Séries Temporelles - cours 2

RCP217

Clément Rambour

Plan

Résumé du dernier cours

Statistical Forecasting

- **Motivation** : utiliser le contexte global
  - Utilisation de toute la série : débruitage
  - Utilisation du passé : prédiction
- Extraire les statistiques du processus sous réserve de stationnarité

Résumé du dernier cours

Prédiction progressive

$$\hat{x}_t = \boldsymbol{\varphi}^T \boldsymbol{x} \quad \text{avec} \quad \boldsymbol{\varphi} = \underset{\boldsymbol{\varphi}}{\text{argmin}}\, \mathbb{E}\left[(x_t - \boldsymbol{\varphi}^T \boldsymbol{x})^2\right]$$

où $\boldsymbol{x} = [x_{t-1}, \cdots, x_{t-p}]$

Résumé du dernier cours

Prédiction progressive

$$\hat{x}_t = \boldsymbol{\varphi}^T \boldsymbol{x} \quad \text{avec} \quad \boldsymbol{\varphi} = \underset{\boldsymbol{\varphi}}{\operatorname{argmin}}\, \mathbb{E}\left[(x_t - \boldsymbol{\varphi}^T \boldsymbol{x})^2\right]$$

où $\boldsymbol{x} = [x_{t-1}, \cdots, x_{t-p}]$

Algorithmes

- Résolution des équations de Yule-Walker
- Gram-Schmidt ou des innovations
- Levinson-Durbin

Résumé du dernier cours

Processus Auto Regréssif Moving Average (ARMA)

Un processus est ARMA d'ordre $(p, q) \in \mathbb{N}^2$ s'il est du stationnaire au sens large et solution de l'équation de récurrence

$$X_t = Z_t + \sum_{k=1}^{p} \varphi_k X_{t-k} + \sum_{k=1}^{q} \theta_k Z_{t-k}$$

où $\forall t, Z_t \sim \mathcal{N}(0, \sigma^2)$ et $\forall i, j, \varphi_i$ and $\theta_j \in \mathbb{R}$.

Résumé du dernier cours

Processus Auto Regréssif Moving Average (ARMA)

Un processus est ARMA d'ordre $(p, q) \in \mathbb{N}^2$ s'il est du stationnaire au sens large et solution de l'équation de récurrence

$$X_t = Z_t + \sum_{k=1}^{p} \varphi_k X_{t-k} + \sum_{k=1}^{q} \theta_k Z_{t-k}$$

où $\forall t, Z_t \sim \mathcal{N}(0, \sigma^2)$ et $\forall i, j, \varphi_i$ and $\theta_j \in \mathbb{R}$.

Existence, causalité et inversion

- Le polynôme $\Phi(z) = \sum_{k=1}^{p} \varphi_k z^k$ n'a pas de racine de module 1
- Si $\Phi(z)$ n'a pas de racine $|z| < 1$ alors la solution est causale
- Si $\Theta(z)$ n'a pas de racine $|z| < 1$ alors la solution est inversible
- Estimation des paramètres par Least Square or Maximum Likelihood

Maintenant

- Système d'état et Filtre de Kalman

Maintenant

- Système d'état et Filtre de Kalman
- Comment comparer des séries temporelles ?
  - Distance
  - Similarité
  - Sur les données brut ou sur des *features*
- Apprentissage profond et séries temporelles
  - Approches auto-regressives
  - Réseaux récurrents
  - RNN Encodeurs-Décodeurs
  - Méchanismes attentionnels
  - Modèles hybrides

Plan

Principe

- Estimer l'état d'un système à partir de mesures incomplètes
- Dans une problématique de prédiction : état = valeurs suivantes du processus
- Pouvoir estimer l'erreur
- Pouvoir générer des estimations online (de façon récursive)

Intuition

L'idée à la base du filtre de Kalman se décompose en deux étapes

Prédiction

D'après un modèle (physique, statistique, signal...), on émet une prédiction

Correction

Une fois la mesure disponible, on règle un paramètre d'ajustement

Pré-requis

Estimateur linéaire non biaisé

Soit $\boldsymbol{x}$ et $\boldsymbol{y}$, deux vecteurs aléatoires. Il existe un unique estimateur orthogonal linéaire non biaisé $\hat{\boldsymbol{x}}$ à partir de $\boldsymbol{y}$ donné par :

$$\hat{\boldsymbol{x}} = \mathbb{E}(\boldsymbol{x}) + \boldsymbol{K}(\boldsymbol{y} - \mathbb{E}(\boldsymbol{y}))$$

$$\boldsymbol{K} = \boldsymbol{\Gamma}_{xy}\boldsymbol{\Gamma}_{y}^{-1}$$

Équations d'état

$$\begin{cases} x_t &= \boldsymbol{A}\boldsymbol{x}_{t-1} + \boldsymbol{u}_{t-1} + \boldsymbol{w}_t \\ y_t &= \boldsymbol{C}\boldsymbol{x}_t + \boldsymbol{n}_t \end{cases}$$

- $\boldsymbol{w}_t$ et $\boldsymbol{n}_t$: bruits blanc gaussiens
- $\boldsymbol{x}_t$: état du système (position, vitesse, valeur boursière, potentiel...)
- $\boldsymbol{u}_t$: commande appliquée sur le système (déterministe)
- $\boldsymbol{y}_t$: mesure du système (capteurs, vidéos...)
- Les matrices $\boldsymbol{A}$ et $\boldsymbol{C}$ peuvent également varier dans le temps mais sont supposées connues

Filtre de Kalman

Objectif : obtenir la meilleure façon d'estimer $\boldsymbol{x}_t$ à partir de l'estimation précédente $\hat{\boldsymbol{x}}_{t-1}$ et de la mesure courante $\boldsymbol{y}_t$

Intuition

En prenant $\hat{\boldsymbol{x}}_{t-1}$ comme un estimateur de la moyenne de $\boldsymbol{x}_t$, on peut faire l'estimation suivante :

$$\hat{\boldsymbol{x}}_t = \hat{\boldsymbol{x}}_{t-1} + \boldsymbol{K}_t(\boldsymbol{y}_t - \boldsymbol{C}\hat{\boldsymbol{x}}_{t-1})$$

où la notation $t|t-1$ correspond à l'estimation de n sachant $[x_1, \cdots, x_{t-1}]$.

Filtre de Kalman

Prédiction

$$\hat{\boldsymbol{x}}_{t|t-1} = \boldsymbol{A}\hat{\boldsymbol{x}}_{t-1|t-1} + \boldsymbol{u}_{t-1}$$
$$\boldsymbol{\Gamma}_{t|t-1} = \boldsymbol{A}\boldsymbol{\Gamma}_{xt-1|t-1}\boldsymbol{A}^T + \boldsymbol{\Gamma}_w$$

Gain de Kalman

$$\boldsymbol{S}_t = \boldsymbol{C}\boldsymbol{\Gamma}_{t|t-1}\boldsymbol{C}^T + \boldsymbol{\Gamma}_n$$
$$\boldsymbol{K}_t = \boldsymbol{\Gamma}_{t|t-1}\boldsymbol{C}^T\boldsymbol{S}_t^{-1}$$

Correction

$$\hat{\boldsymbol{x}}_{t|t} = \hat{\boldsymbol{x}}_{t|t-1} + \boldsymbol{K}_t(\boldsymbol{y}_t - \boldsymbol{C}\hat{\boldsymbol{x}}_{t|t-1})$$
$$\hat{\boldsymbol{\Gamma}}_{t|t} = (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{C})\boldsymbol{\Gamma}_{t|t-1}$$

Plan

Comparaison de ST

Comment comparer deux séries temporelles ?

