IA et Nous

TALN 2ème partie

Traitement Automatique Du Langage Naturel

NLP Natural Language Processing

Plan

1. *TALN c'est quoi ?*
2. *Brève histoire*
3. *Séquences*
4. *Prétraitements*
    - *Codage*
    - *Tokenisation*
    - *Word Embedding*
5. ***Réseaux de neurones récurrents RNN et LSTM***
6. ***Encodeur Décodeur - Seq2seq***
7. ***Mécanisme d'attention***
8. ***Transformer***
9. ***Modèles de langage***

Applications

Types d'applications

- Traduction automatique
- Classification d'émotions
- Description automatique d'images
- Résumé automatique de texte
- Chatbot
- Génération de texte

Traduction automatique

Traduction automatique

Google Translate DeepL

- **Problème**: Traduire un texte d'une langue source vers une langue cible
- **Données**:
  - Ens. de couples de textes (Source, Cible)

Ex :
Lois européennes
Lois canadiennes Montréal
….

Traduction automatique

- **Objectif**:
  - Maximiser la probabilité de génération de la bonne traduction
- **Fonction de coût**:
  - Minimiser la vraisemblance logarithmique négative (NLL)

Comment réaliser ces applications ?

Sources :
Pirmin Lemberger Directeur Scientifique OnePoint
[LUO] ***Neural Machine Translation****, Minh-Thang Luong, PhD Dissertation Stanford University 2016*
IVADO Ecole Hiver César Franck et Arsène Fansi Université de Montréal

Evolution des réseaux de neurones

1. Pourquoi
2. Les réseaux récurrents RNN
3. Les réseaux à mémoire long terme LSTM

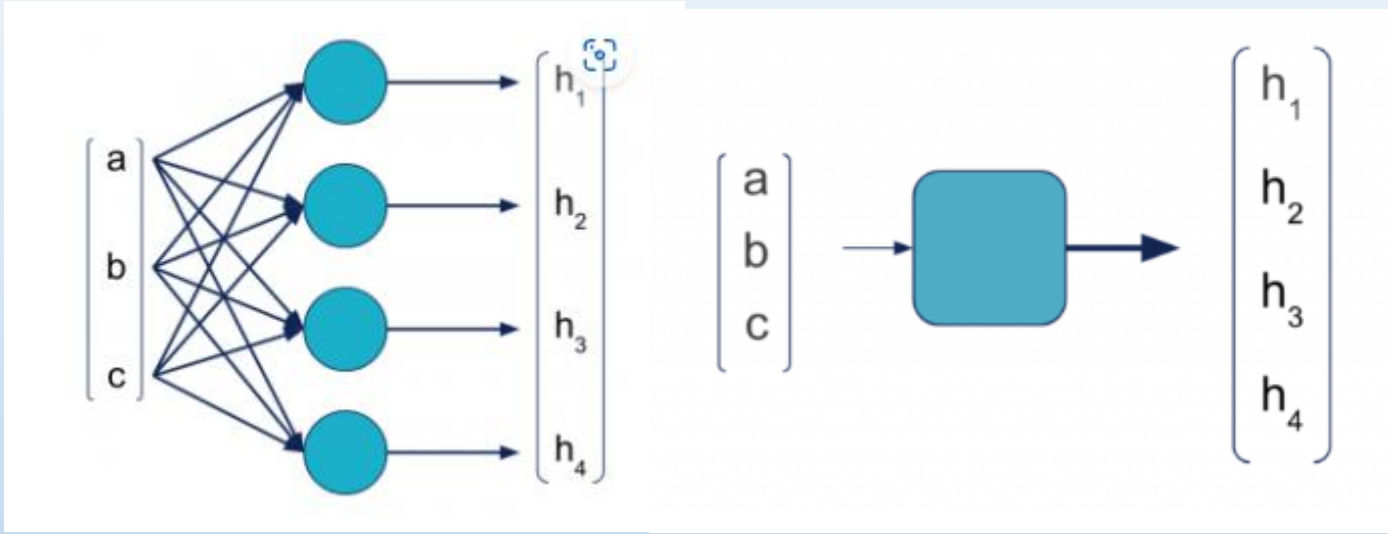
Pourquoi

Réseaux de neurones classiques

Données de taille fixe

	Variable 1	Variable 2	Variable 3	Variable 4	Variable 5
Échantillon 1					
Échantillon 2					
Échantillon 3					
⋮					
Échantillon n					

Schéma MLP



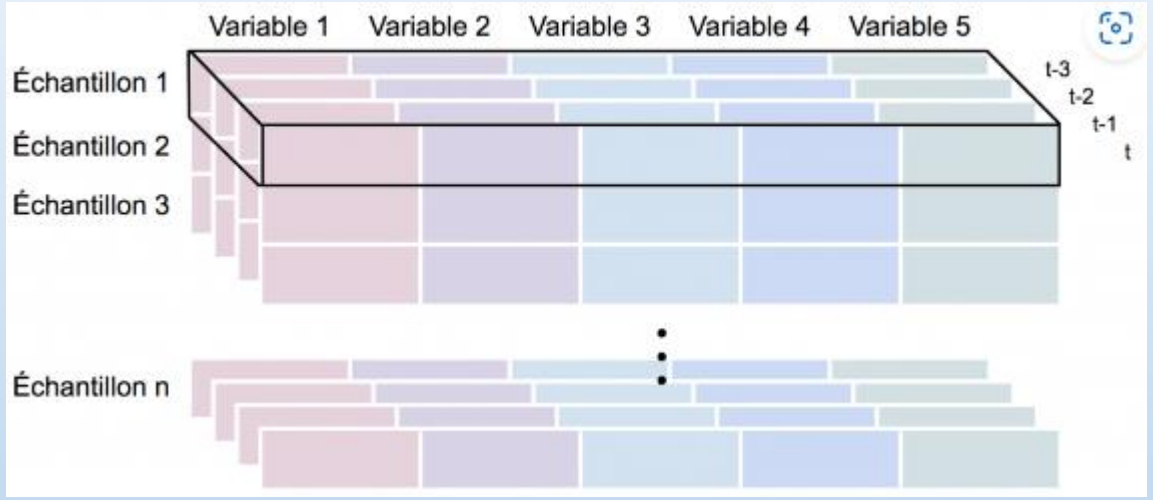
Entrée : vecteurs numériques

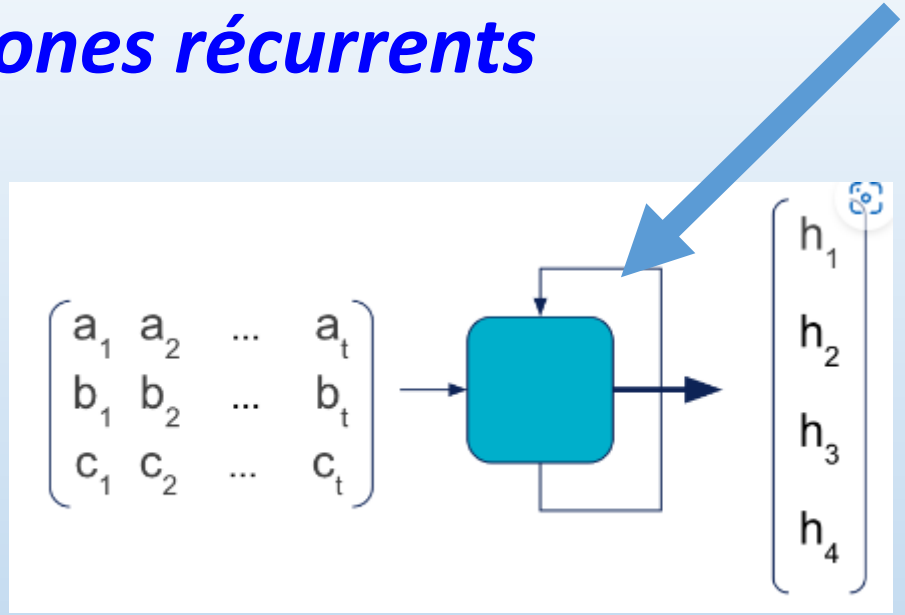
Sortie Vecteurs numériques

Pourquoi

Données séquentielles

Réseaux de neurones récurrents





Entrée : vecteurs numériques séquentiels

Séquence de vecteurs

La variable a est une suite chronologique de n valeurs (a1, a2, …, an).

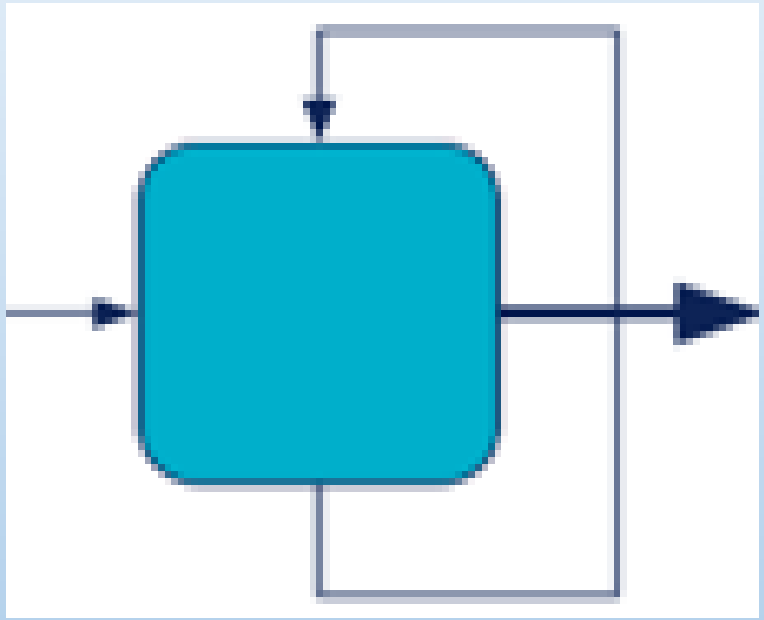
Pourquoi

Données séquentielles

La variable d'entrée est une suite chronologique de n valeurs

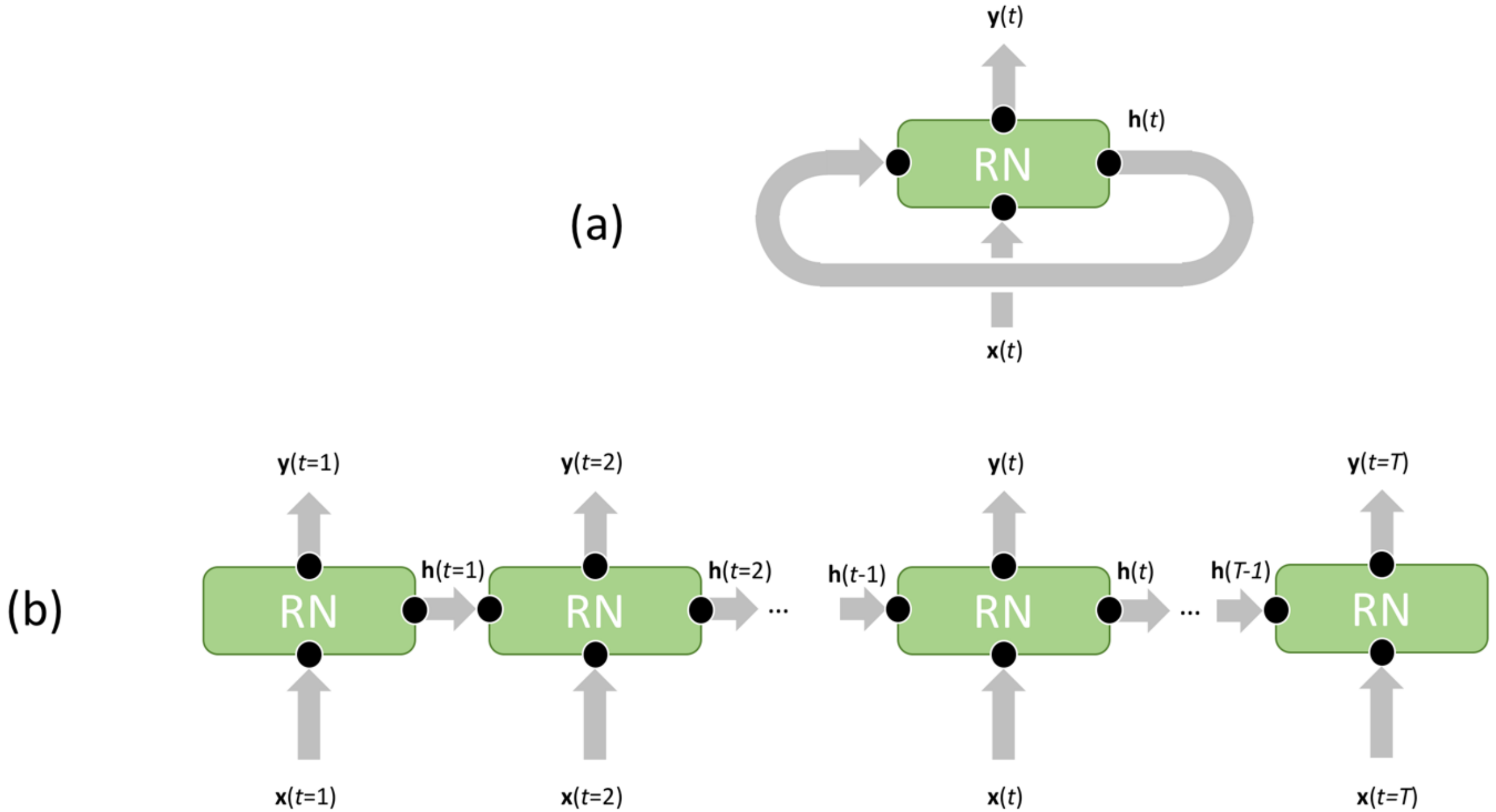
Entrée à instant t-1

Entrée à instant t



Sortie à instant t

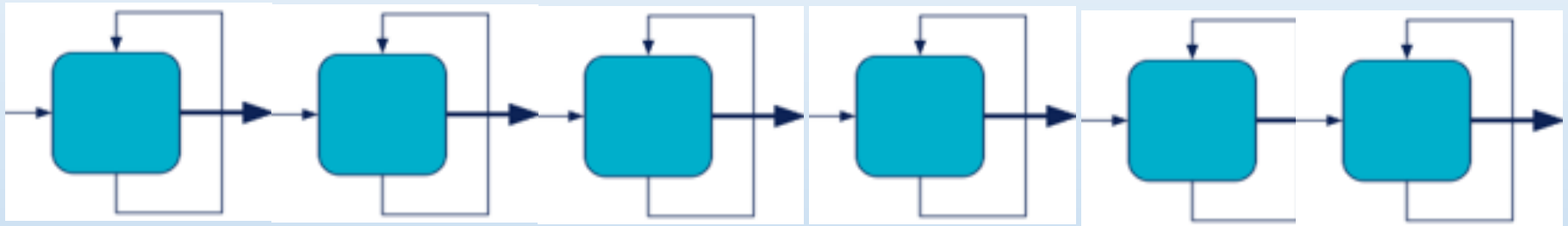
n est variable


(a)
y(t)
h(t)
RN
x(t)
(b)
y(t=1)
y(t=2)
y(t)
y(t=T)
h(t=1)
h(t=2)
h(t-1)
h(t)
h(T-1)
RN
RN
RN
RN
...
...
x(t=1)
x(t=2)
x(t)
x(t=T)

