UNIVERSITÉ LAVAL

GLO-4030/7030
APPRENTISSAGE PAR RÉSEAUX DE NEURONES PROFONDS

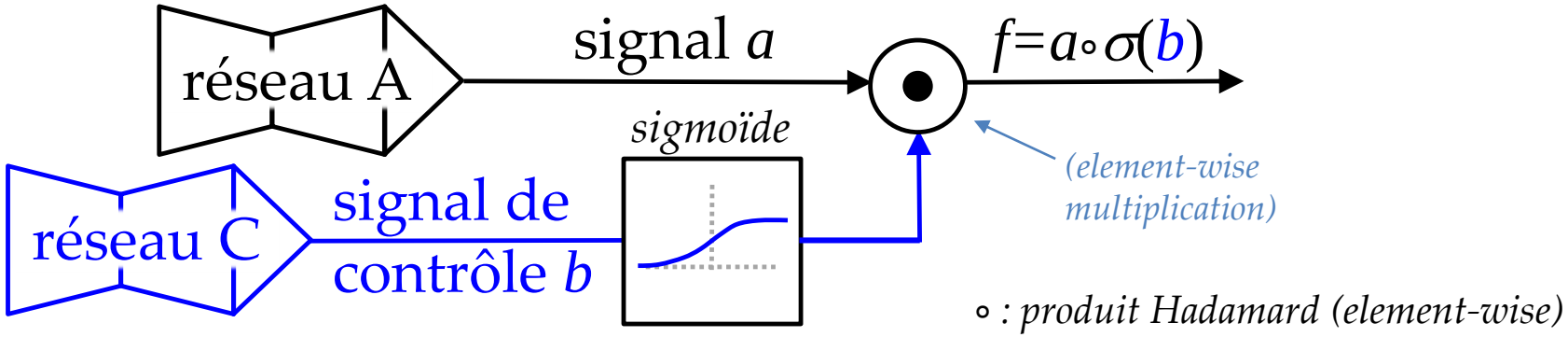
Réseaux Récurrents avec *gate* (LSTM et GRU)

Philippe Giguère

LSTM (1997)

- Toujours d'actualité
- Résoudre les problèmes du RNN :
  - difficulté de la longue portée
  - *vanishing gradient*
- Idée maîtresse : **cellule(s) à état (cell state)** c_t
  - peut y ajouter/retirer/exposition de l'information via des *gates* (contrôle flux d'information)

Rappel

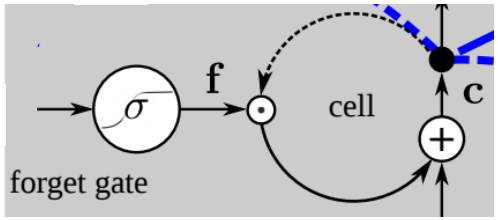


  - similitude avec highway network/ResNet

LSTM : cellule + 3 gates

(beaucoup de diagrammes)

Cellule : (déroulée)



h_t

C_t (linéaire, peu modifiée, highway)