Problèmes courants

De nombreuses causes peuvent rendre cette tâche difficile :

- Translations
- Différence d'échelle (*scaling*)
- Décalages (*shift*)
- Longueur
- Occlusions
- Complexité

Exemples

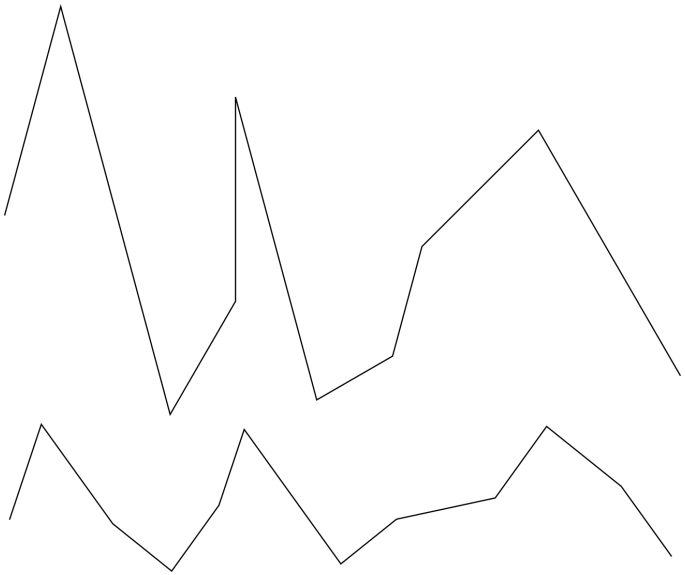


Figure – échelle

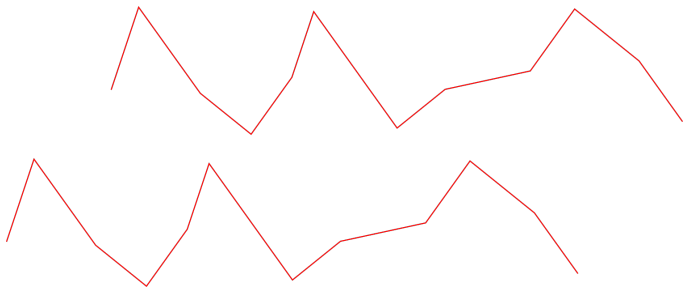


Figure – translation

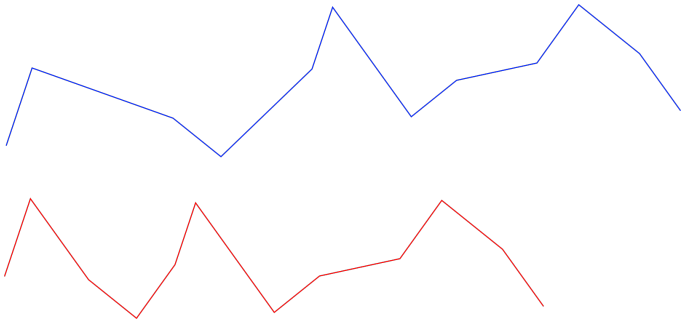


Figure – décalage

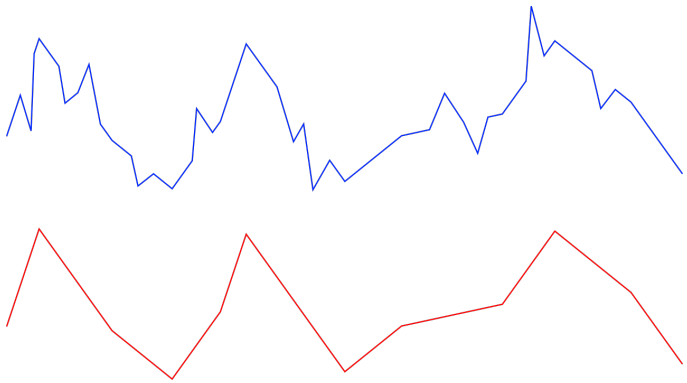


Figure – complexité

Beyond Raw Data

Features

Une première solution peut être de trouver une représentation compacte de taille fixe

- ARMA/ARIMA coefficients
- coefficients spectraux : Discrete Cosine Transform / Discrete Fourier Transform
- Décomposition en ondelette

On peut toujours indiquer le nombre coefficient max ou l'ordre max du modèle ARMA que l'on considère.

Beyond Raw Data

Représentations

Une autre approche est d'avoir des valeurs normalisées

- Autocorrélogramme
- Indicateur des minima ou maxima

Distances ℓ_1 et ℓ_2

Si l'on souhaite comparer les données brutes, on peut s'intéresser aux distances associées aux normes ℓ_1 et ℓ_2

Définition : distance euclidienne et de Manhattan

Soient deux séries temporelles $\{x_t\}_{0\leq t\leq T}$ et $\{y_t\}_{0\leq t\leq T}$, on définit

- la distance euclidienne ou ℓ_2 par :

$$\ell_2 = \sqrt{\sum_{t=1}^{N}(x_t - y_t)^2}$$

- la distance de Manhattan ou ℓ_1 par :

$$\ell_1 = \sum_{t=1}^{N}|x_t - y_t|$$

Distances ℓ_1 et ℓ_2

- Facile à interpréter
- Rapide à implémenter et calculer
- Différentiable

- Nécessite des prétraitements pour palier la variance à l'échelle, la translation, aux décalages ou à la longueur
- Information point à point : pas de notion de forme

Comparaison de forme

Cross-Corrélation : rappel

Soit la cross-corrélation entre deux times series $\{x_t\}_{0\leq t\leq T}$ et $\{y_t\}_{0\leq t\leq T}$ de moyenne respectives μ_x et μ_y :

$$c_{xy}(h) = \frac{\mathbb{E}[(x_t - \mu_x)(y_{t+h} - \mu_y)]}{\sqrt{\mathbb{E}[(x_t - \mu_x)^2]\mathbb{E}[(y_t - \mu_y)^2]}} = \frac{\gamma_{xy}(h)}{\sqrt{\gamma_{xx}(0)\gamma_{yy}(0)}}.$$

$c_{xy}(h)$ donne une information de dépendance linéaire entre $\{x_t\}_{0\leq t\leq T}$ et $\{y_t\}_{0\leq t\leq T}$:

- Si $c_{xy}(h) = 1$, alors $\exists\, \alpha, \beta \in \mathbb{R}$ tels que $x_t = \alpha y_{t+h} + \beta$
- Si $c_{xy}(h) = -1$, alors $\exists\, \alpha, \beta \in \mathbb{R}$ tels que $x_t = \alpha(\mu_y - y_{t+h}) + \beta$
- Si $c_{xy}(h) = 0$, les deux séries sont indépendantes pour un lag de h

La cross-corrélation est généralement estimée par la cross-corrélation empirique $\{\hat{c}_{xy}(h)\}$ sur un ensemble de N échantillons compris dans des fenêtres glissantes décallée d'un lag de h.

