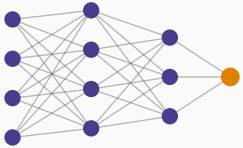
# Apprentissage profond

## Chapitre 7 : Initialisation et normalisation



Jaouad Madkour

Faculté d'Économie et de Gestion de Tanger · 27 août 2026

# Plan du chapitre

# L’initialisation

# Deux façons de rendre l'entraînement impossible

## Tout initialiser à zéro

Si tous les poids d'une couche valent zéro, tous les neurones calculent la **même** chose, reçoivent le **même** gradient, et sont mis à jour **identiquement**.

Ils resteront indistinguables pour toujours. La couche entière se comporte comme un neurone unique.

## Le problème de symétrie

Il faut donc **briser la symétrie** par un tirage aléatoire. Mais l'échelle de ce tirage n'est pas indifférente.

## Trop grand ou trop petit

Le chapitre 5 l'a établi : remonter $L$ couches multiplie $L$ facteurs.

Des poids initiaux **trop grands** font croître les activations couche après couche : saturation des activations, gradients qui explosent.

Des poids **trop petits** font décroître le signal : les couches profondes ne reçoivent presque rien.

Il faut donc une échelle telle que la **variance du signal se conserve** d'une couche à l'autre.

# Les deux initialisations usuelles

## Xavier et He

| Nom | Variance des poids | Pour |
|---|---|---|
| **Xavier** (Glorot) | $\frac{2}{n_{\text{entrée}} + n_{\text{sortie}}}$ | sigmoïde, tangente hyperbolique |
| **He** | $\frac{2}{n_{\text{entrée}}}$ | **ReLU** et variantes |

## Pourquoi He est plus large

ReLU annule la moitié des sorties en moyenne. Elle divise donc la variance transmise par deux.

L'initialisation de He compense exactement cette perte, en doublant la variance des poids. **C'est un ajustement calculé, non un réglage empirique.**

**Application en gestion**

Toutes les bibliothèques appliquent aujourd'hui une initialisation raisonnable par défaut, généralement adaptée à l'activation choisie.

Il est rare d'avoir à y toucher. Il faut en revanche savoir qu'un réseau qui n'apprend pas dès la première époque, ou dont la perte devient indéfinie, a souvent un problème de ce côté — ou de taux d'apprentissage.

# La normalisation des entrées

# Une étape préalable non négociable

## Centrer et réduire

Chaque variable d'entrée est ramenée à une moyenne nulle et un écart-type unitaire.

## Pourquoi c'est indispensable ici

Avec des variables d'échelles très différentes — un revenu en dizaines de milliers, un ratio entre zéro et un — la surface de la perte devient très allongée.

La descente de gradient y **zigzague** : le taux qui convient à une direction est trop grand ou trop petit pour l'autre. Normaliser rend la surface plus sphérique, et la descente directe.

C'est la même exigence qu'au chapitre 4 d'*Apprentissage statistique* pour ridge, et qu'au chapitre 11 pour les SVM.

## Le piège du calcul des statistiques

Moyennes et écarts-types doivent être calculés sur l'ensemble d'**apprentissage seulement**, puis appliqués tels quels à la validation et au test.

Les calculer sur toutes les données est une **fuite** — exactement le cas cité au chapitre 1 d'*Apprentissage statistique*. Elle est discrète, fréquente, et fausse toute l'évaluation.

# La normalisation par lot

# Normaliser aussi à l'intérieur du réseau

## Batch normalization

Normaliser les activations **à chaque couche**, sur les observations du lot courant, puis les remettre à l'échelle par deux paramètres appris.

## Ce qu'elle apporte

1. Entraînement **plus rapide** — on peut employer un taux d'apprentissage plus élevé.
2. **Moins de sensibilité** à l'initialisation.
3. Un effet **régularisant** léger, dû au bruit du lot.
4. Elle atténue l'évanouissement du gradient.

## Une différence à ne pas manquer

En **entraînement**, la normalisation utilise les statistiques du lot courant.

En **inférence**, elle utilise des moyennes accumulées pendant l'entraînement — sinon la prédiction dépendrait des autres observations du lot, ce qui n'aurait aucun sens.

Oublier de basculer le modèle en mode évaluation est une erreur classique, et elle produit des prédictions incohérentes d'un appel à l'autre.



## Les variantes

# Ce qu’il faut retenir

# Le résumé du chapitre

## Cinq points

1. Initialiser à **zéro** rend tous les neurones identiques : il faut briser la symétrie.
2. L'échelle doit **conserver la variance** du signal d'une couche à l'autre.
3. **He** pour ReLU — variance doublée pour compenser les sorties annulées ; **Xavier** pour les activations bornées.
4. **Normaliser les entrées** est indispensable ; les statistiques se calculent sur l'apprentissage **seulement**.
5. La **normalisation par lot** accélère l'entraînement ; **layer norm** pour les séquences.

## La suite

Tout le nécessaire est en place pour entraîner un réseau entièrement connecté. Les trois chapitres suivants introduisent des architectures **spécialisées**, en commençant par celle qui traite les images.