Deep learning avec Pytorch

Partie 10 : Recurrent Nerual Network

Présenté par **Morgan Gautherot**

Pourquoi des modèles de séquence ?

	X	Y
Reconnaissance vocale		“Never stoop learning”
Génération de musique	∅	
Classification des sentiments	“There is nothing to like in this movie.”	
Analyse des séquences d'ADN	AGCCCCTGTGAGGAACTAG	AGCCCCTGTGAGGAACTAG
Traduction automatique	Artificial Intelligence is the New Electricity	L'intelligence artificielle est la nouvelle électricité
Reconnaissance d'activité		Running
Reconnaissance d’un nom	Elon Musk is the founder of SpaceX	Elon Musk is the founder of SpaceX

Notation

Reconnaissance de nom

x: “Harry Potter and Hermione Granger invented a new spell.”

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ … $x^{<t>}$ … $x^{<8>}$ $x^{<9>}$

y: [1 , 1 , 0 , 1 , 1 , 0 , 0 , 0 , 0]

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ … $y^{<t>}$ … $y^{<9>}$

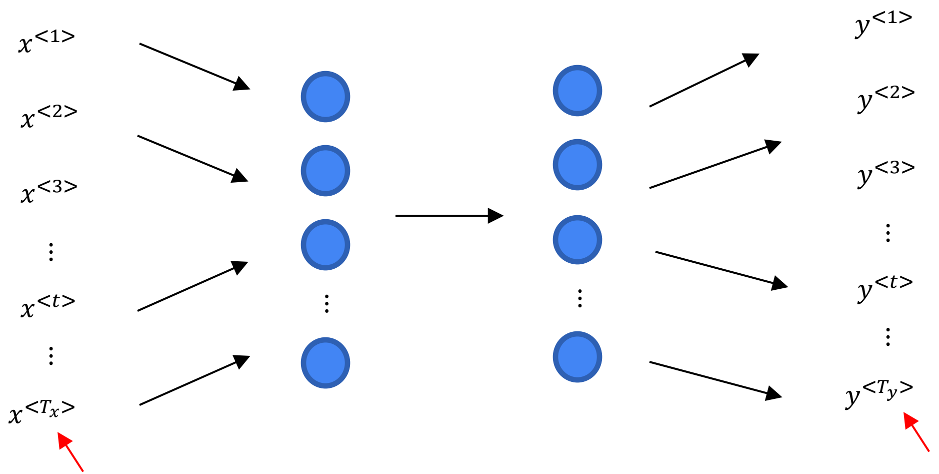
$X^{(i)<t>}$ observation de la séquence t du $i^{ème}$ exemple.

$y^{(i)<t>}$ target de la séquence t du $i^{ème}$ exemple.

${T_x}^{(i)}$ est la longueur de la séquence d'observations.

${T_y}^{(i)}$ est la longueur de la séquence des valeurs cibles.

Pourquoi pas un réseau dense ?



Problèmes :

- Les entrées et les sorties peuvent être de longueurs différentes dans différents exemples.

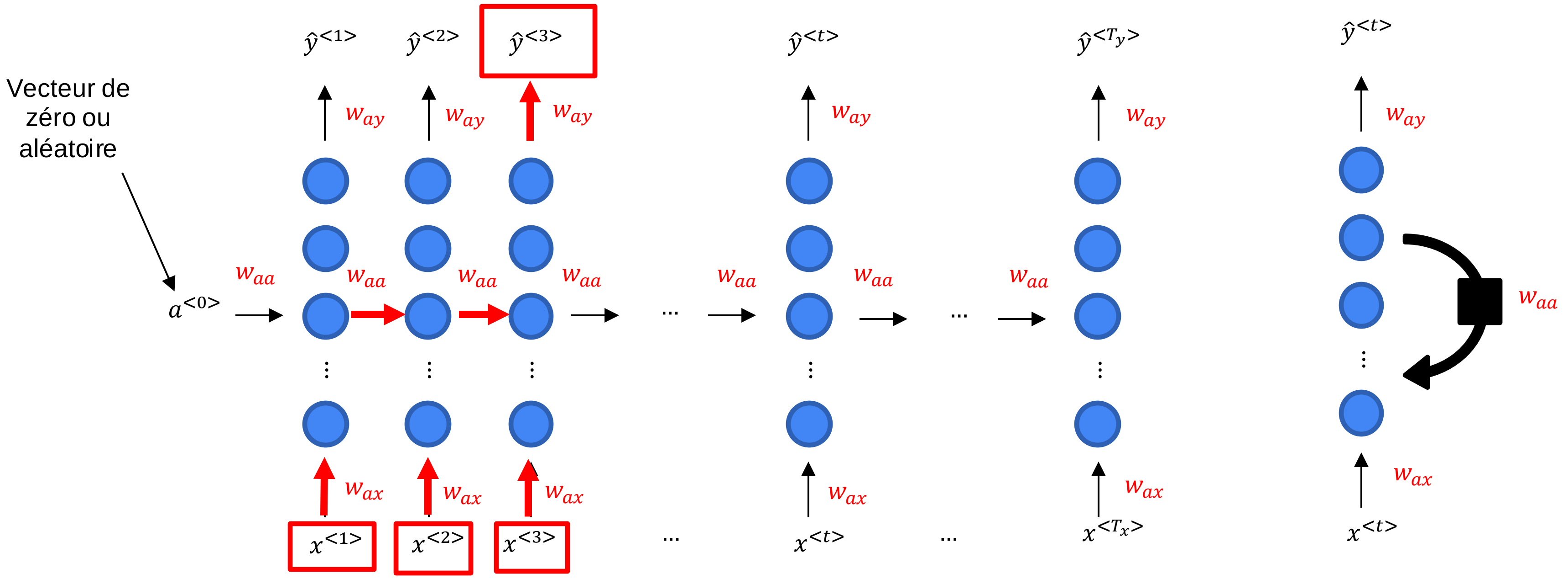
- Ne partage pas les caractéristiques apprises entre les différentes positions de la séquence.