Pourquoi

Données séquentielles

Le chat mange goulûment la souris

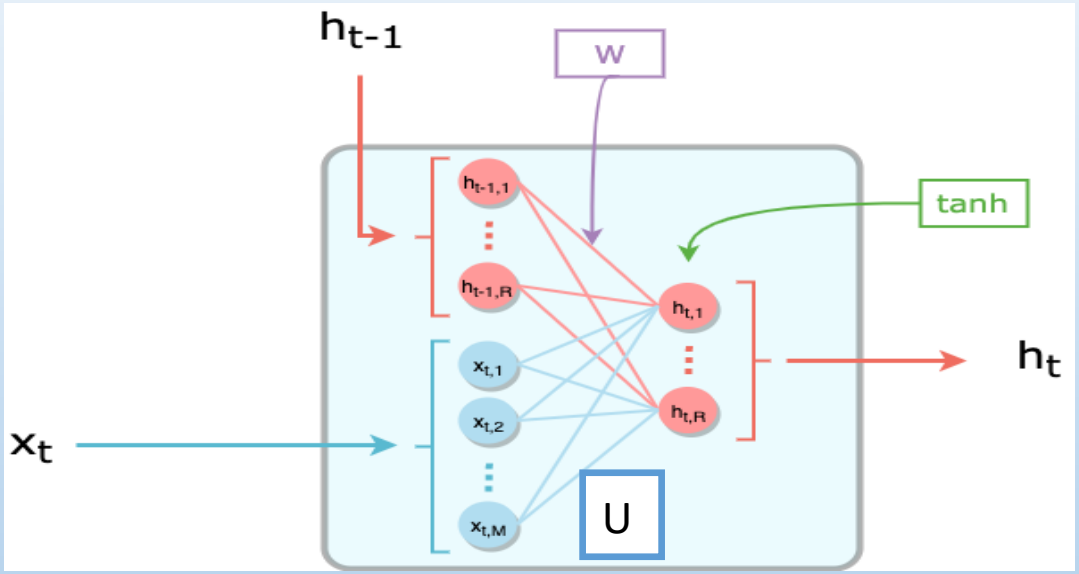


Vecteur de Pensée (Geoffrey Hilton)

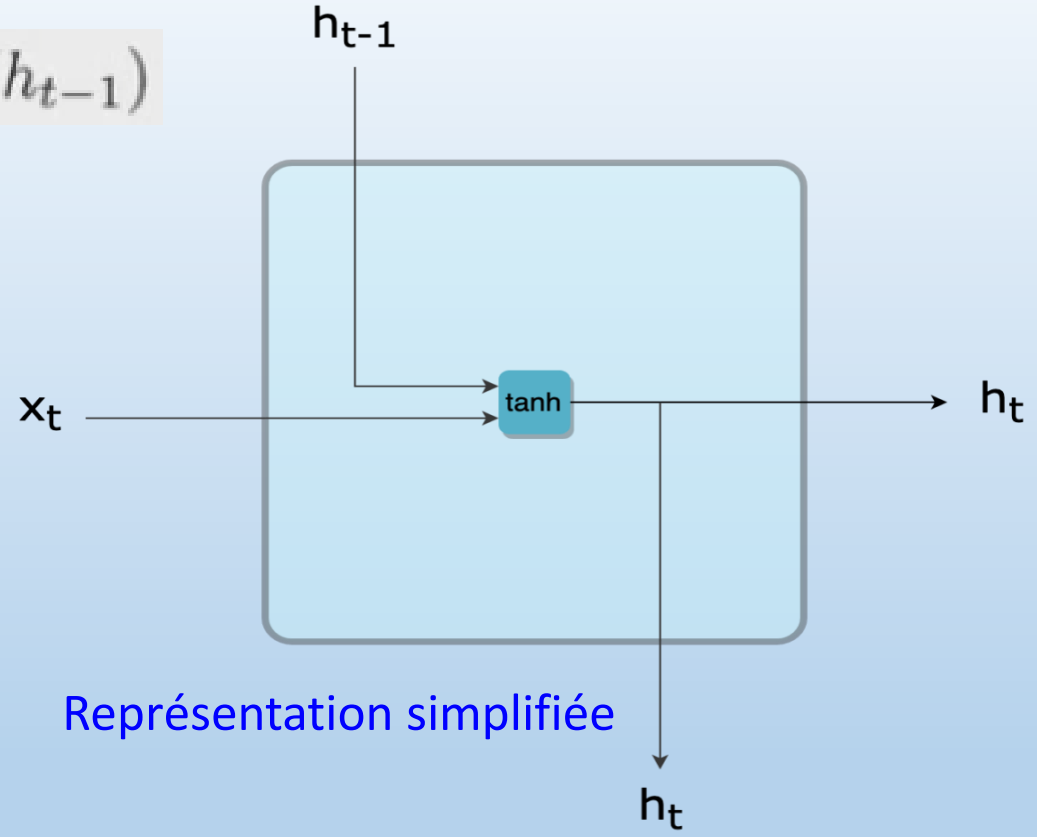
Réseau de Neurones Récurrent
Recurrent Neural Network

*Un RNN applique une fonction à une **séquence d'entrée** pour produire une **séquence de sortie** tout en conservant un **état interne***

$$h_t = \tanh(Ux_t + Wh_{t-1})$$



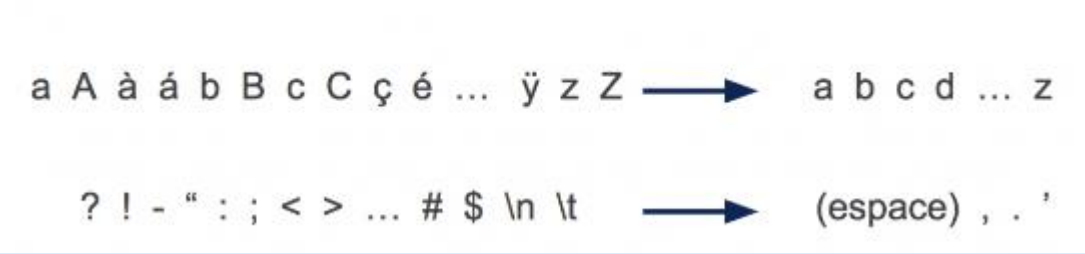
Détail



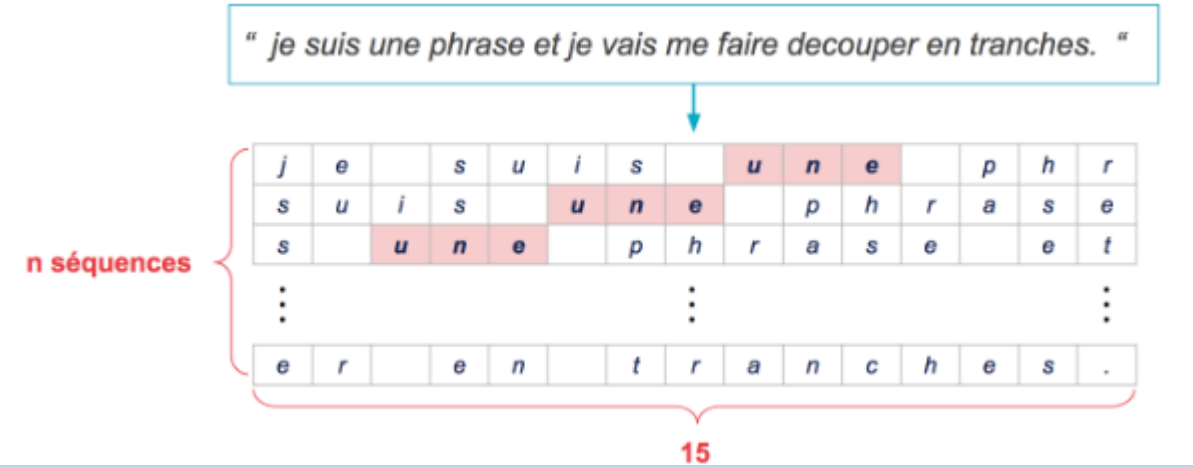
Représentation simplifiée

Préparation des données

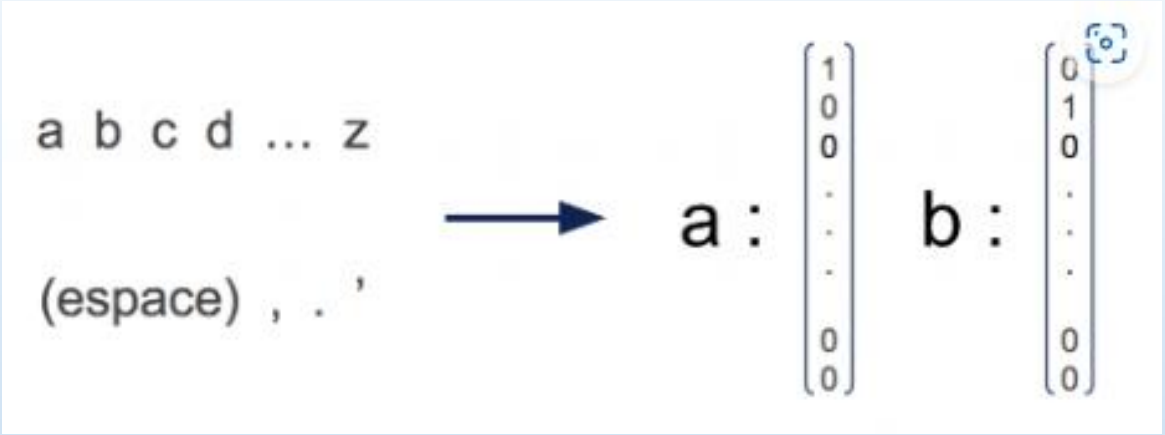
1 . Réduction vocabulaire



2 . Choix des séquences

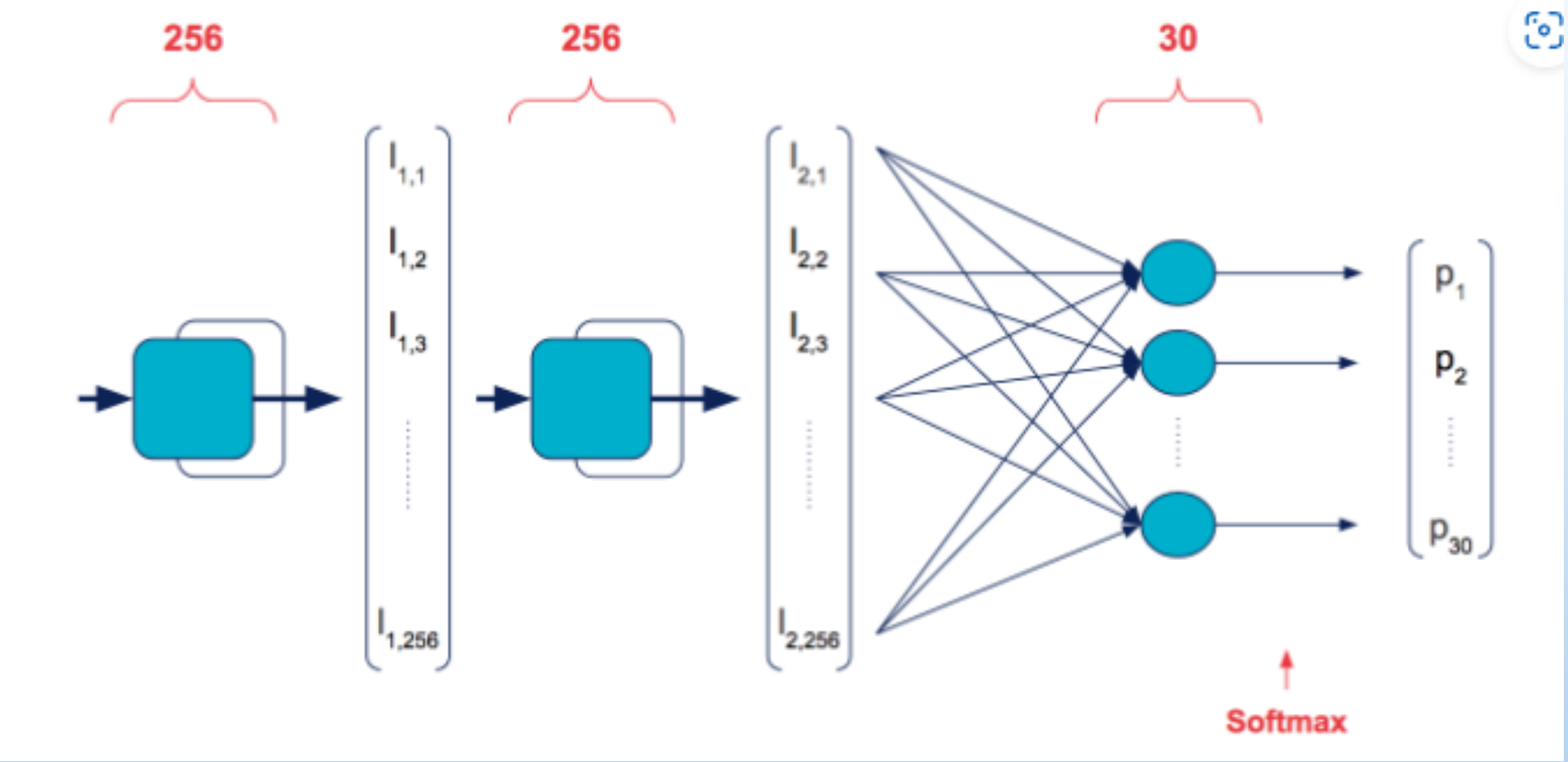


3. Encodage one-hot

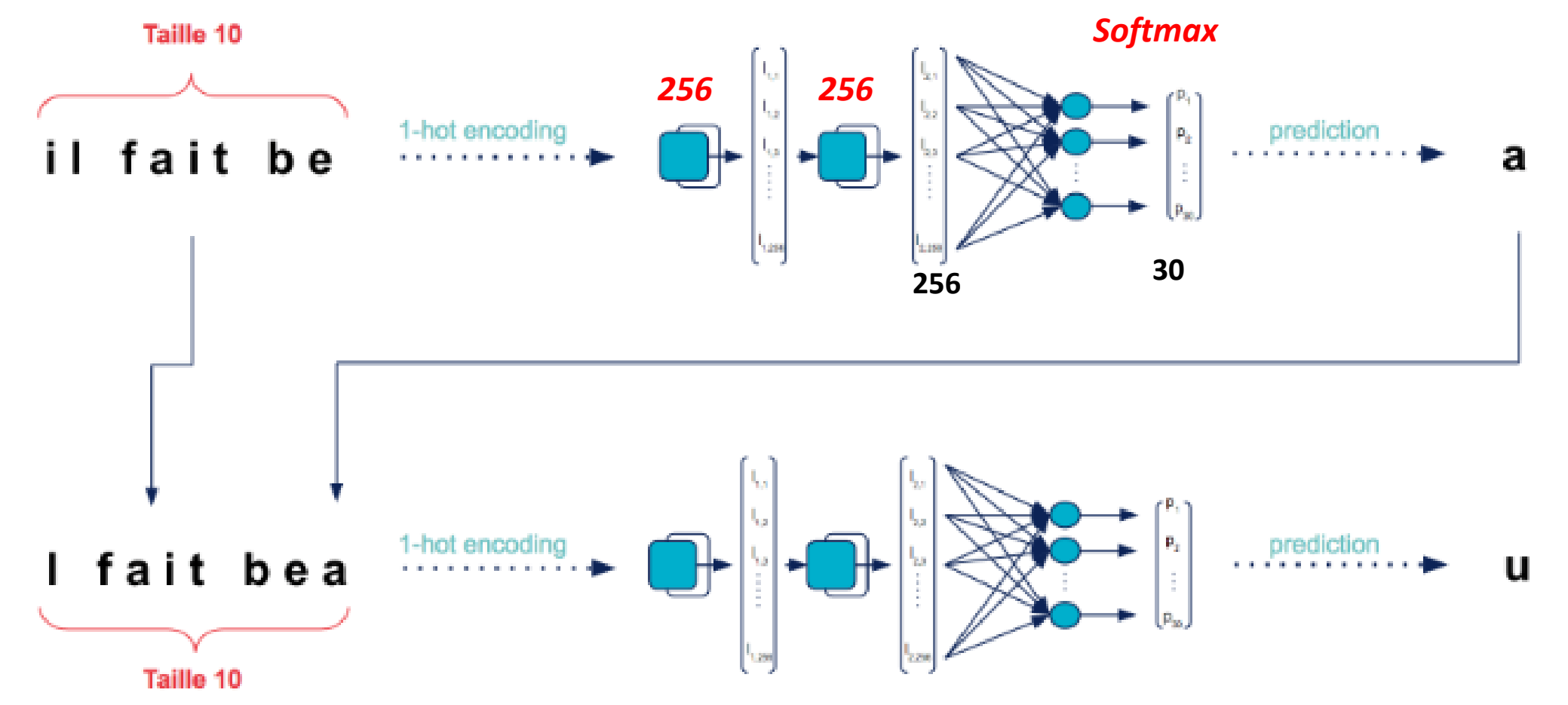


Exemple : Séquences de 15 caractères coulissant d’un pas de 3
Longueur et pas sont des hyper-paramètres

