



# Attention

Mohamed Abbas KONATE

# Plan

Réseaux de neurones récurrents

Modèle Sequence à Séquence ( SeqToSeq)

Introduction Mécanismes d'attention

**But du chapitre : Comprendre le mécanisme d'attention**

# Principes de base : réseaux neuronaux récurrents

**Question principale : comment un modèle fini peut-il modéliser un contexte long (infini) ?**

Solution : la récursivité ! (utilisation récursive d'un modèle)

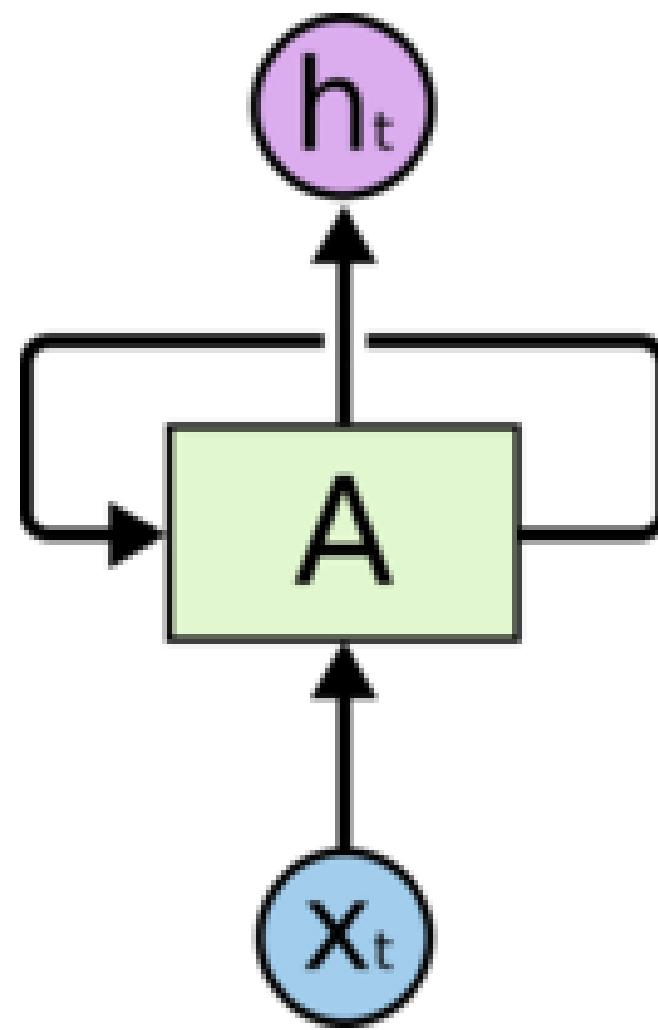
# Principes de base : réseaux neuronaux récurrents

Nouvel  
état

Ancien  
État

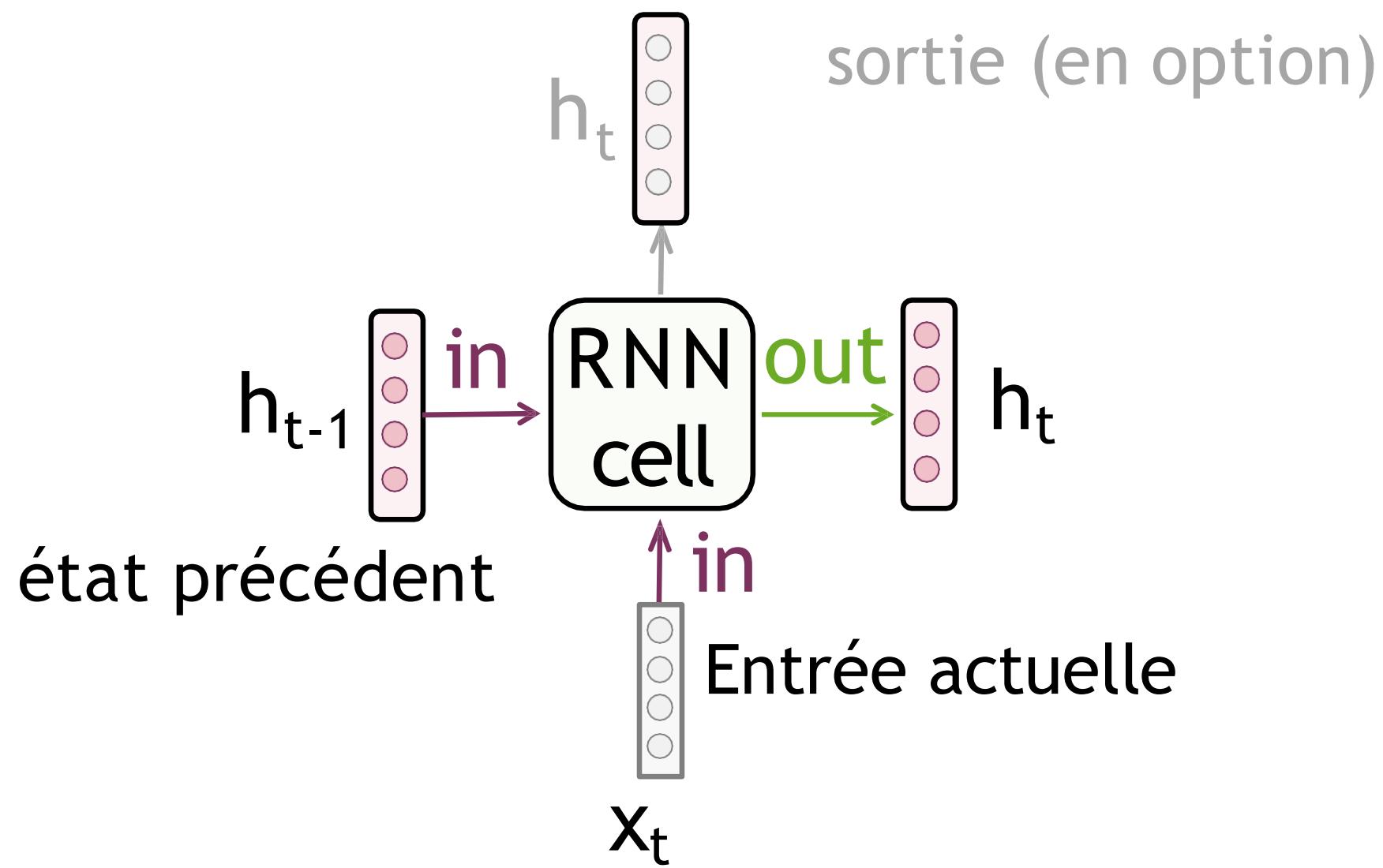
Vecteur  
d'entrée en t

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



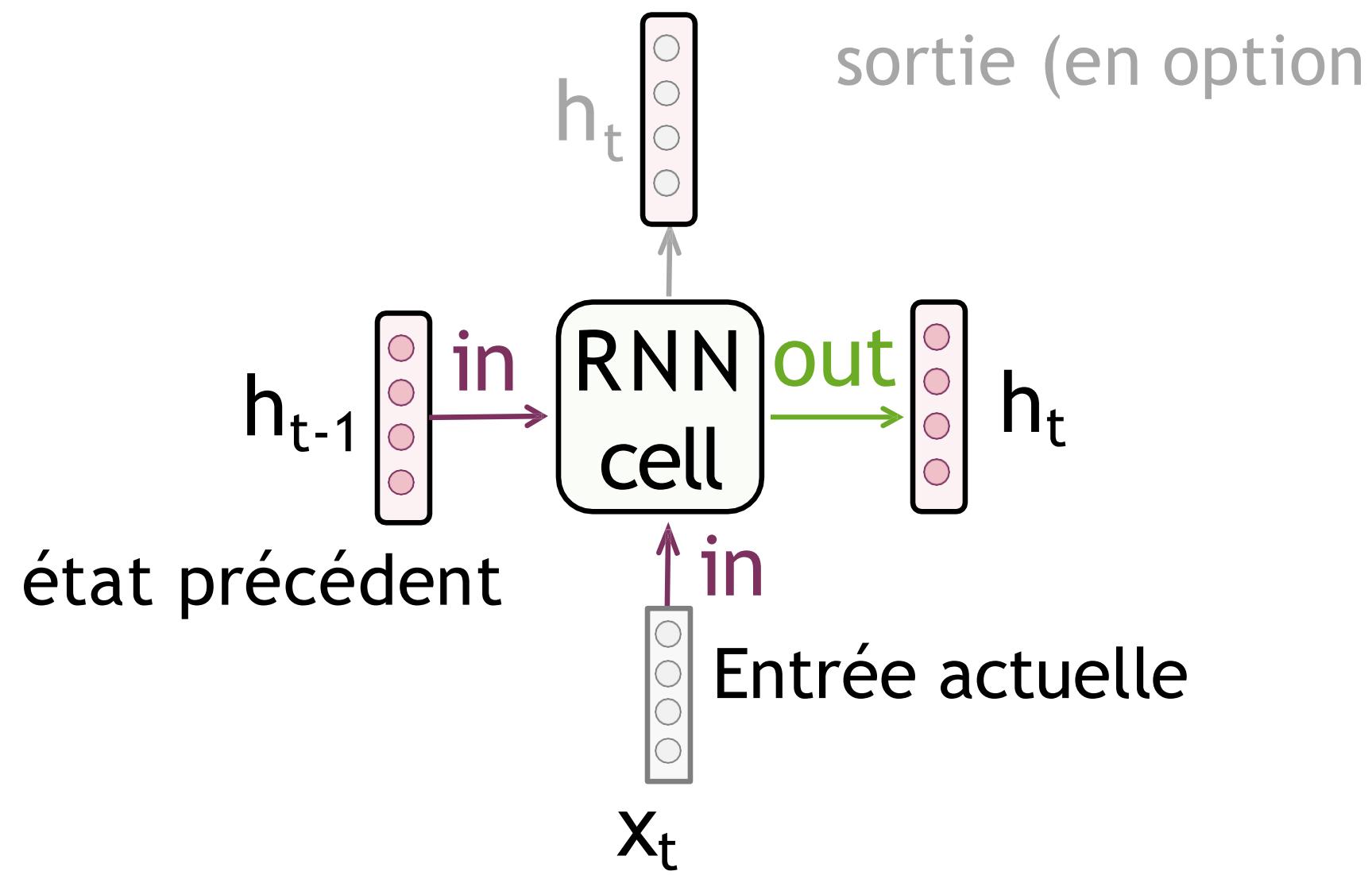
# Principes de base : réseaux neuronaux récurrents

- cellule récurrente



# Principes de base : réseaux neuronaux récurrents

- cellule récurrente



- « vanilla » RNN

$$h_t = \tanh(h_{t-1}W_h + x_tW_x)$$

A detailed diagram of the 'vanilla' RNN cell computation. It shows the hidden state  $h_{t-1}$  being multiplied by the weight matrix  $W_h^T$  (indicated by a purple arrow) and the current input  $x_t$  being multiplied by the weight matrix  $W_x^T$  (indicated by a grey arrow). The results of these two multiplications are summed together. This sum is then passed through a tanh activation function (indicated by a green arrow with a circular symbol) to produce the hidden state  $h_t$ .

# Principes de base : réseaux neuronaux récurrents

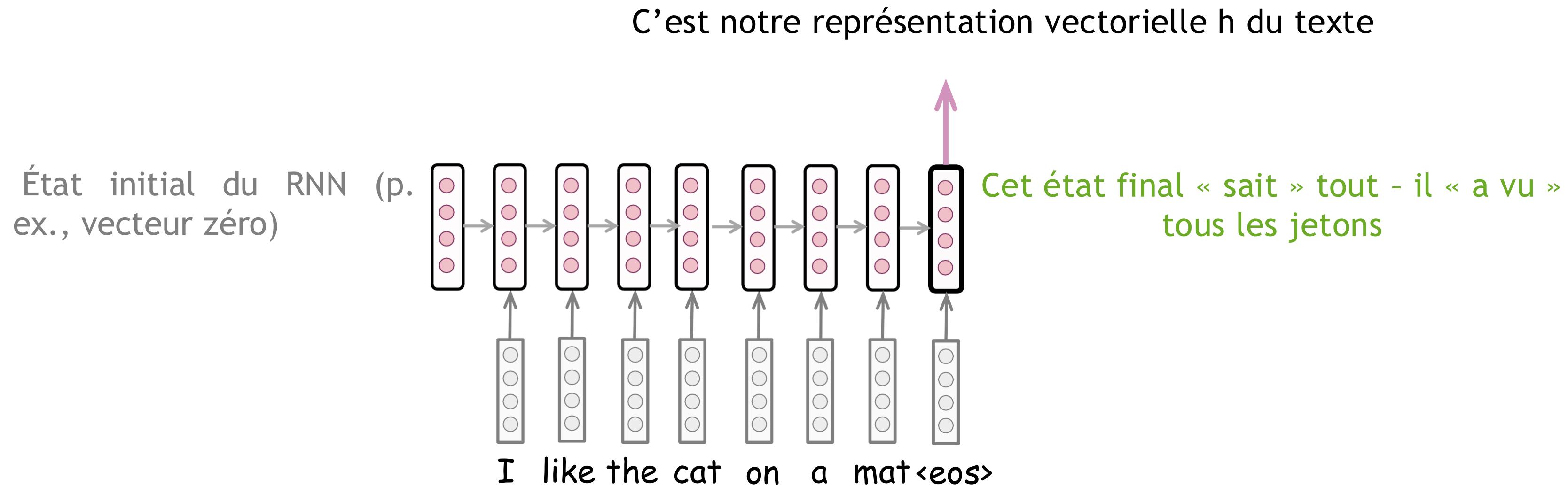
- RNN lit un texte



Text: I like the cat on a mat <eos>  
not read yet

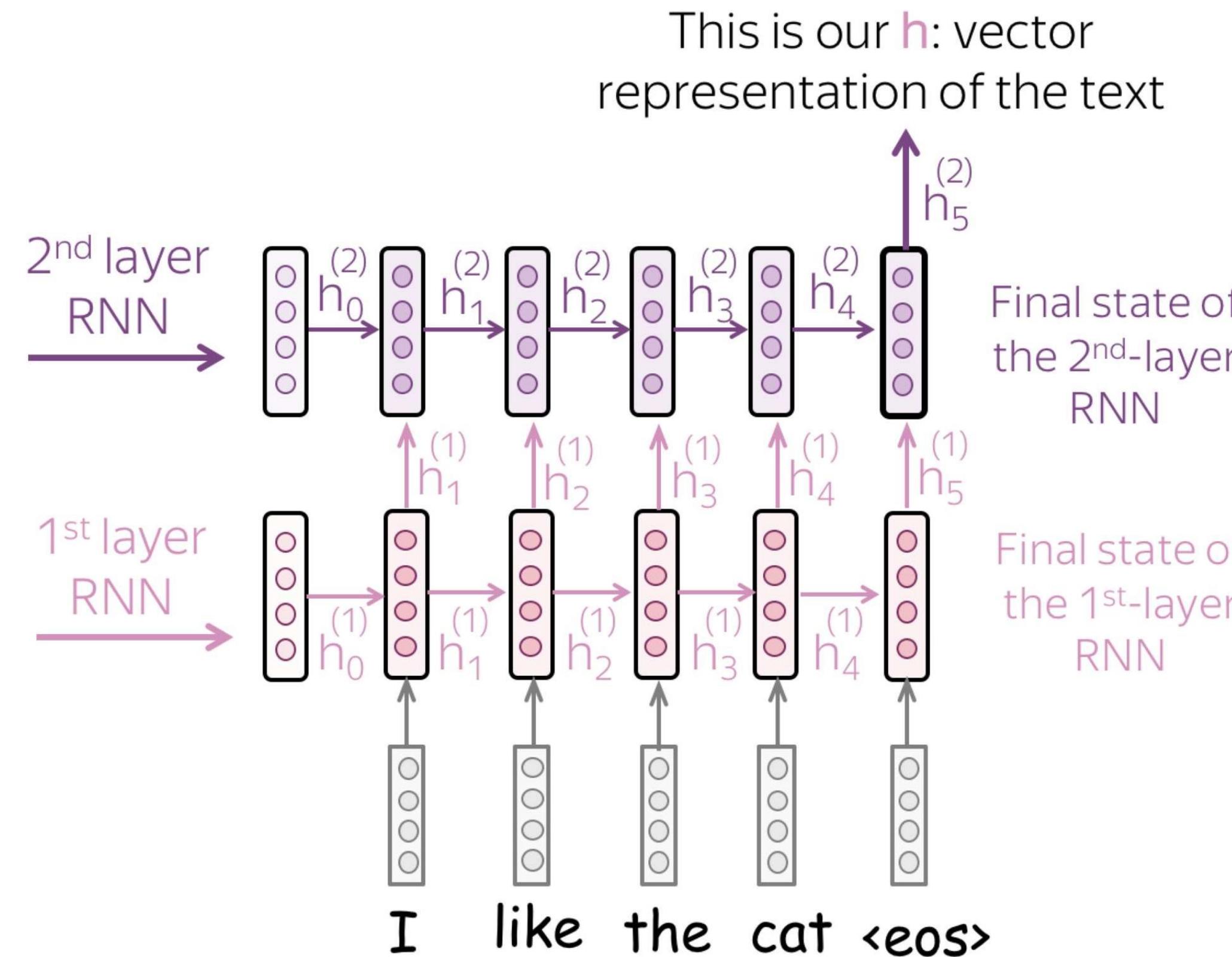
# Modèles récurrents pour la classification de texte

- Simple : lire un texte, prendre l'état final



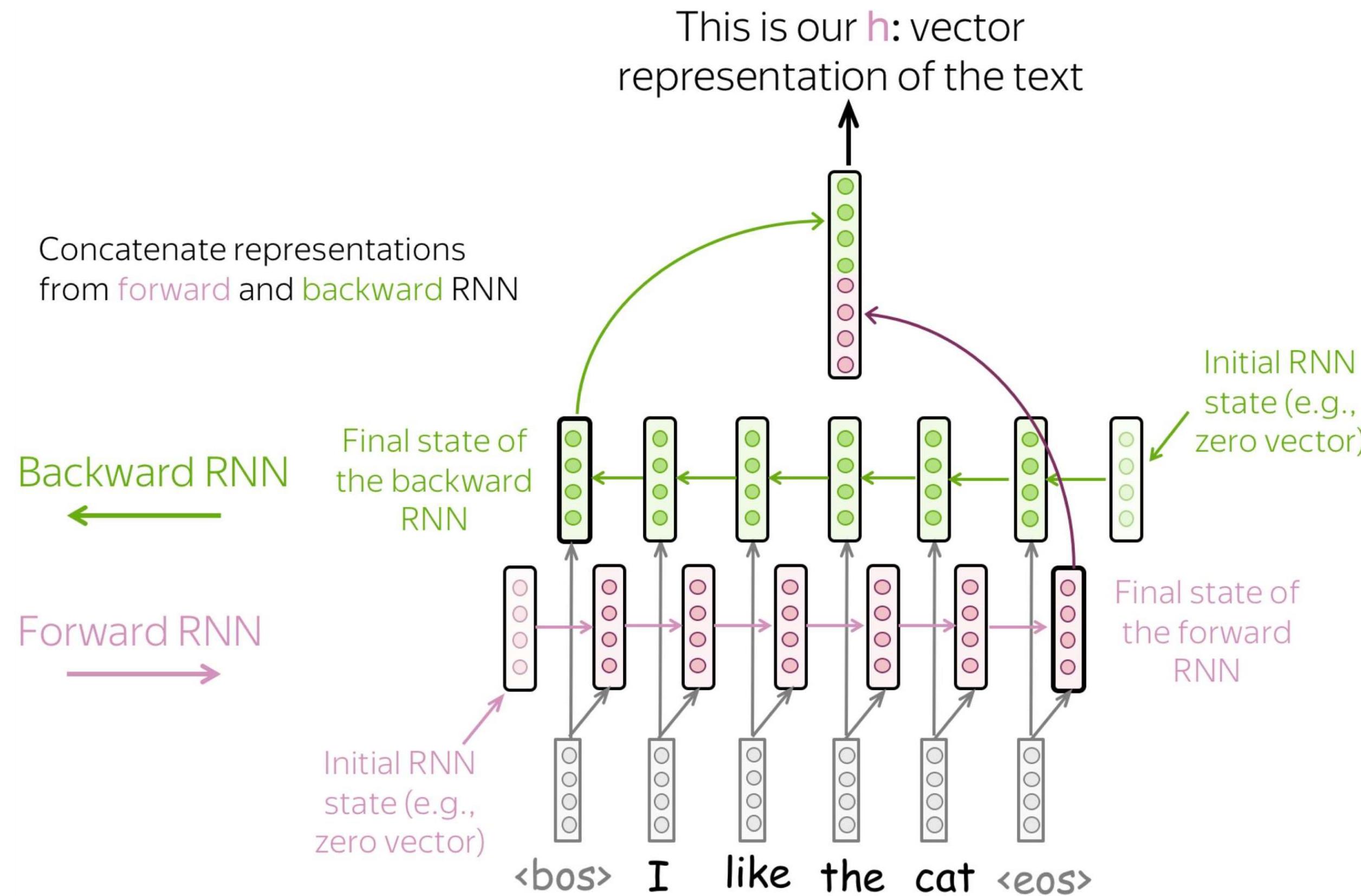
# Modèles récurrents pour la classification de texte

- Couches multiples



# Modèles récurrents pour la classification de texte

- bidirectionnel : utiliser les états finaux des RNN avant et arrière



# Tâche de séquence à séquence

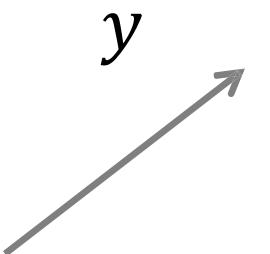
- Traduction entre langues naturelles
- Plus généralement, la traduction entre n’importe quelle séquence



# Traduction

## Traduction humaine

$$y^* = \arg \max p(y|x)$$

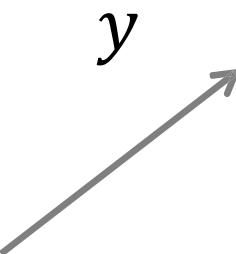


La « probabilité »  
est intuitive et est  
donnée par  
l'expertise d'un  
traducteur humain

# Traduction

## Traduction humaine

$$y^* = \arg \max p(y|x)$$



La « probabilité »  
est intuitive et est  
donnée par  
l'expertise d'un  
traducteur humain

## Traduction automatique

$$y'' = \arg \max_y p(y|x, \theta)$$

model

parameters

# Traduction

## Traduction humaine

$$y^* = \arg \max_y p(y|x)$$

La « probabilité » est intuitive et est donnée par l'expertise d'un traducteur humain

## Traduction automatique

$$y'' = \arg \max_y p(y|x, \theta)$$

modèle

Paramètres

Questions auxquelles nous devons répondre

- Modélisation

Quelle architecture pour estimer  $p(y|x, \theta)$ ?