Comparaison de forme

Cross-Corrélation : rappel

Soit la cross-corrélation entre deux times series $\{x_t\}_{0\leq t\leq T}$ et $\{y_t\}_{0\leq t\leq T}$ de moyenne respectives μ_x et μ_y :

$$c_{xy}(h) = \frac{\mathbb{E}[(x_t - \mu_x)(y_{t+h} - \mu_y)]}{\sqrt{\mathbb{E}[(x_t - \mu_x)^2]\mathbb{E}[(y_t - \mu_y)^2]}} = \frac{\gamma_{xy}(h)}{\sqrt{\gamma_{xx}(0)\gamma_{yy}(0)}}.$$

$c_{xy}(h)$ donne une information de dépendance linéaire entre $\{x_t\}_{0\leq t\leq T}$ et $\{y_t\}_{0\leq t\leq T}$:

- Si $c_{xy}(h) = 1$, alors $\exists\, \alpha, \beta \in \mathbb{R}$ tels que $x_t = \alpha y_{t+h} + \beta$
- Si $c_{xy}(h) = -1$, alors $\exists\, \alpha, \beta \in \mathbb{R}$ tels que $x_t = \alpha(\mu_y - y_{t+h}) + \beta$
- Si $c_{xy}(h) = 0$, les deux séries sont indépendantes pour un lag de h

La cross-corrélation est généralement estimée par la cross-corrélation

Équivalent à une similarité cosinus pour le produit scalaire induit par $< x, y >= \mathbb{E}[xy]$ définie sur des séries de moyennes nulles.

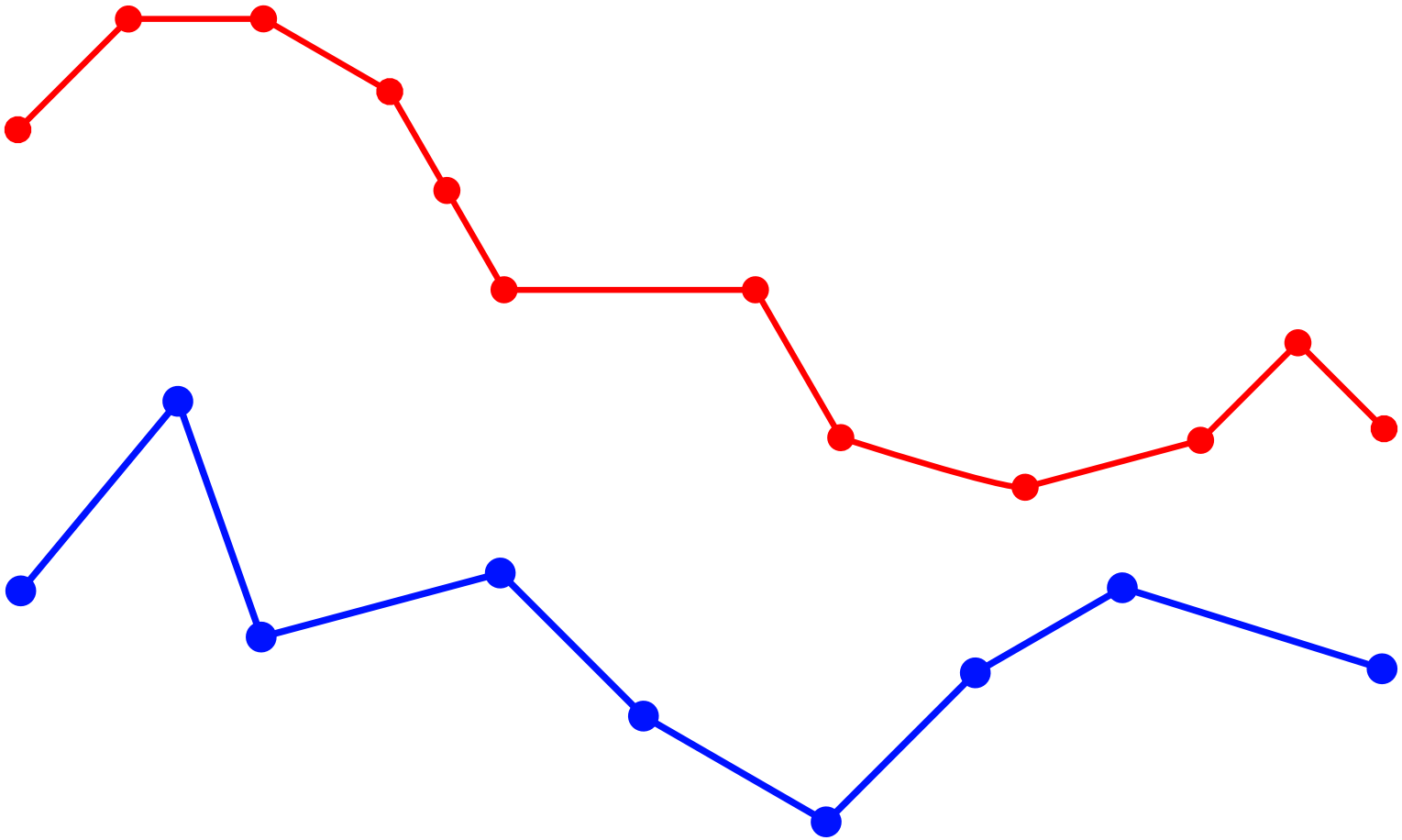
Corrélation

- Facile à interpréter
- Donne une notion de forme
- Robuste aux scaling et à la complexité.

- Pas invariant aux translations ou shifts
- Nécessite un nombre d'échantillons N suffisamment grand pour être calculé mais complexité quadratique en $N \Rightarrow$ long à calculer.
- Problème aux bords : nécessite une stratégie de *padding*
- Information purement de forme : indépendant à la différence de magnitude entre les séries.