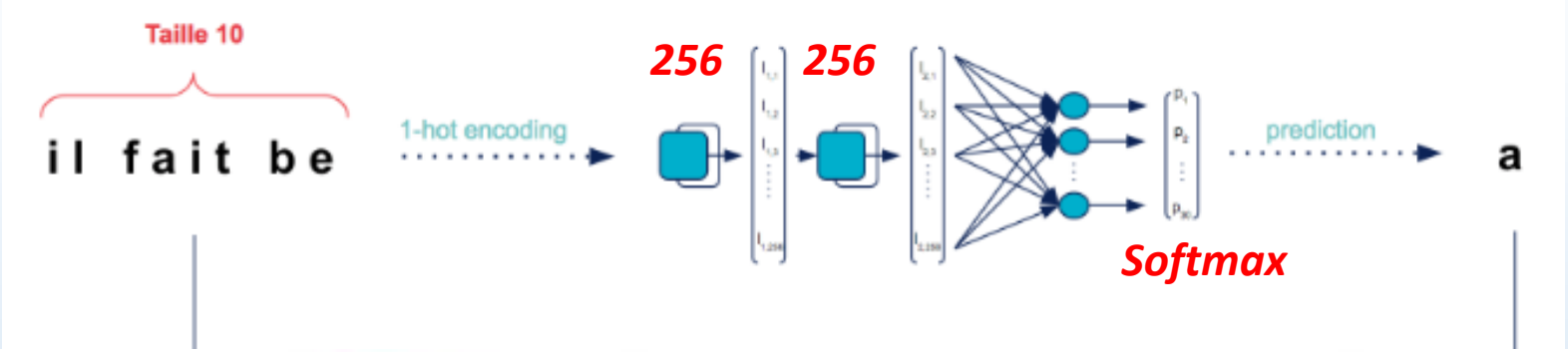
Exemple de réseau de RNN



Exemple : Génération de texte



Exemple : Génération de texte

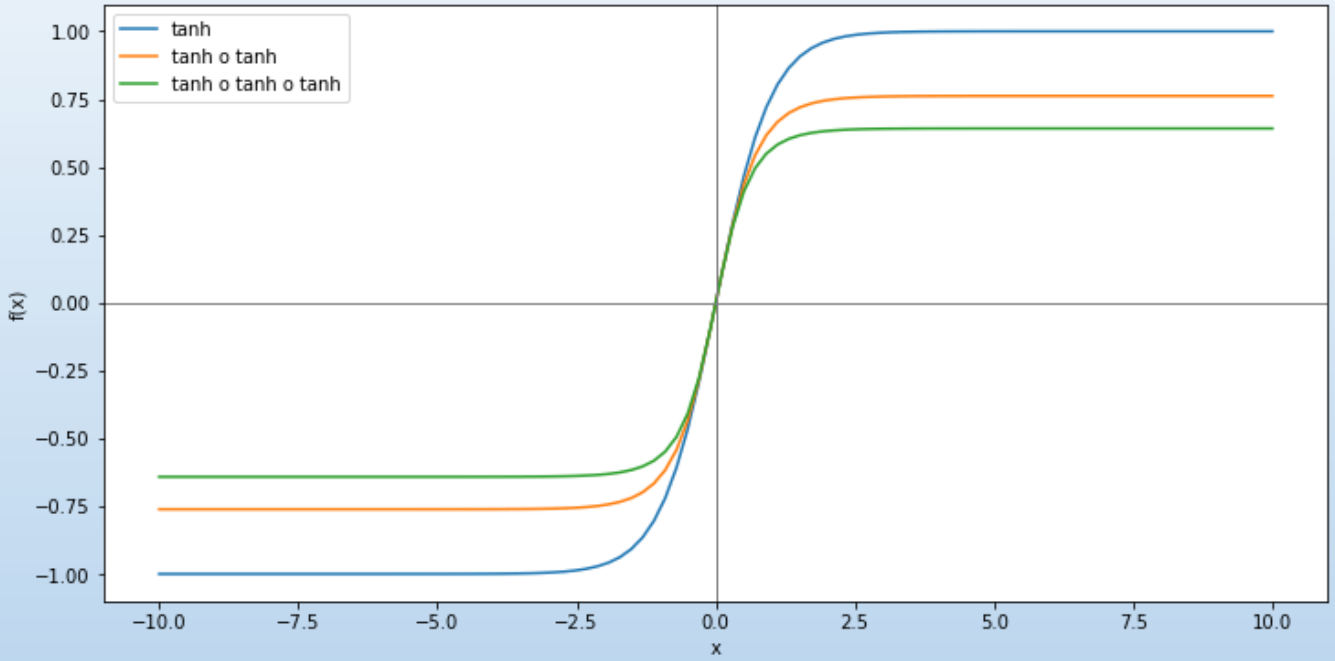



```
from keras.models import Sequential
from keras.layers import LSTM, Dense, Dropout, Activation

SEQ_LEN = 50
NB_CHARS = 30

model = Sequential()
model.add(LSTM(256, input_shape=(SEQ_LEN, NB_CHARS), return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(256))
model.add(Dropout(0.2))
model.add(Dense(NB_CHARS))
model.add(Activation('softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
```

Limites des RNN
1. Mémoire courte



.

“il prévien ».

Pour prédire la prochaine lettre, qui correspond à l’accord du verbe, il est nécessaire de connaître le sujet. Or, cette information se trouve en début de séquence. Le modèle aura potentiellement du mal à différencier cette séquence d’une autre qui lui ressemble:

« tu prévien ».

*Si $\mathbf{x}_1$ est cruciale pour la tâche de prédiction, son influence est **mécaniquement réduite** par les **passages successifs par la tangente hyperbolique**.*

La pensée s’est évanouie….

Vecteur de Pensée

Limites des RNN
2 . Modèle difficile à entraîner

Pour un réseau à 3 couches on va être amené à calculer le gradient de 3 fonctions composées de tanh, dont la dérivée st comprise entre 0 et 1

Or, plus on multiplie des valeurs entre 0 et 1 entre elles, plus le résultat se rapproche de 0.

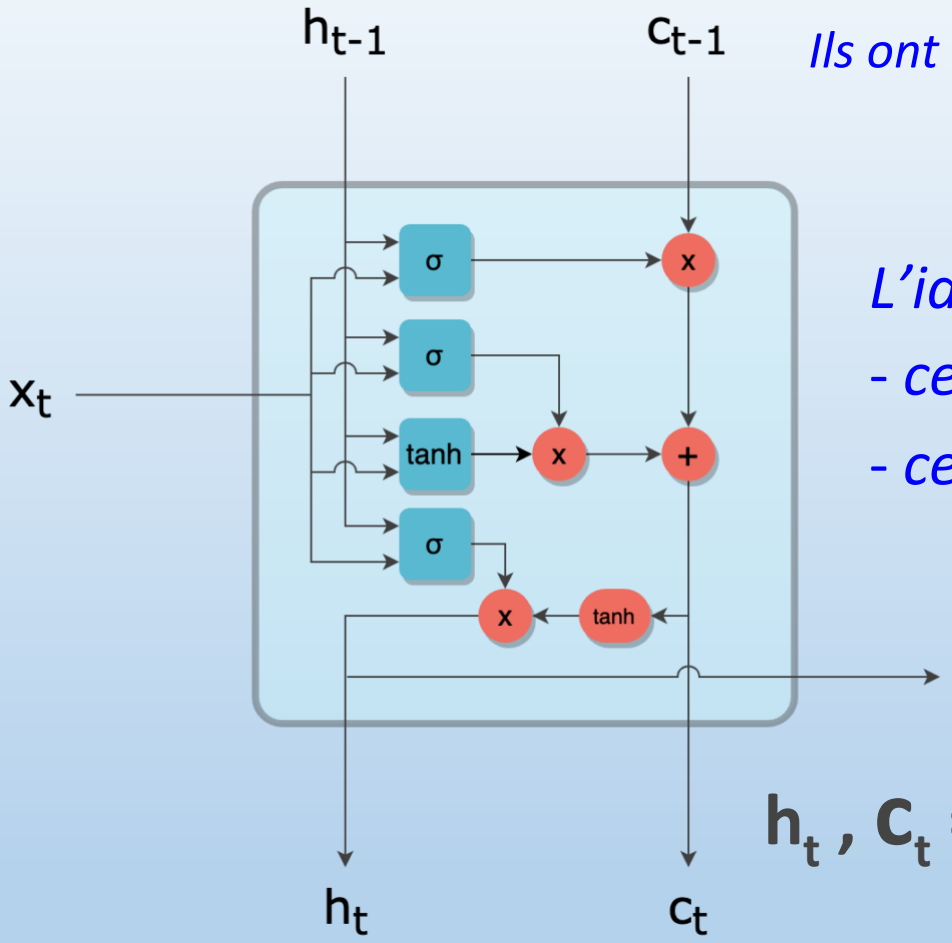
Le gradient prend donc des valeurs très petites lorsque les séquences sont longues.*

➔ *La mise à jour des paramètres devient donc* ***très lente*** *et l'entraînement du modèle est mis à mal.*

* *C'est ce qu'on appelle le problème du gradient évanescent*

Long Short Term Memory LSTM

Ce type de RNN est très utilisé en traitement du langage naturel



Ils ont été introduits par ***Hochreiter et Schmidthuber en 1997.***

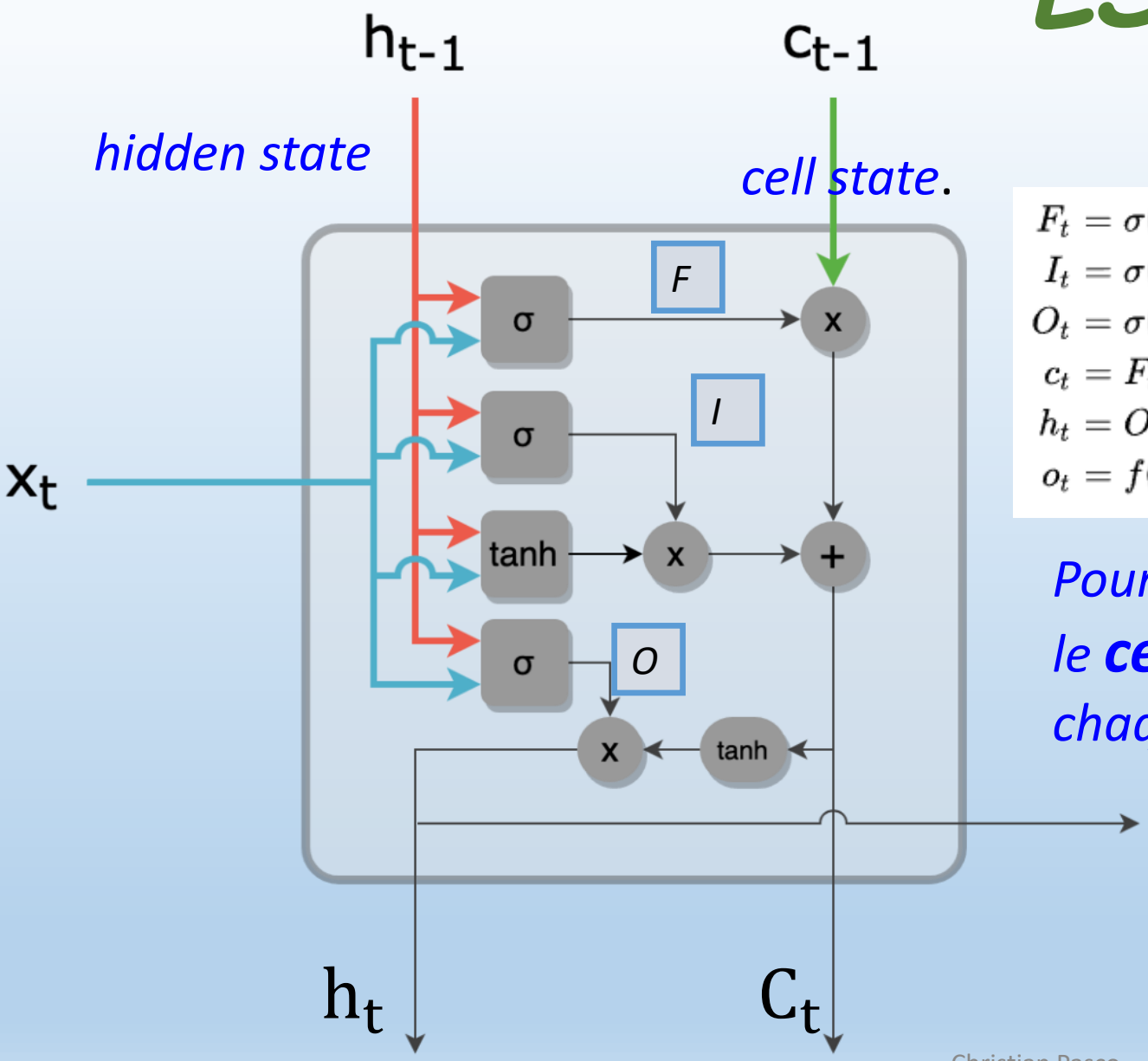
L'idée est de diviser le signal entre :