# Traduction

## Traduction humaine

$$y^* = \arg \max_y p(y|x)$$

La « probabilité » est intuitive et est donnée par l'expertise d'un traducteur humain

## Traduction automatique

$$y'' = \arg \max_y p(y|x, \theta)$$

modèle

Paramètres

Questions auxquelles nous devons répondre

- Modélisation

Quelle architecture pour estimer  $p(y|x, \theta)$ ?

- Apprentissage

Comment trouver  $\theta$ ?

# Traduction

## Traduction humaine

$$y^* = \arg \max_y p(y|x)$$

La « probabilité » est intuitive et est donnée par l'expertise d'un traducteur humain

## Traduction automatique

$$y'' = \arg \max_y p(y|x, \theta)$$

modèle

Paramètres

### Questions auxquelles nous devons répondre

- **Modélisation**

Quelle architecture pour estimer  $p(y|x, \theta)$ ?

- **Apprentissage**

Comment trouver  $\theta$ ?

- **Maximisation**

Comment trouver argmax?

# Traduction

## Traduction humaine

$$y^* = \arg \max_y p(y|x)$$

La « probabilité » est intuitive et est donnée par l'expertise d'un traducteur humain

## Traduction automatique

$$y'' = \arg \max_y p(y|x, \theta)$$

model

parameters

### Questions auxquelles nous devons répondre

- Modélisation

Quelle architecture pour estimer  $p(y|x, \theta)$ ?

- Apprentissage

Comment trouver  $\theta$ ?

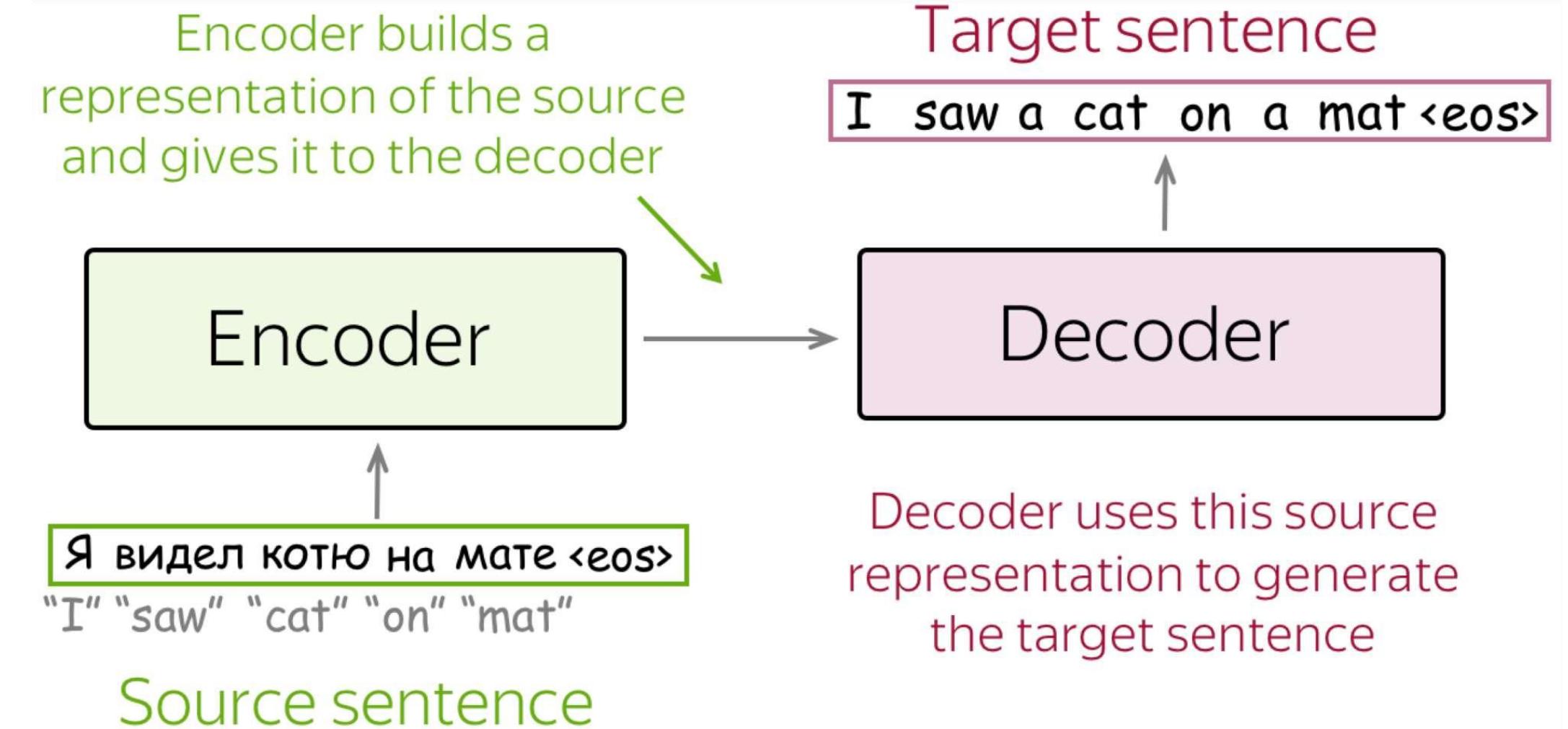
- Maximisation

Comment trouver argmax?

# Encoder-Decoder Framework

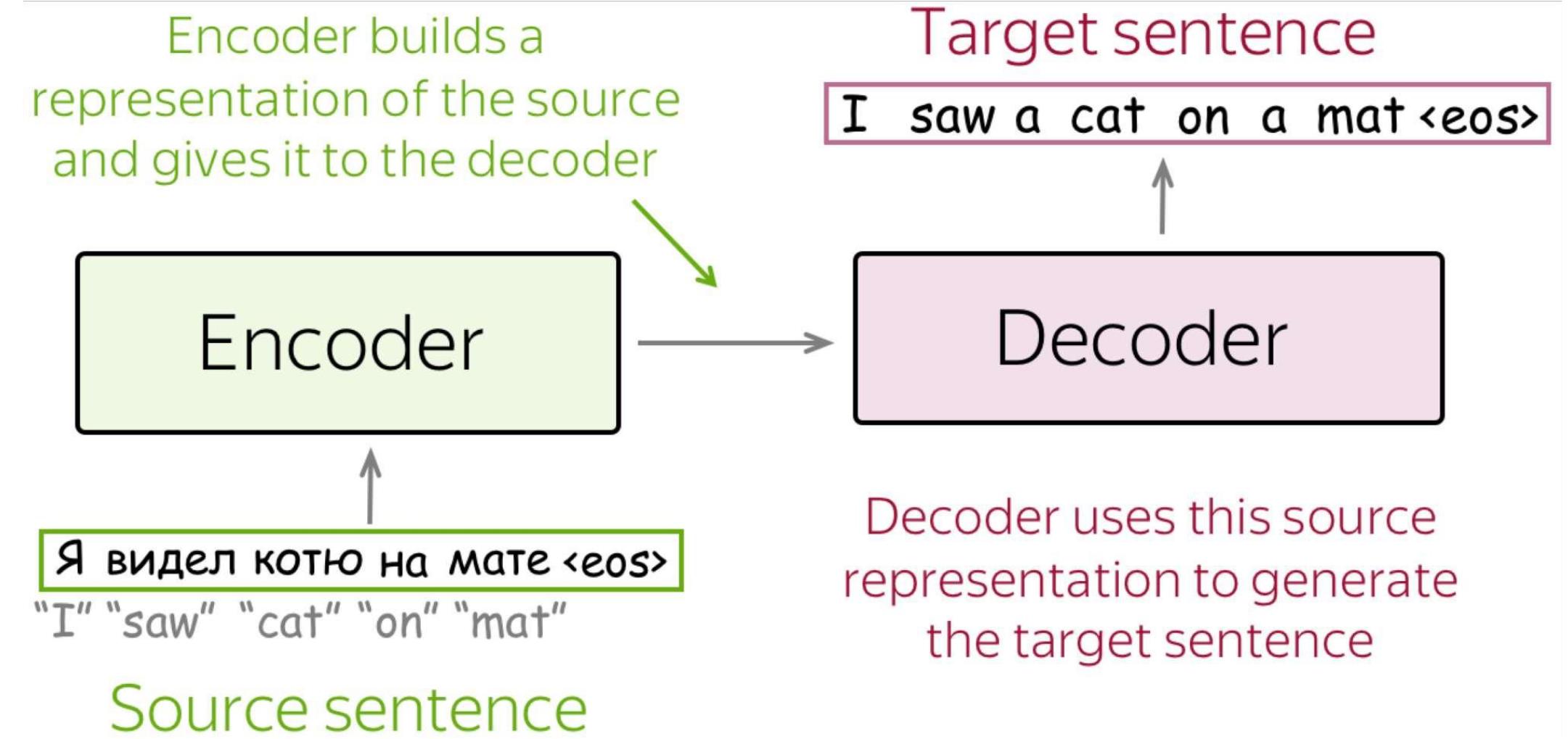
**IMPORTANT**

- Encodeur - lit la phrase source et produit sa représentation

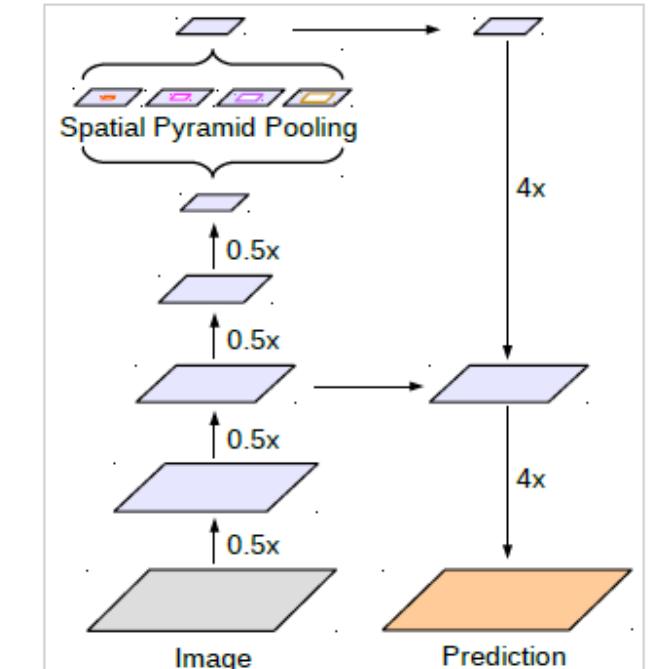
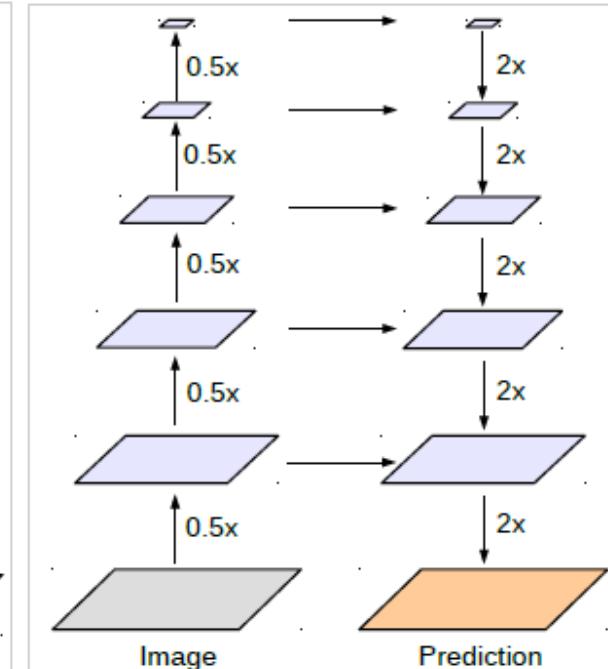
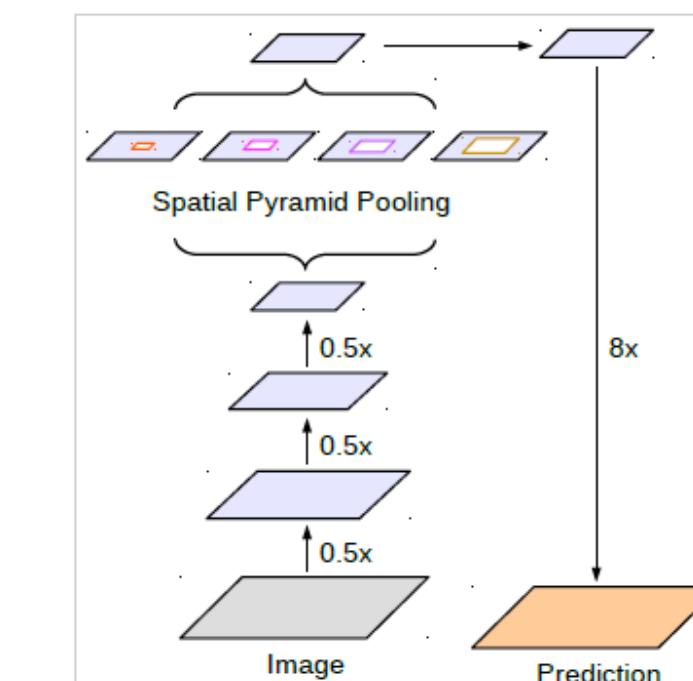
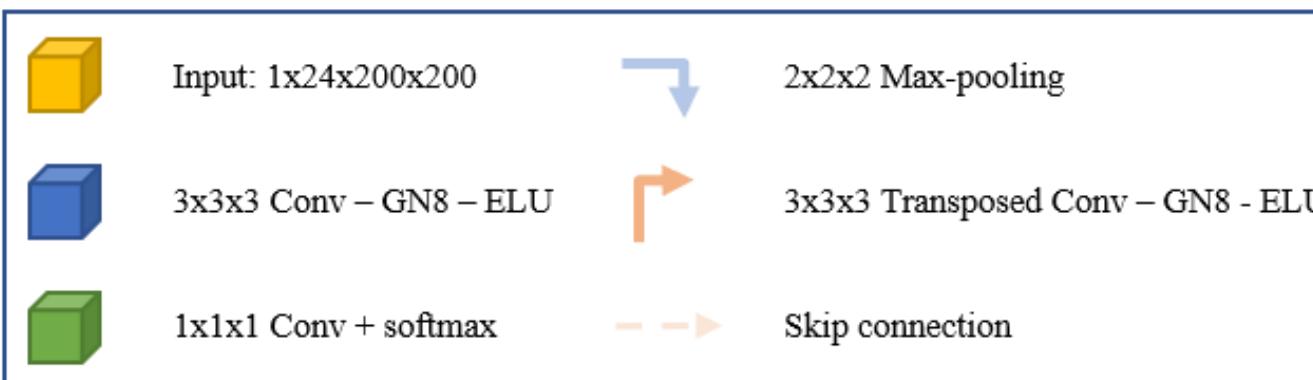
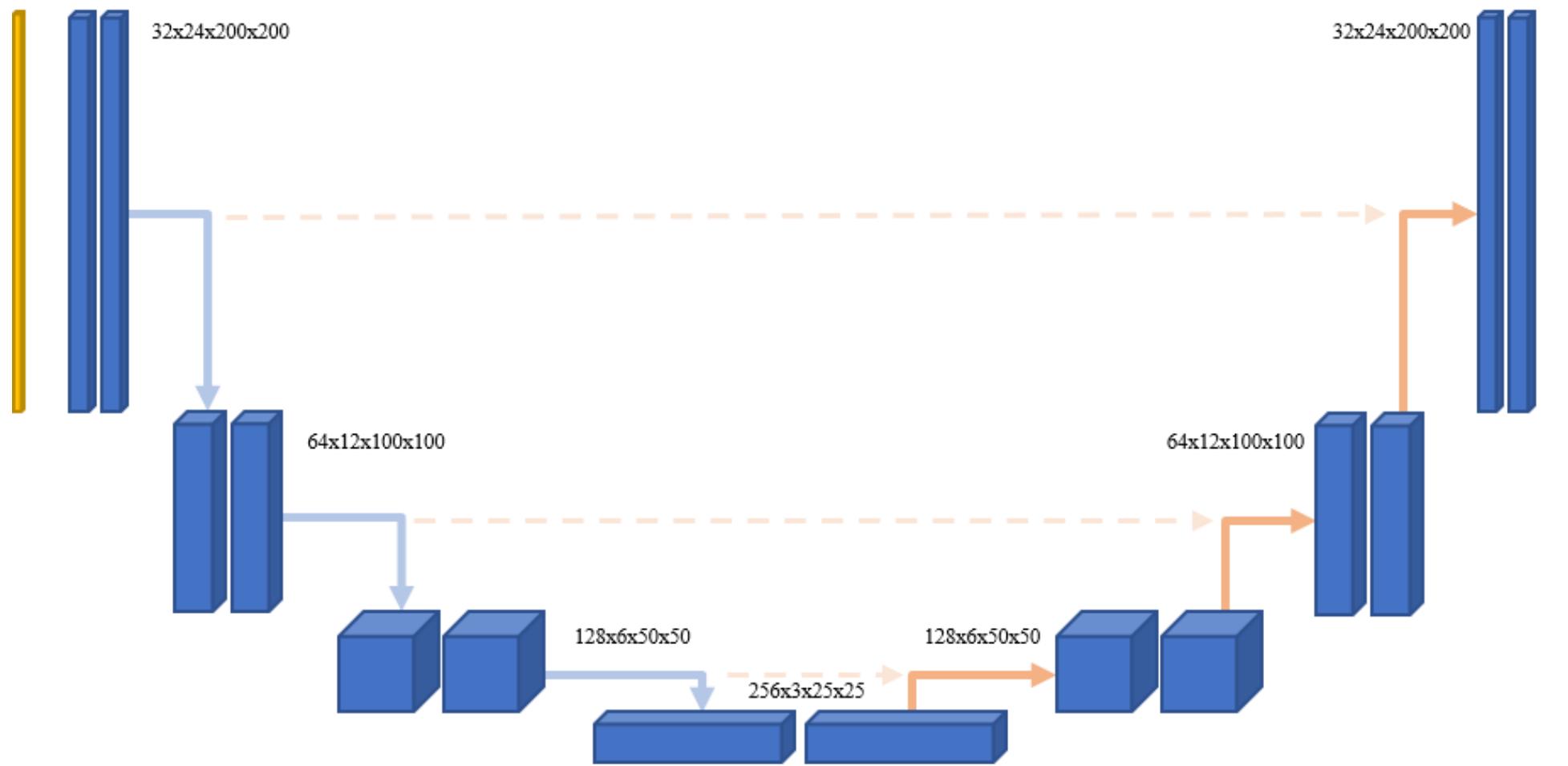


# Encoder-Decoder Framework

- Encodeur - lit la phrase source et produit sa représentation
- Décodeur : utilise la représentation de la source de l'encodeur pour générer la séquence cible.