Forget gate

h_{t-1}

Input gate

Output gate

X_t

tanh

Neural Network Layer

Pointwise Operation

Vector Transfer

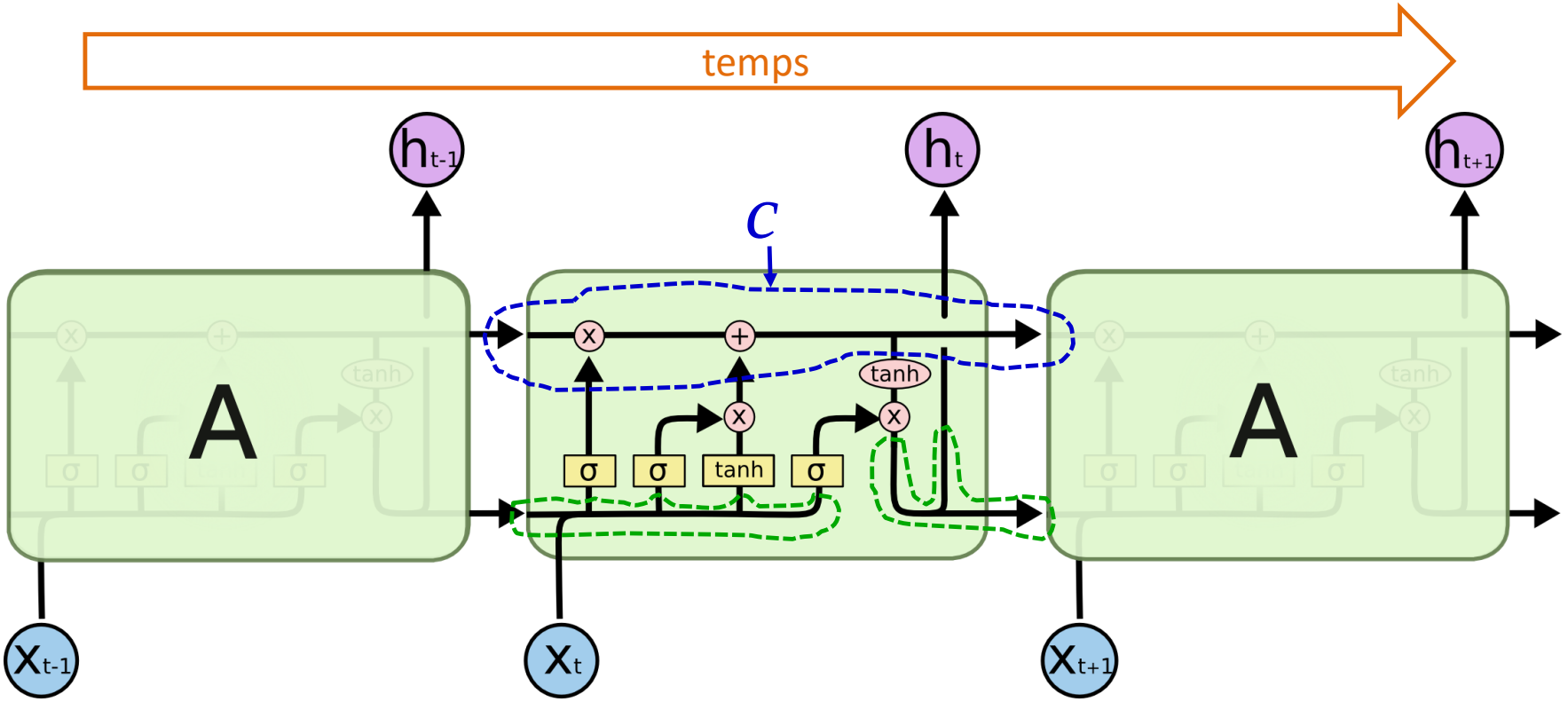
Concatenate

Copy

Illustrations : Christopher Olah, avec permission

LSTM : récursivité déroulée

- « État caché » est (h_t, c_t)