- ce qui est important à ***court terme hidden state*** *: h(t)*

- ce qui l'est à ***long terme****, à travers le* ***cell state : c(t)***

$$h_t , c_t = f(x_t , h_{t-1} , c_{t-1})$$

LSTM

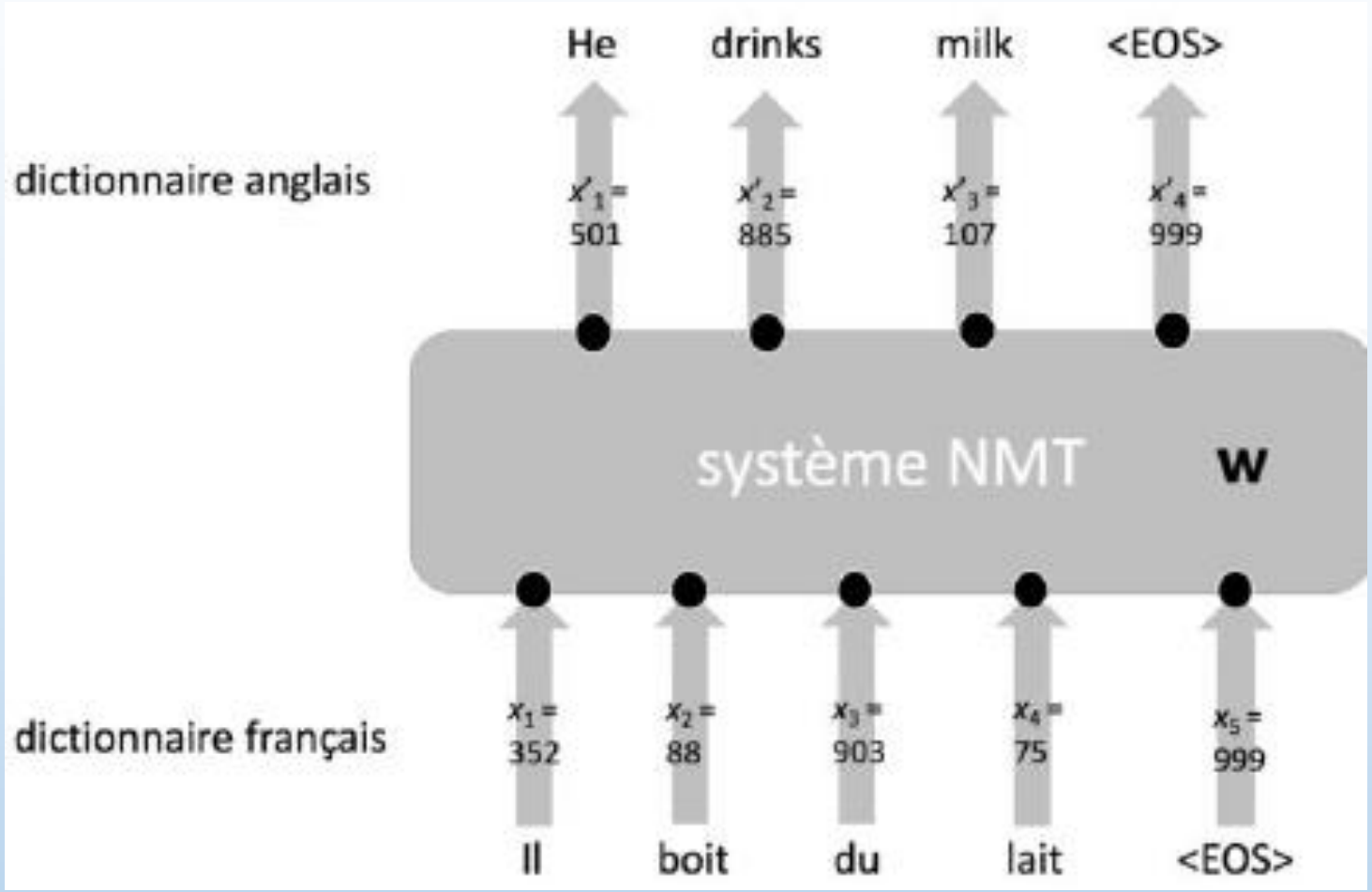


$$
\begin{aligned}
F_t &= \sigma(W_F x_t + U_F h_{t-1} + b_F) && \text{(forget gate)} \\
I_t &= \sigma(W_I x_t + U_I h_{t-1} + b_I) && \text{(input gate)} \\
O_t &= \sigma(W_O x_t + U_O h_{t-1} + b_O) && \text{(output gate)} \\
c_t &= F_t \circ c_{t-1} + I_t \circ \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= O_t \circ \tanh(c_t) \\
o_t &= f(W_o h_t + b_o)
\end{aligned}
$$

*Pour éviter le problème du vanishing gradient, le **cell state** est mis à jour de façon **additive** à chaque étape, sans passer par une activation.*

Système de traduction mot à mot

NMT : Neural Machine Translation

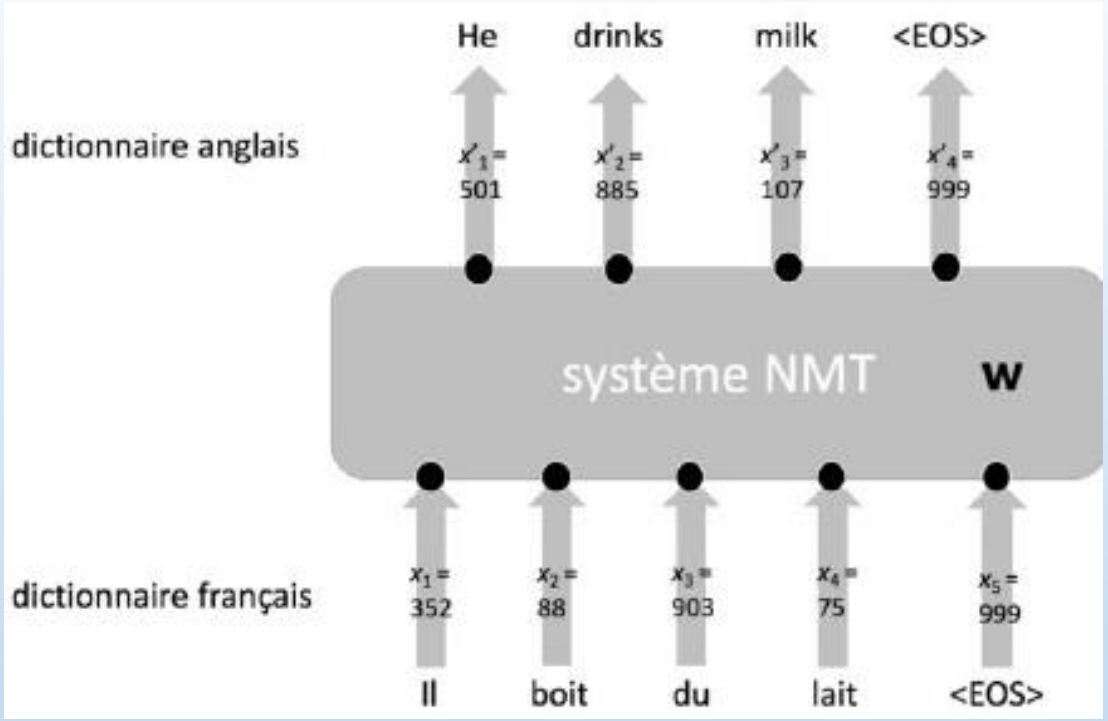


$$x'_1, x'_2, x'_3, x'_4, \ldots, x'_n = T(x_1, x_2, x_3, x_4, \ldots, x_m)$$

Entrainement du NMT

*On lui présente en **entrée des mots et des phrases** ainsi que leur **traduction correcte**.*
*Chaque **traduction correcte** d'un mot est une **distribution qui attribue une probabilité 1 au mot correct et 0 à tous les autres***
Il s'agit par conséquent de comparer, pour chaque mot du vocabulaire source, cette distribution correcte à celle prédite par le système NMT.

*Entraîner le système revient dès lors à chercher des paramètres **w tels que, en moyenne sur tous les mots du vocabulaire source, ces deux distributions soient aussi proches que possible***



Couches complètement connectées

Principe

- *définit une fonction (avec des paramètres ajustables w) qui transforme un vecteur en entrée x=(x1, x2, …, xn) en un autre vecteur h=(h1, h2, …, hm) en sortie et ainsi de suite*

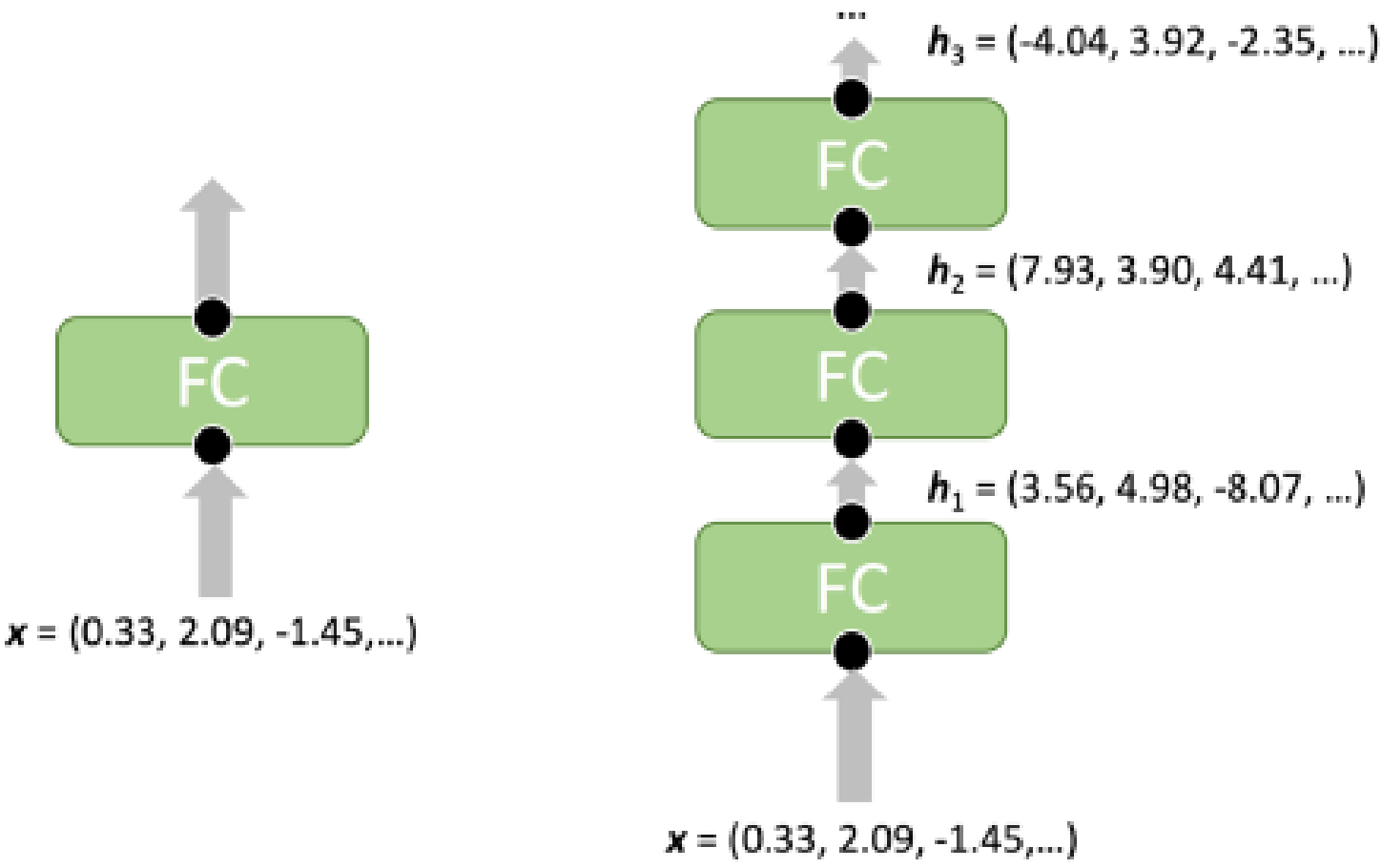


Figure 4 Un module FC et une pile de 3 module FC

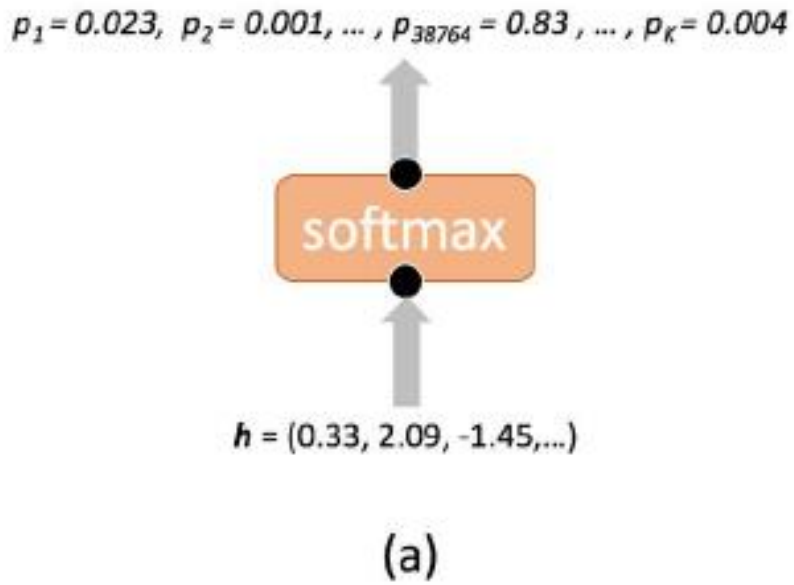
Couche Softmax

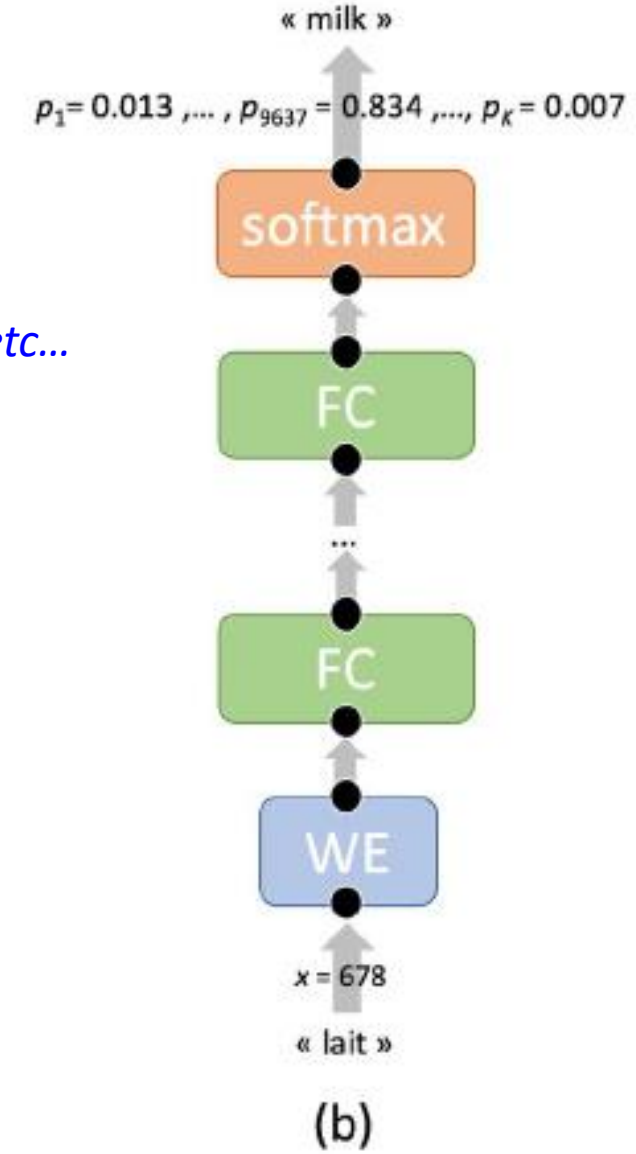
Principe

• *Calcule une distribution de probabilité sur les K mots du vocabulaire parmi lesquels on retiendra le plus probable.*

