. Oui pour certains <Nom de l'artiste> ces pas chanter ni rapper mais qu'est ce qu'on s'en fiche les goûts et les couleurs ne se discutent pas" (commentaire Amazon)*

# Apprentissage multimodal : avis

*"On parle de quelqu'un qui avant de penser à sois-même pense d'abord à ces fans en leur faisant cadeaux d'album. Oui pour certains Jul ces pas chanter ni rapper mais qu'est ce qu'on s'en fiche les goûts et les couleurs ne se discutent pas" (commentaire Amazon)*

# Apprentissage multimodal : avis

*"Le premier atout de cette intégrale se situe dans la direction d'orchestre d'Antal Dorati, vive, énergique, et même percutante, dès "L'Ouverture" de l'opéra et lors des transitions orchestrales entre les Actes. Il sait aussi parfaitement mener les différents chœurs, masculins ou féminins : ainsi, au début de l'Acte I (marins norvégiens), puis à celui de l'Acte II ("Chœur des Fileuses"), et enfin au tout début de l'Acte III (chœurs des marins et des jeunes femmes norvégiens, "tuilés" avec ceux des marins hollandais)." (commentaire Amazon)*

# Apprentissage multimodal : avis

*"Le premier atout de cette intégrale se situe dans la direction d'orchestre d'Antal Dorati, vive, énergique, et même percutante, dès "L'Ouverture" de l'opéra et lors des transitions orchestrales entre les Actes. Il sait aussi parfaitement mener les différents chœurs, masculins ou féminins : ainsi, au début de l'Acte I (marins norvégiens), puis à celui de l'Acte II ("Chœur des Fileuses"), et enfin au tout début de l'Acte III (chœurs des marins et des jeunes femmes norvégiens, "tuilés" avec ceux des marins hollandais)."*
*(commentaire Amazon)*

# Apprentissage multimodal : paroles

*"No one of them is innocent*
*Smell the stench of their fear*
*Sequestred in the dark*
*Humiliation and torture*
*Destroy the human rights*
*Desecrate the creation of god*
*And when you lose the taste*
*When supplies don't your eyes anymore*
*End your misery life*
*Suicide for Satan"*

# Apprentissage multimodal : paroles

*"No one of them is innocent*
*Smell the stench of their fear*
*Sequestred in the dark*
*Humiliation and torture*
*Destroy the human rights*
*Desecrate the creation of god*
*And when you lose the taste*
*When supplies don't your eyes anymore*
*End your misery life*
*Suicide for Satan"*

From *Destroy your life for Satan* by Mütiilation

# Apprentissage multimodal : paroles

*"Je n'ai aucune peine, j'te nique ta race*
*Dans les veines je n'ai que de la glace*
*J'veux savoir c'que ça fait de prendre leur place*
*Je m'entraîne à sourire devant ma glace*
*La capitale dans le barillet*
*Tout arrive de Colombie akhi*
*J'rappe sale tellement avarié*
*Que même ces putains de rats attrapent la diarrhée*
*J'encule Brandon et Dylan*
*Si ces pédés crament au napalm j'veux la palme*
*Je m'en bats les couilles de qui rend l'âme*
*J'trempe mes cookies dans tes larmes"*

# Apprentissage multimodal : paroles

*"Je n'ai aucune peine, j'te nique ta race*
*Dans les veines je n'ai que de la glace*
*J'veux savoir c'que ça fait de prendre leur place*
*Je m'entraîne à sourire devant ma glace*
*La capitale dans le barillet*
*Tout arrive de Colombie akhi*
*J'rappe sale tellement avarié*
*Que même ces putains de rats attrapent la diarrhée*
*J'encule Brandon et Dylan*
*Si ces pédés crament au napalm j'veux la palme*
*Je m'en bats les couilles de qui rend l'âme*
*J'trempe mes cookies dans tes larmes"*

From *Zoo* by Kaaris

# Apprentissage multimodal

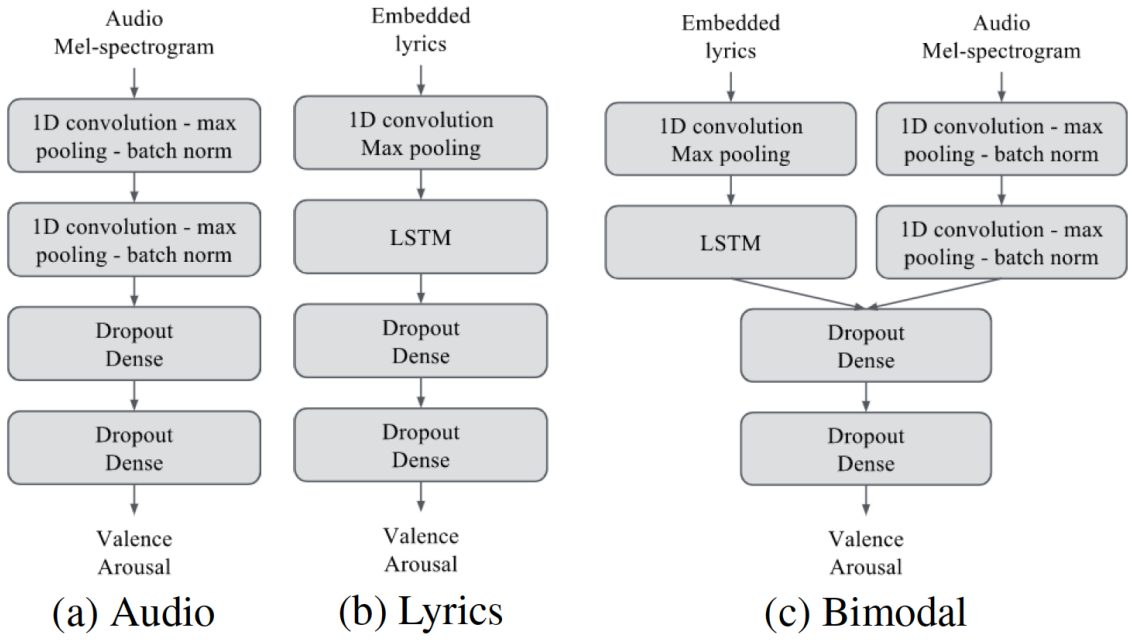
Fusion de modalités :

- Early fusion : fusion des features.
- Late fusion : fusion des sorties des classifieurs.

# Apprentissage multimodal

Fusion de modalités :

- Early fusion : fusion des features.
- Late fusion : fusion des sorties des classifieurs.
- Deep learning permet une fusion mi-niveau.




Source : Delbouys et al. *Music Mood Detection Based On Audio And Lyrics With Deep Neural Net*

# Conclusion

# Deep Learning

- Basé sur un empilement de couches de neurones artificiels.
- Peut modéliser de nombreux types de fonction complexe.
- Etat de l'art pour de nombreuses tâches de MIR.

# Limites du Deep Learning

- Nécessite une grande quantité de données annotées.
- Nécessite une forte puissance de calcul pour l'apprentissage.
- La compréhension des systèmes profonds est encore mal maîtrisée.
- Difficile de proposer des améliorations sur un système existant.
- Pas encore état de l'art en "end-to-end" pour certaines tâches : exemple détection de pulsation et de premier temps de mesure.