Le recurrent Neural Network

Partie 9 : Recurrent Nerual Network

Recurrent Neural Networks

Ici $T_x = T_y$

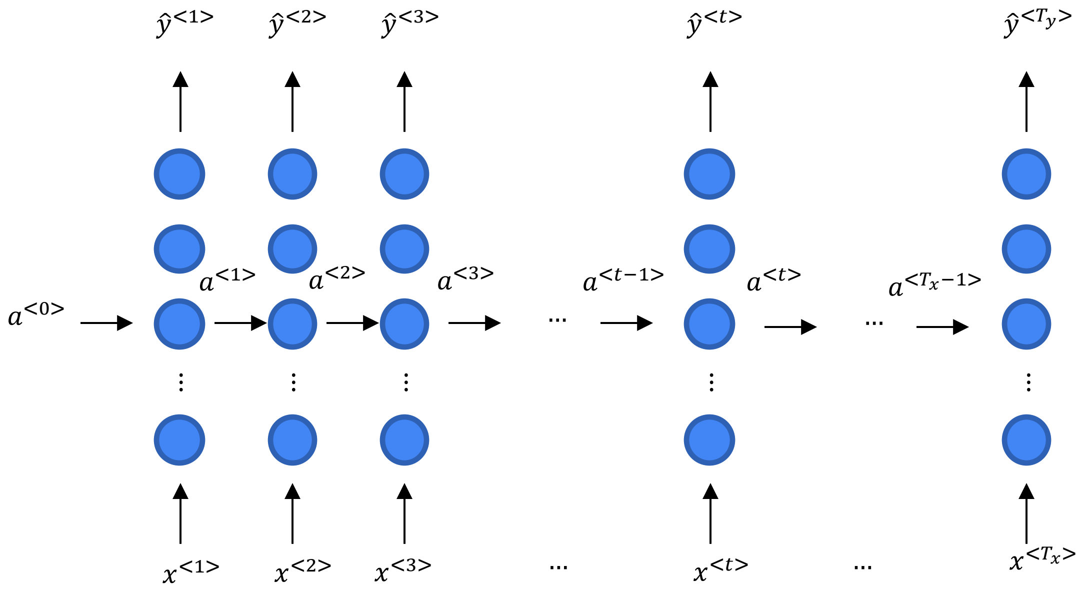


Froward et backward propagation

Partie 9 : Recurrent Nerual Network

Forward propagation

Here $T_x = T_y$



$$a^{<0>} = \vec{0}$$

$$a^{<1>} = g(W_{aa}a^{<0>} + W_{ax}x^{<1>} + b_a) \longleftarrow \text{Relu}$$

$$\hat{y}^{<1>} = \sigma(W_{ya}a^{<1>} + b_y) \longleftarrow \text{Sigmoid}$$

$$a^{<t>} = g(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$

$$\hat{y}^{<t>} = \sigma(W_{ya}a^{<t>} + b_y)$$

Notation

$a^{<t>} = g(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$

$\hat{y}^{<t>} = g(W_{ya}a^{<t>} + b_y)$

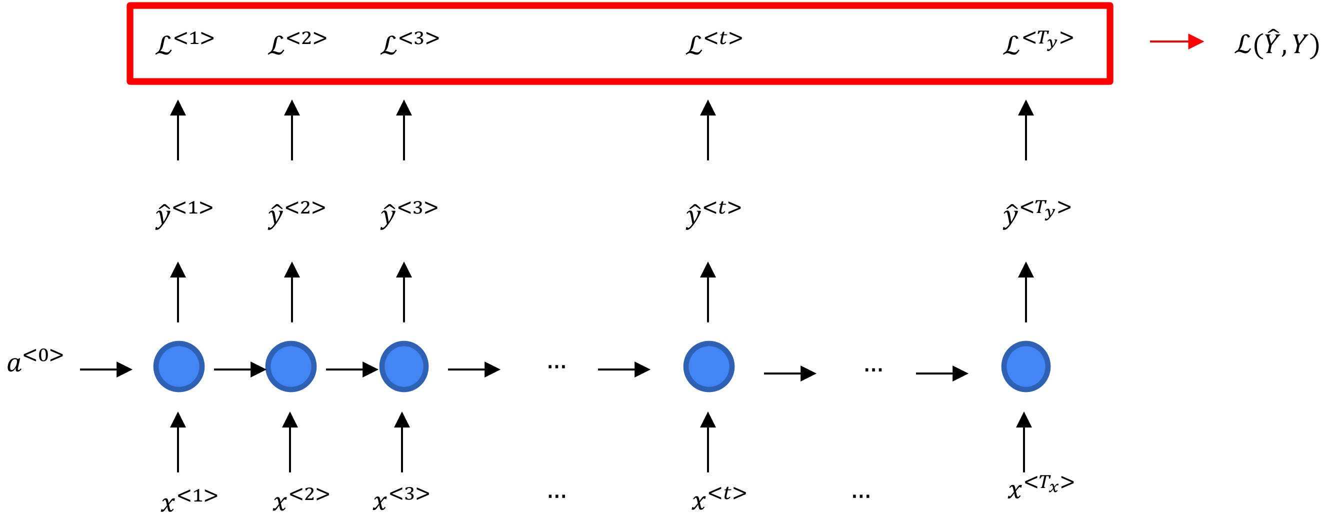
W_a=$[W_{aa}, W_{ax}]$

$a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$

$\hat{y}^{<t>} = g(W_y a^{<t>} + b_y)$

$[W_{aa}, W_{ax}] \quad \begin{bmatrix} a^{<t-1>} \\ x^{<t>} \end{bmatrix} = W_{aa}a^{<t-1>} + W_{ax}x^{<t>}$