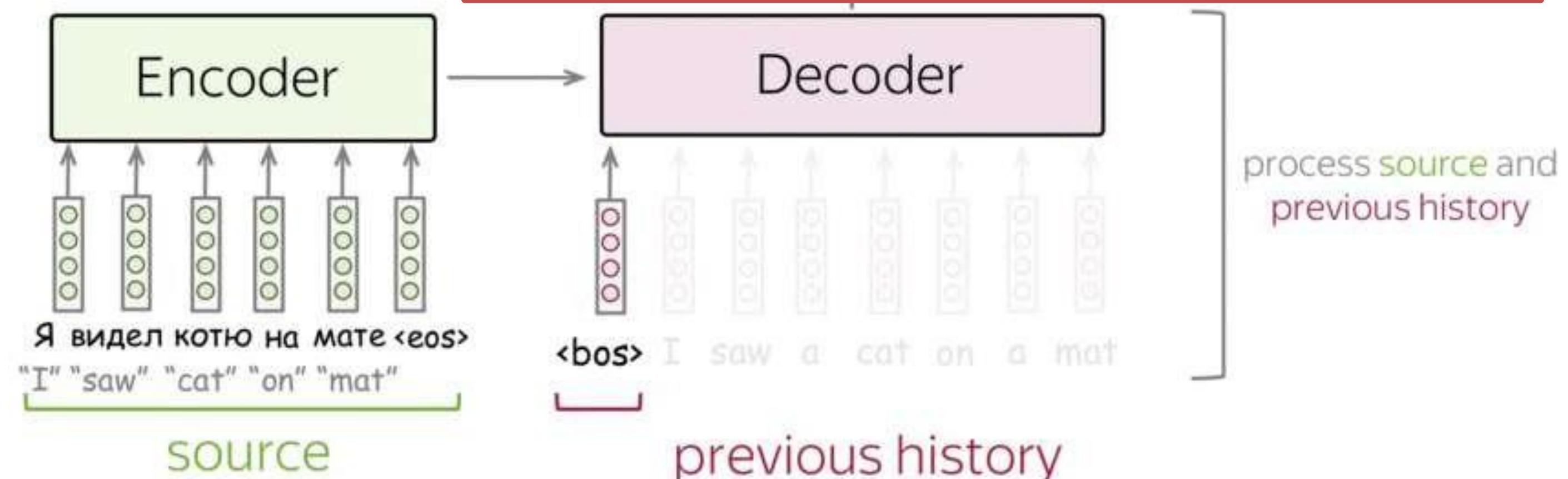


# Encoder-Decoder Framework

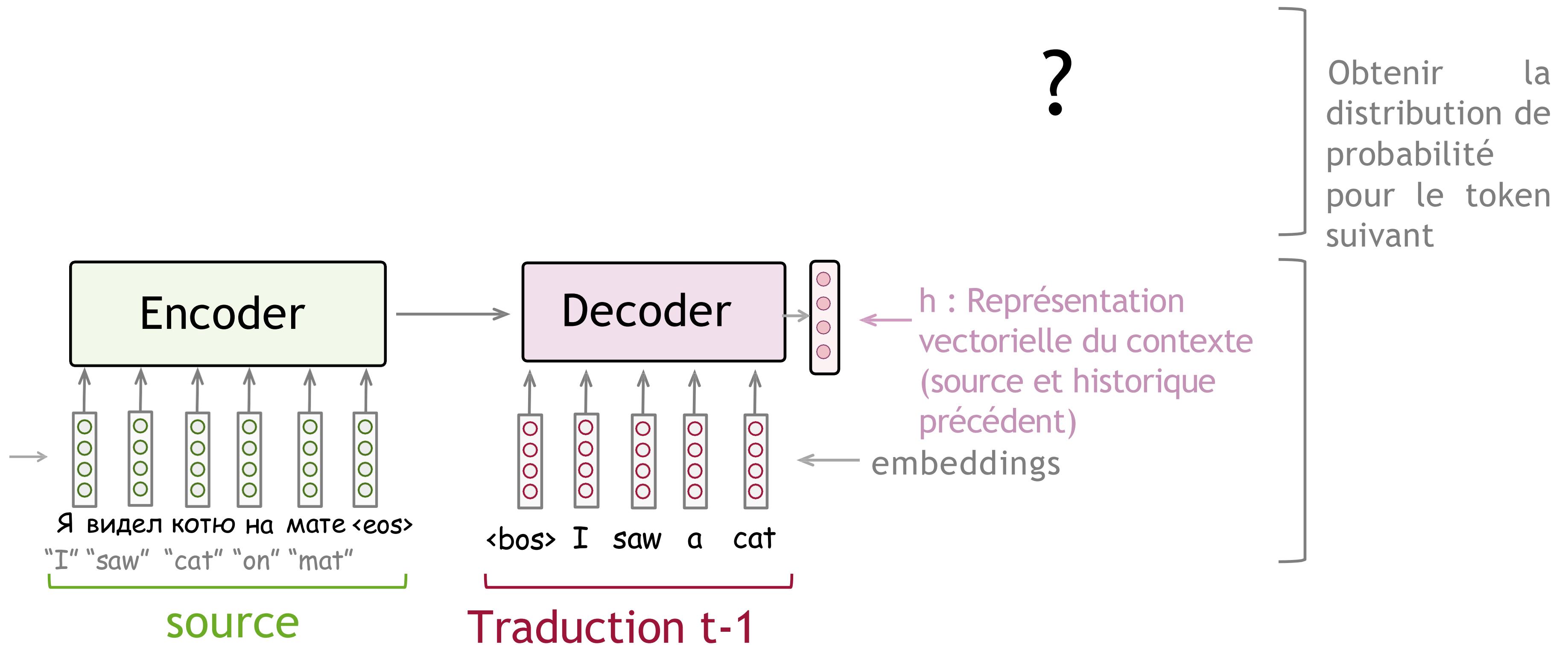


# Vue générale

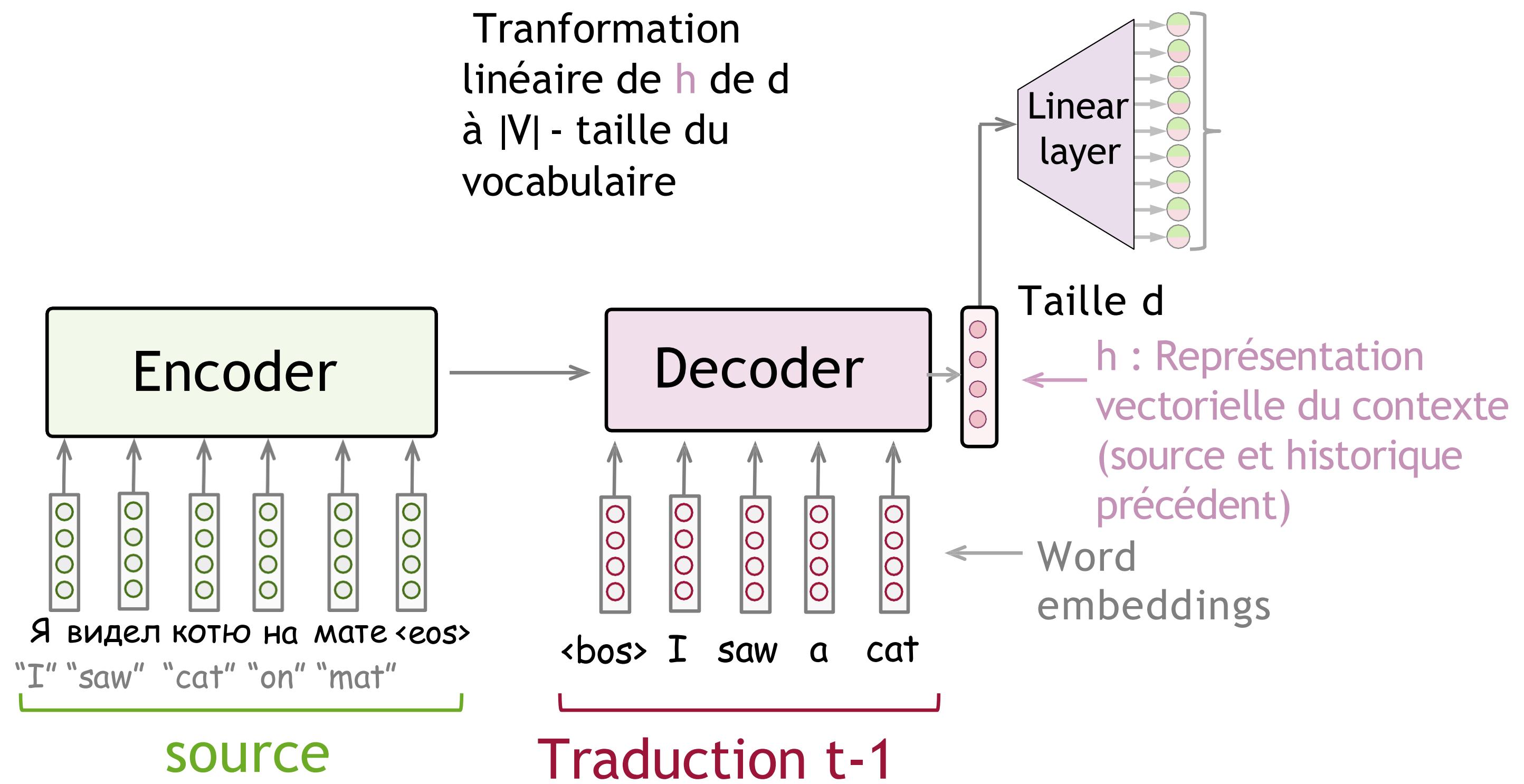
Prédire la distribution de probabilité pour le token cible suivant



# Vue générale



# Vue générale



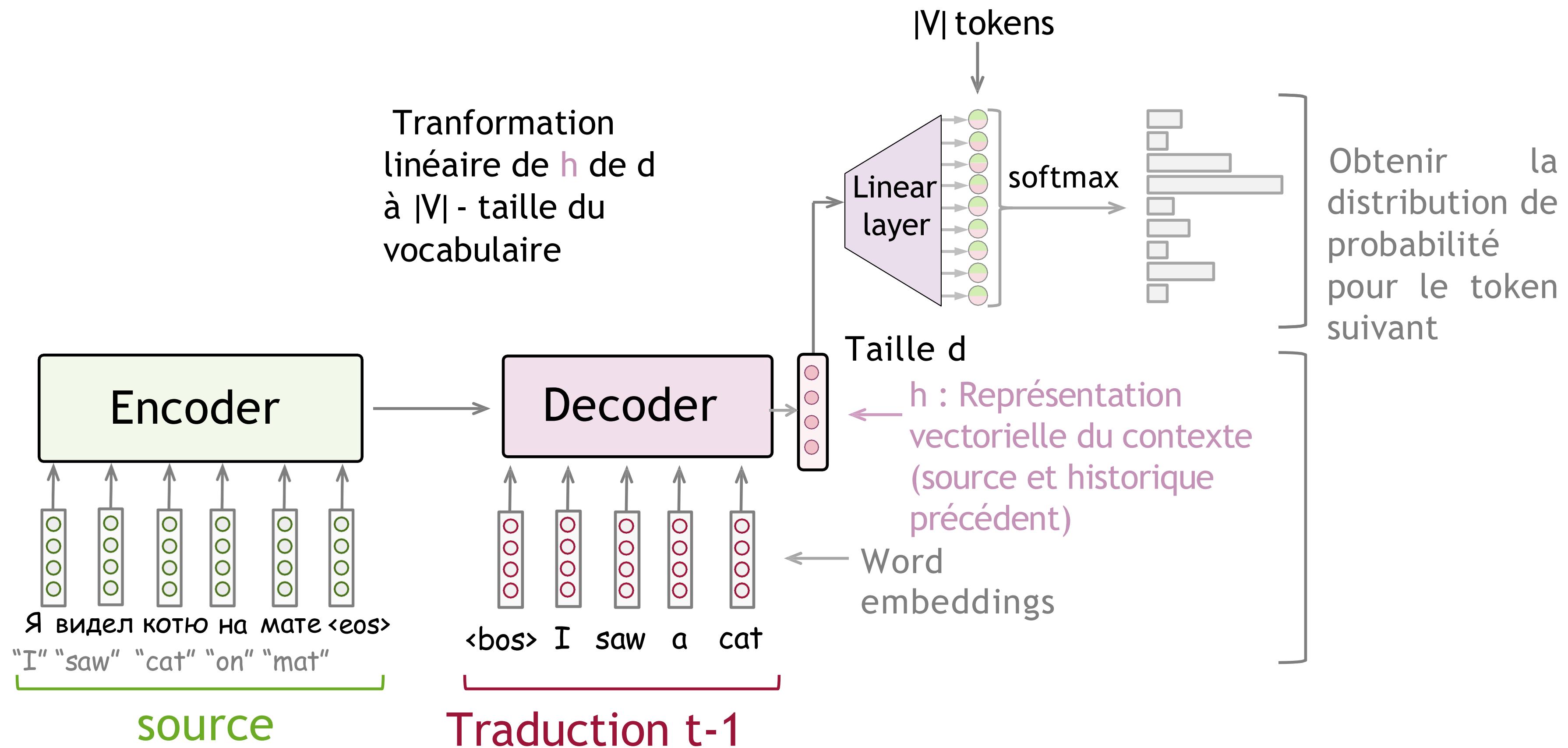
$$h_t = \tanh(h_{t-1}W_h + x_tW_x)$$

A detailed diagram of the recurrent unit update equation. It shows the calculation of the hidden state  $h_t$  from the previous hidden state  $h_{t-1}$  and the input  $x_t$ . The equation is:

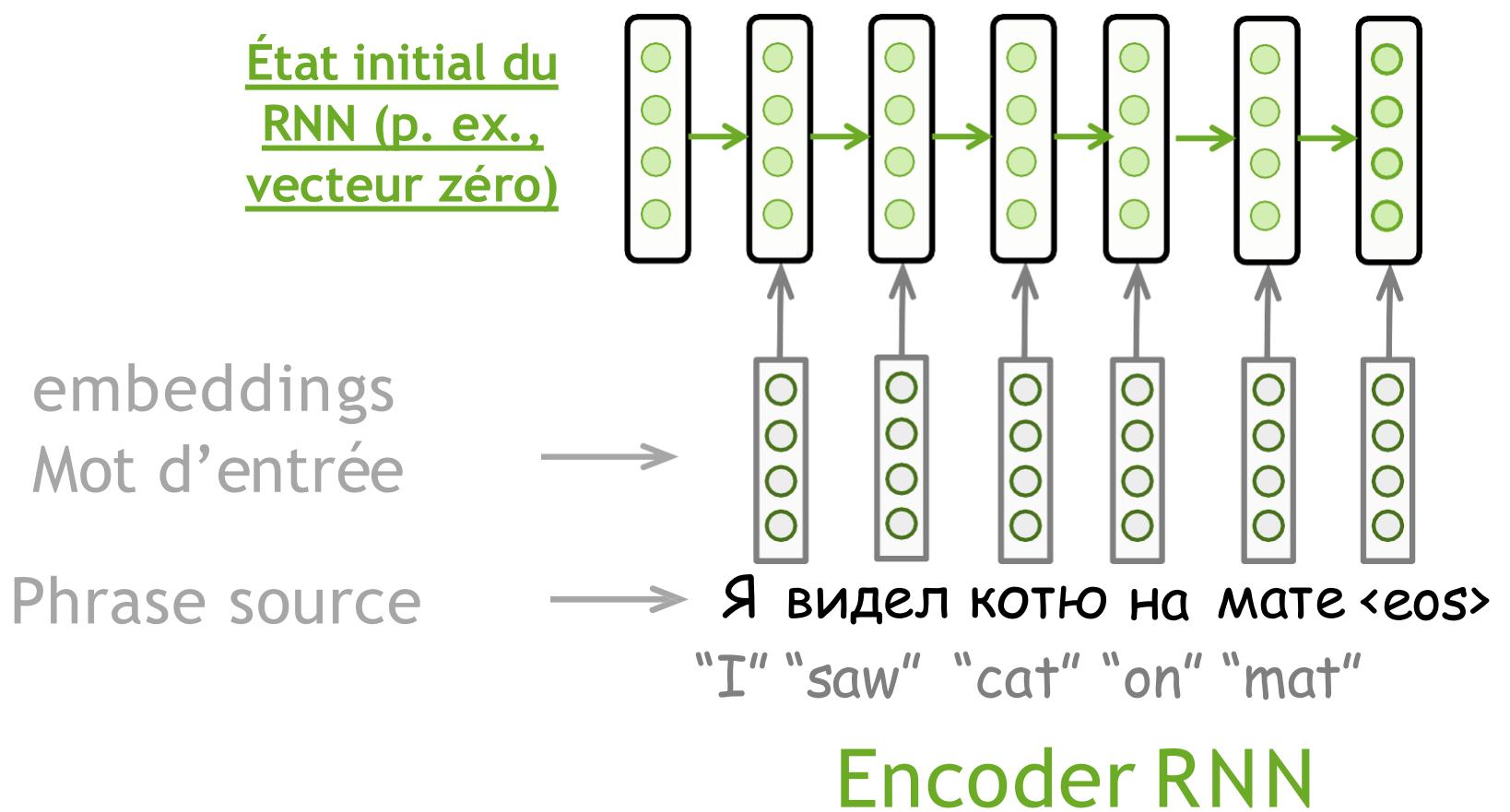
$$h_t = \tanh(h_{t-1}^T W_h + x_t^T W_x)$$

The diagram illustrates the matrix multiplication and the application of the tanh activation function.

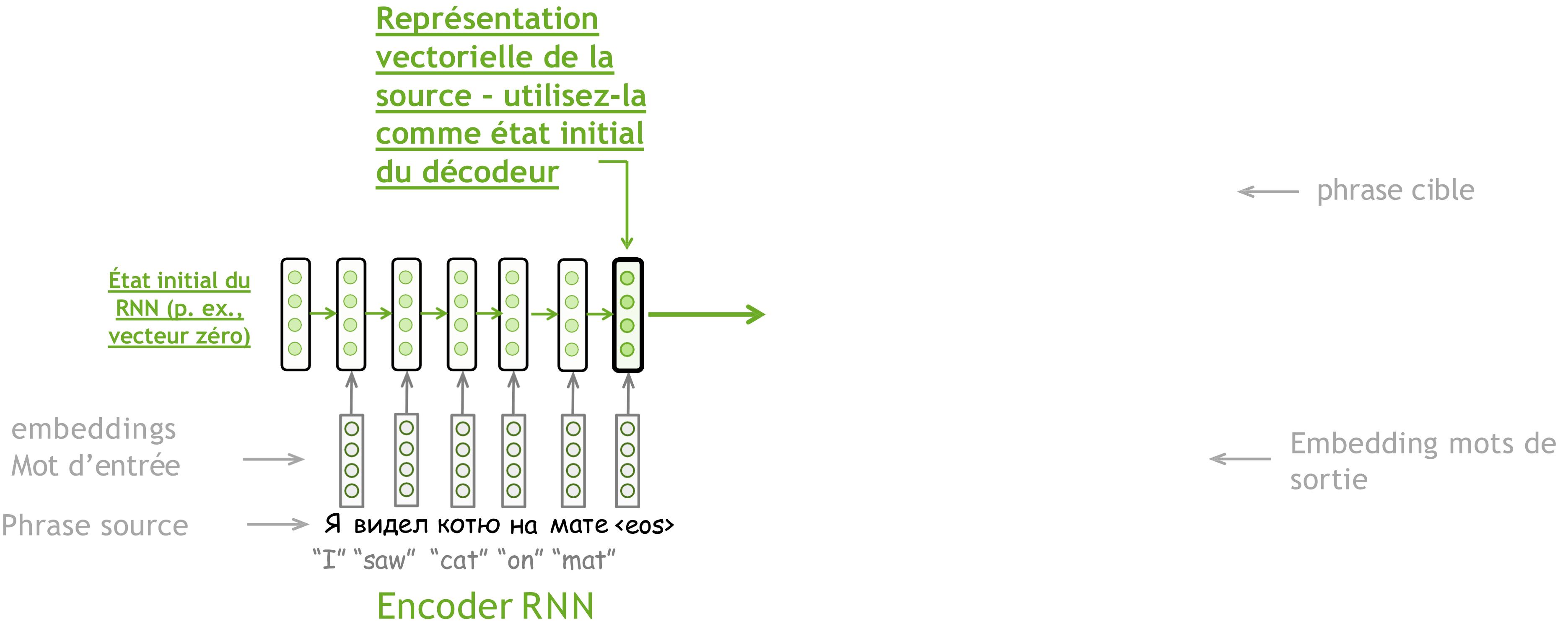
# Vue générale



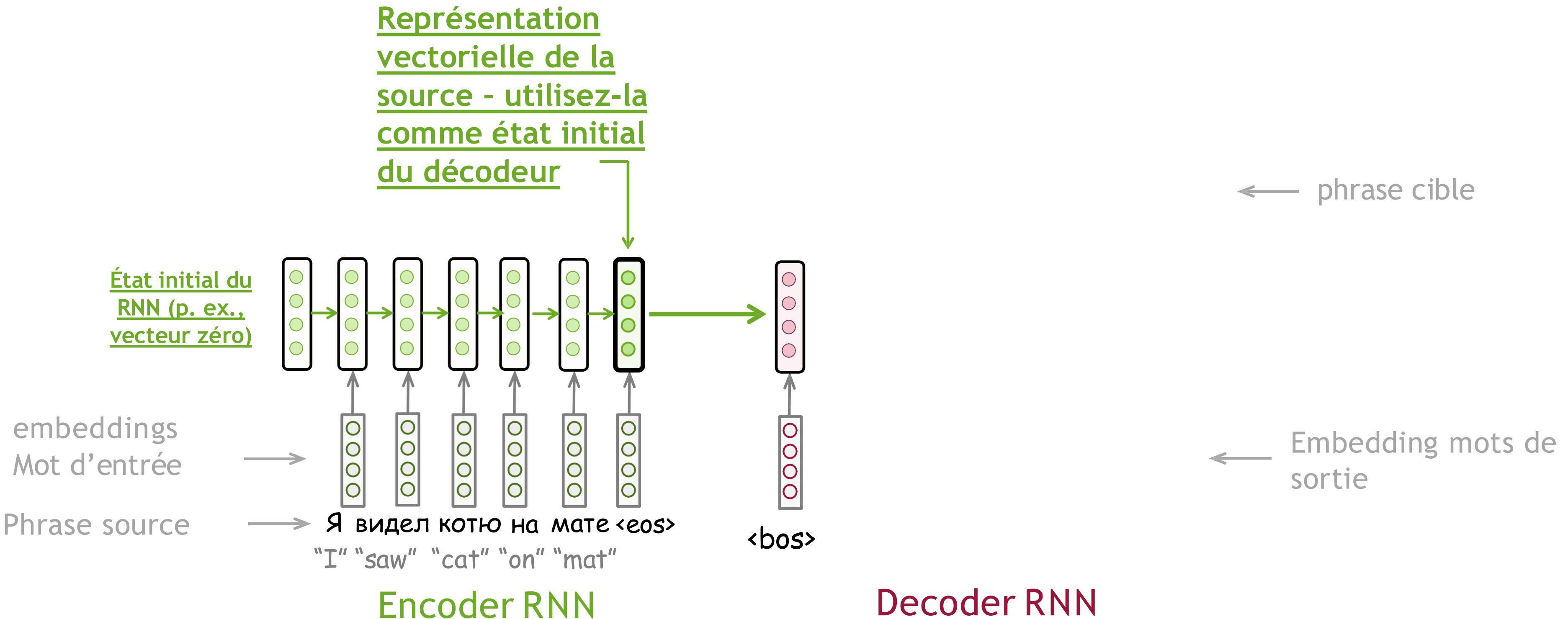
# Le modèle le plus simple : encodeur et décodeur RNN