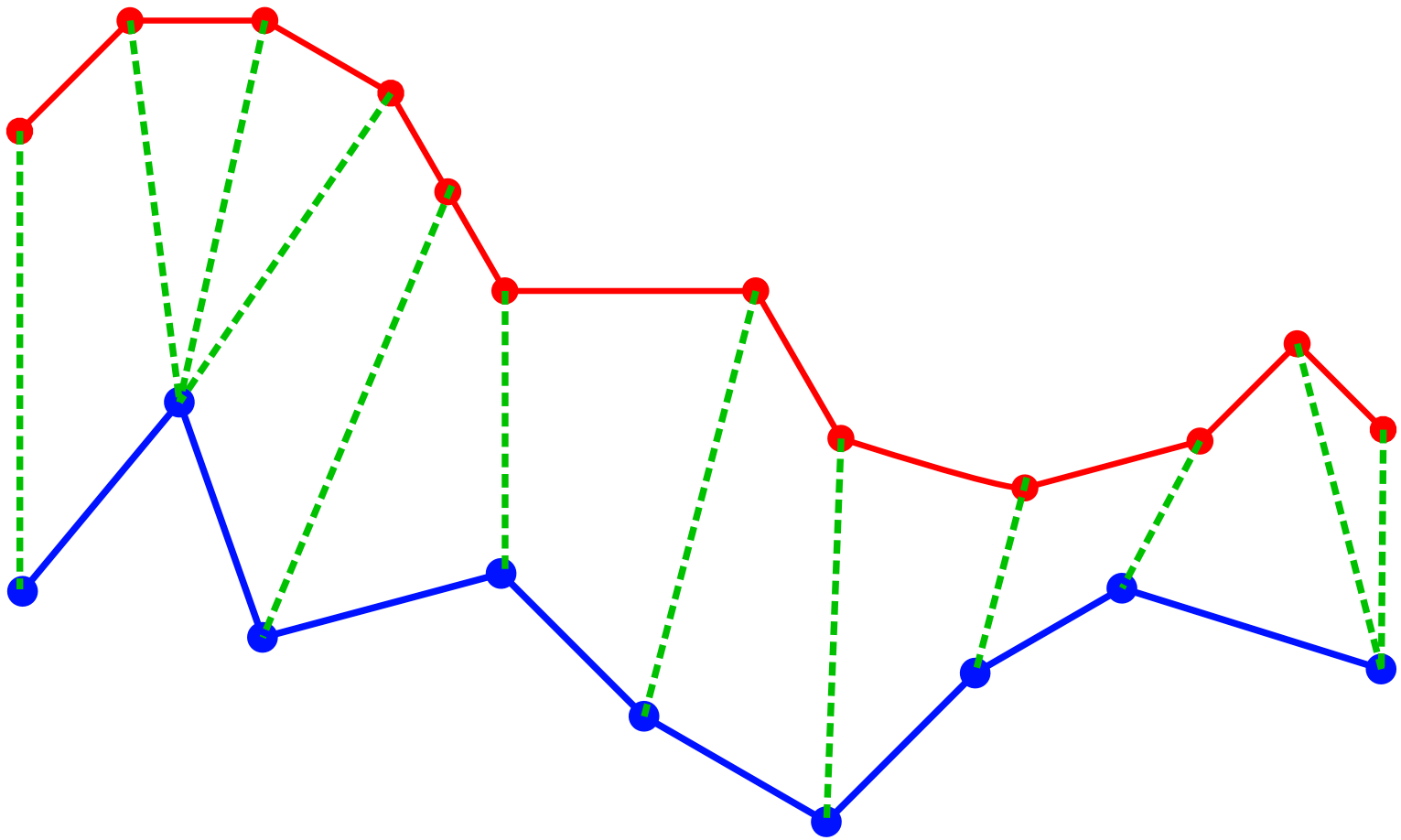
Dynamic Time Warping

Motivation : S'affranchir des problèmes mentionnés précédemment
Intuition : Relier chaque point avec son plus proche voisin dans la série comparée



Dynamic Time Warping

Motivation : S'affranchir des problèmes mentionnés précédemment et garder une information de forme et de distance. **Intuition** : Relier chaque point avec son plus proche voisin dans la série comparée pour calculer une "distance"



Dynamic Time Warping

Quelques contraintes dans la comparaison des séries :

- Chaque élément de $\{x_t\}$ doit être associé avec un élément de $\{y_t\}$
- Les premiers et derniers éléments de $\{x_t\}$ et $\{y_t\}$ doivent être liés (pas forcément leur seule association)
- Les appariements ne doivent pas se "croiser" : si $i < j$ sont des indices de $\{x_t\}$ alors il ne doit pas exister $k < l$ tels que y_l soit associé à x_i et y_k soit associé à x_j

En respectant ces considérations DTW$(\boldsymbol{x}, \boldsymbol{y})$ peut être calculé de façon itérative :

Input: $\boldsymbol{x} = \{x_t\}_{0 \leq t \leq T_1}, \boldsymbol{y} = \{y_t\}_{0 \leq t \leq T_2}$, distance function δ
$D_{0,0} = 0$, and $\forall (i,j) \in \{1, ..., T_1\} \times \{1, ..., T_2\}, D_{i,0} = D_{0,j} = \infty$
for $j = 1, ..., T_2$ **do**
  for $i = 1, ..., T_1$ **do**
    $D_{i,j} = \delta(x_i, y_j) + \min\{D_{i-1,j-1}, D_{i-1,j}, D_{i,j-1}\}$
  end
end
Output: DTW$(\boldsymbol{x}, \boldsymbol{y}) = \mathrm{D}_{n,m}$

Dynamic Time Warping

On calcule donc itérativement les coefficients d'une matrice d'alignement $\boldsymbol{D}$:

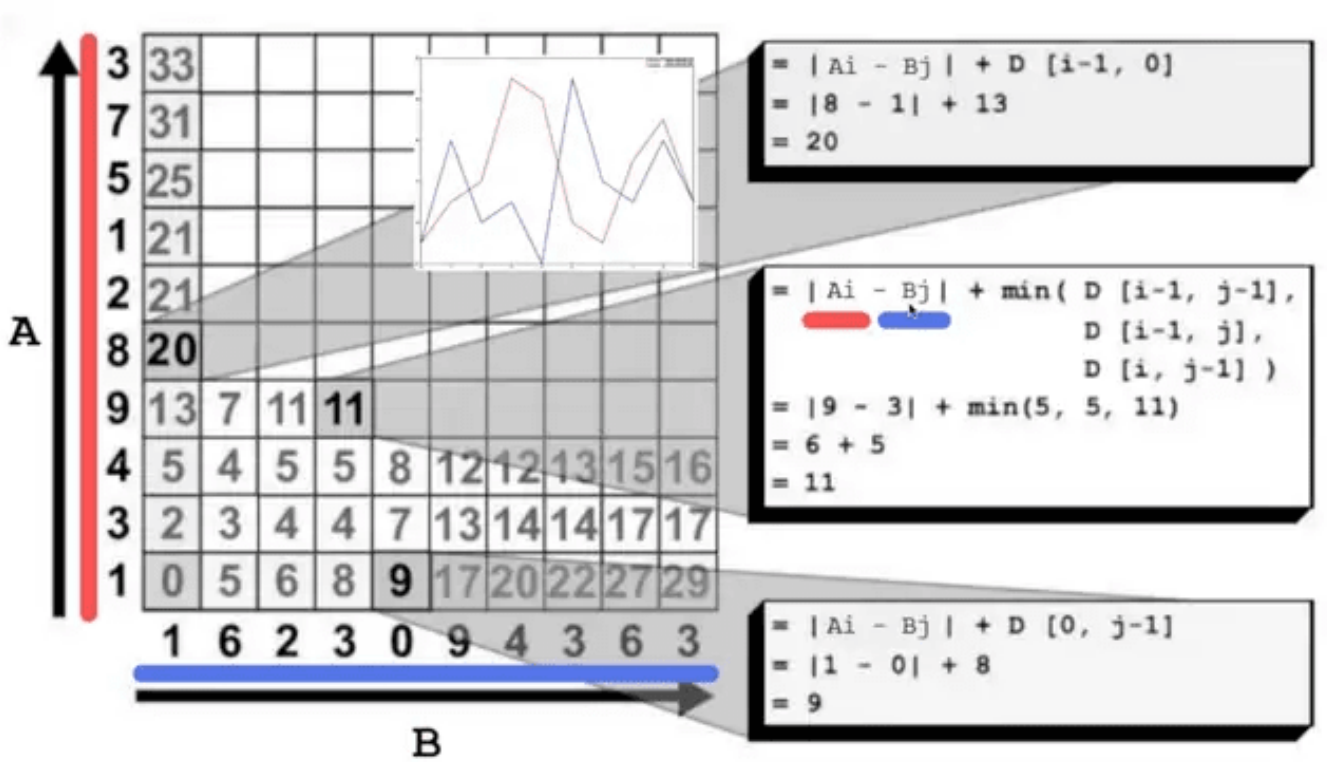


Figure – Thales Sehn Körting - DTW vidéo

Soft Dynamic Time Warping

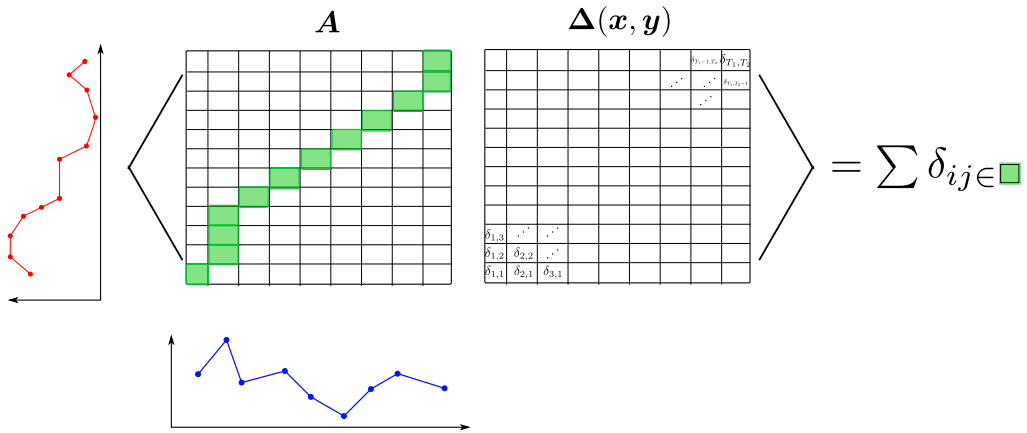
Problème DTW n'est pas dérivable mais on peut l'approcher par une version *soft-DTW*