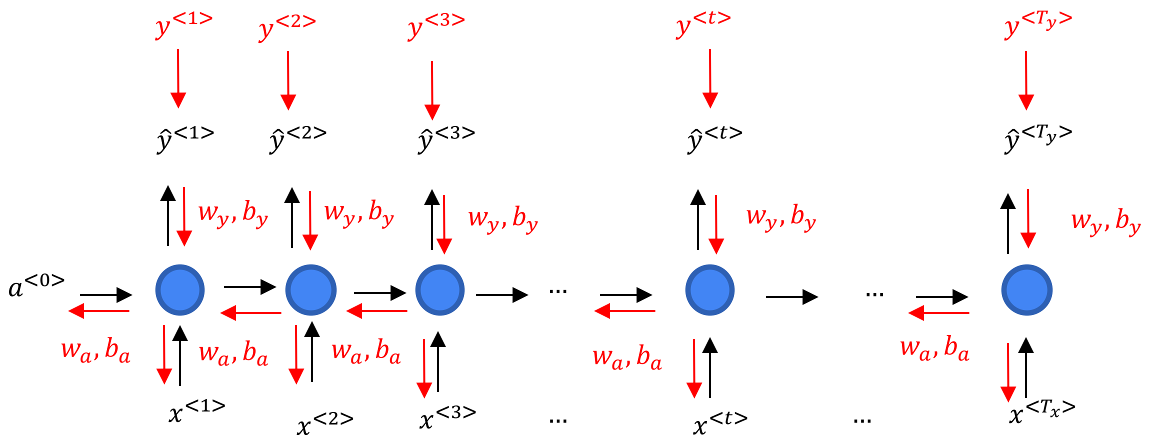
Fonction de coût



$$\mathcal{L}^{<t>}(\hat{y}^{<t>}, y^{<t>}) = -y^{<t>} \log(\hat{y}^{<t>}) - (1 - y^{<t>})\log((1 - \hat{y}^{<t>})$$

$$\mathcal{L}(\hat{y}, y) = \sum_{t=1}^{T_y} \mathcal{L}^{<t>}(\hat{y}^{<t>}, y^{<t>})$$