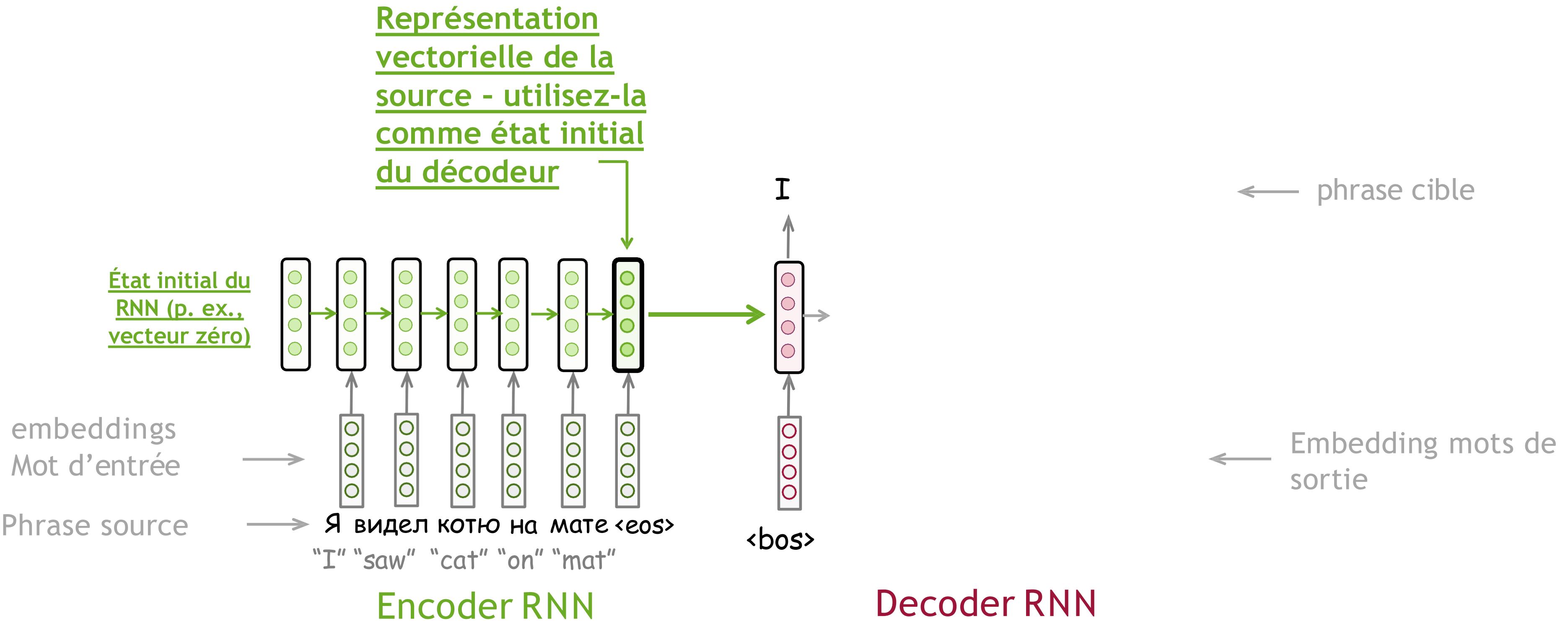
# Encodeur et décodeur RNN



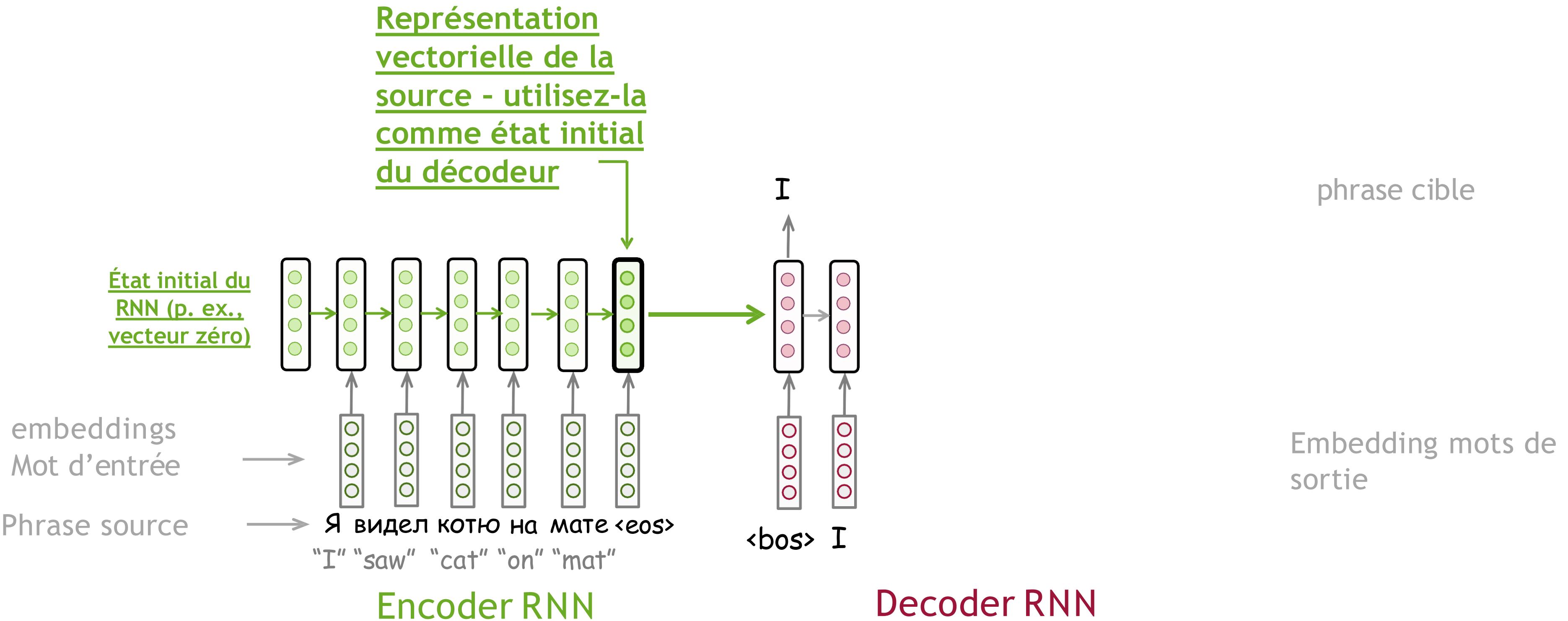
# Encodeur et décodeur RNN



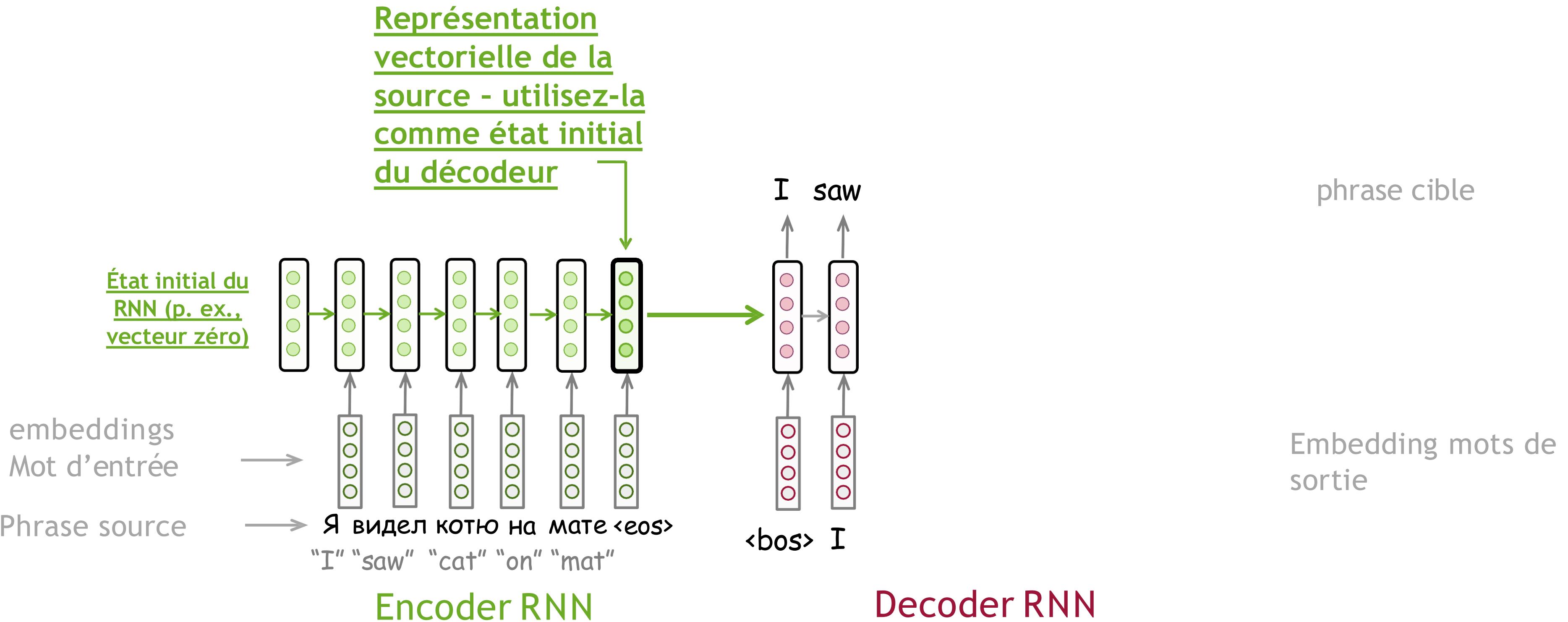
# Encodeur et décodeur RNN



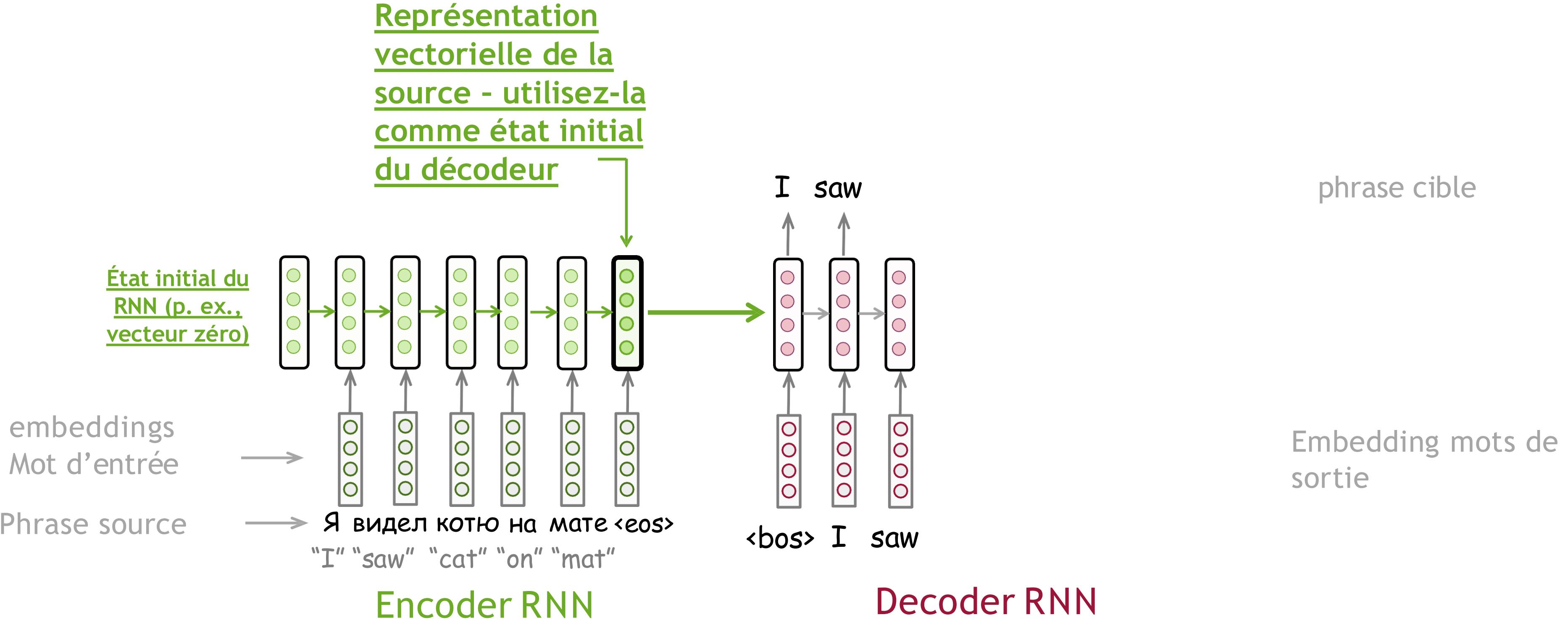
# Encodeur et décodeur RNN



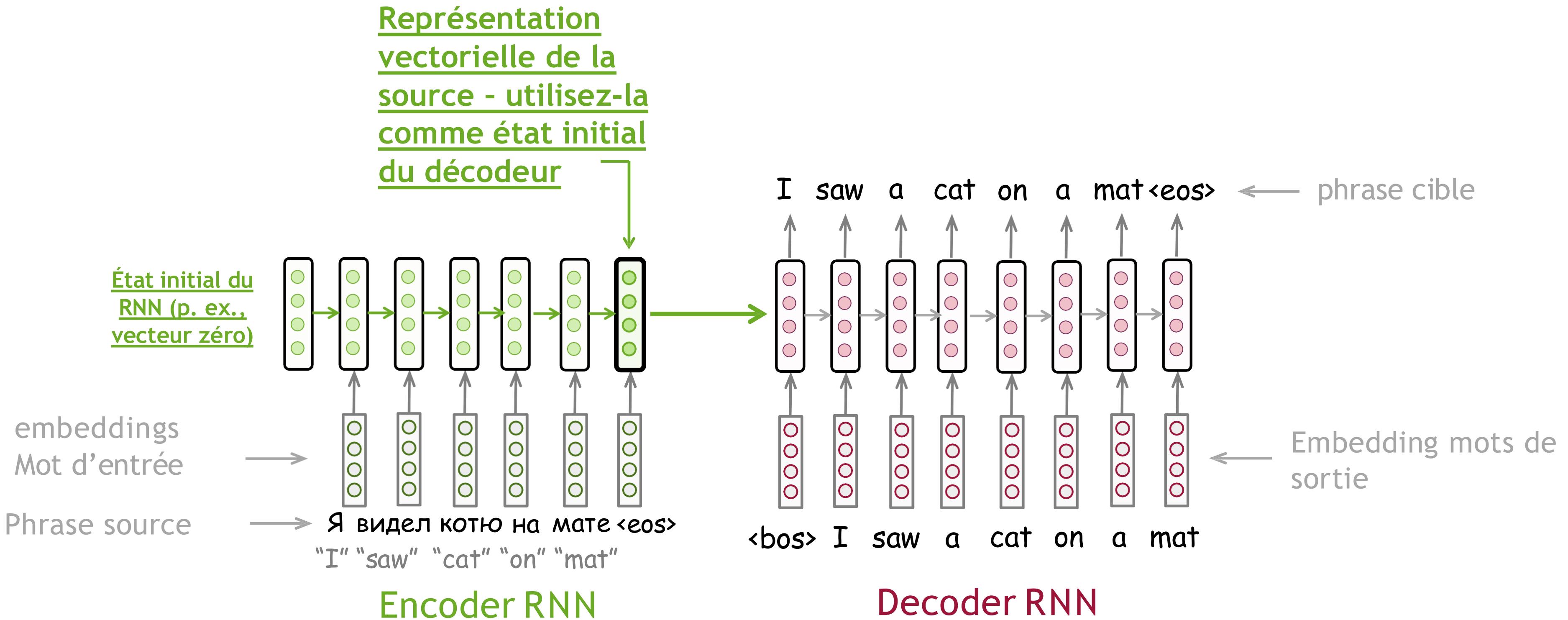
# Encodeur et décodeur RNN



# Encodeur et décodeur RNN



# Encodeur et décodeur RNN



# Traduction

## Traduction humaine

$$y^* = \arg \max_y p(y|x)$$

La « probabilité » est intuitive et est donnée par l'expertise d'un traducteur humain

## Traduction automatique

$$y'' = \arg \max_y p(y|x, \theta)$$

model

parameters

### Questions auxquelles nous devons répondre

- **Modélisation**

Quelle architecture pour estimer  $p(y|x, \theta)$ ?

- **Apprentissage**  
Comment trouver  $\theta$ ?

- **Maximisation**  
Comment trouver argmax?

# Translation

## Human Translation

$$y^* = \arg \max_y p(y|x)$$

The “probability” is  
intuitive and is given  
by a human  
translator’s expertise

## Machine Translation

$$y' = \arg \max_y p(y|x, \theta)$$

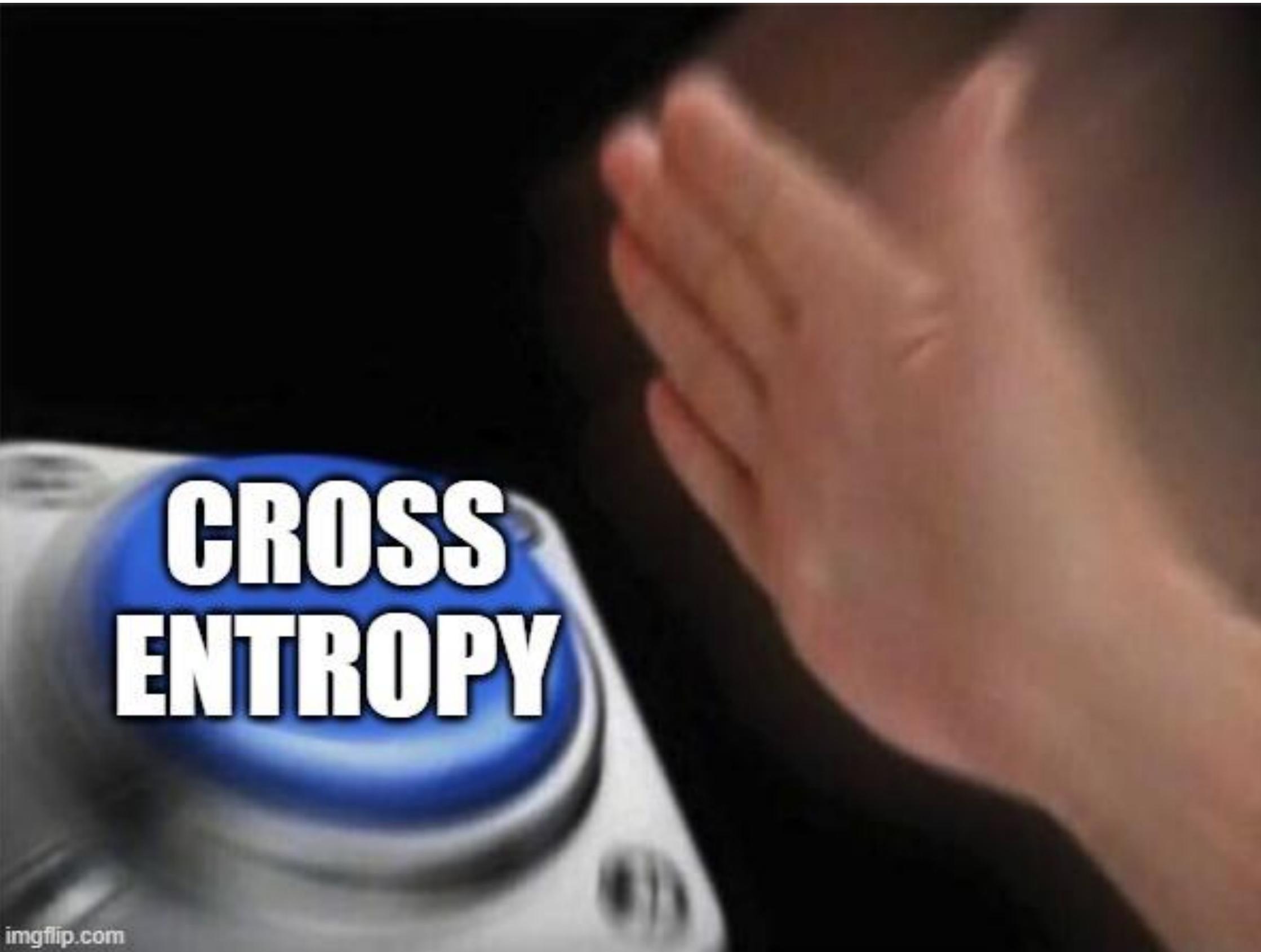
### Questions we need to answer

- **modeling**  
How does the model  
for  $p(y|x, \theta)$  look like?