Illustrations : Christopher Olah, avec permission

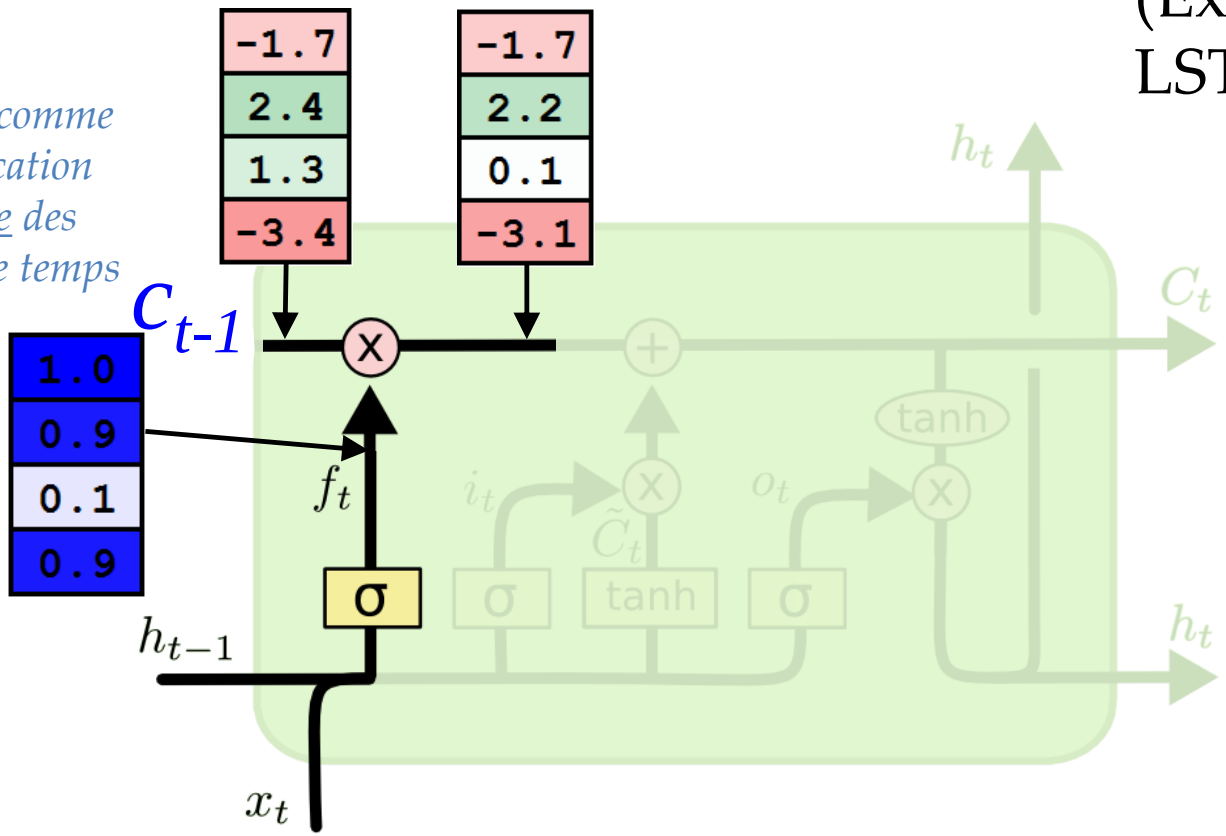
LSTM : Étape 1

- Quelle information retirer de la cellule ?
- forget gate (pensez plus : *remember gate*)

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

(Exemple pour LSTM à 4 unités)

Peut le voir comme une modification dynamique des constantes de temps



Illustrations : Christopher Olah, avec permission

LSTM : Étape 2

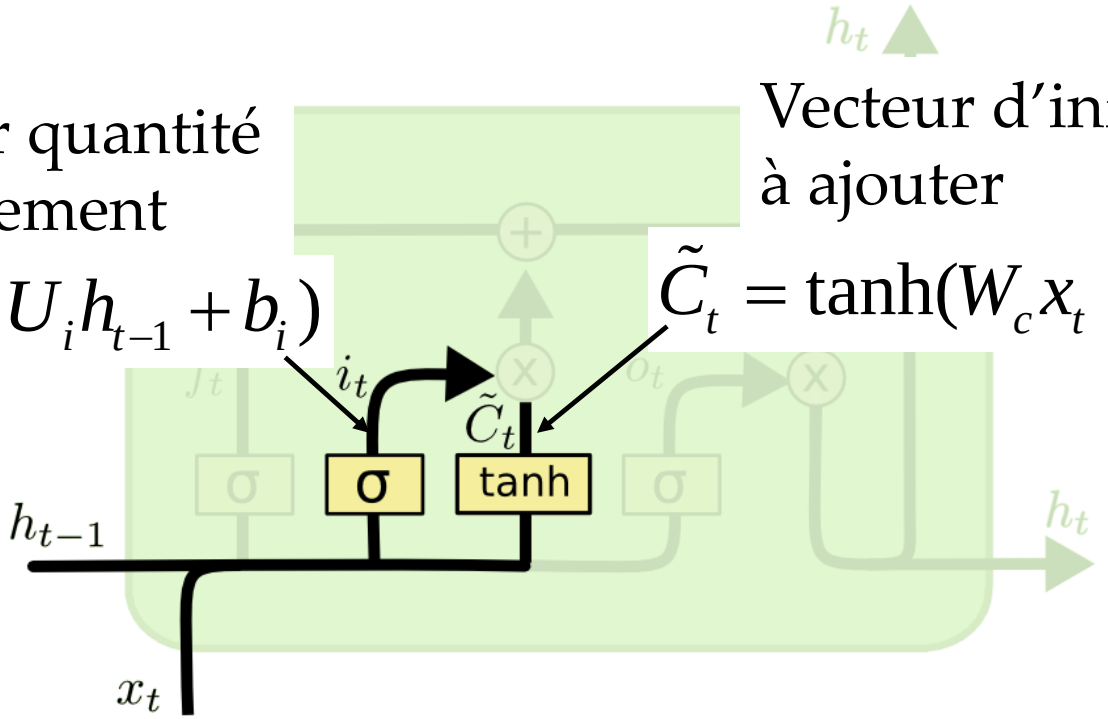
- Choisir l'information à ajouter à la cellule