- Version soft du *min* :

$$\min^{\gamma}\{x_1, ..., x_T\} = \begin{cases} \min\{x_1, ..., x_T\} \text{ si } \gamma = 0 \\ -\gamma \log \sum_{t=1}^{T} e^{-x_i/\gamma} \text{ sinon} \end{cases}$$

- Donne le *soft-DTW* :

$$DTW_{\gamma}(\boldsymbol{x}, \boldsymbol{y}) = \min^{\gamma} < \boldsymbol{A}, \Delta(\boldsymbol{x}, \boldsymbol{y}) >, \boldsymbol{A} \in \mathcal{A}$$



Soft Dynamic Time Warping : Dérivation

Soft-DTW :
Input: $\boldsymbol{x} = \{x_t\}_{0 \leq t \leq T_1}, \boldsymbol{y} = \{y_t\}_{0 \leq t \leq T_2}$, distance function δ
$\boldsymbol{D}_{0,0} = 0$, and $\forall (i,j) \in \{1, ..., T_1\} \times \{1, ..., T_2\}, \boldsymbol{D}_{i,0} = D_{0,j} = \infty$
for $j = 1, ..., T_2$ **do**
  for $i = 1, ..., T_1$ **do**
    $D_{i,j} = \delta(x_i, y_j) + \min^{\gamma}\{\boldsymbol{D}_{i-1,j-1}, \boldsymbol{D}_{i-1,j}, \boldsymbol{D}_{i,j-1}\}$
  end
end
Output: DTW($\boldsymbol{x}$, $\boldsymbol{y}$) = $\boldsymbol{D}_{n,m}$

$\boldsymbol{D}_{n,m}$ dépend de $\boldsymbol{D}_{i,j}$ seulement à travers $\boldsymbol{D}_{i-1,j}, \boldsymbol{D}_{i,j-1}$ et $\boldsymbol{D}_{i-1,j-1}$.
La *chain rule* donne :

$$\frac{\partial \boldsymbol{D}_{n,m}}{\partial \boldsymbol{D}_{i,j}} = \frac{\partial \boldsymbol{D}_{n,m}}{\partial \boldsymbol{D}_{i-1,j}} \frac{\partial \boldsymbol{D}_{i-1,j}}{\partial \boldsymbol{D}_{i,j}} + \frac{\partial \boldsymbol{D}_{n,m}}{\partial \boldsymbol{D}_{i,j-1}} \frac{\partial \boldsymbol{D}_{i,j-1}}{\partial \boldsymbol{D}_{i,j}} + \frac{\partial \boldsymbol{D}_{n,m}}{\partial \boldsymbol{D}_{i-1,j-1}} \frac{\partial \boldsymbol{D}_{i-1,j-1}}{\partial \boldsymbol{D}_{i,j}}$$

Plan

Deep Learning strategies

Architecture

- Feed Forward Network ou Convolutional Neural Networks : sous-séquence de taille τ fixe $\{x_t\}_{T-\tau \leq t \leq T}$ en entrée
- Recurrent Neural Networks : le modèle conserve un vecteur d'état en mémoire
- Modèles attentionnels : toute l'information passée est utilisée pour biaiser les prédictions

Deep Learning strategies

Architecture

- Feed Forward Network ou Convolutional Neural Networks : sous-séquence de taille τ fixe $\{x_t\}_{T-\tau \leq t \leq T}$ en entrée
- Recurrent Neural Networks : le modèle conserve un vecteur d'état en mémoire
- Modèles attentionnels : toute l'information passée est utilisée pour biaiser les prédictions

Nature des entrées

- (Monovariée) $x \in \mathbb{R}^{1 \times T}$
- (Multivariée) $\boldsymbol{x} \in \mathbb{R}^{d \times T}$
- Prédiciton conditonnelle : la prédiction est basée sur la série d'entrée ainsi que d'autres paramètres

Deep Learning strategies

Architecture

- Feed Forward Network ou Convolutional Neural Networks : sous-séquence de taille τ fixe $\{x_t\}_{T-\tau \leq t \leq T}$ en entrée
- Recurrent Neural Networks : le modèle conserve un vecteur d'état en mémoire
- Modèles attentionnels : toute l'information passée est utilisée pour biaiser les prédictions

Nature des entrées

- (Monovariée) $x \in \mathbb{R}^{1 \times T}$
- (Multivariée) $\boldsymbol{x} \in \mathbb{R}^{d \times T}$
- Prédiciton conditonnelle : la prédiction est basée sur la série d'entrée ainsi que d'autres paramètres

Nature des prédictions

- Estimation $n+1$: une seule trajectoire future envisagée pour les $n+1$ prochains évènements
- Multi-horizon : plusieurs futurs considérés
- Probabilistic outputs : prédiction stochastique donnant variabilité et incertitude des prédictions. Possibilité d'échantillonner différents futurs $\Rightarrow$ des simulations.

Réseaux linaires et convolutifs

Modèles convolutifs : taille de la série d'entrée fixée ⇒ la prédiction dépend d'un passé relativement proche

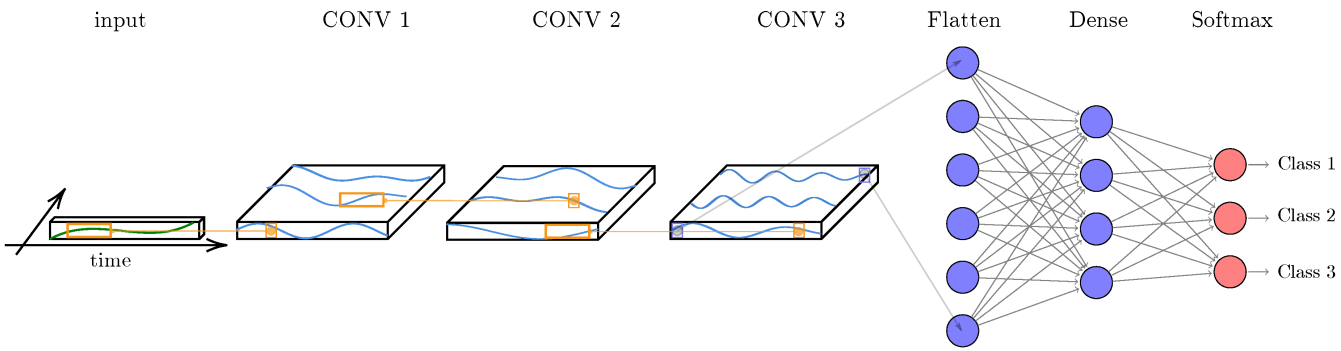


Figure – C. Pelletier et Al. 2019

La dépendance dans le passé peut être étendue en choisissant des convolutions dilatées