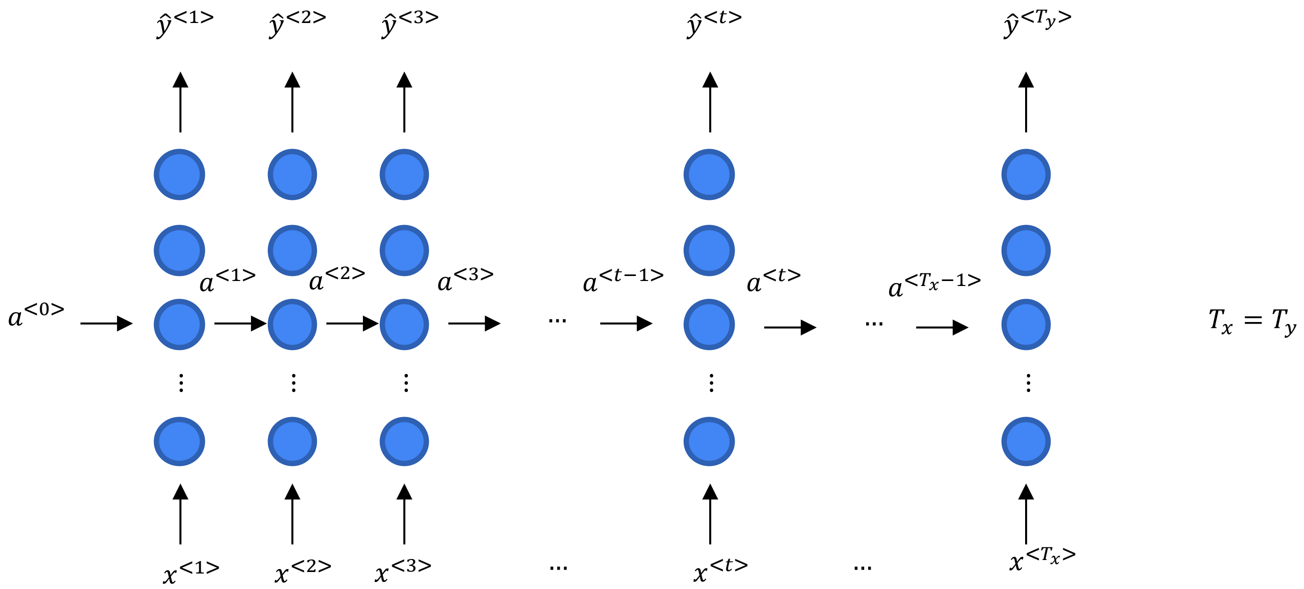
Backpropagation



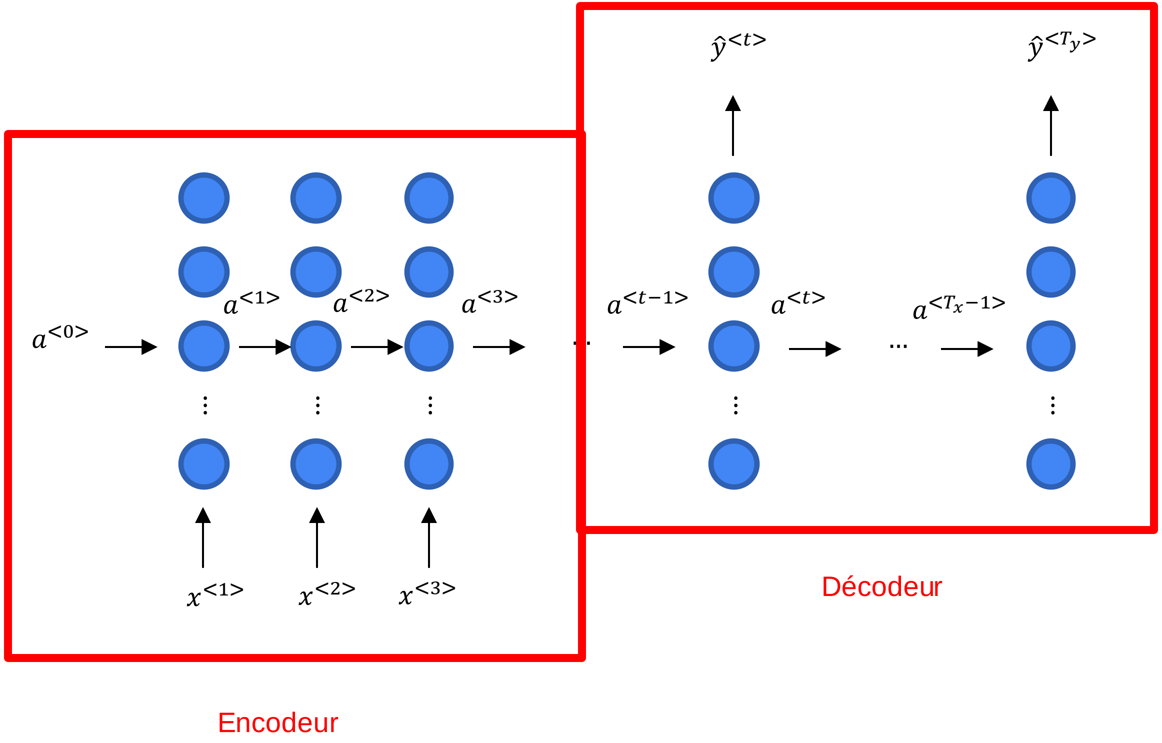
Les différentes architectures

Partie 9 : Recurrent Nerual Network

Many to many

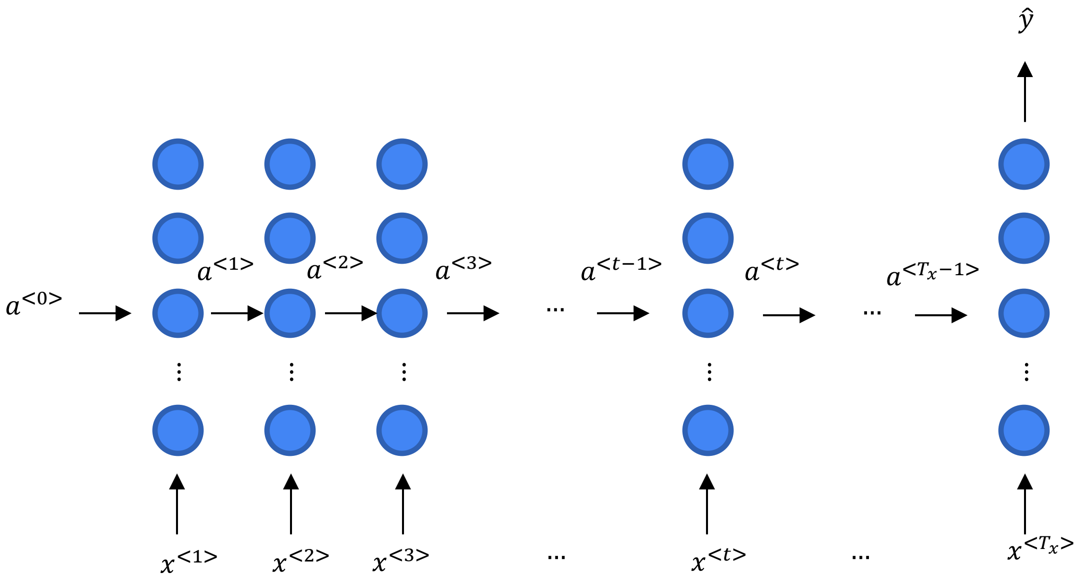

$\hat{y}^{<1>}$
$\hat{y}^{<2>}$
$\hat{y}^{<3>}$
$\hat{y}^{<t>}$
$\hat{y}^{<T_y>}$
$a^{<0>}$
$a^{<1>}$
$a^{<2>}$
$a^{<3>}$
...
$a^{<t-1>}$
$a^{<t>}$
...
$a^{<T_x-1>}$
$x^{<1>}$
$x^{<2>}$
$x^{<3>}$
...
$x^{<t>}$
...
$x^{<T_x>}$
$T_x = T_y$