Vecteur pour quantité de changement

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

Vecteur d'information à ajouter

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c)$$

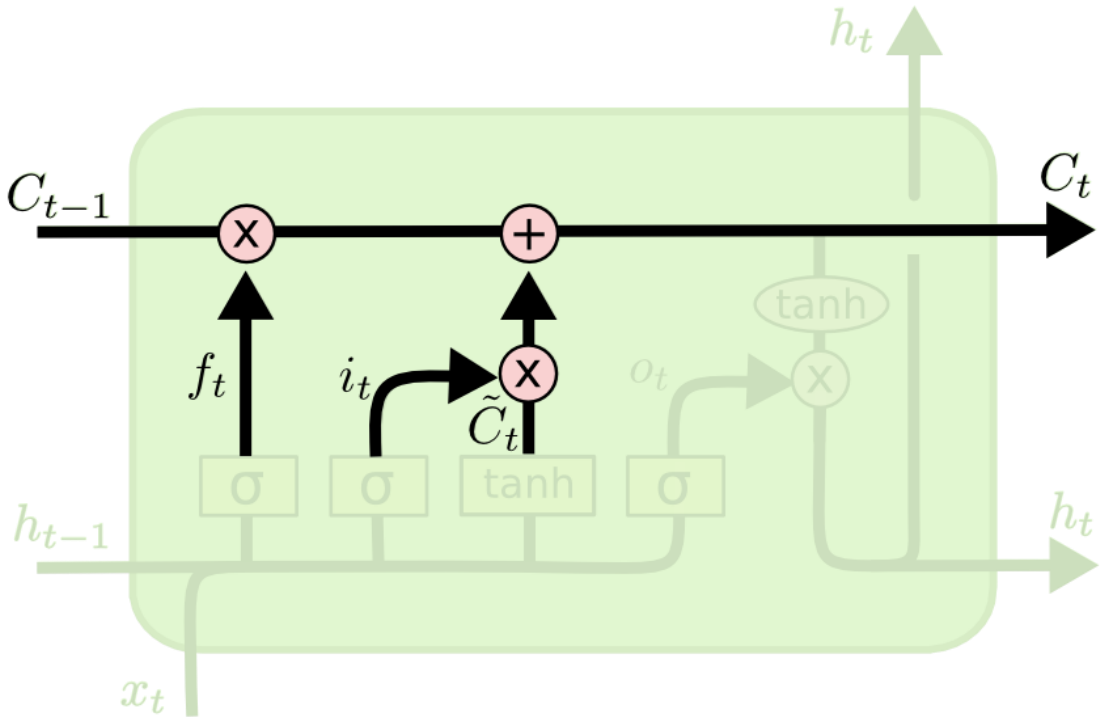


Illustrations : Christopher Olah, avec permission

LSTM : Étape 3

- La cellule est mise-à-jour indirectement par l'état caché
  - voir comme des ajustements incrémentaux (résiduel)