Exemple

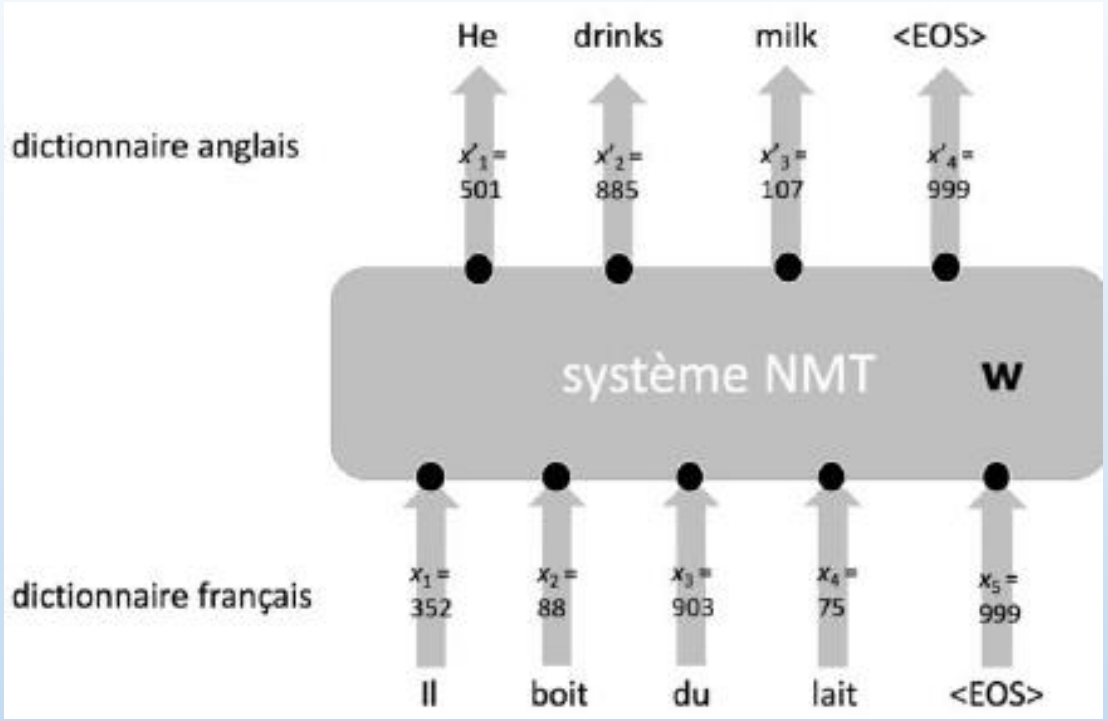
• *token n° 357 « milk », ➔ p357=0.834*
• *token n° 503 « wacky » ➔ p503=0.007 etc…*





Limites de la traduction mot à mot

*Ce **système de traduction mot à mot** n'a évidemment pas grand intérêt pratique car on pourrait tout aussi bien lui substituer un **dictionnaire bilingue !***



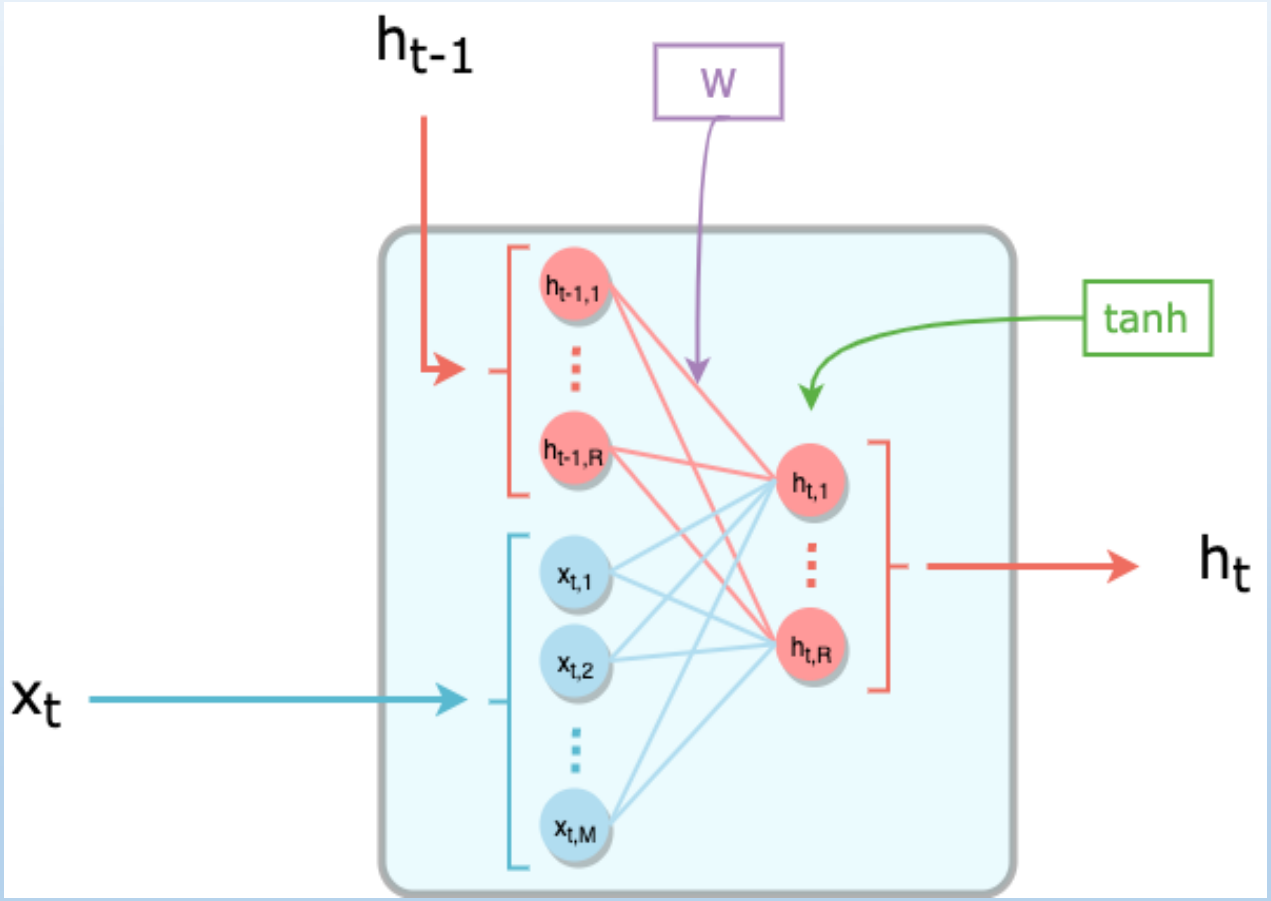
Comment prendre en compte le contexte d'un mot ?

La traduction automatique
NMT
Neural Machine Translation

Utilisation des Réseaux de neurones récurrents RNN et des Long Short Term Memory LSTM

RNN

Dans un réseau récurrent la sortie y(t) dépend à la fois de l'entrée x(t) et de variables cachées h(t-1) définies à l'instant précédent.

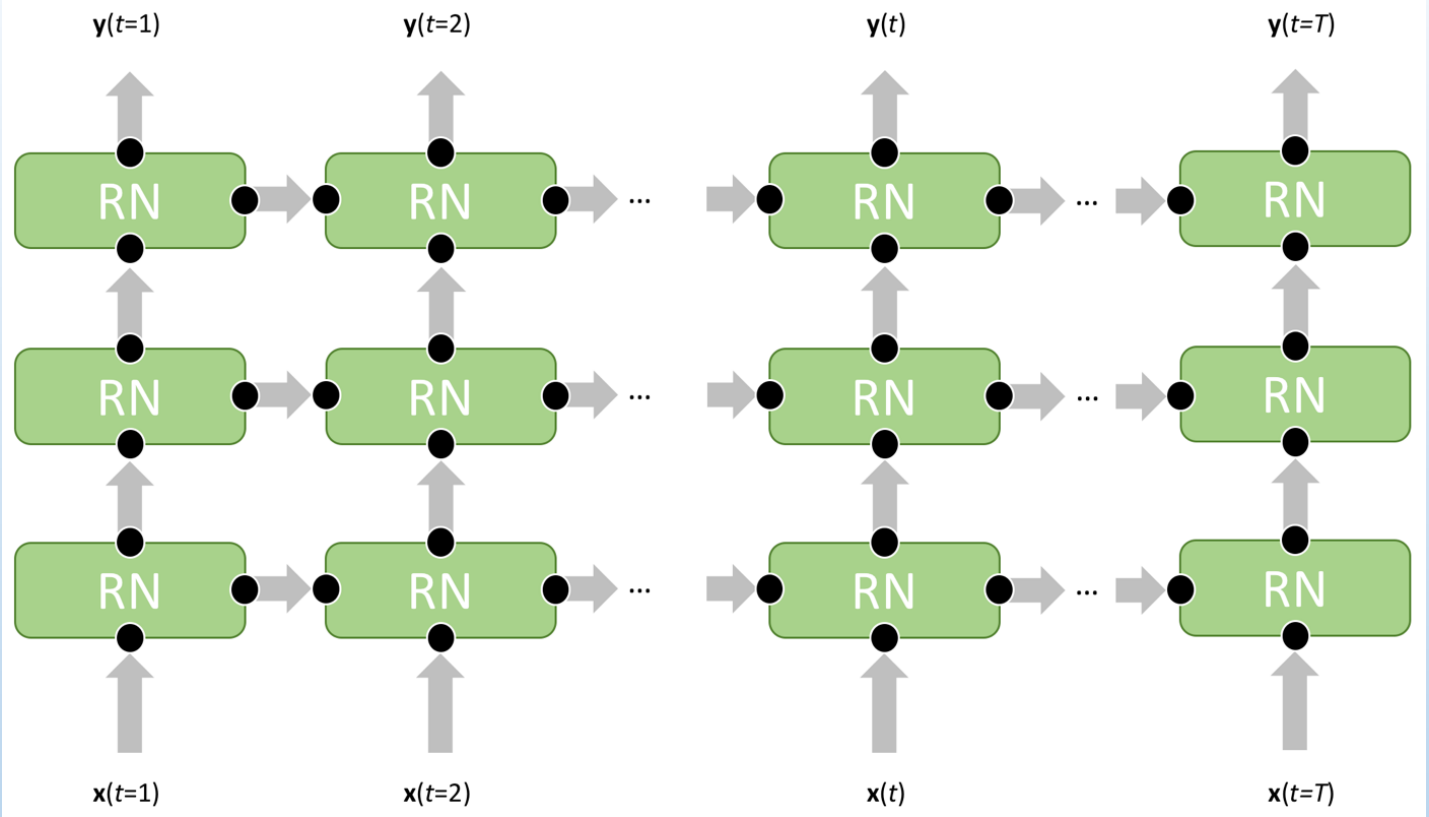


$$Y(t) = f(x_t\ h_{t-1})$$

Empilement de RNN

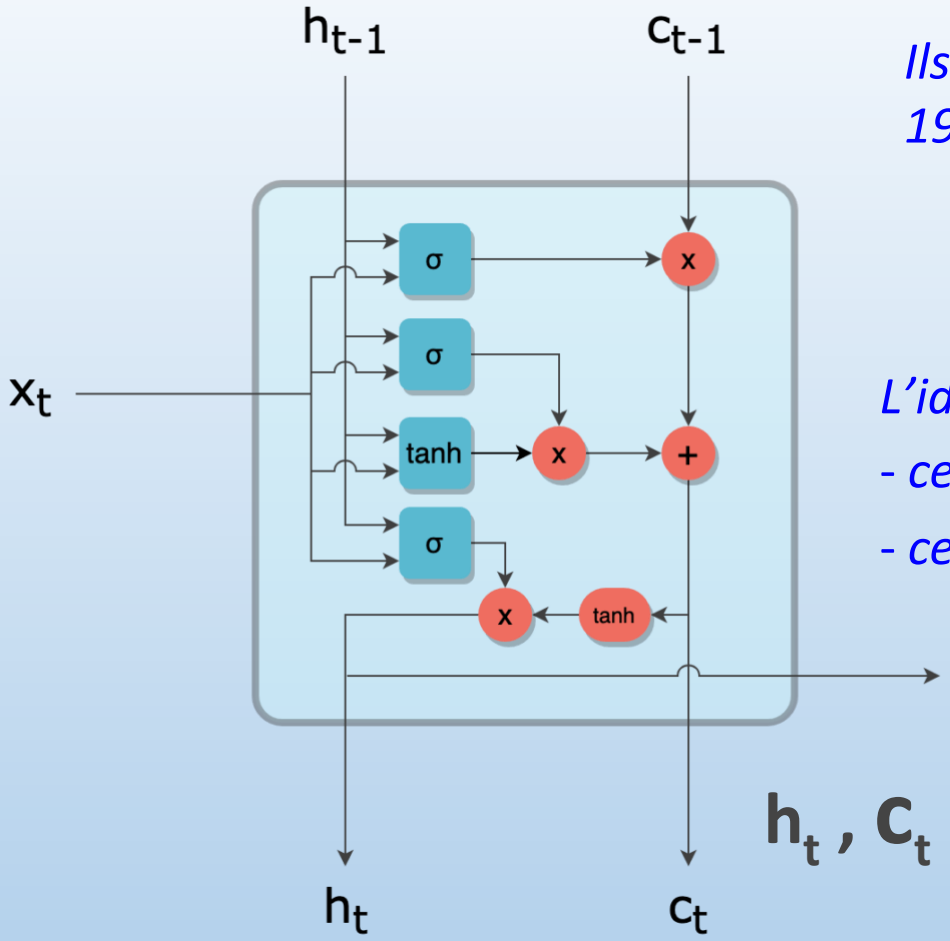
On peut empiler des RNN pour apprendre des transformations complexes de suites de vecteurs.

L'expérience montrent que ce RNN fonctionne mal car le réseau a tendance à ***perdre la mémoire*** *:*
un événement x(t') qui survient dans un passé lointain (t' << t) d'un événement x(t) n'a que très peu d'influence sur la sortie y(t) ce qui est t un handicap pour la traduction de phrases longues..



Long Short Term Memory LSTM

Ce type de RNN est très utilisé en traitement du langage naturel



Ils ont été introduits par Hochreiter et Schmidthuber en 1997.