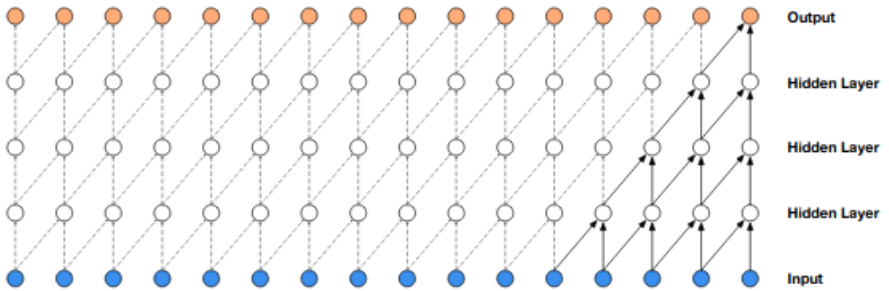


Figure – Van Den Oord et Al. 2016

Réseaux linaires et convolutifs

Modèles convolutifs : taille de la série d'entrée fixée ⇒ la prédiction dépend d'un passé relativement proche

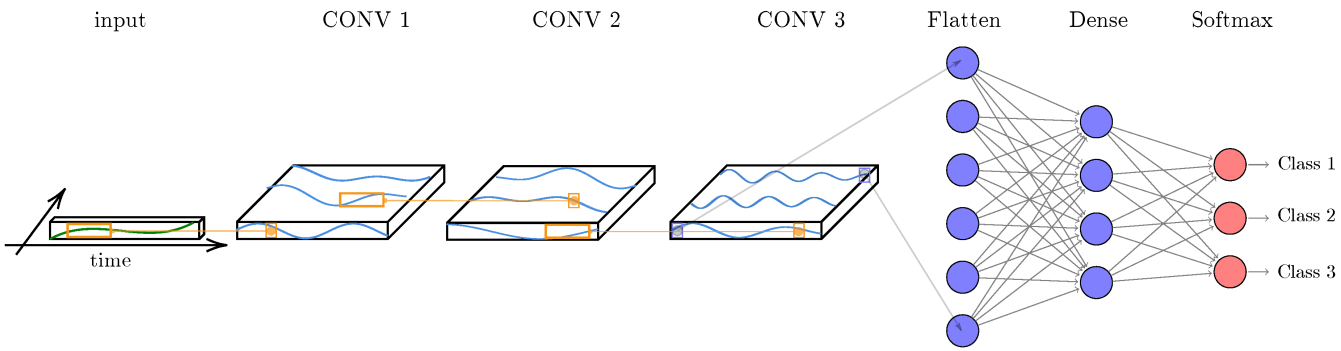


FIGURE – C. Pelletier et Al. 2019

La dépendance dans le passé peut être étendue en choisissant des convolutions dilatées

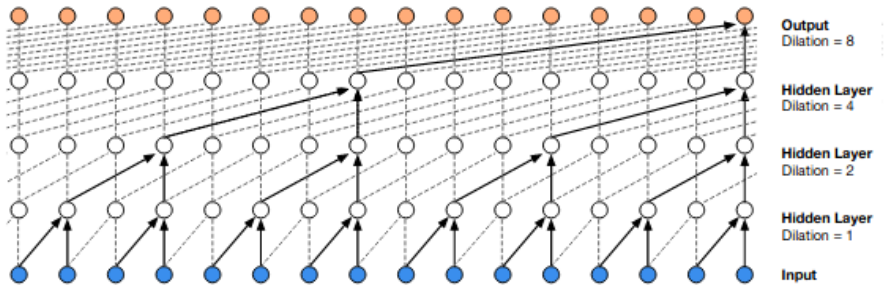


FIGURE – Van Den Oord et Al. 2016

Réseaux linaires et convolutifs

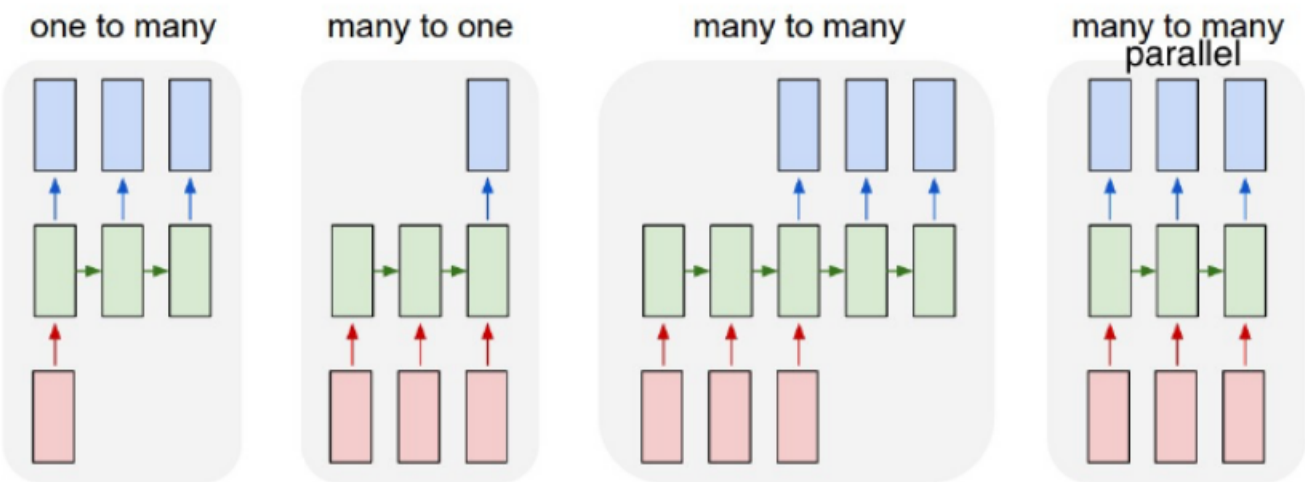
- CNN plus stable, meilleure estimation locale et plus compact que FCN

- Taille des entrées limitée
- Décisions indépendantes entre différents pas
- Ne peut pas prendre des séquences de taille variable

RNN

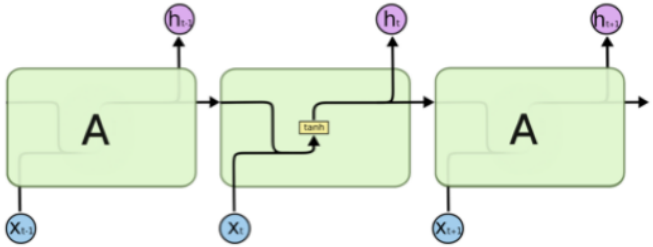
Différentes correspondances (*mapping*) pour différentes tâches

- one-to-many : image captioning
- many-to-one : prédiction
- many-to-many : traduction, sous-titrage



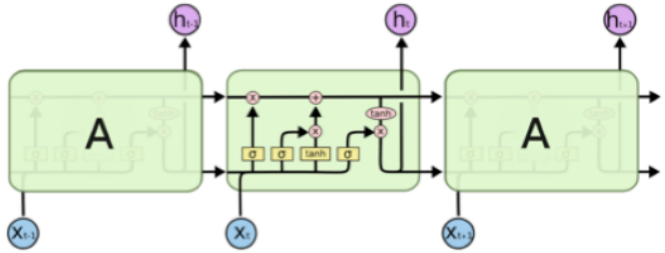
Vanilla RNN, LSTM et GRU

- Recap: Vanilla RNN cell



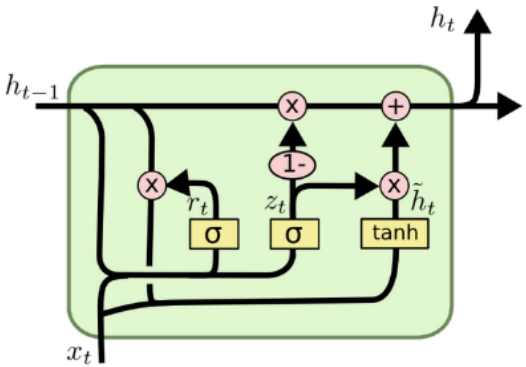
$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

- Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997]



$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$
$$c_t = f \odot c_{t-1} + i \odot g$$
$$h_t = o \odot \tanh(c_t)$$

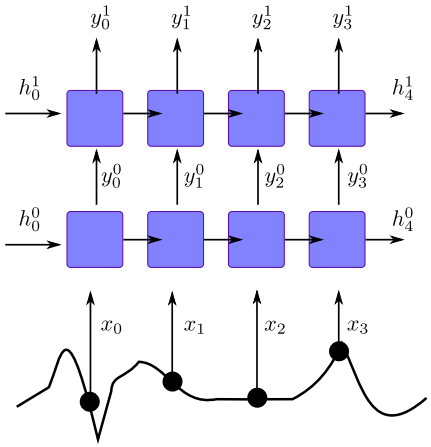
Credit: C. Olah



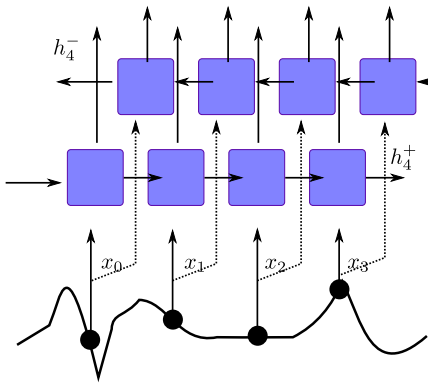
$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

RNN

- Comme pour les CNN, il est possible d'empiler des blocs récurrents
- La sortie d'un bloc à un niveau est transmise en entrée du bloc correspondant au niveau suivant

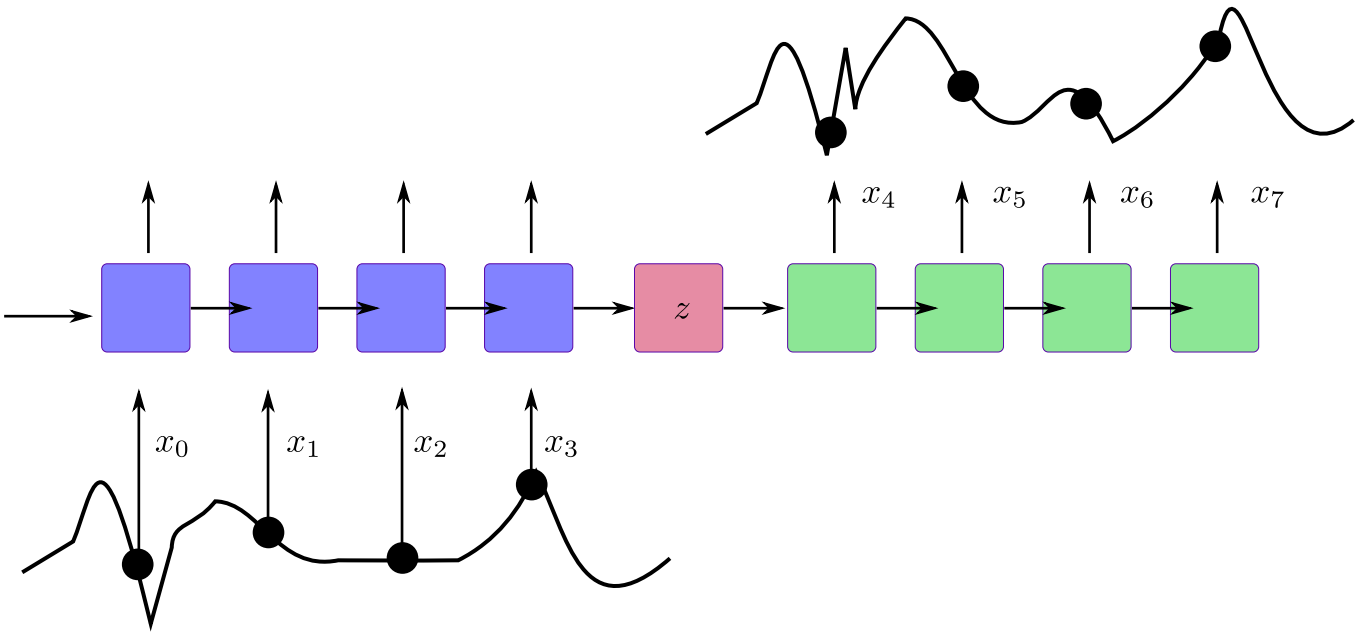


- Plus de contexte peut être obtenu si le réseau parcourt la séquence dans les deux sens
- On a alors deux vecteurs contextuels h^+ et h^-



seq2seq

- L'architecture en *encoder-decoder* est généralisable aux RNN
- L'encodage se fait à travers le vecteur caché transmis à chaque étape de lecture de la séquence.
- Le décodage est effectué à partir du contexte global
- Dans une optique de prédiction, la dernière entrée peut aussi servir d'entrée au décodeur



Mechanismes attentionnels

Problème des seq2seq :

- Compression d'information très contraignante
- Difficile de trop augmenter la taille du vecteur caché

Attention

- Utiliser tous les états cachés
- Pondérer leur utilisation en fonction de leur utilité

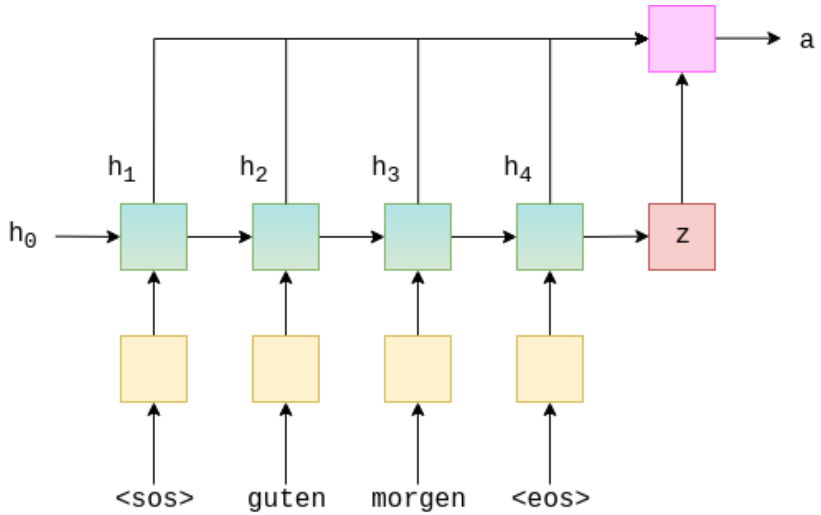


Figure – Ref : https ://github.com/bentrevett/pytorch-seq2seq

Transformers

- Architecture complètement connectée
- Toute l'information est mise en correspondance avec elle-même
- Permet de repondérer les données en fonction de leur redondance et intérêt pour une tâche donnée