Many to many



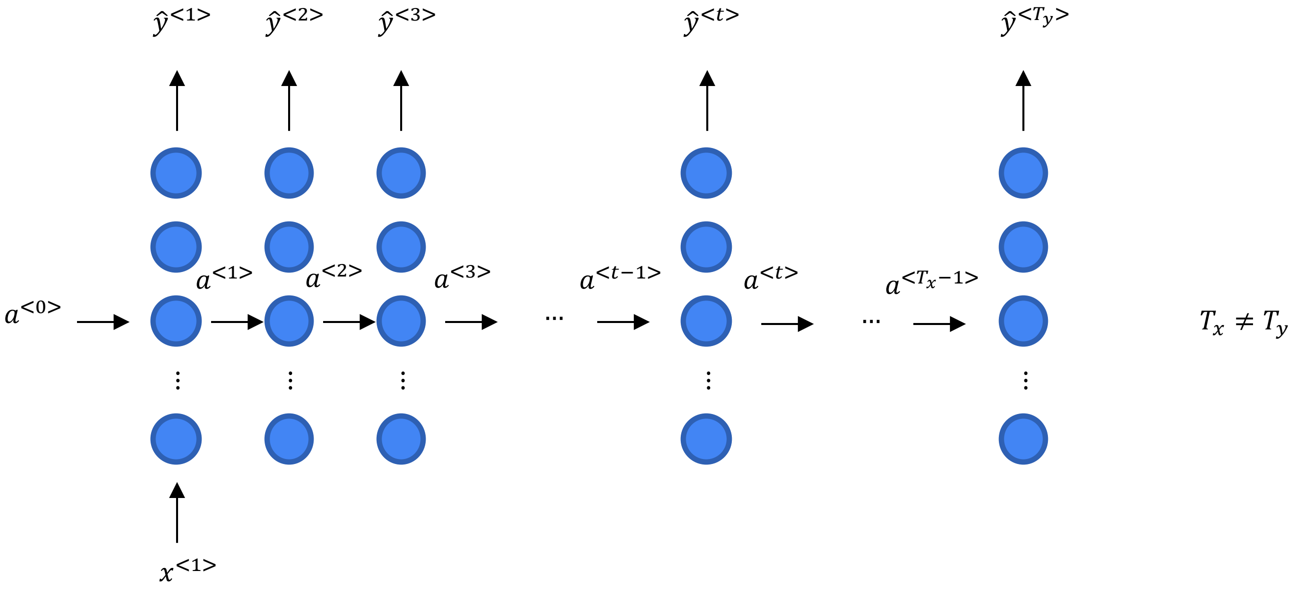
$T_x \neq T_y$

Many to one



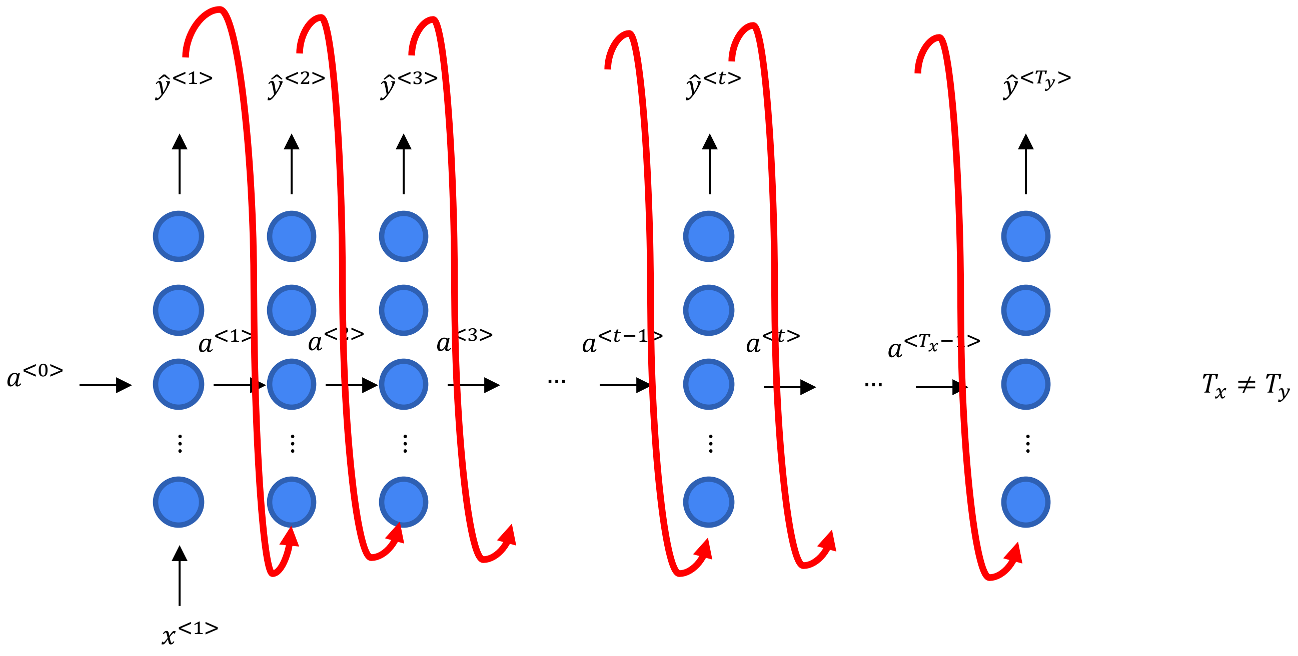
$T_x \neq T_y$

One to many



$\hat{y}^{<1>}$
$\hat{y}^{<2>}$
$\hat{y}^{<3>}$
$\hat{y}^{<t>}$
$\hat{y}^{<T_y>}$
$a^{<0>}$
$a^{<1>}$
$a^{<2>}$
$a^{<3>}$
...
$a^{<t-1>}$
$a^{<t>}$
...
$a^{<T_x-1>}$
$x^{<1>}$
$T_x \neq T_y$

One to many



$\hat{y}^{<1>}$
$\hat{y}^{<2>}$
$\hat{y}^{<3>}$
$\hat{y}^{<t>}$
$\hat{y}^{<T_y>}$
$a^{<0>}$
$a^{<1>}$
$a^{<2>}$
$a^{<3>}$
$a^{<t-1>}$
$a^{<t>}$
$a^{<T_x-1>}$
$x^{<1>}$
$T_x \neq T_y$

Bidirectional RNN (BRNN)

Partie 9 : Recurrent Nerual Network

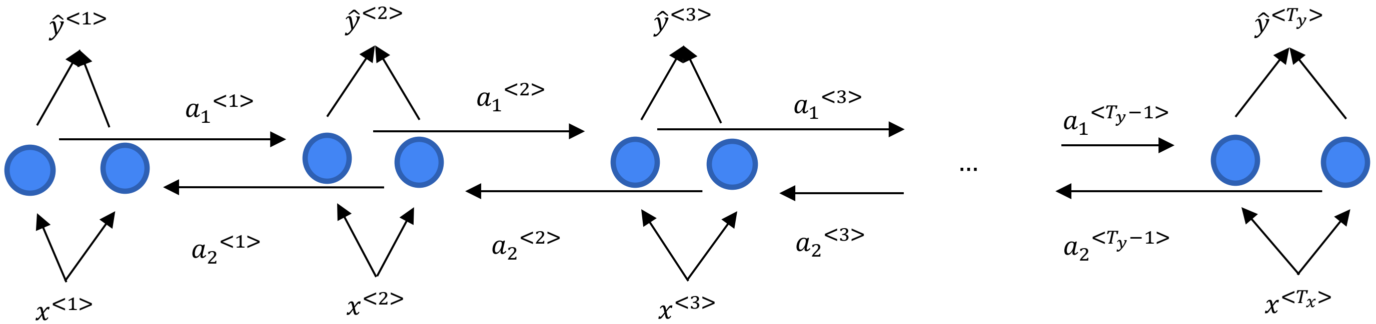
De l’information manquantes

He said, “Teddy Roosevelt was a great President.”

He said, “Teddy bears are on sale!”

Bidirectional RNN (BRNN)