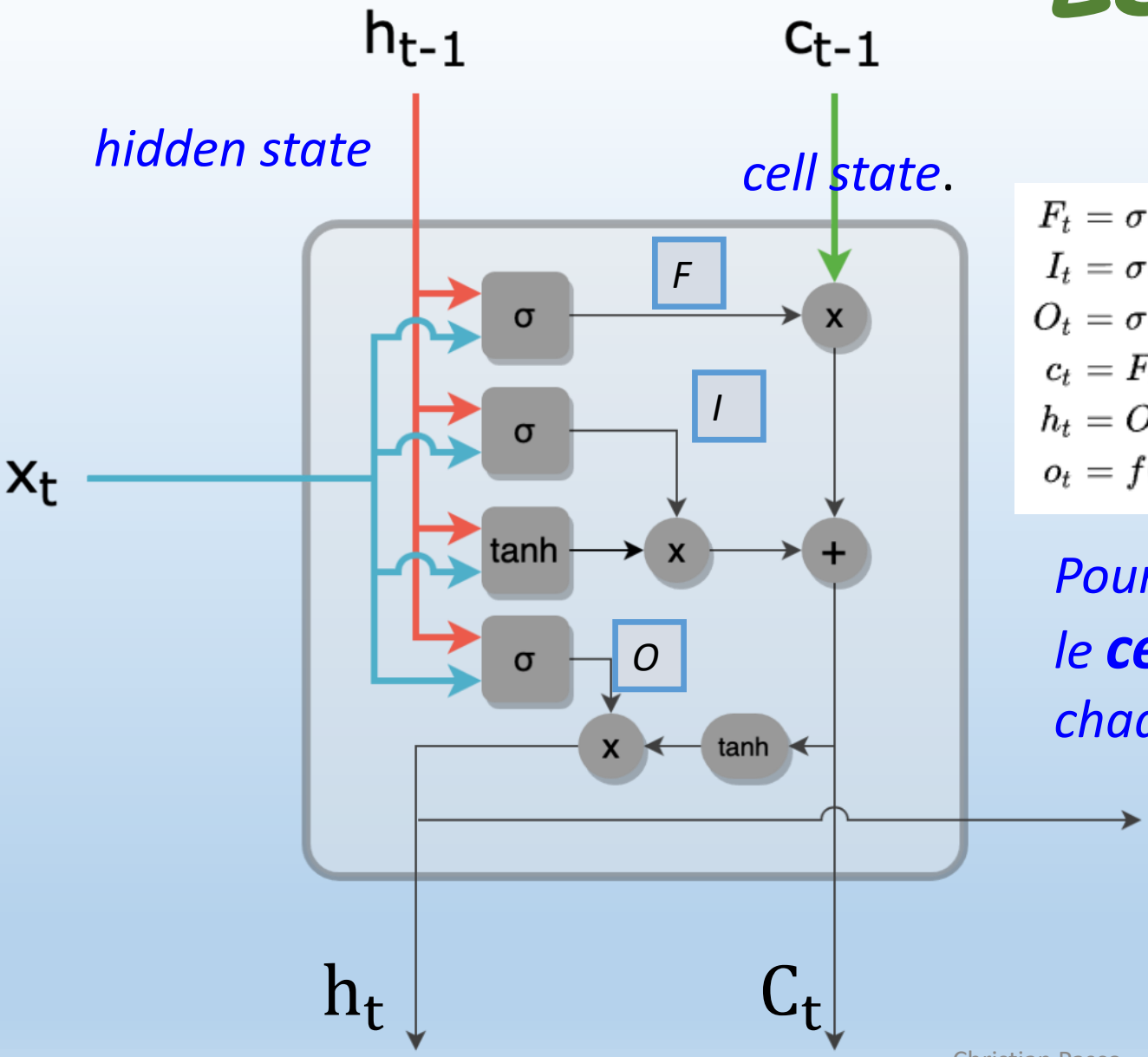
L'idée est de diviser le signal entre :

- ce qui est important à ***court terme*** ***hidden state*** *: h(t)*

- ce qui l'est à ***long terme****, à travers le* ***cell state : c(t)***

$$h_t , c_t = f(x_t , h_{t-1} , c_{t-1})$$

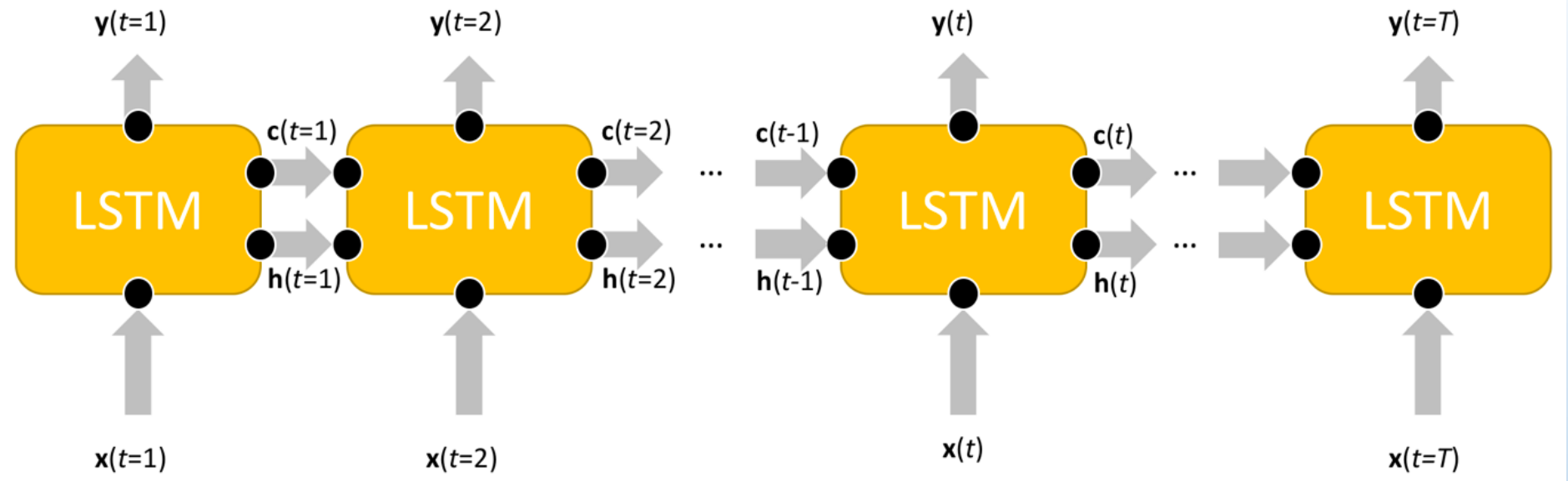
LSTM en détail



$$
\begin{aligned}
F_t &= \sigma(W_F x_t + U_F h_{t-1} + b_F) && \text{(forget gate)} \\
I_t &= \sigma(W_I x_t + U_I h_{t-1} + b_I) && \text{(input gate)} \\
O_t &= \sigma(W_O x_t + U_O h_{t-1} + b_O) && \text{(output gate)} \\
c_t &= F_t \circ c_{t-1} + I_t \circ \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= O_t \circ \tanh(c_t) \\
o_t &= f(W_o h_t + b_o)
\end{aligned}
$$

*Pour éviter le problème du vanishing gradient, le **cell state** est mis à jour de façon **additive** à chaque étape, sans passer par une activation.*

Les réseaux avec mémoire longue LSTM



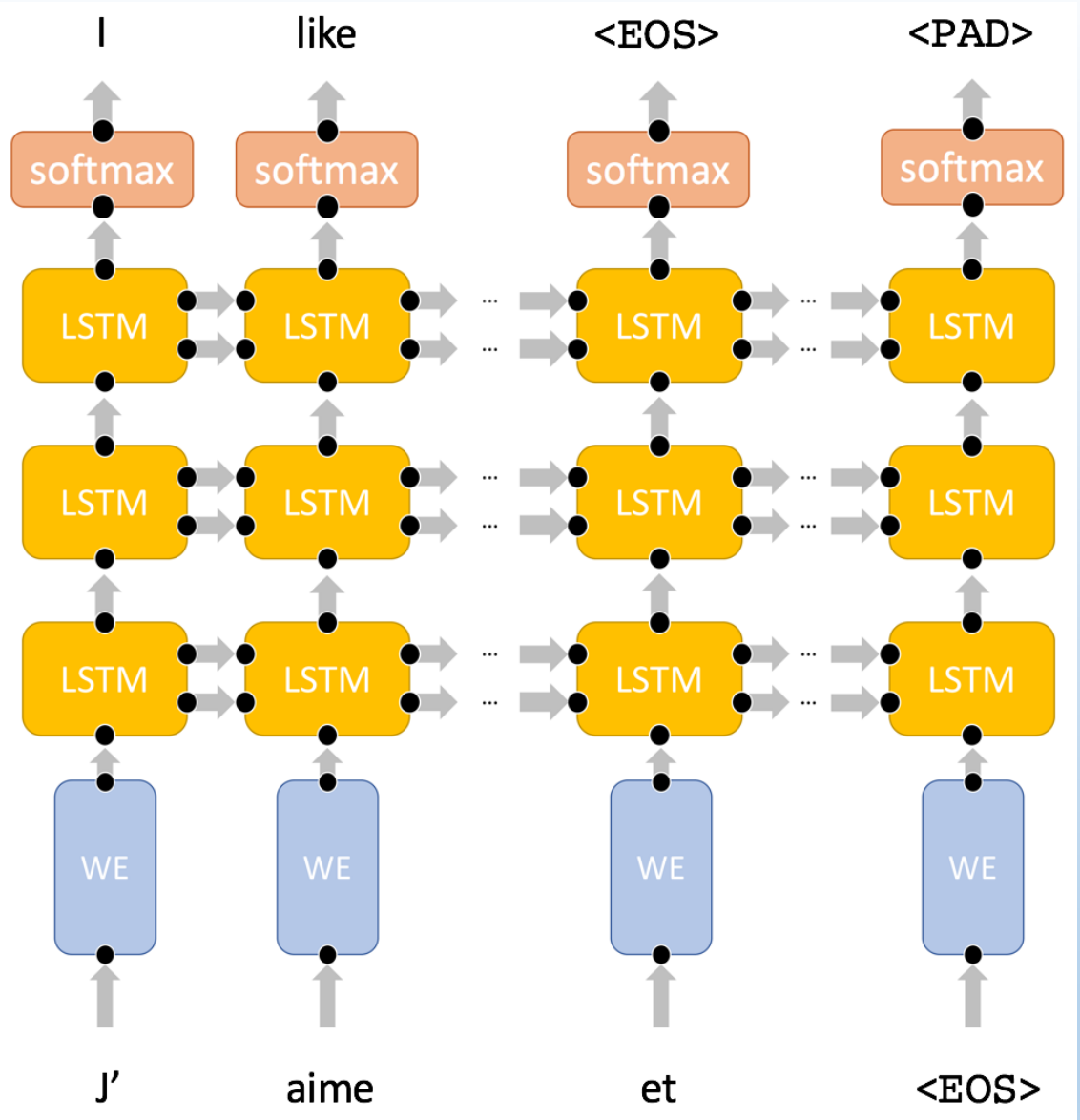
h(t) : mémoire à court terme,

c(t) : mémoire à long terme.

Les paramètres sont « appris » lors de l'entraînement du système :

➔ quelles parties de l'information dans c(t) il convient d'oublier, d'ajouter et de transmettre pour créer c(t+1).

RNN LSTM



*Une limitation importante des **RNN** tient au fait que la longueur des suites en sortie est toujours égale à la longueur des suites en entrée du système.*

*Avec les **deux symboles<EOS> et <PAD>,** désignant la fin d'une phrase et un symbole vide, et en choisissant pour la durée T celle **de la plus longue** des phrases en entrée ou en sortie, on résout en partie ce problème*

Entrainement du NMT

*On lui présente en **entrée** des **mots** et des **phrases** ainsi que leur **traduction correcte**.*
*Chaque **traduction correcte** d'un mot est une **distribution qui attribue une probabilité 1 au mot correct et 0 à tous les autres***
Il s'agit par conséquent de comparer, pour chaque mot du vocabulaire source, cette distribution correcte à celle prédite par le système NMT.

*Entraîner le système revient dès lors à chercher des paramètres **w tels que, en moyenne sur tous les mots du vocabulaire source, ces deux distributions soient aussi proches que possible***

RNN LSTM

*Ces structures qui traitent des données **séquentielles,** sont à la base de grandes avancées en NLP :*

- *modélisation du langage,*
- *traduction*
- *résumé automatique de texte.*

Limitations des RNN LSTM

L'un des défauts des LSTM est qu'il est nécessaire de lire entièrement une séquence pour produire une prédiction.
En traduction par exemple, cette démarche reviendrait à lire entièrement une phrase en mémorisant tous ses mots, pour ensuite produire une phrase traduite d'un seul coup.
On imagine mal un humain procéder de la même façon.

En effet avec les séquentiels, on ne peut contextualiser avec un mot suivant …
*D'où les **LSTM bidirectionnels** **Bi RNN et BI LSTM très utilisés***

.

Limitations des RNN LSTM

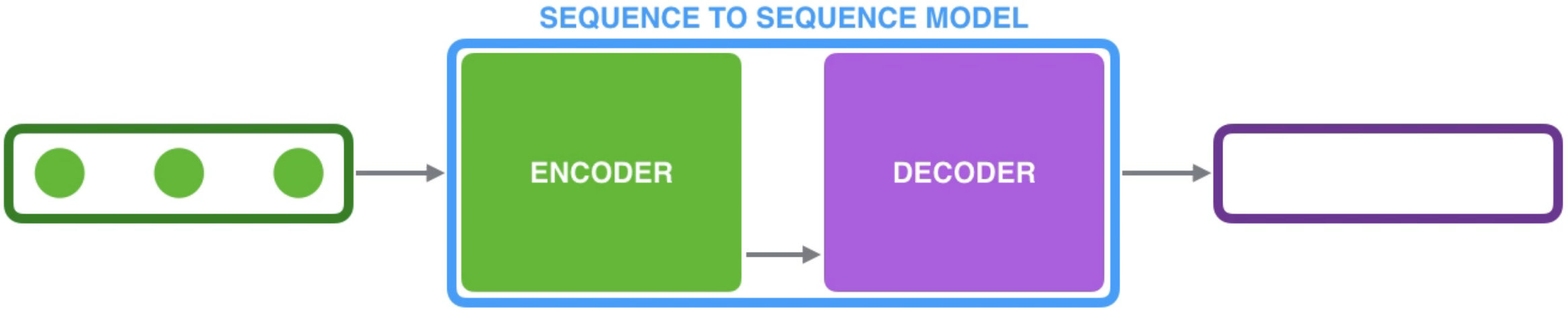
L'expérience a toutefois montré que cette architecture fonctionne mal car les mots en sortie du RNN restent ***trop corrélés aux mots en entrée au même instant t,*** *conduisant à une* ***traduction de piètre qualité.***
Dans les phrases longues, le premier mot de la phrase est vite oublié

La pensée est incomplète et se perd dans le temps ….