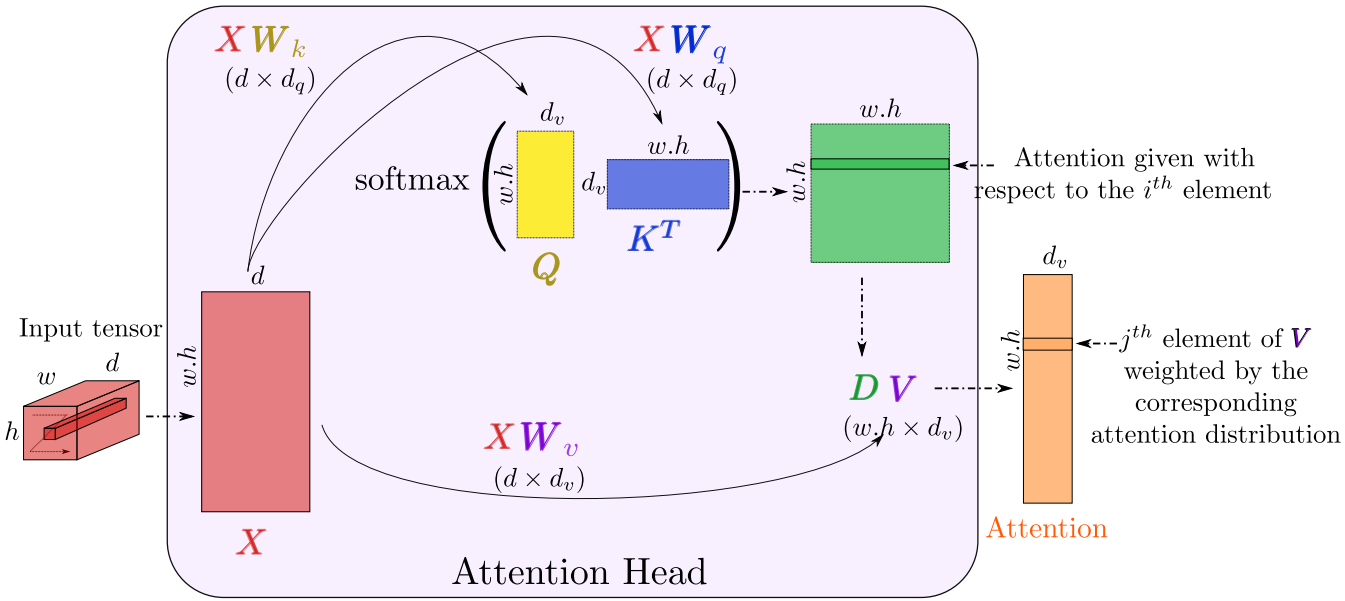


FIGURE – Ref : U-Net Transformer, Petit O et al., MLMI 2021

Forecasting Loss

Losses

- Quantized loss :

$$E(\hat{x}) = -\frac{1}{T}\sum_{k=1}^{T}\log(\hat{x}_t)$$

- Regression loss :

$$E(x, \hat{x}) = \frac{1}{T}\sum_{k=1}^{T}(x_{t+k} - \hat{x}_{t+k})^2$$

Métriques

Absolute Error

- Mean Absolute Error :

$$E(x, \hat{x}) = \frac{1}{T} \sum_{k=1}^{T} |x_{t+k} - \hat{x}_{t+k}|$$

- Max Absolute Error :

$$E(x, \hat{x}) = \max_{k \in \{1,\dots,T\}} |x_{t+k} - \hat{x}_{t+k}|$$

Plan

Déterministe vs. Stochastique

Prévision

- **Déterministe** Le réseaux donne en sortie la prédiction jusqu'à un certain horizon h :

$$\hat{\boldsymbol{x}}_{T+h} = f_\theta(\boldsymbol{x}_{T-l:T})$$

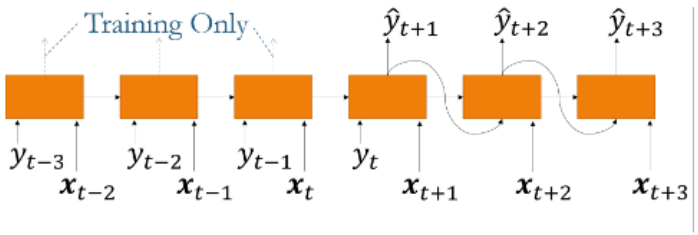
- **Stochastique** Le réseau donne en sortie les paramètres d'une loi de distribution jusqu'à un certain horizon h :

$$\hat{\boldsymbol{x}}_{T+h} \sim \mathcal{N}(\boldsymbol{\mu}_{T+h}, \boldsymbol{\Sigma}_{T+h})$$
$$(\boldsymbol{\mu}_{T+h}, \boldsymbol{\Sigma}_{T+h}) = f_\theta(\boldsymbol{x}_{T-l:T})$$

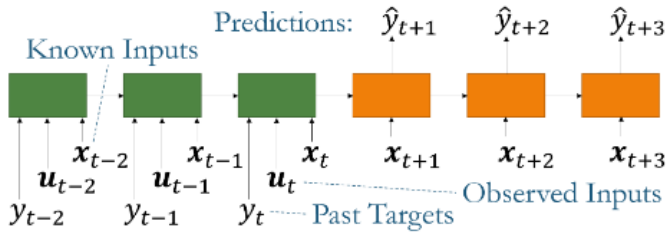
Horizon

Deux choix **close loop** ou **open loop**

- Close loop ou iterative : on prédit de façon autoregressive en suivant les valeurs déjà prédites
- Open loop ou directe : on prédit directement sur plusieurs pas de temps



(a) Iterative Methods



(b) Direct Methods

Modèles hybrides

Certaines approches combinent des méthodes statistiques précédemment développées avec de l'apprentissage.

Rappel Holt-Winters

- Approximation $x_t \simeq L_t S_t$
- Estimation des composantes par :

$$L_t = \alpha \frac{x_t}{S_{t-\tau}} + (1-\alpha)\left(L_{t-1} + T_{t-1}\right)$$
$$T_t = \gamma\left(L_t - L_{t-1}\right) + (1-\gamma) T_{t-1}$$
$$S_t = \delta \frac{x_t}{L_t} + (1-\delta) S_{t-\tau}$$

- Prédiction donnée par $\hat{x}_{t+h} = \left(L_t + hT_t\right) S_{t-\tau+h}$ avec τ la période

Slawek Smyl, A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting, International Journal of Forecasting, 2020

Modèles hybrides

Exponential Smoothing-Recurrent Neural Networks (ES-RNN)

ES-RNN tire parti de la simplicité et des bonnes performances du forecasting Holt-Winters.

- On garde $x_t \simeq L_t S_t$

$$L_t = \alpha \frac{x_t}{S_{t-\tau}} + (1-\alpha)(L_{t-1})$$

$$S_t = \delta \frac{x_t}{L_t} + (1-\delta) S_{t-\tau}$$

- Traitement des données :

$$x_t \leftarrow \log(\frac{x_t}{L_t * S_t})$$

- Prédiction donnée par :

$$\hat{x}_{t+h} = \exp f_\theta(x_{t-l:T}) L_t S_{t-\tau+h}$$

- Pinball loss pour diminuer le biais positif introduit par le log :

$$\mathcal{L}(x_t, \hat{x}_t) = \begin{cases} (x_t - \hat{x}_t) \cdot \epsilon \text{ si } x_t > \hat{x}_t \\ (\hat{x}_t - x_t) \cdot (1-\epsilon) \text{ sinon} \end{cases}$$

Slawek Smyl, A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting, International Journal of Forecasting, 2020

Modèles hybrides

- Holt-Winters + RNN
- Deep states models
- Deep Kalman
- Estimation d'un modèle SARIMA par un CNN/RNN
- ...

Références

- *The Analysis of Time Series* Chris Chatfield, Chapman & Hall/CRC, 2004
- Soft-DTW : a Differentiable Loss Function for Time-Series, Marco Cuturi, Mathieu Blondel, 2017
- A benchmark study on time series clustering, Javed et al., 2020
- Time Series Forecasting With Deep Learning : A Survey, Bryan Lim and Stefan Zohren, 2020