$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t$$



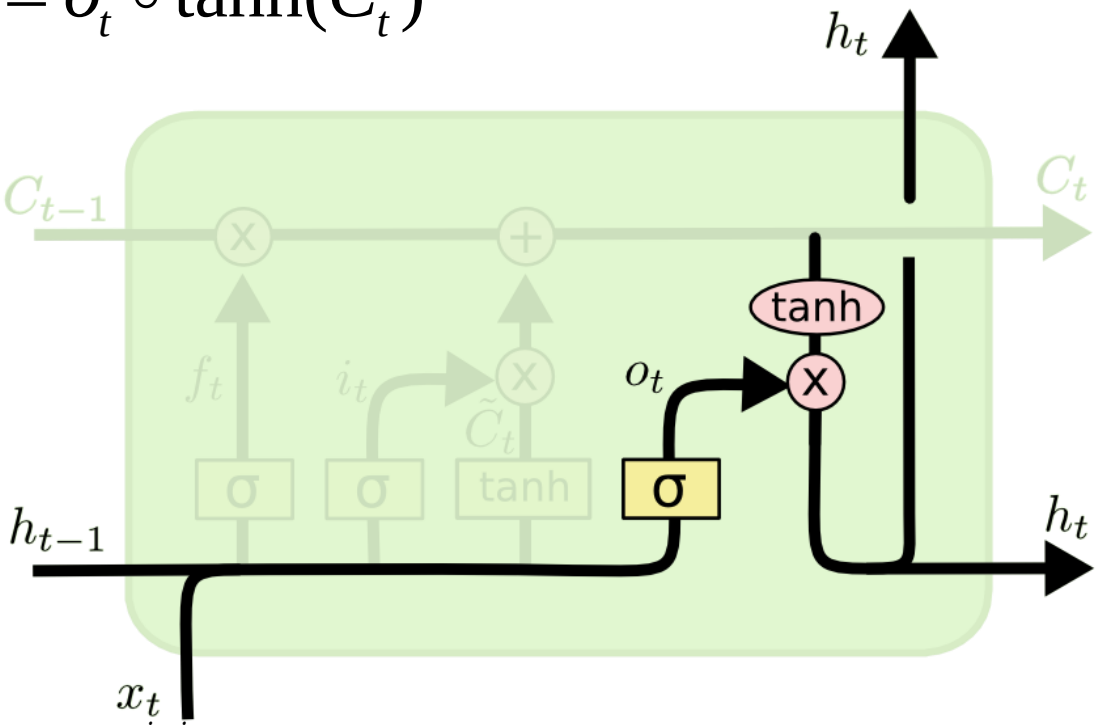
Illustrations : Christopher Olah, avec permission