Vecteur de Pensée

Pour traiter les phrases de longueur variable
➔ encodeur décodeur

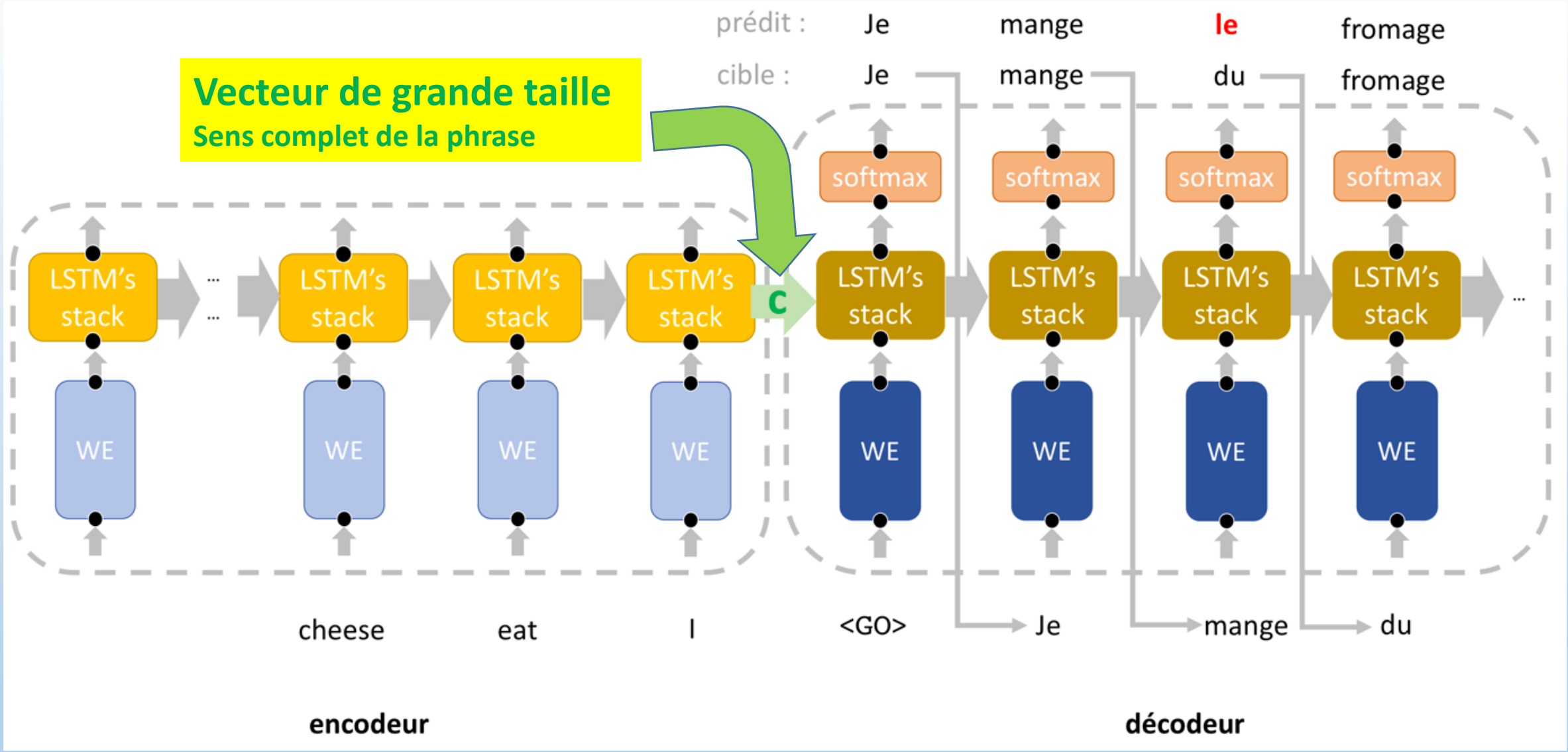
Le schéma encodeur décodeur



La traduction mot à mot ne donne pas de bons résultats
Il est préférable d'avoir « digéré » la phrase entière avant de commencer la traduction

➔ *encodeur décodeur*

Le schéma encodeur décodeur



Limitations

Limitation de la mémoire

Prédiction du prochain mot

Boston est une très belle ville, j'y ai vécu trois ans, ce fut une expérience très enrichissante et surtout ça m'a permis d'apprendre à bien mieux parler ...

Pour les phrases longues, la pensée reste encore

incomplète et se perd dans le temps ….

Vecteur de Pensée

Mécanisme d'Attention

Le mécanisme d'attention pour la traduction automatique

Vecteur de contexte c de dimension fixe en sortie de l'encodeur

➔*Sens d'une phrase longue plus difficile à encapsuler que celui d'une phrase courte*
➔*Qualité traduction décroît avec longueur des phrases*

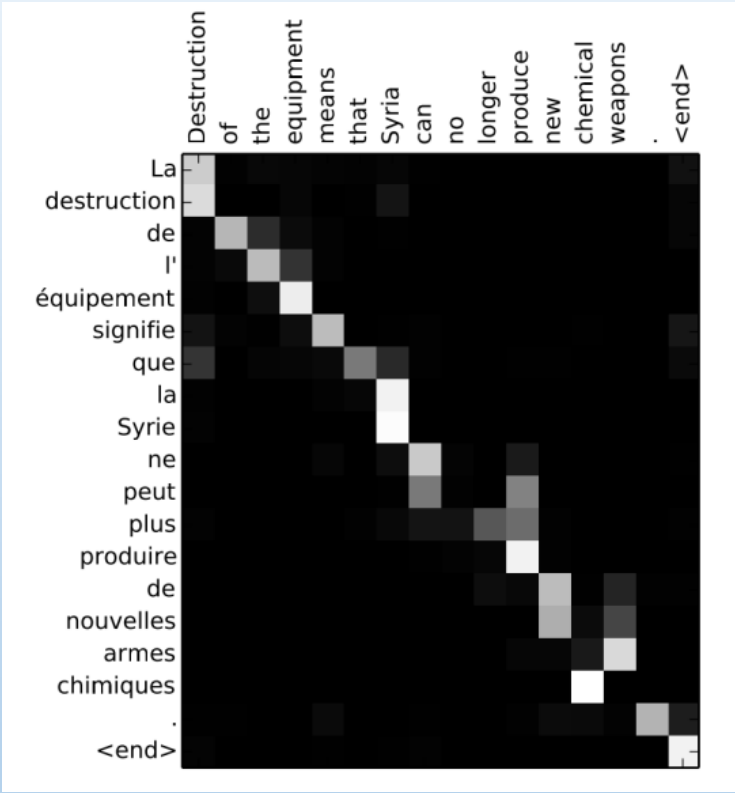
***Idée** : Vecteur de contexte dynamique c(t), différent à chaque instant t*

Le contenu d'un tel vecteur c(t) devrait correctement refléter le ***contexte sémantique*** *des mots les plus pertinents dans la phrase source selon le mot à générer par le décodeur à l'instant t.*

C'est là précisément le cœur d'un mécanisme d'attention (MA).

Le mécanisme d'attention en IA

- *Le mécanisme d'attention (MA), peut clairement être assimilé à un mécanisme librement inspiré du fonctionnement de notre propre cortex cérébral*.
- *Il s'agit d'un mécanisme capable d'apprendre les contextes entre les mots d'un texte*

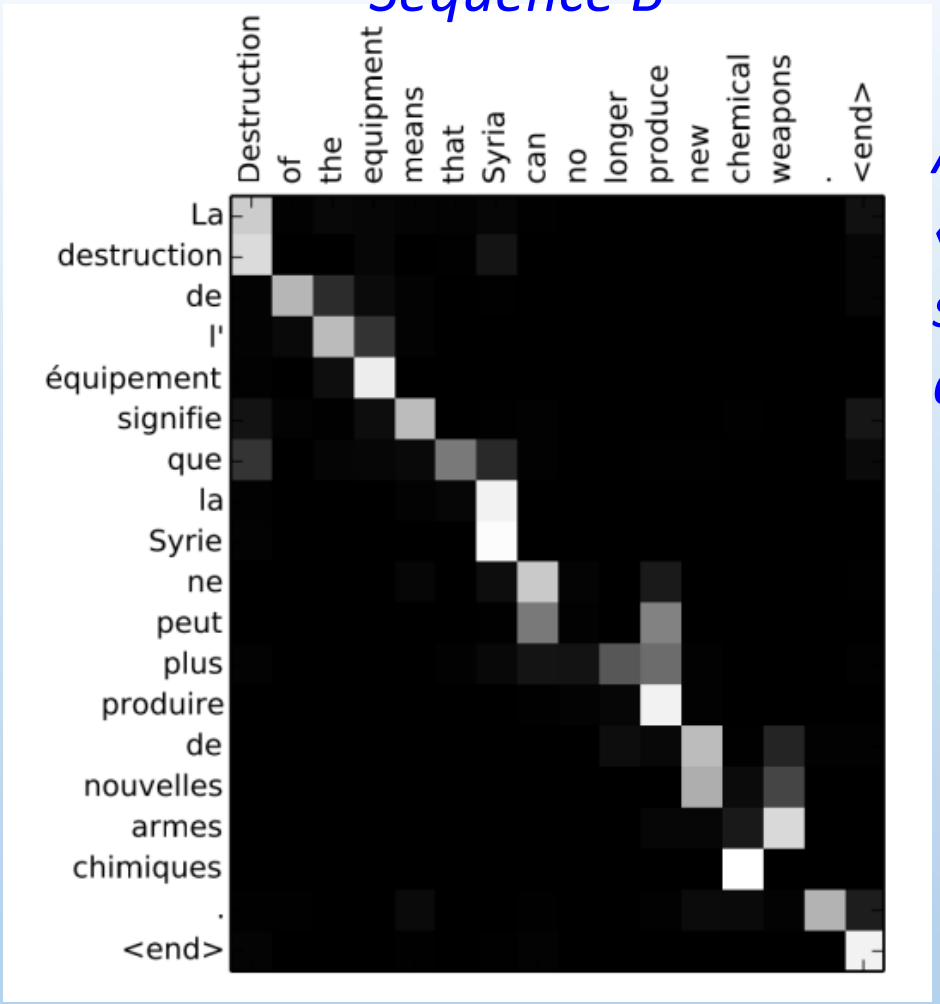


Pirmin Lemberger
Directeur Scientifique Onepoint
2018

Mécanisme d'attention

Séquence B

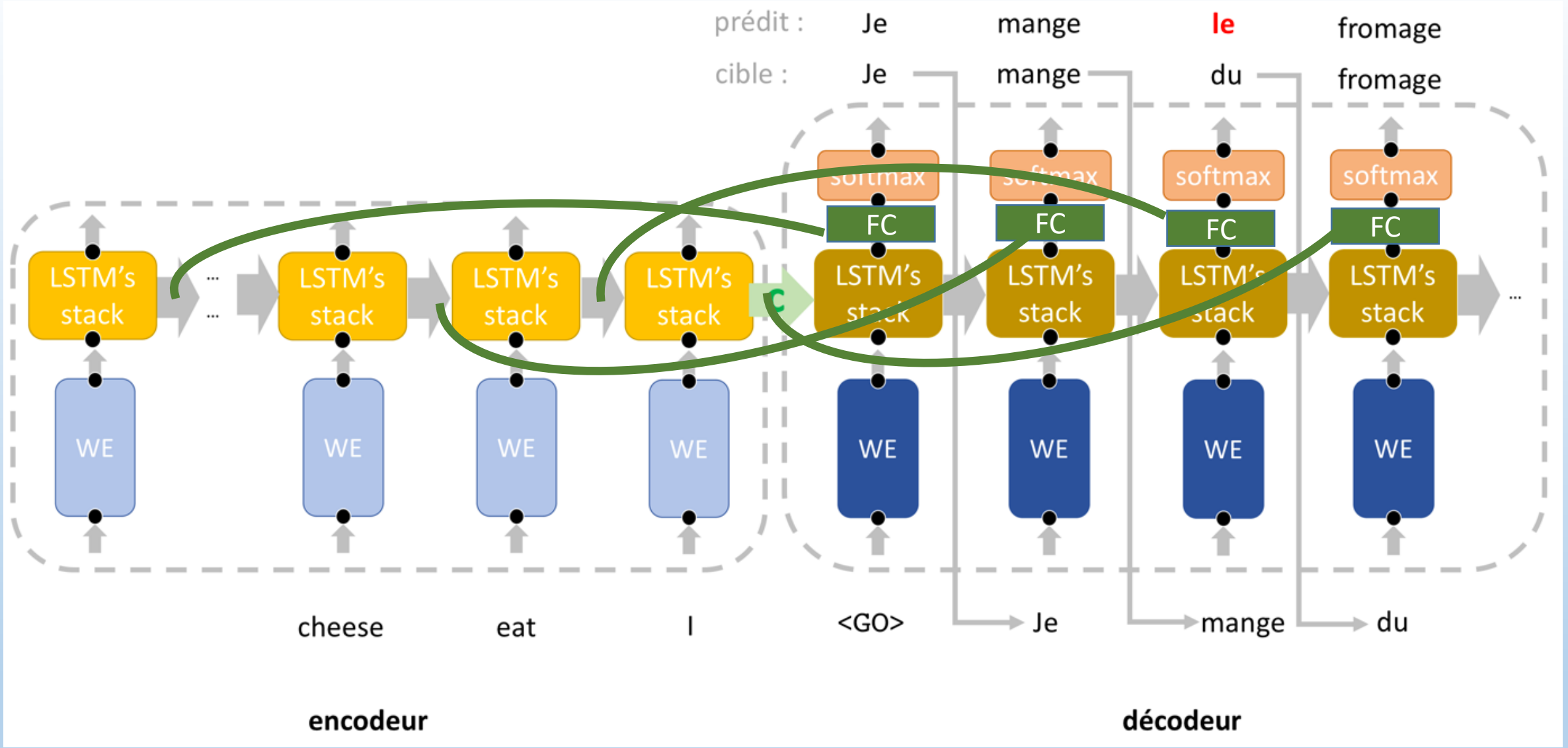
Séquence A



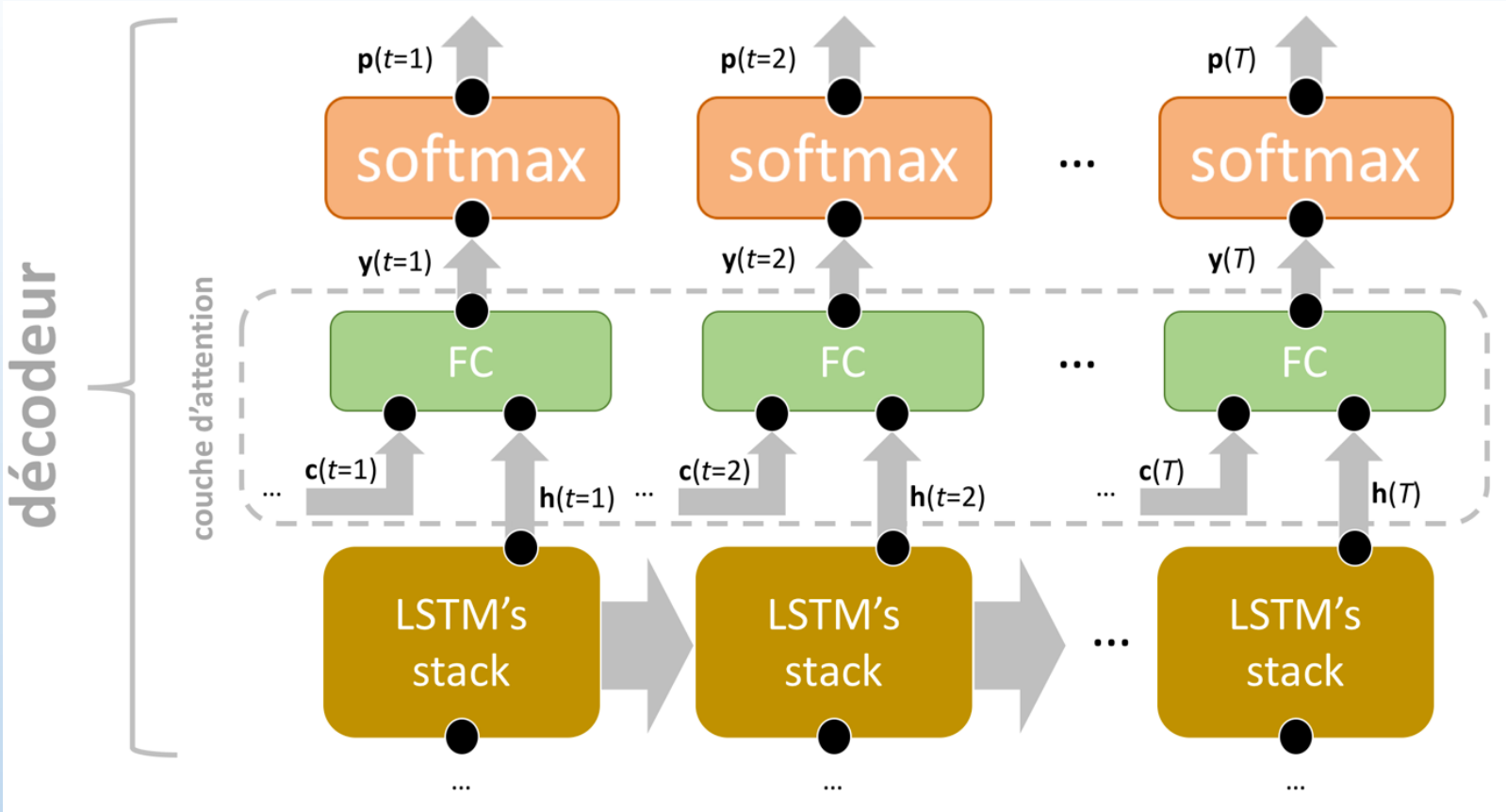
A pour but de
« dire, au reste du modèle, à quels mots de la séquence B il faut porter le plus d'attention quand on traite un mot de la séquence A ».