$$\hat{y}^{<t>} = g(w_y[a_1{}^{<t>}, a_2{}^{<t>}] + b_y)$$



Deep RNN

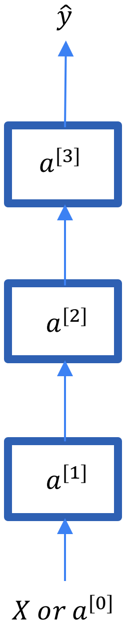
Partie 9 : Recurrent Nerual Network

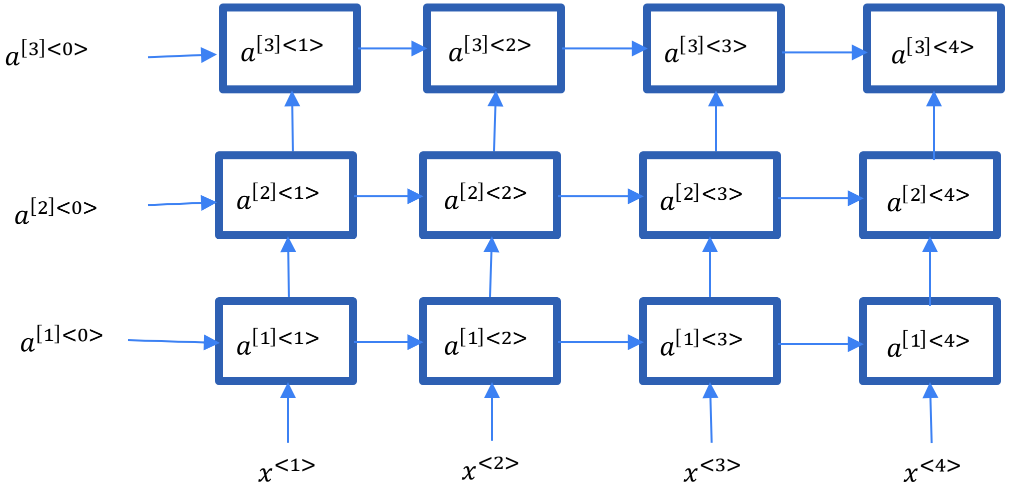
Deep RNN

couche

temps

$$a^{[l]<t>}$$

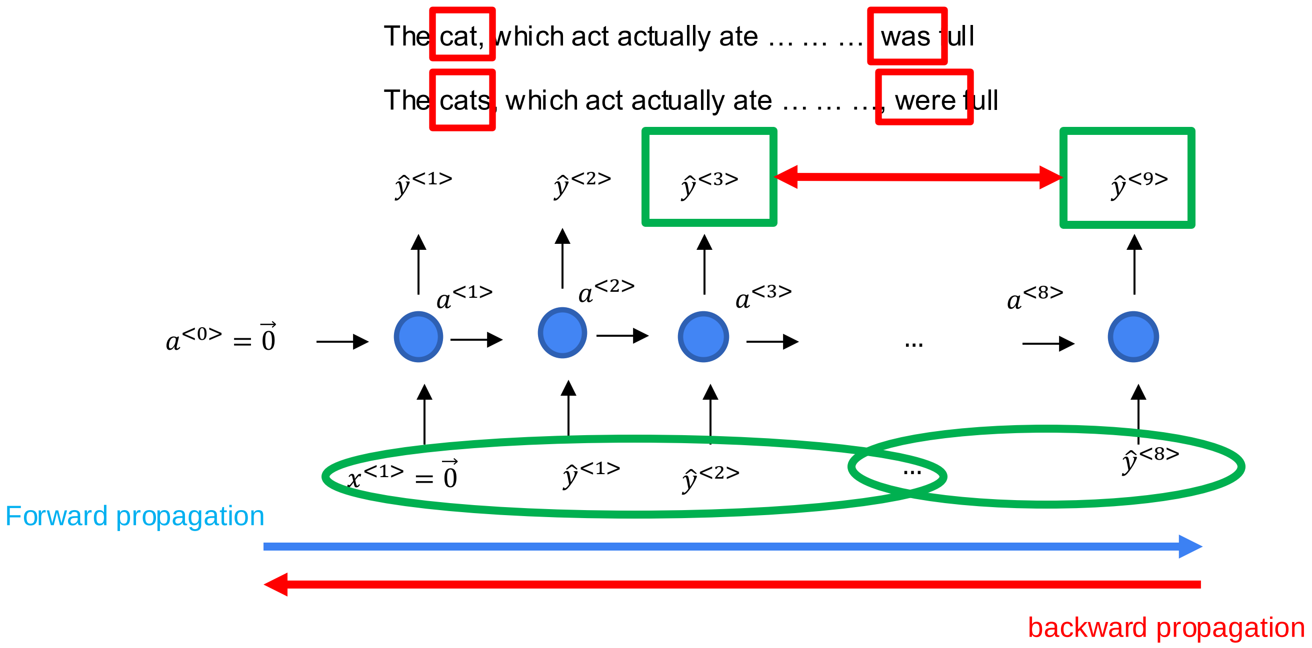




Problème de mémoire

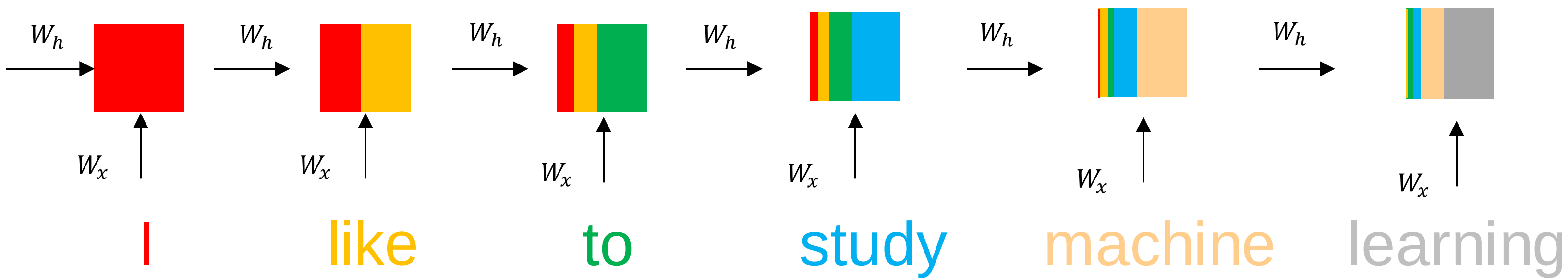
Partie 9 : Recurrent Nerual Network

Le problème du RN classique



L'influence des premiers termes

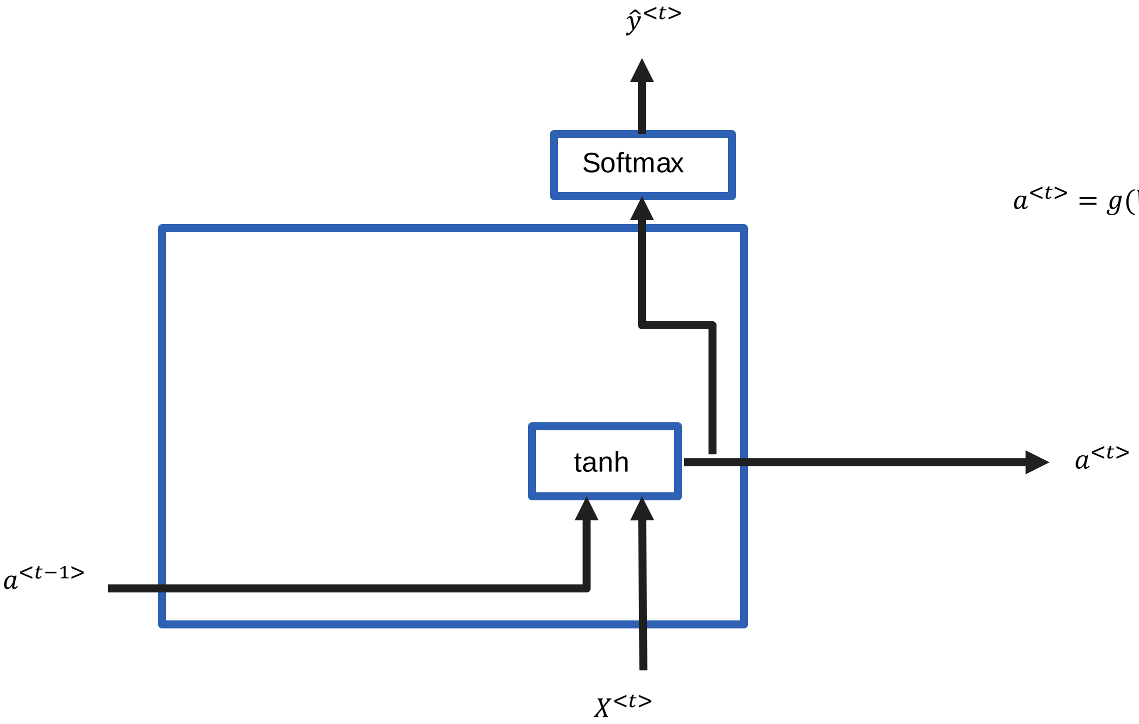