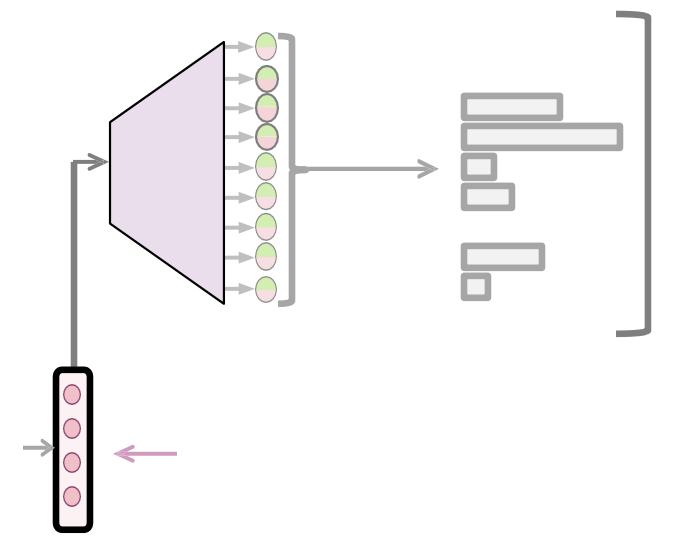
- **learning**  
How to find  $\theta$ ?

- **search**  
How to find  
the argmax?

# Cross-Entropy - Encore!



# Cross-Entropy - Encore!



Source :

Я видел котю на мате <eos>  
"I" "saw" "cat" "on" "mat"

Cible :

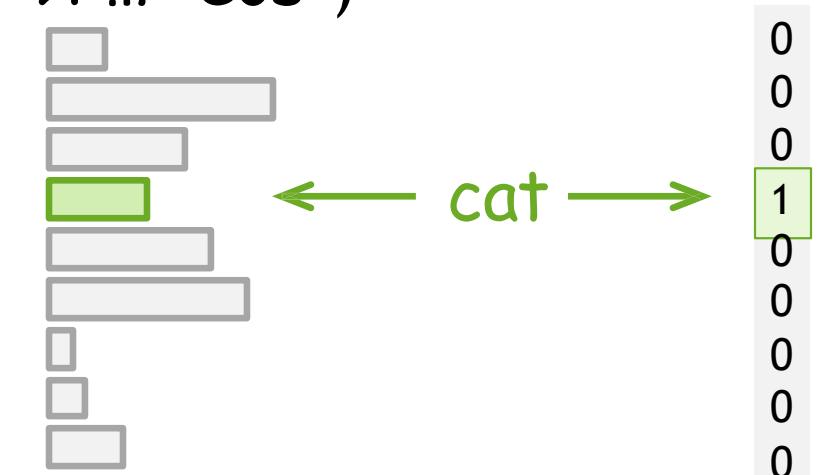
I saw a cat on a mat <eos>

Anciens tokens

Prédiction attendue du modèle

Prédiction :

$p(* | I \text{ saw } a,$   
 $\text{я ... } <\text{eos}>)$



Cible

Loss =  $-\log(p(\text{cat})) \rightarrow \min$



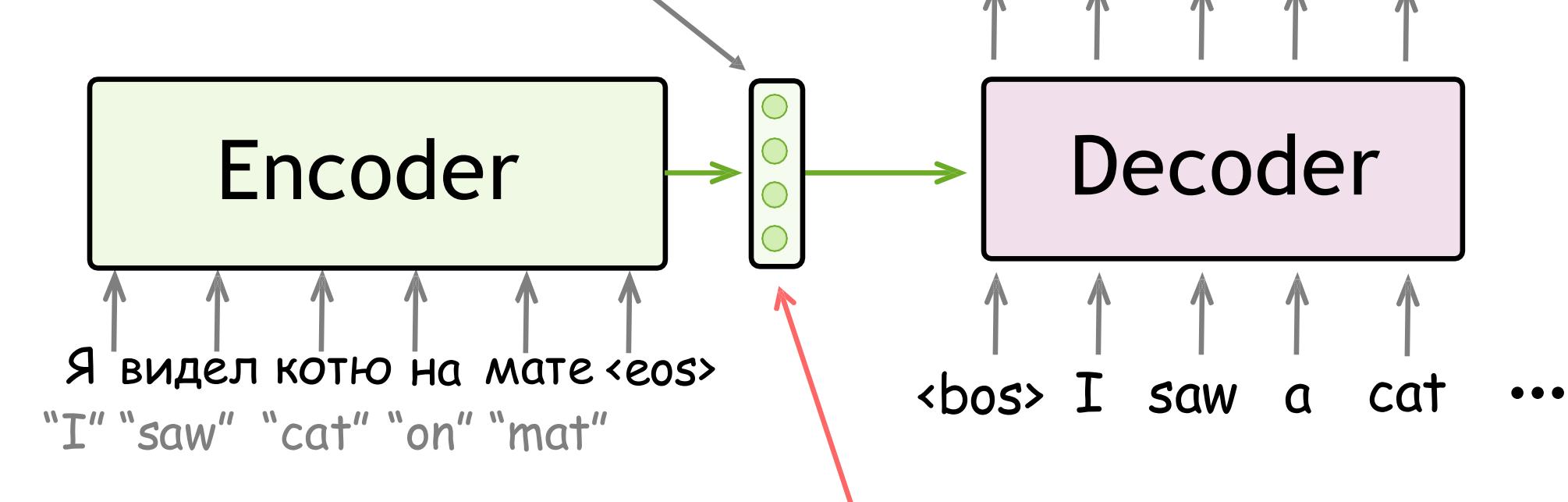
# Le problème de représentation fixe des encodeurs

La représentation fixe est problématique

Pour l'encodeur, il est difficile de compresser une longue phrase

Pour le décodeur, des informations différentes peuvent être nécessaires à différentes étapes

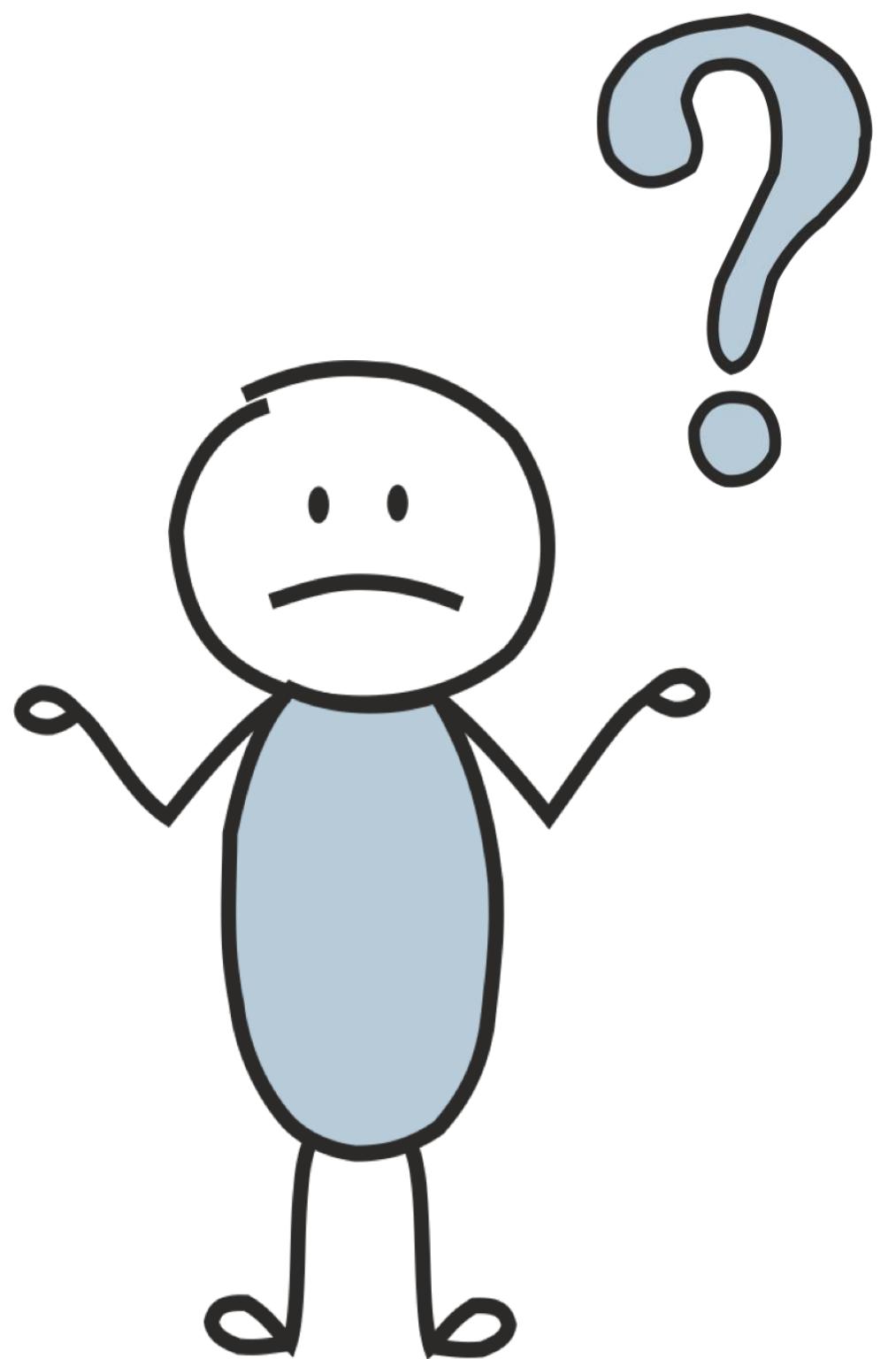
Nous l'avons vu :  
l'encodeur  
comprime la source  
en un seul vecteur



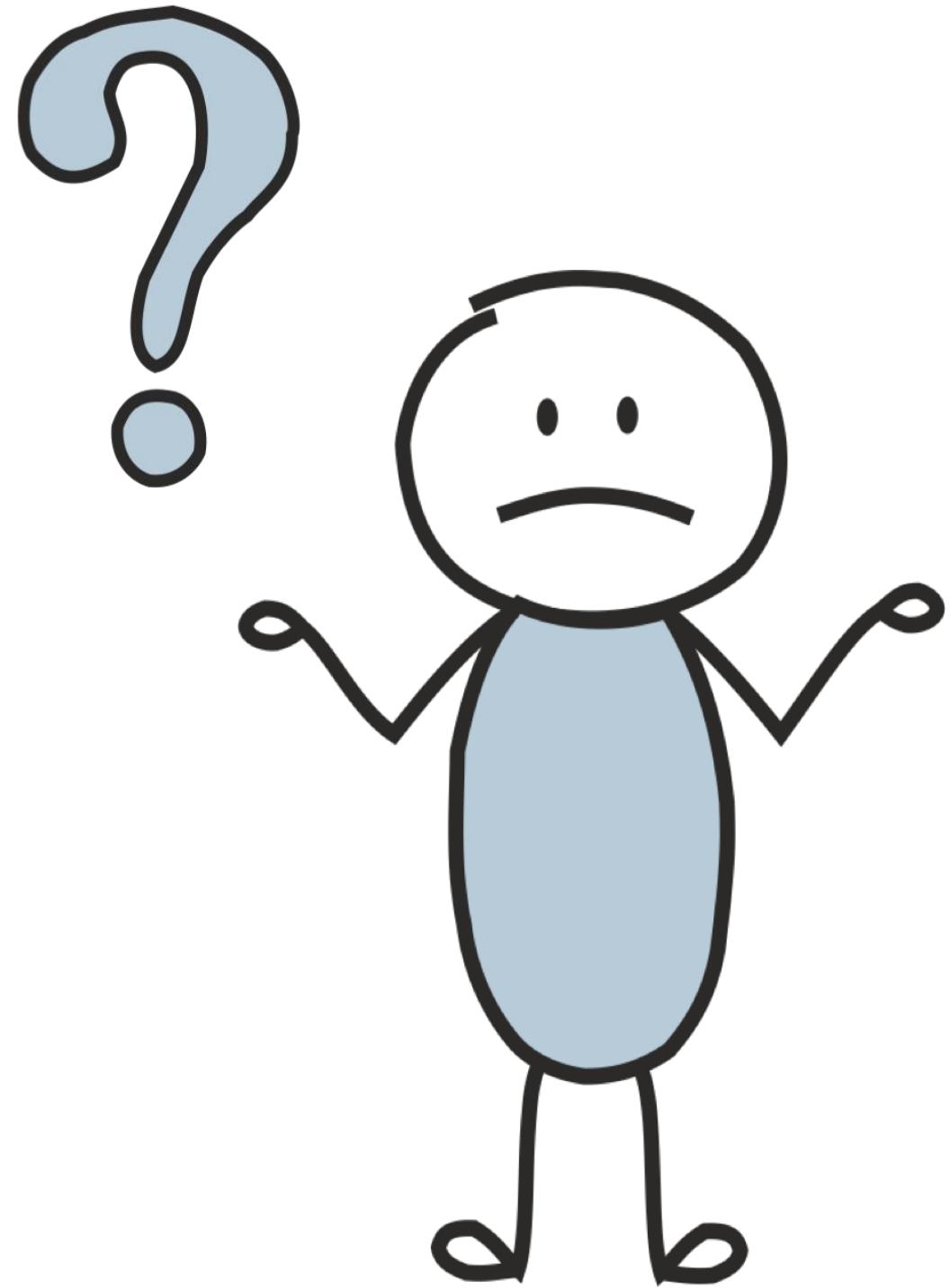
Problème : il s'agit d'un goulot d'étranglement !

**IMPORTANT**

# Des idées ?



# Des idées ?



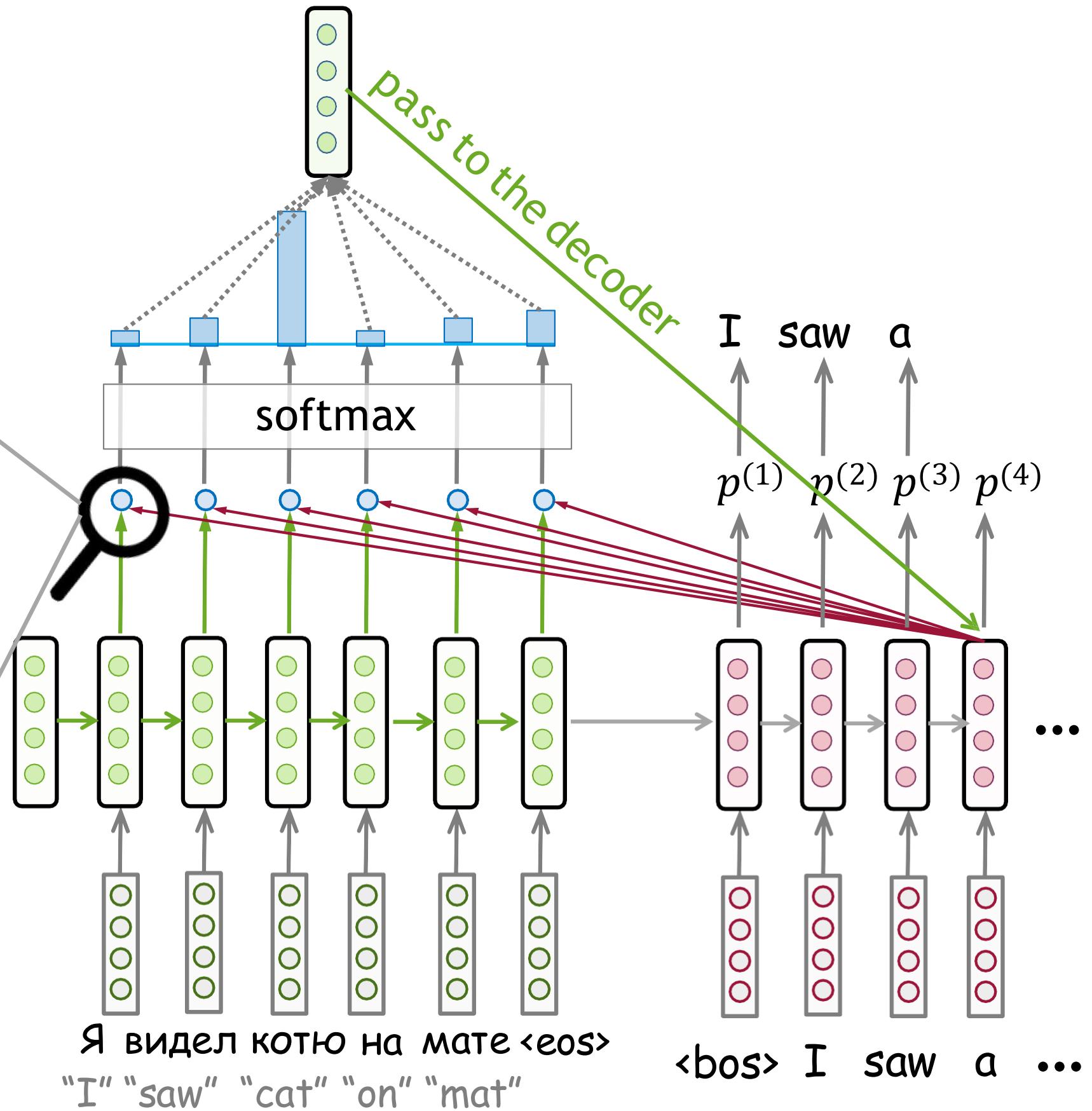
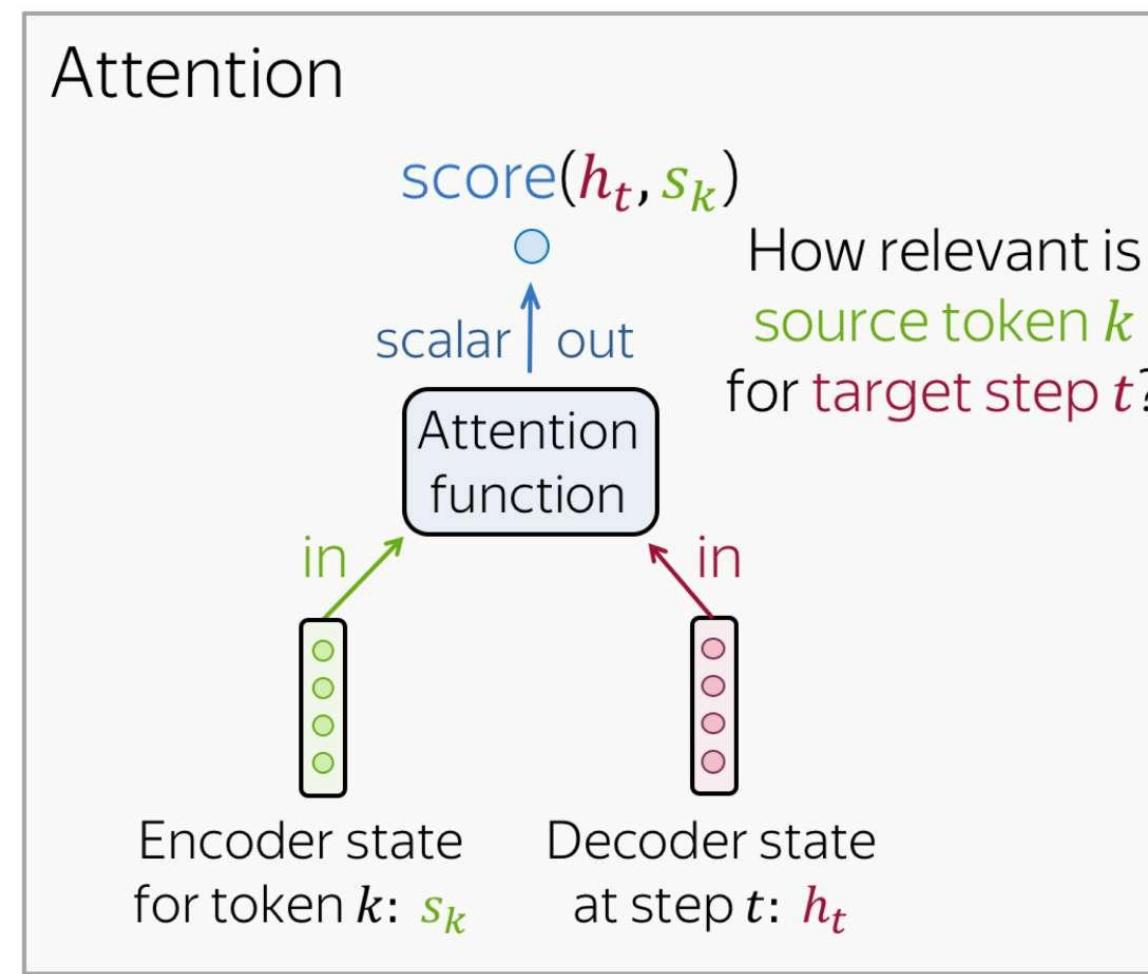
Moins compresser  
l'information