Les carrés clairs sont ceux pour lesquels l'attention est importante c.à.d. que le poids est proche de 1

L'attention dans le schéma encodeur décodeur

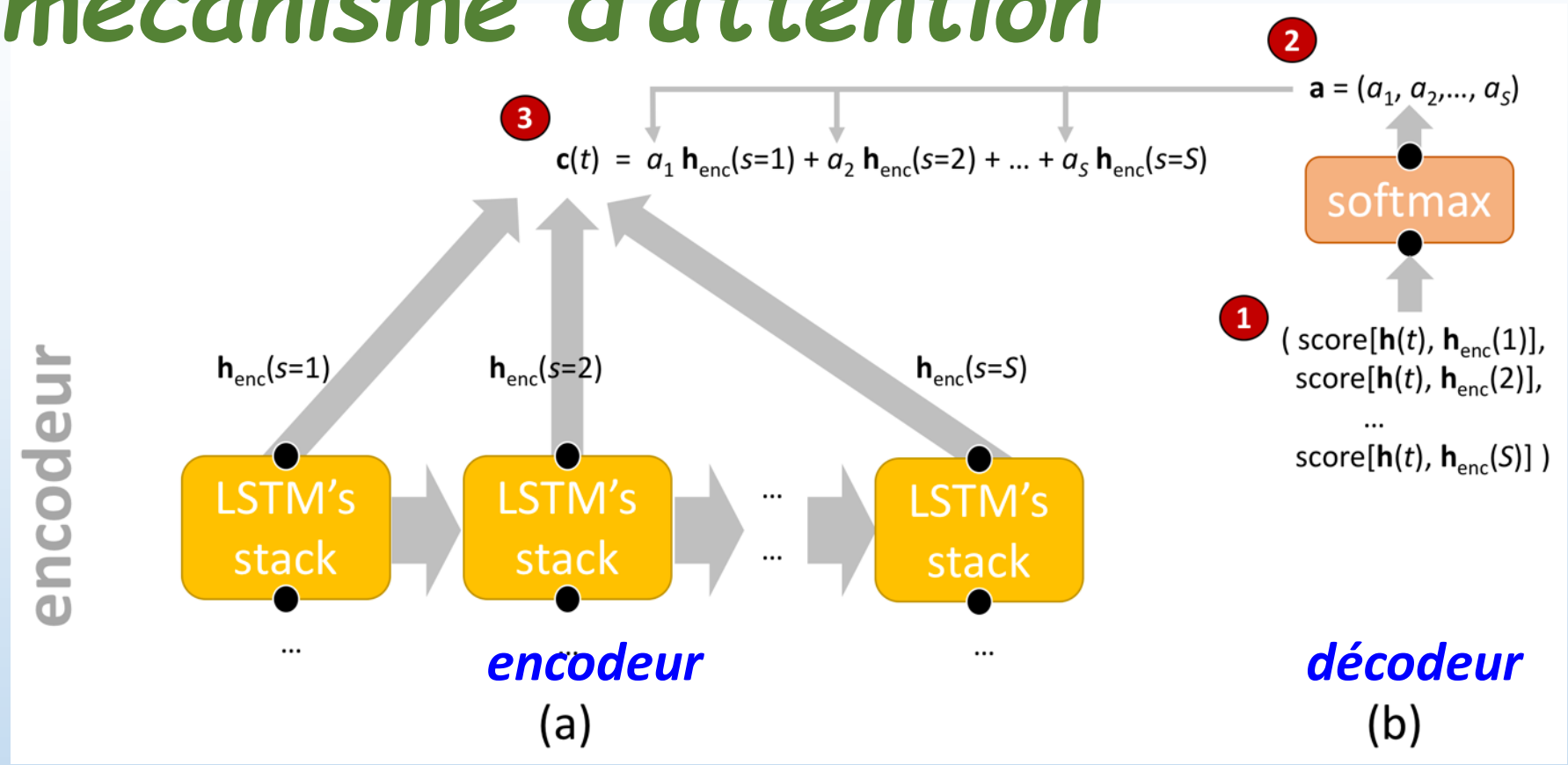


Le mécanisme d'attention



Une couche d'attention intercalée entre la dernière couche de LSTM et les softmax permet de tenir compte à la fois du contexte ***h(t) local au décodeur et d'un contexte c(t) calculé à partir de la phrase source***

Le mécanisme d'attention



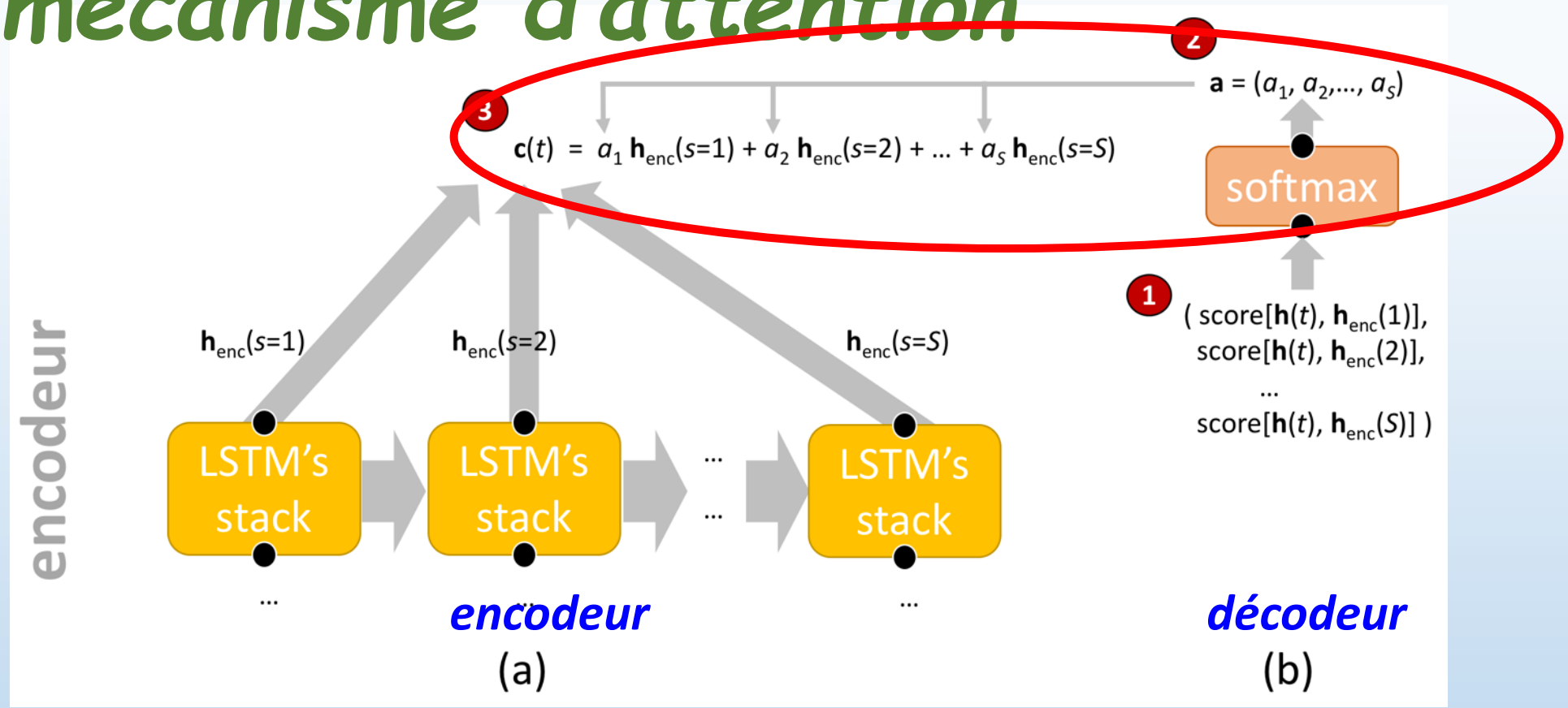
Le vecteur de contexte c(t) utilisé est une somme pondérée des S vecteurs de contexte côté encodeur, il est calculé en 3 étapes :

(1) on calcule des scores de proximité entre le contexte h(t) coté décodeur et les S contextes henc(s) associés à chaque mot côté encodeur,

(2) on transforme ces scores en probabilités avec un softmax

(3) on utilise ces probabilités pour calculer c(t) comme une somme pondérée des S contextes henc(s).

Le mécanisme d'attention



Les pondérations α_s *représentent la* ***proximité sémantique*** *entre le contexte cible* $\mathbf{h}(t)$ *et chacun des contextes sources* $\mathbf{h}_{enc}(s)$.

Exemples

Traduction automatique

- **Problème**: Traduire un texte d'une langue source vers une langue cible
- **Données**:
  - Ens. de couples de textes (Source, Cible)

Ex :
Lois européennes
Lois canadiennes Montréal

Google Translate :

encodeur décodeur avec mécanisme d'attention

Nouvelle approche Transformers

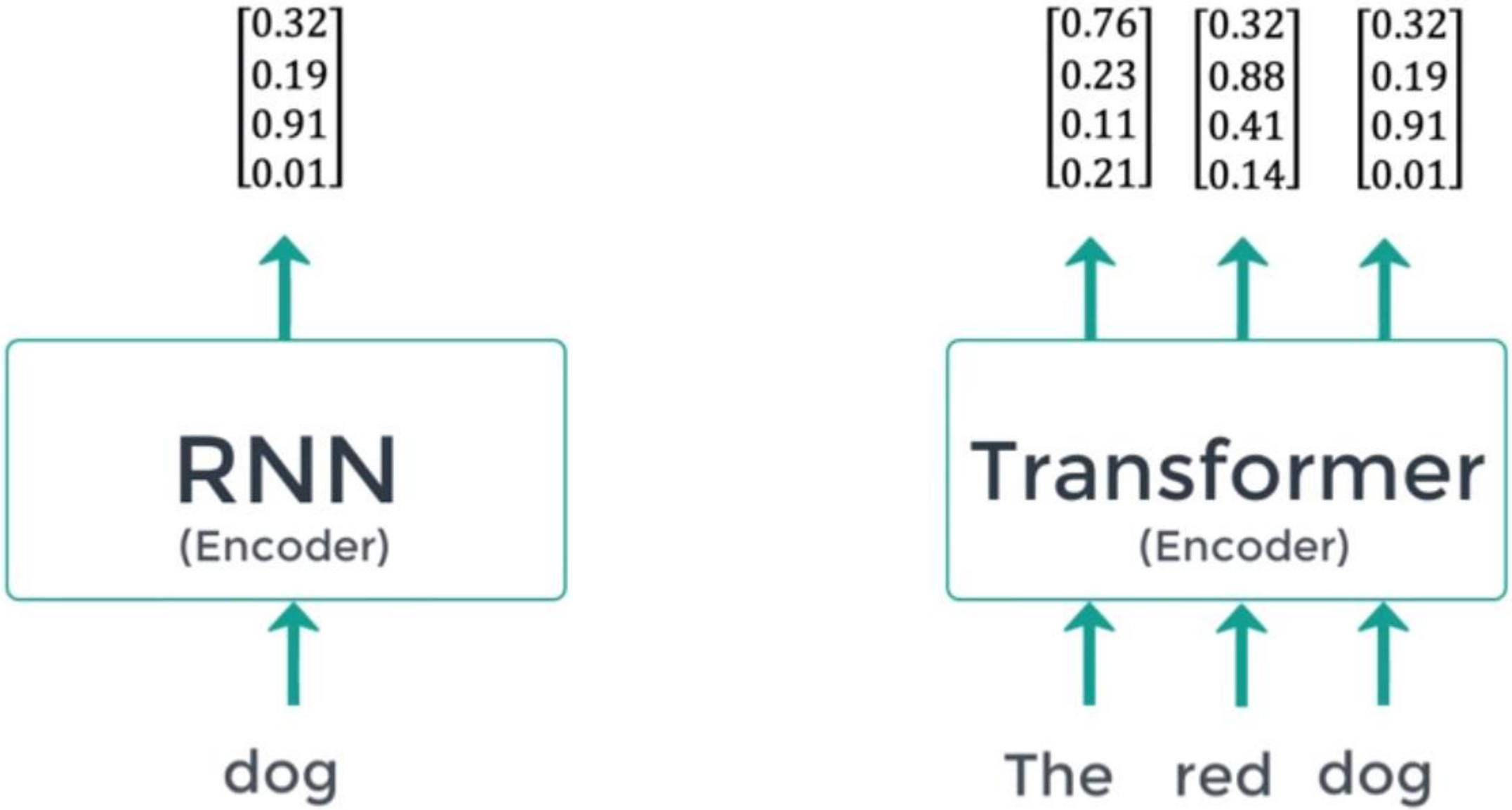
On va donner au réseau toute la phrase en encodant la position de chaque mot

Contrairement aux RNN et LSTM, les transformers ne traitent pas les données sous forme de flux continu en respectant l'ordre des mots des phrases.

Ce qui leur permet de découper les traitements et de paralléliser les calculs de la phase d'apprentissage.

Résultat : ils sont beaucoup plus rapides à entraîner que les RNN.

English-French Translation



Transformer

Boston est une très belle ville, j'y ai vécu trois ans, ce fut une expérience très enrichissante et surtout ça m'a permis d'apprendre à bien mieux parler ...

Vecteur de Pensée représente toute la phrase

Le mécanisme d'attention en détail
Les Transformers

Cf. présentation spécifique