LSTM : Étape 4

- h_t est une sortie de tanh(c_t)
- Modulée par l'output gate

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$

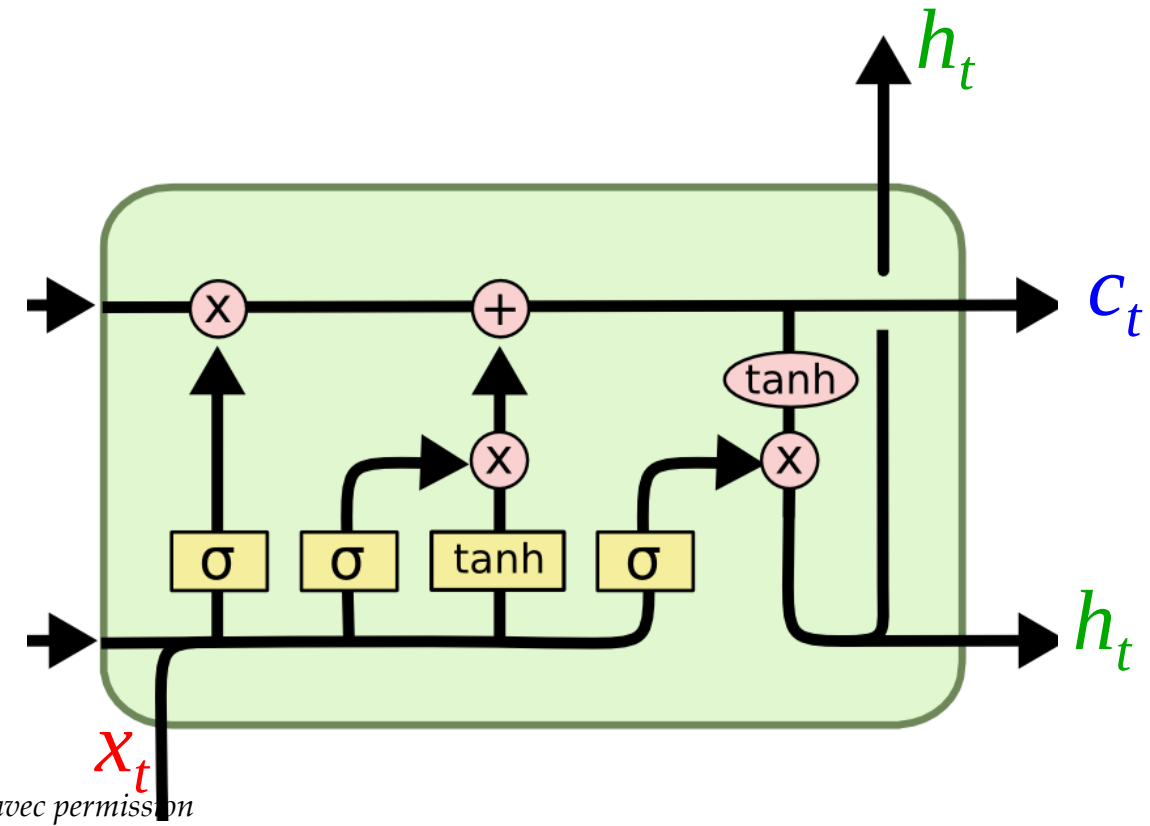
$$h_t = o_t \circ \tanh(C_t)$$



Illustrations : Christopher Olah, avec permission

LSTM

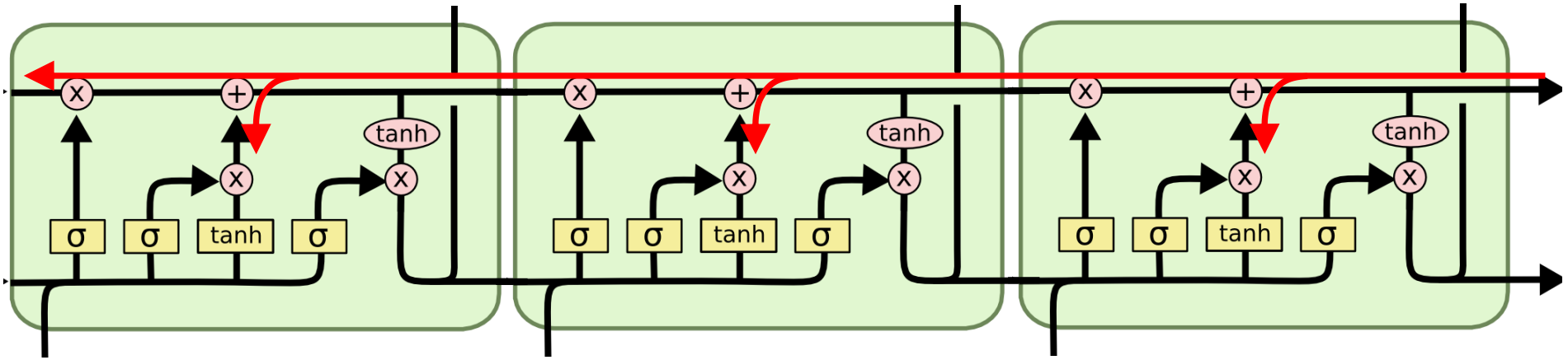
- La cellule est affectée lentement (**slow state**)
- L'état *h* est affecté plus rapidement (**fast state**)



Illustrations : Christopher Olah, avec permission

Flot du gradient

- Gradient se propage mieux que RNN
  - via la cellule
  - pensez ResNet



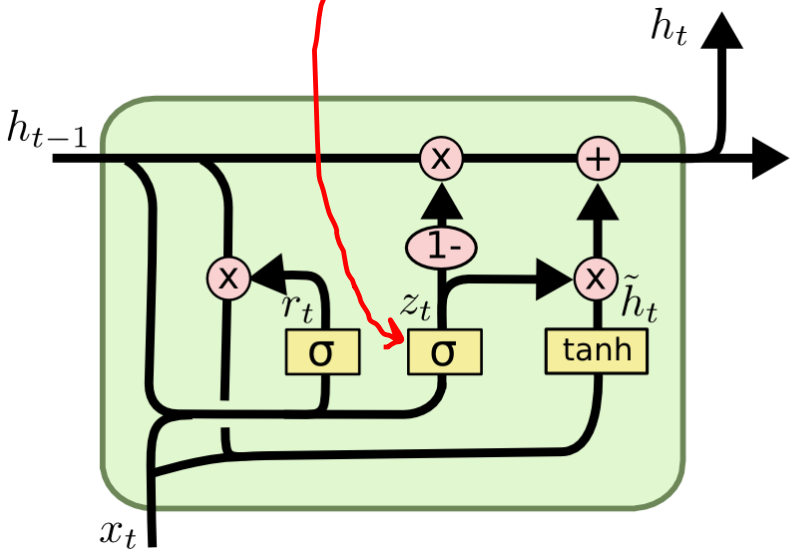
(Aussi à condition que *forget gate* ait des entrées proches de 1)