L'info juste nécessaire  
au bon moment

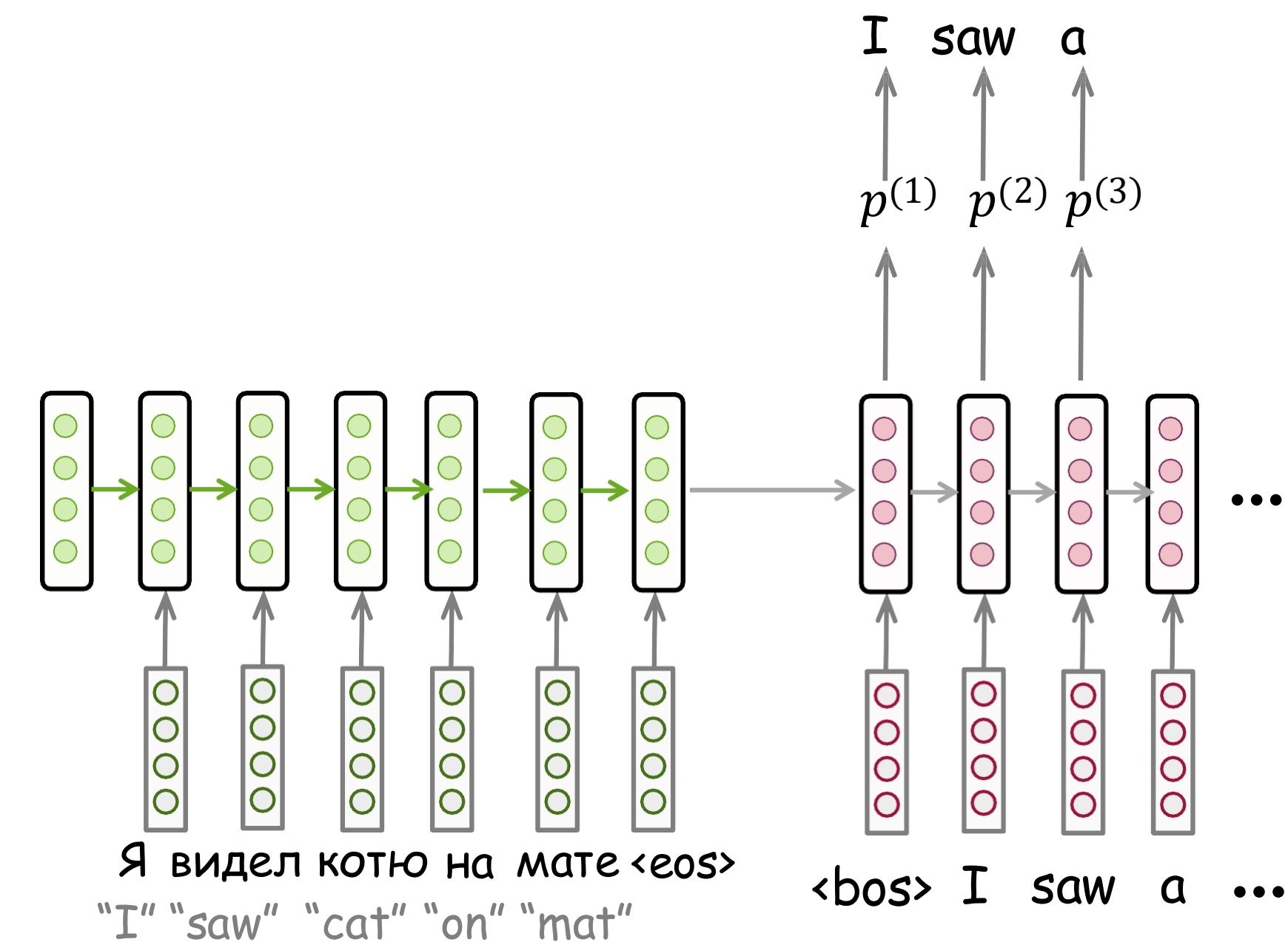
# Attention dans les RNN avec Encodeur-Decodage

A différentes étapes de la traduction, se concentrer sur différentes parties de la source