I like to study machine learning ________________



Le Gate Recurrent Unit (GRU)

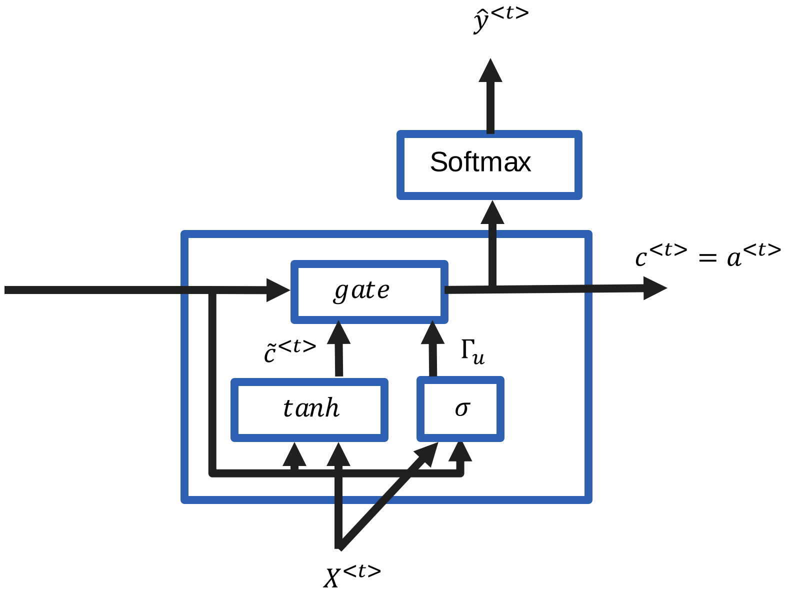
Partie 9 : Recurrent Nerual Network

Recurrent Neural Network



$$a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

Gate Recurrent Unit (GRU) (simplified)



σ est la fonction sigmoïd

c = unité de mémoire

$c^{<t>} = a^{<t>}$

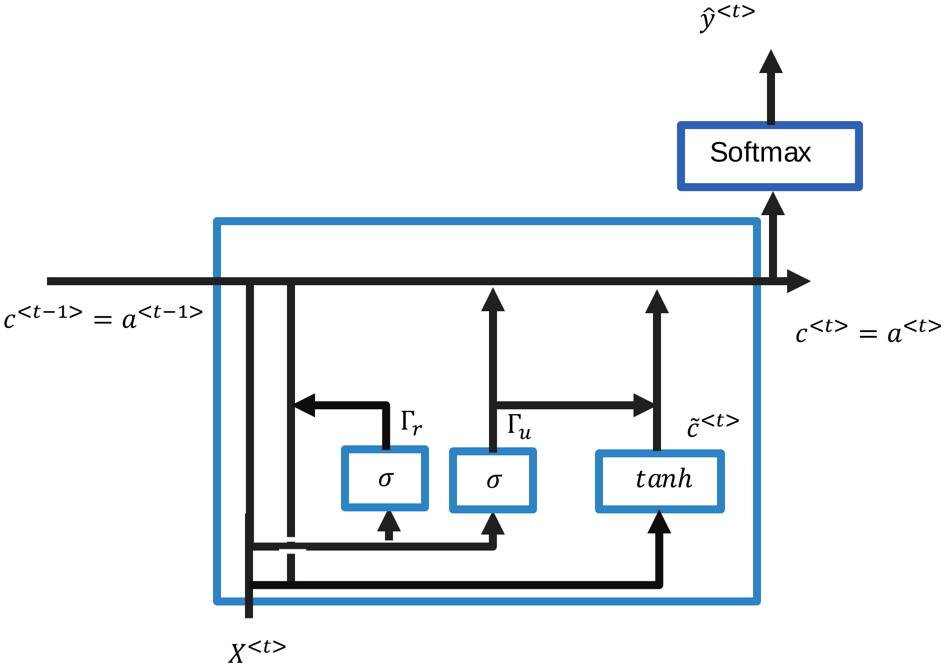
$\tilde{c}^{<t>} = \tanh(w_c[c^{<t-1>}, X^{<t>}] + b_c$