Illustrations adaptée de Christopher Olah, avec permission

variante de LSTM

GRU : Gated Recurrent Unit

- Combine forget et input gate ensemble
  - update gate
- Plus de séparation hidden/cell



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

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

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

- Moins de paramètres

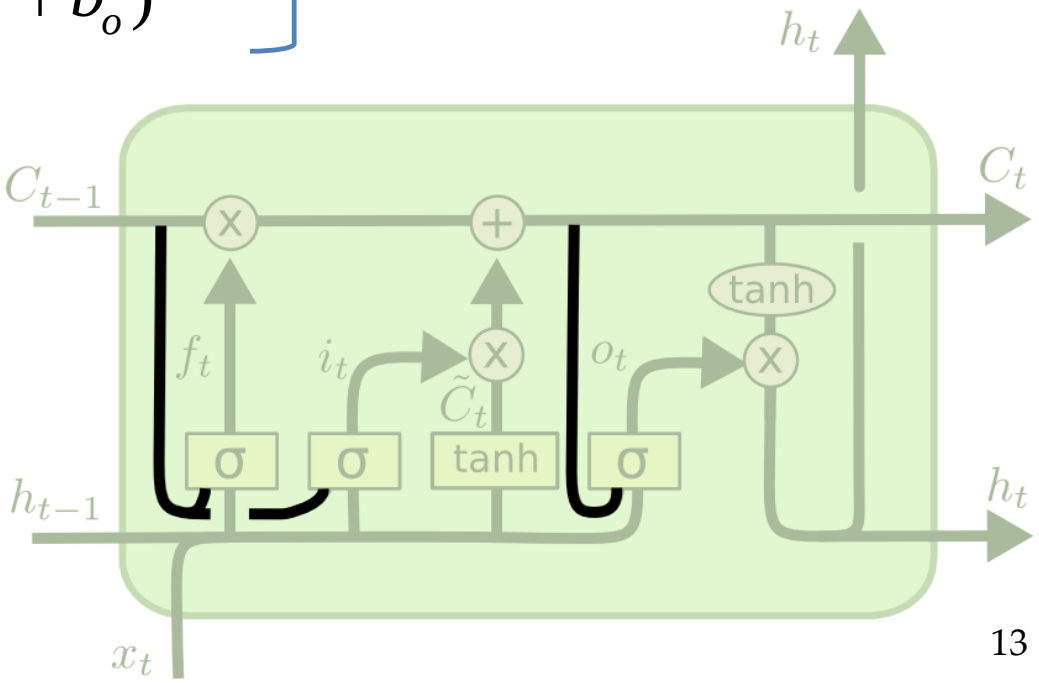
Illustrations : Christopher Olah, avec permission

Peephole connection

- Pour permettre à l'état de la cellule de contrôler les *gates*

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + P_f c_{t-1} + b_f)$$
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + P_i c_{t-1} + b_i)$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + P_o c_t + b_o)$$

Note : certaines variantes, U = 0



Illustrations : Christopher Olah, avec permission

LSTM: A Search Space Odyssey

- 5400 tests de 8 variantes d'architecture sur 3 tâches :
  - modélisation acoustique
  - reconnaissance d'écriture manuscrite
  - modélisation musique polyphonique
- Aucune variante ne domine réellement
  - variante GRU a l'avantage d'avoir moins de paramètres
- Parties les plus importantes :
  - forget gate
  - output activation

An Empirical Exploration of Recurrent Network Architectures

- Essais de 10,000 architectures trouvées par processus d'évolution
- Ont identifié une architecture qui parfois dépasse le LSTM et le GRU, mais sur certaines tâches seulement
- Bref, LSTM/GRU encore compétitif!
  - réduit l'écart GRU-LSTM en ajoutant un biais de b_f=1 pour le *forget gate* du LSTM

Évolution

RNN → LSTM → Attention