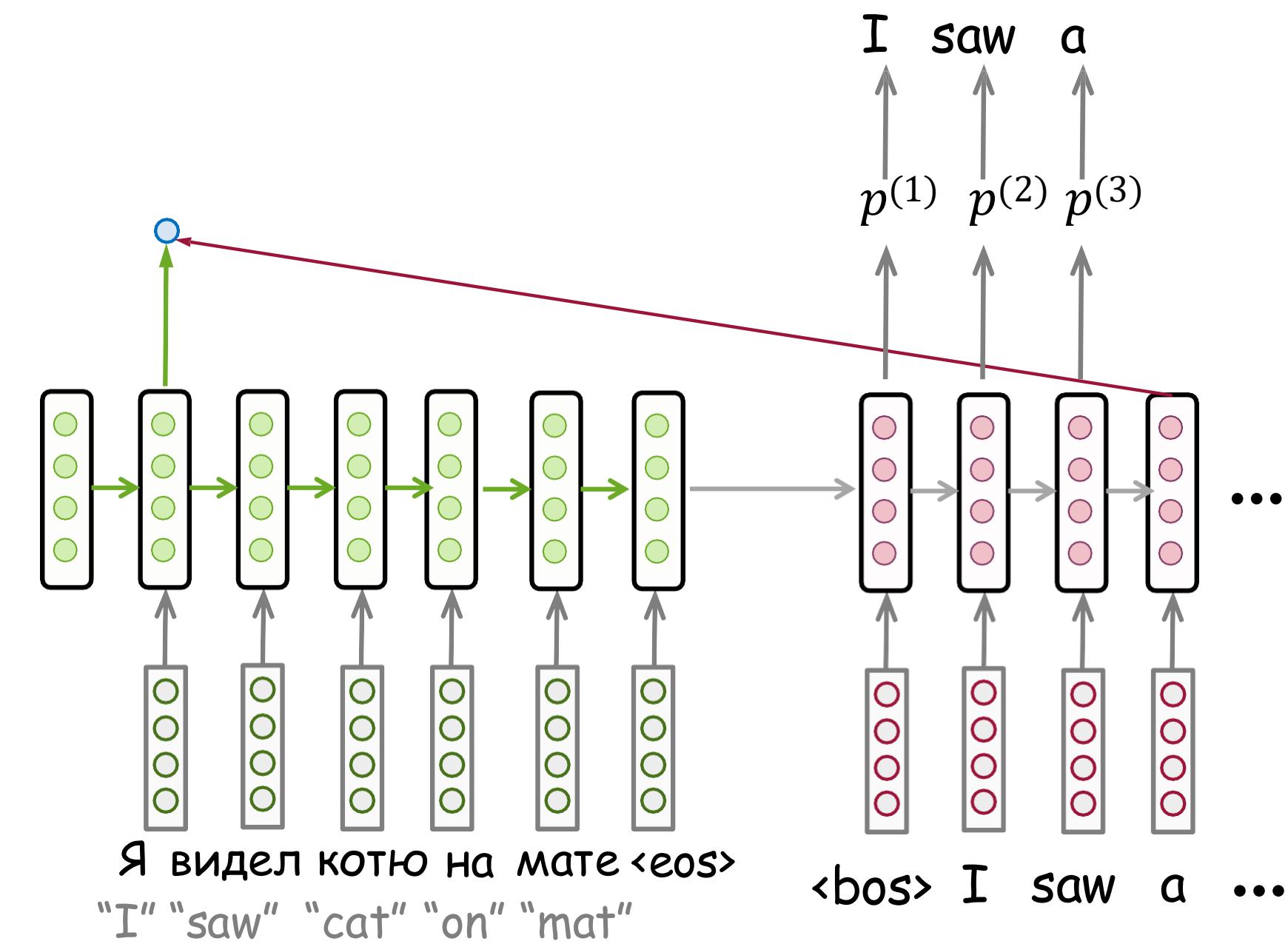
**IMPORTANT**



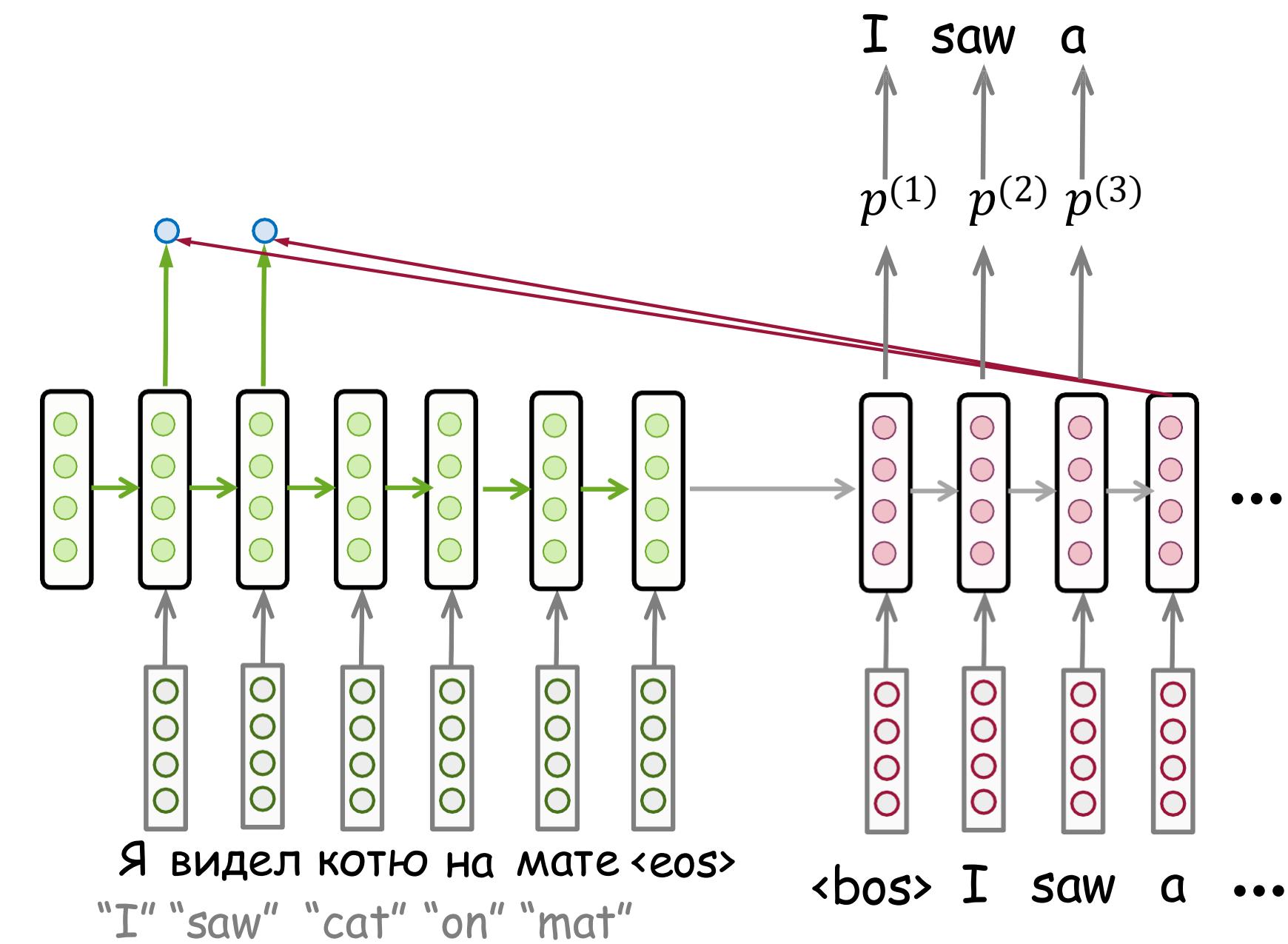
# Attention : Vue de haut niveau



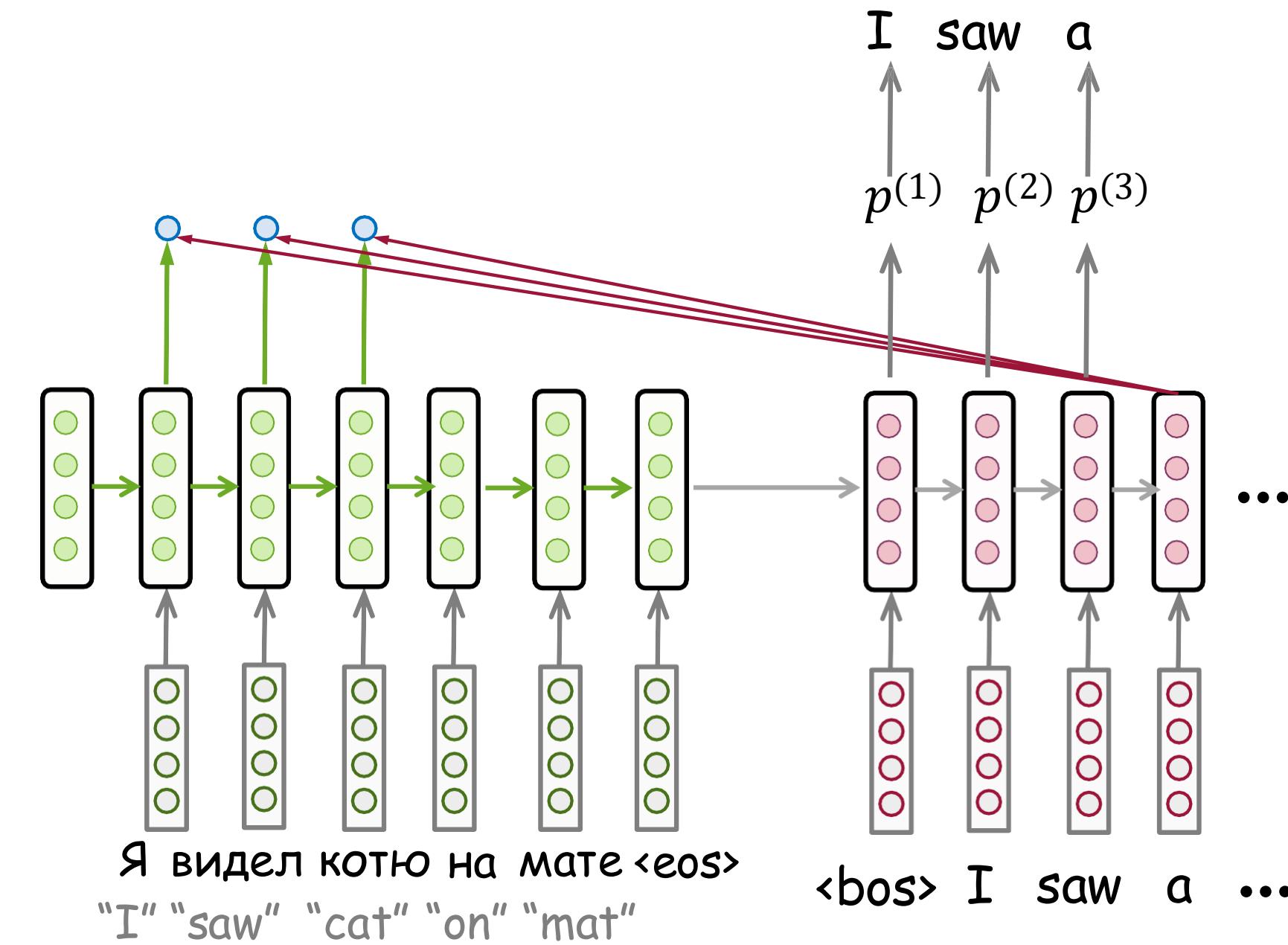
# Attention : Vue de haut niveau



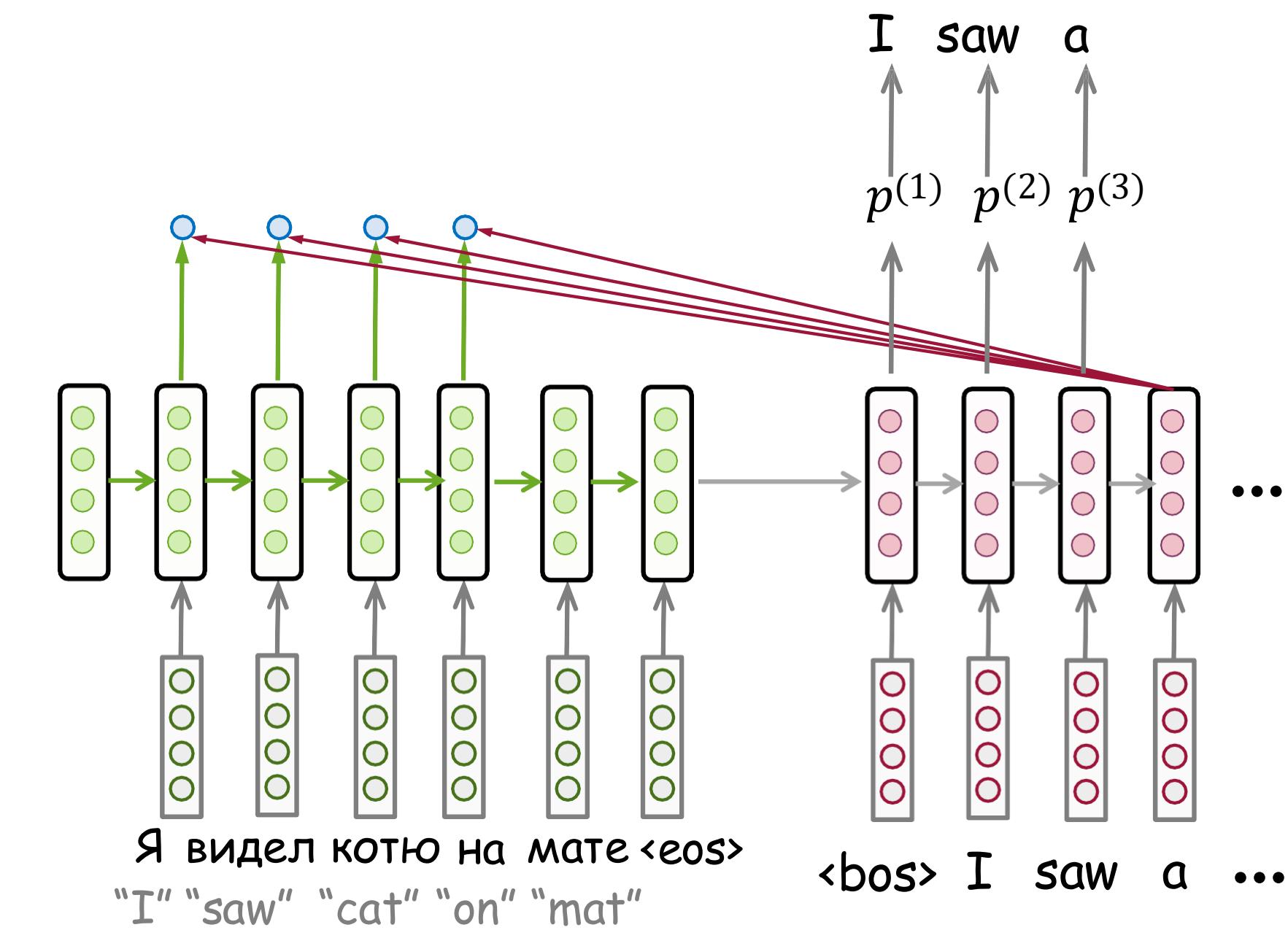
# Attention : Vue de haut niveau



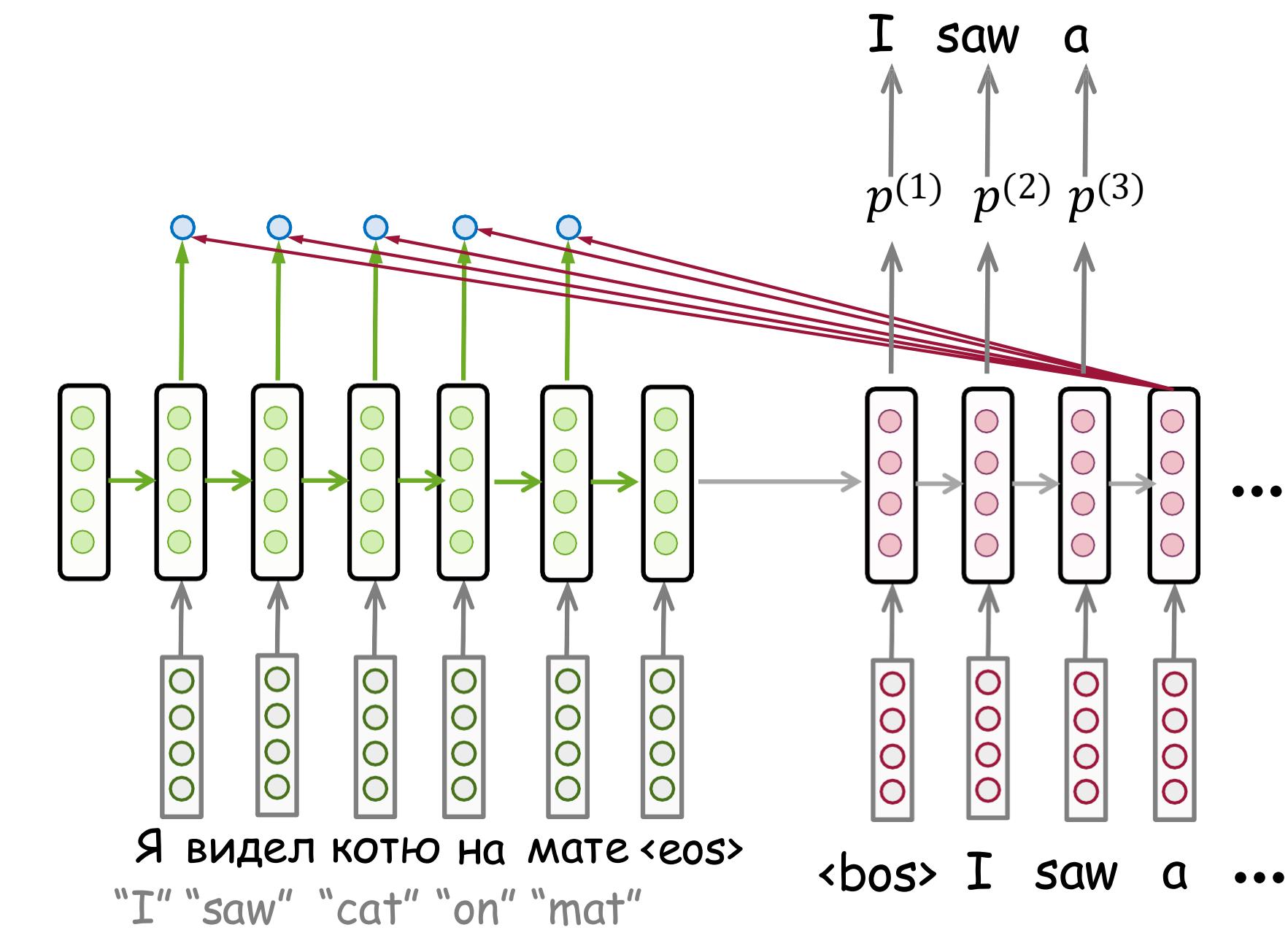
# Attention : Vue de haut niveau



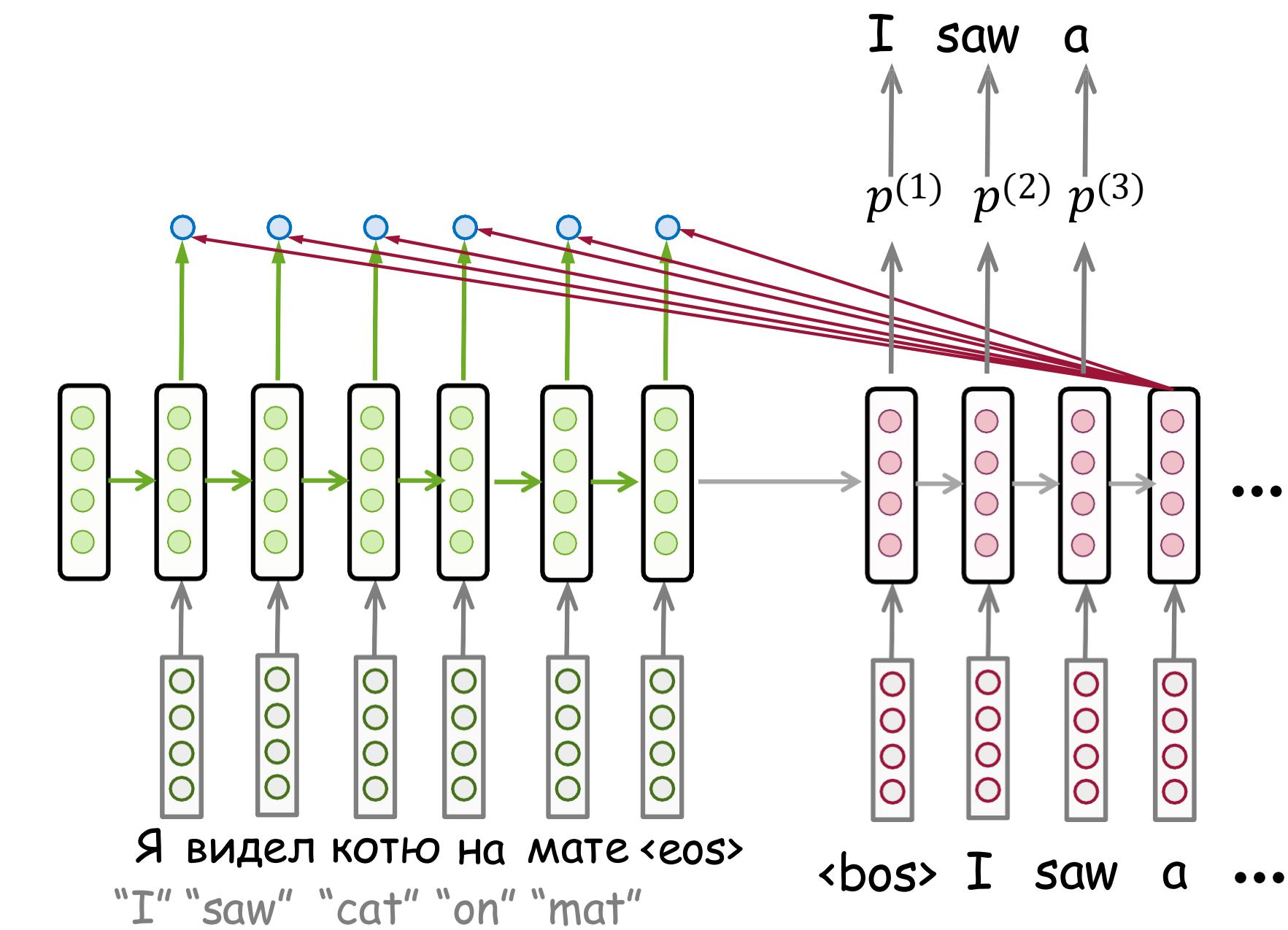
# Attention : Vue de haut niveau



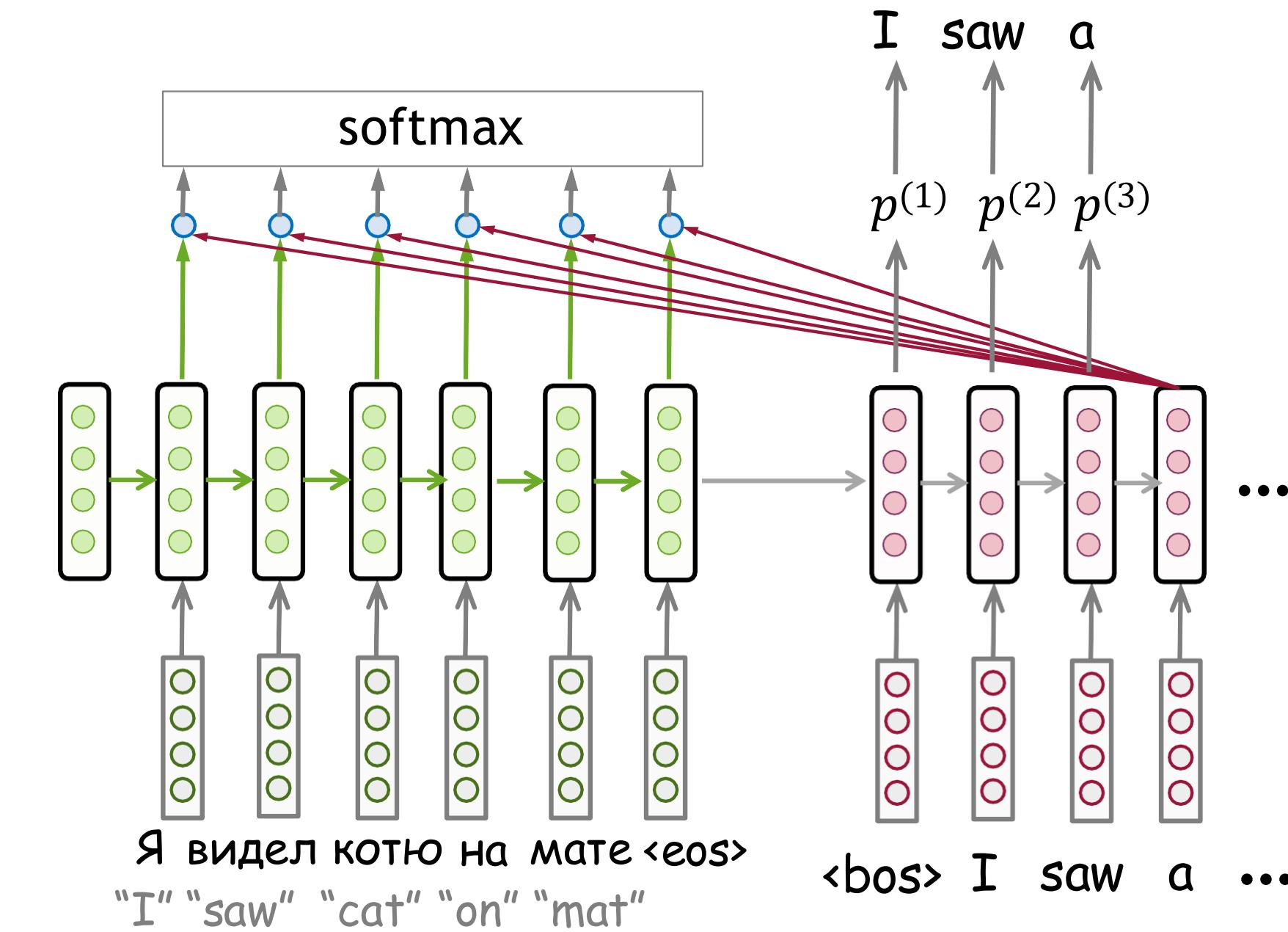
# Attention : Vue de haut niveau



# Attention : Vue de haut niveau



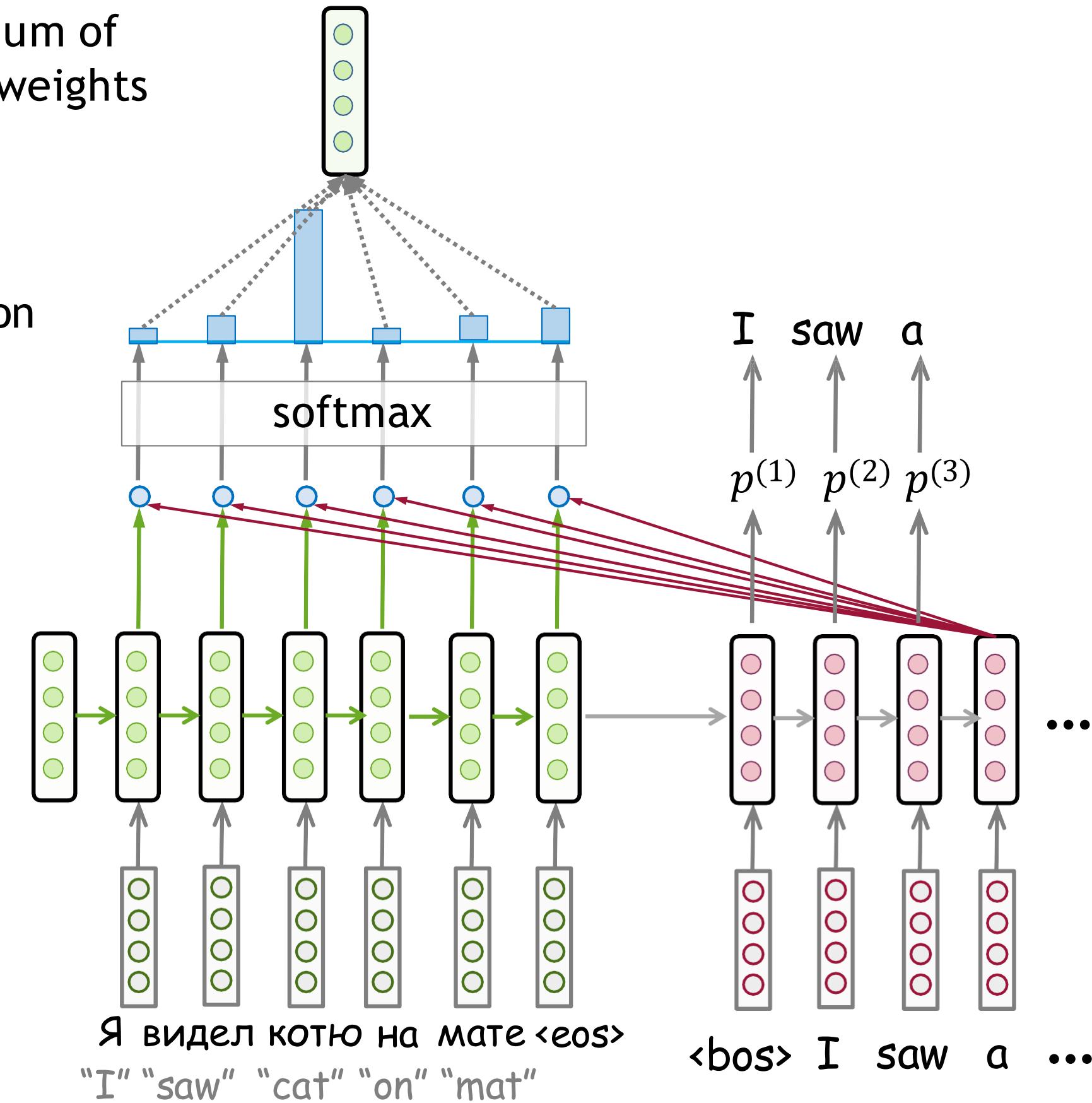
# Attention : Vue de haut niveau



# Attention : Vue de haut niveau

Attention output: weighted sum of encoder states with attention weights

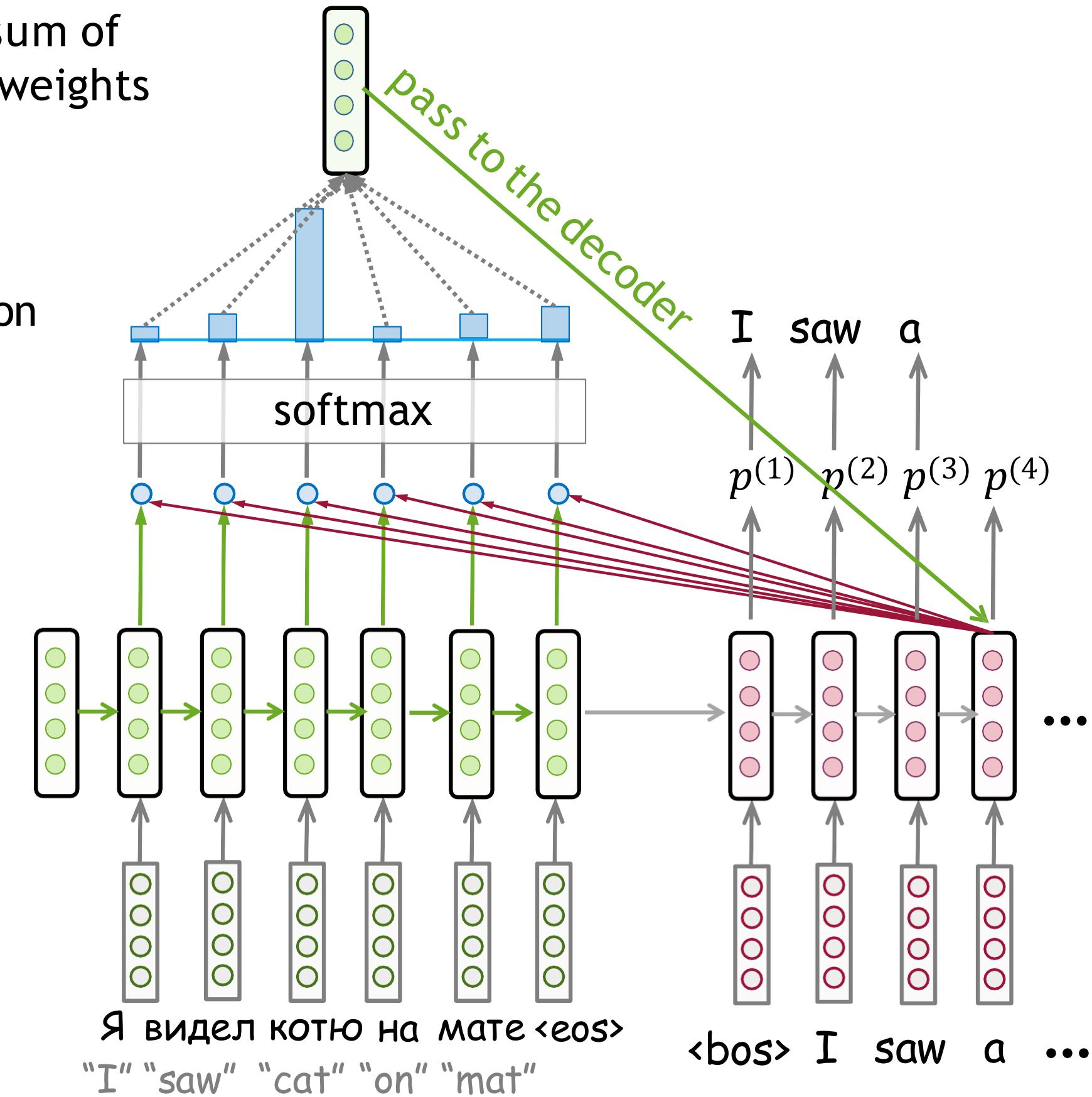
Attention weights: distribution over source tokens