$\Gamma_u = \sigma(w_u[c^{<t-1>}, X^{<t>}] + b_u)$

$0 < \Gamma_u < 1$

$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$

Gate Recurrent Unit (GRU)



c = unité de mémoire

$$c^{<t>} = a^{<t>}$$

$$\tilde{c}^{<t>} = \tanh(w_c[\Gamma_r * c^{<t-1>}, X^{<t>}] + b_c$$

$$\Gamma_u = \sigma(w_u[c^{<t-1>}, x^{<t>}] + b_u)$$

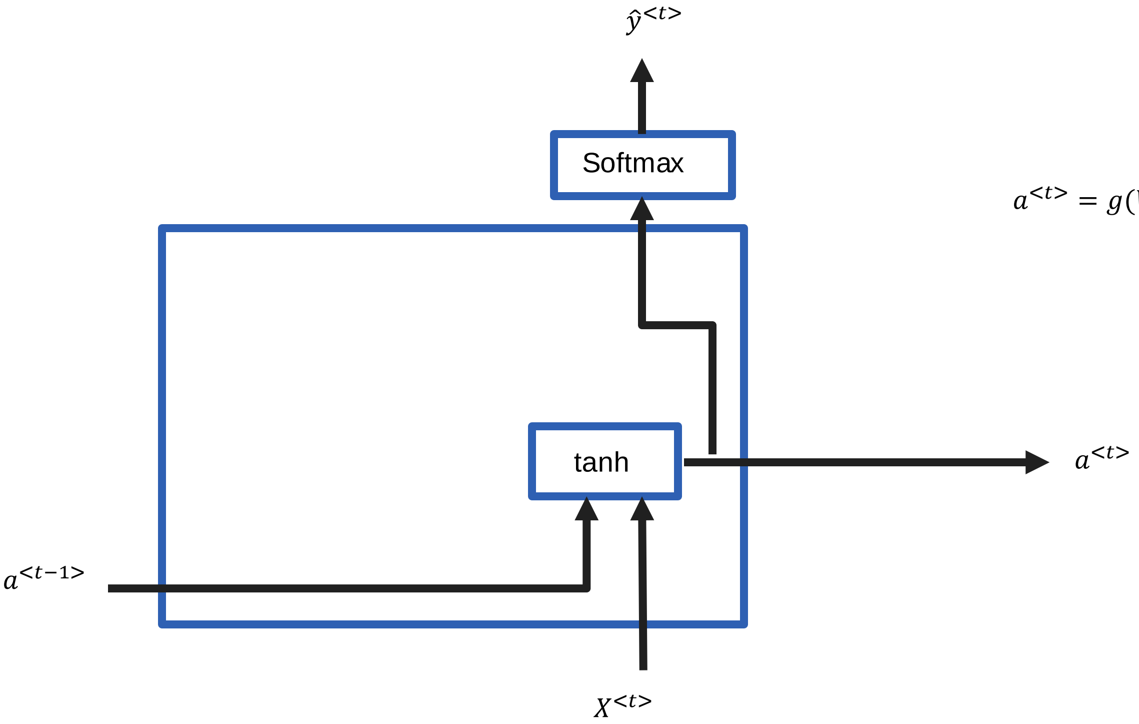
$$\Gamma_r = \sigma(w_r[c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

Le Gate Recurrent Unit (GRU)

Partie 9 : Recurrent Nerual Network

Recurrent Neural Network



$$a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

Gate Recurrent Unit and Long Short Term Memory

GRU

$$\tilde{c}^{<t>} = \tanh(w_c[\Gamma_r * c^{<t-1>}, X^{<t>}\} + b_c$$

$$\Gamma_u = \sigma(w_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(w_r[c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

LSTM

$$\tilde{c}^{<t>} = \tanh(w_c[a^{<t-1>}, X^{<t>}\} + b_c$$

$$\Gamma_u = \sigma(w_u[a^{<t-1>}, x^{<t>}] + b_u)$$

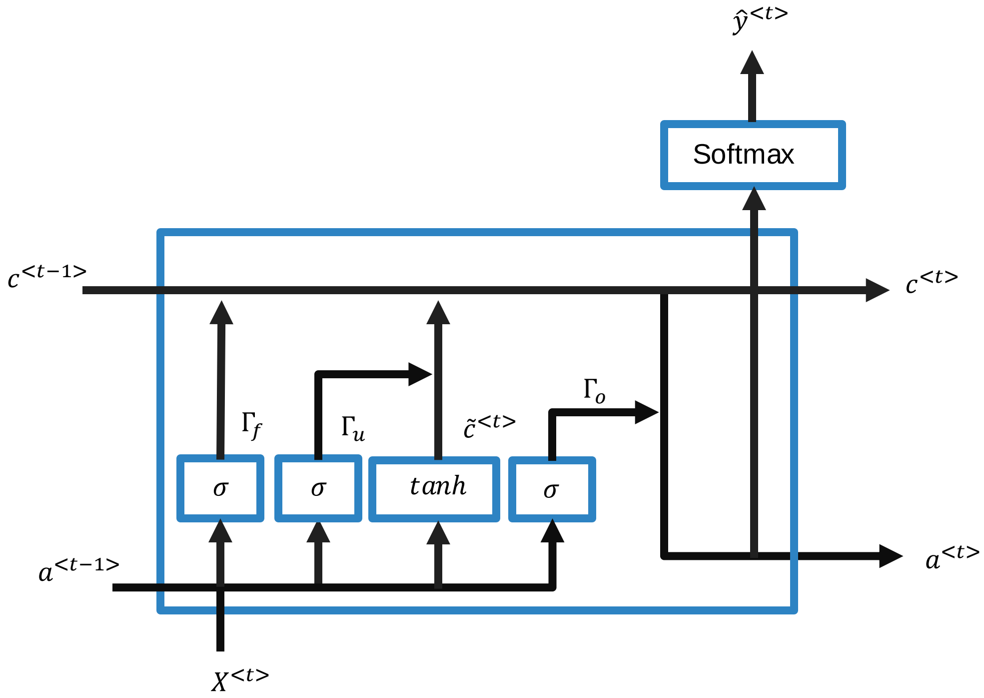
$$\Gamma_f = \sigma(w_f[c^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(w_o[c^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_0 * \tanh(c^{<t>})$$

Long Short Term Memory (LSTM)



$$\tilde{c}^{<t>} = \tanh(w_c[a^{<t-1>}, X^{<t>}\} + b_c$$

$$\Gamma_u = \sigma(w_u[a^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(w_f[a^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(w_o[a^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_0 * \tanh(c^{<t>})$$

LSTM en série