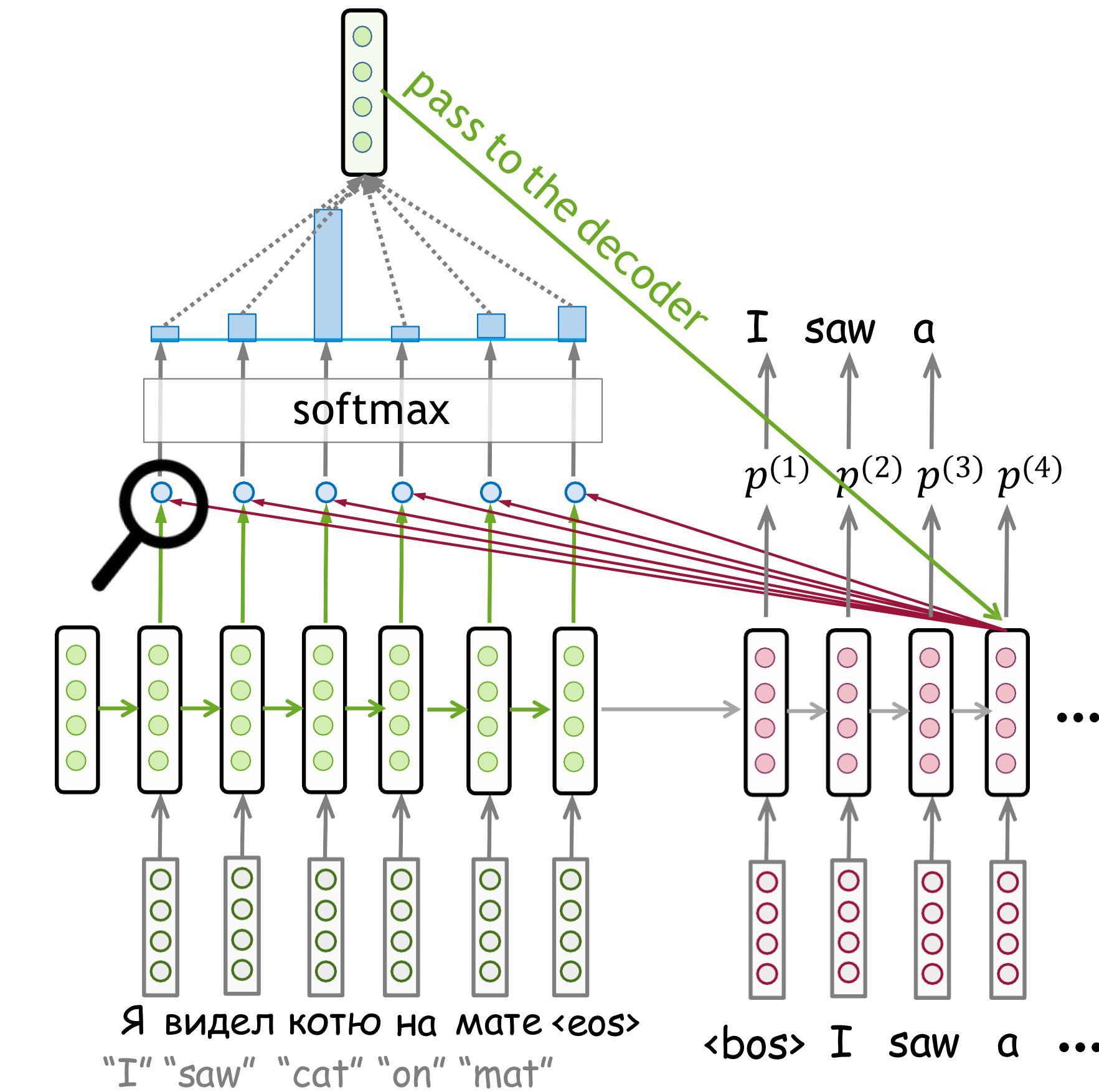
# Attention : Vue de haut niveau

Attention output: weighted sum of encoder states with attention weights

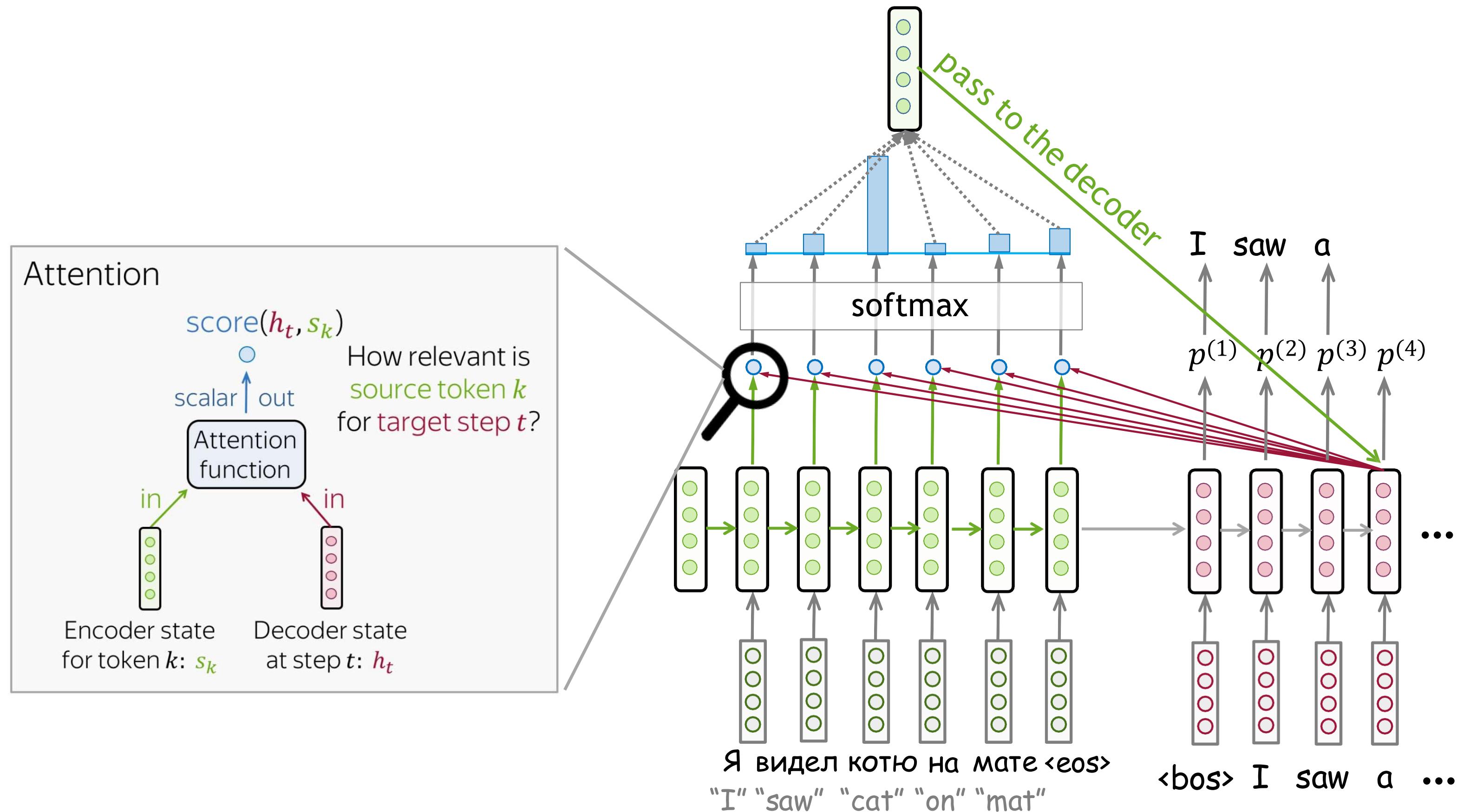
Attention weights: distribution over source tokens



# Attention : Vue de haut niveau



# Attention : Vue de haut niveau



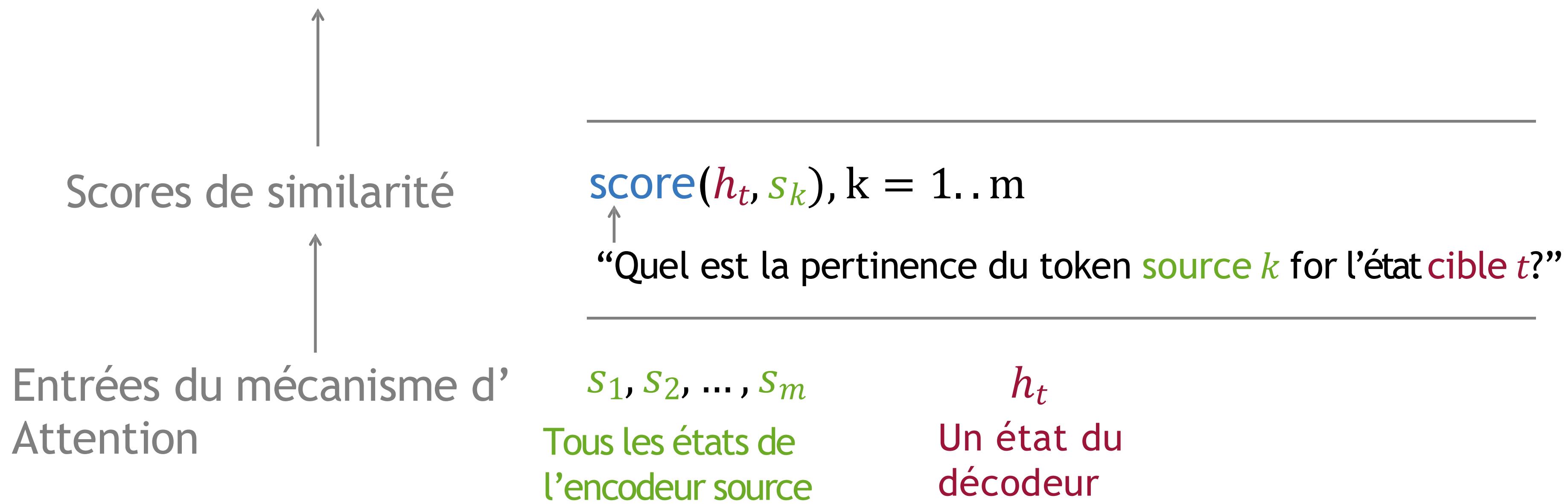
# Pipeline de calcul

Entrées du mécanisme d'  
Attention

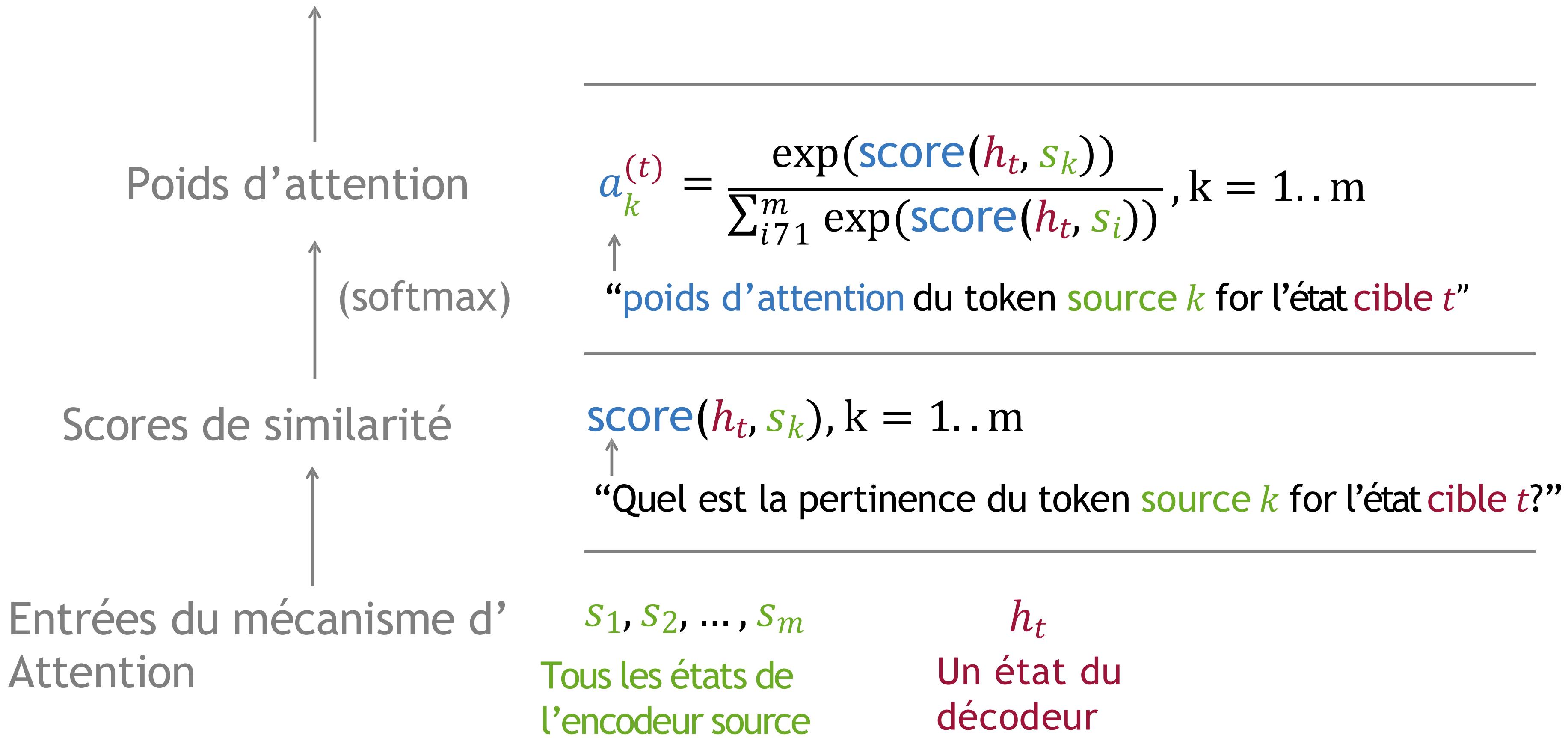
$s_1, s_2, \dots, s_m$   
Tous les états de  
l'encodeur source

$h_t$   
Un état du  
décodeur

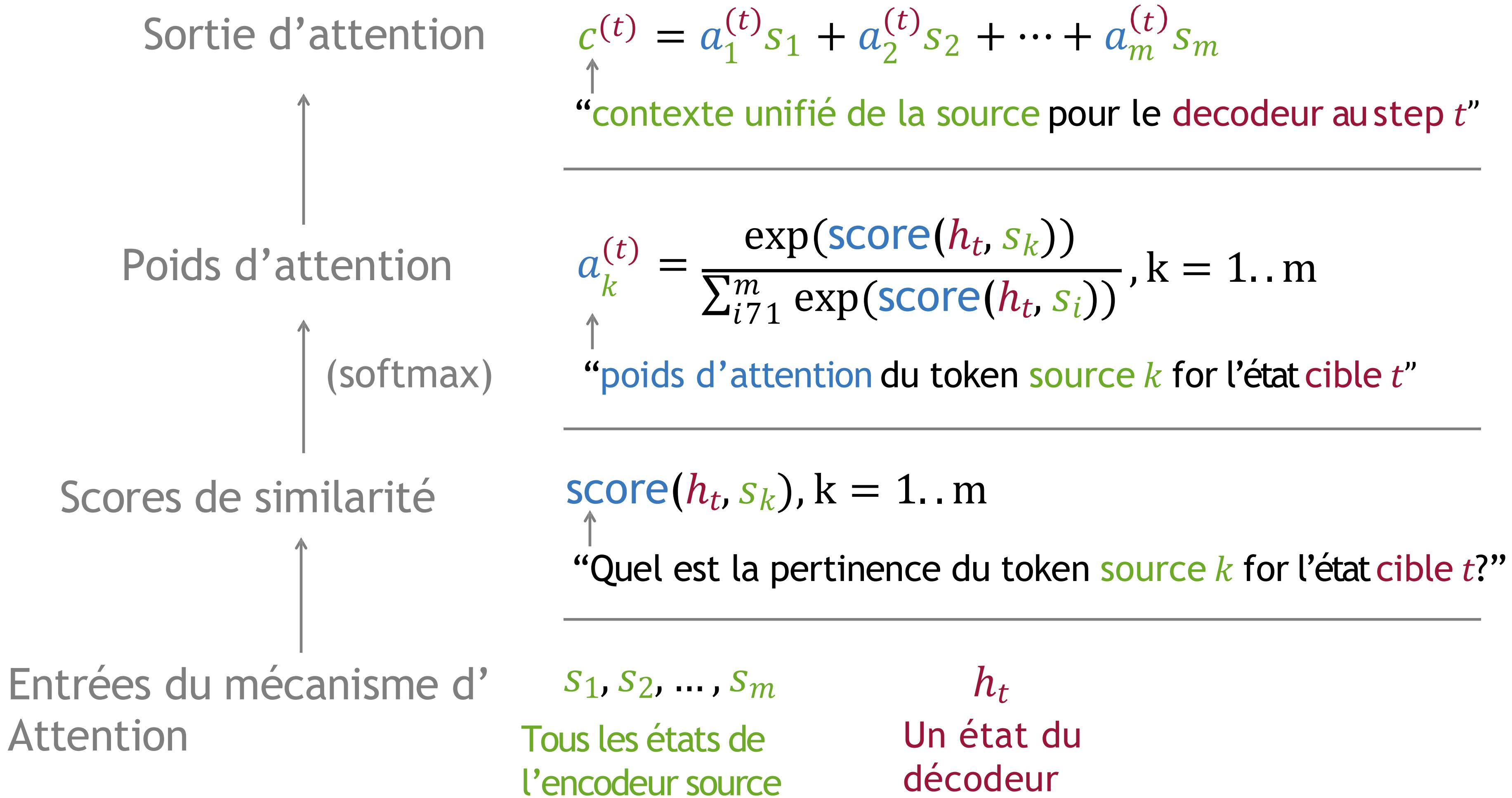
# Pipeline de calcul



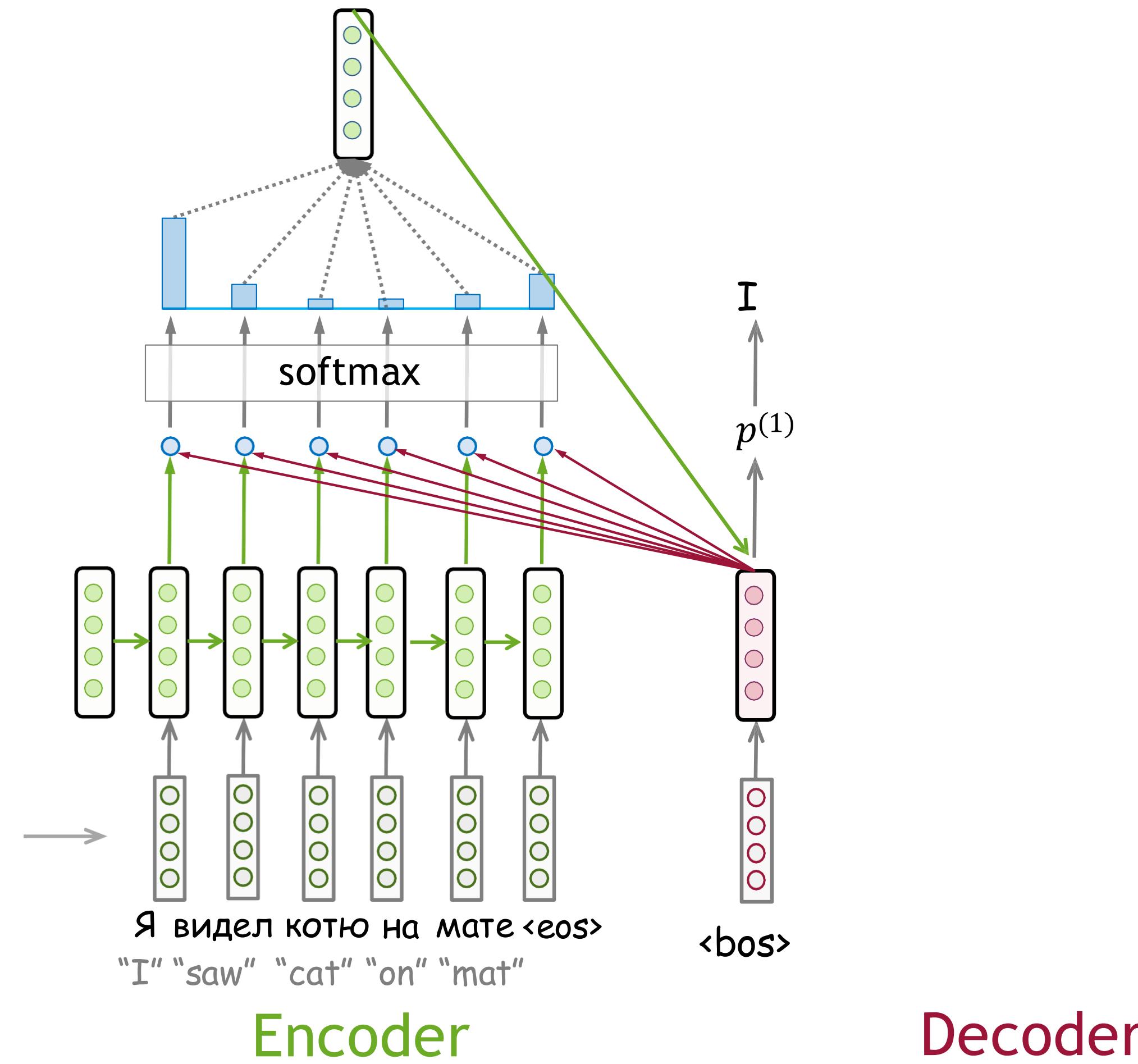
# Pipeline de calcul



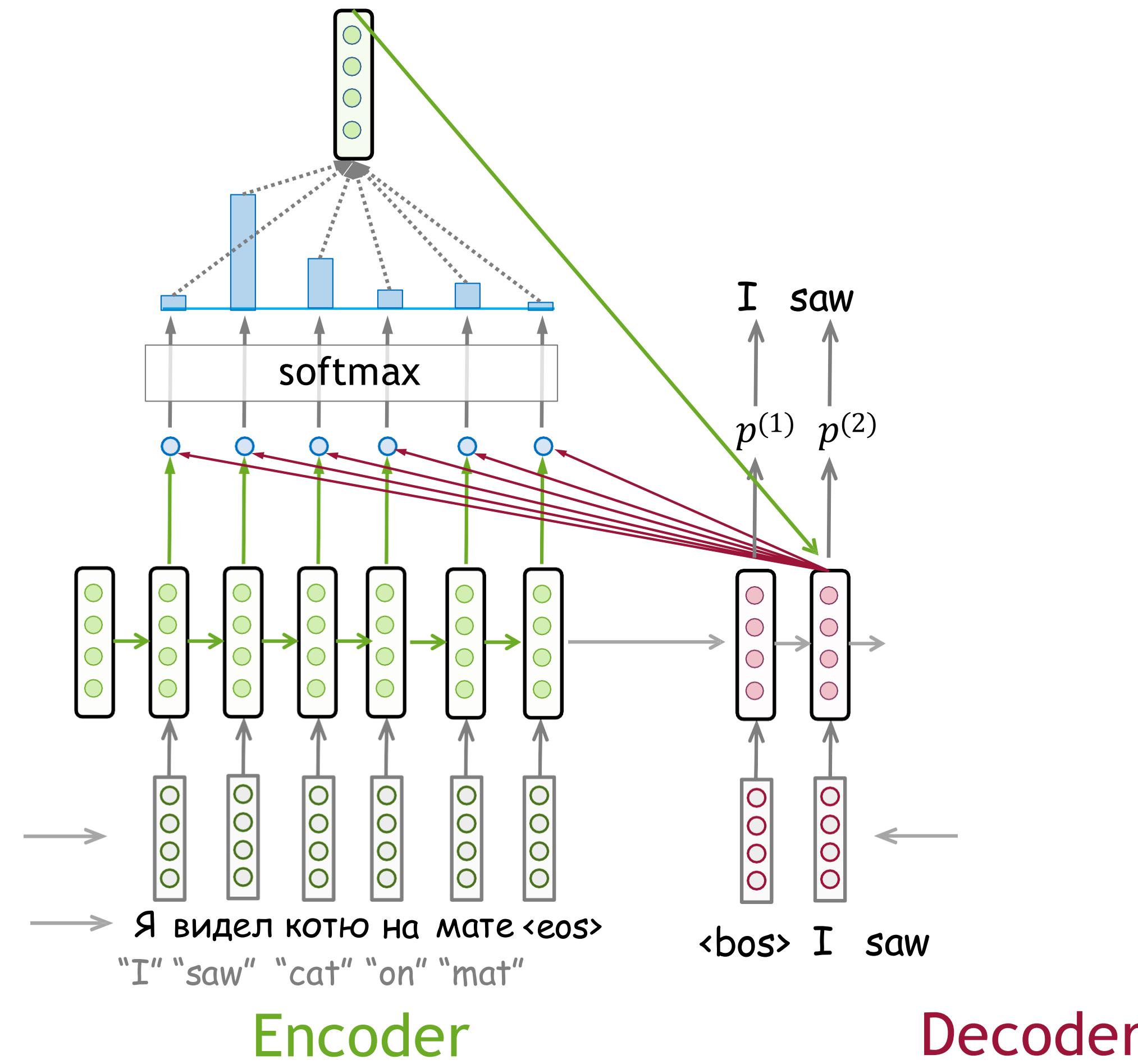
# Pipeline de calcul



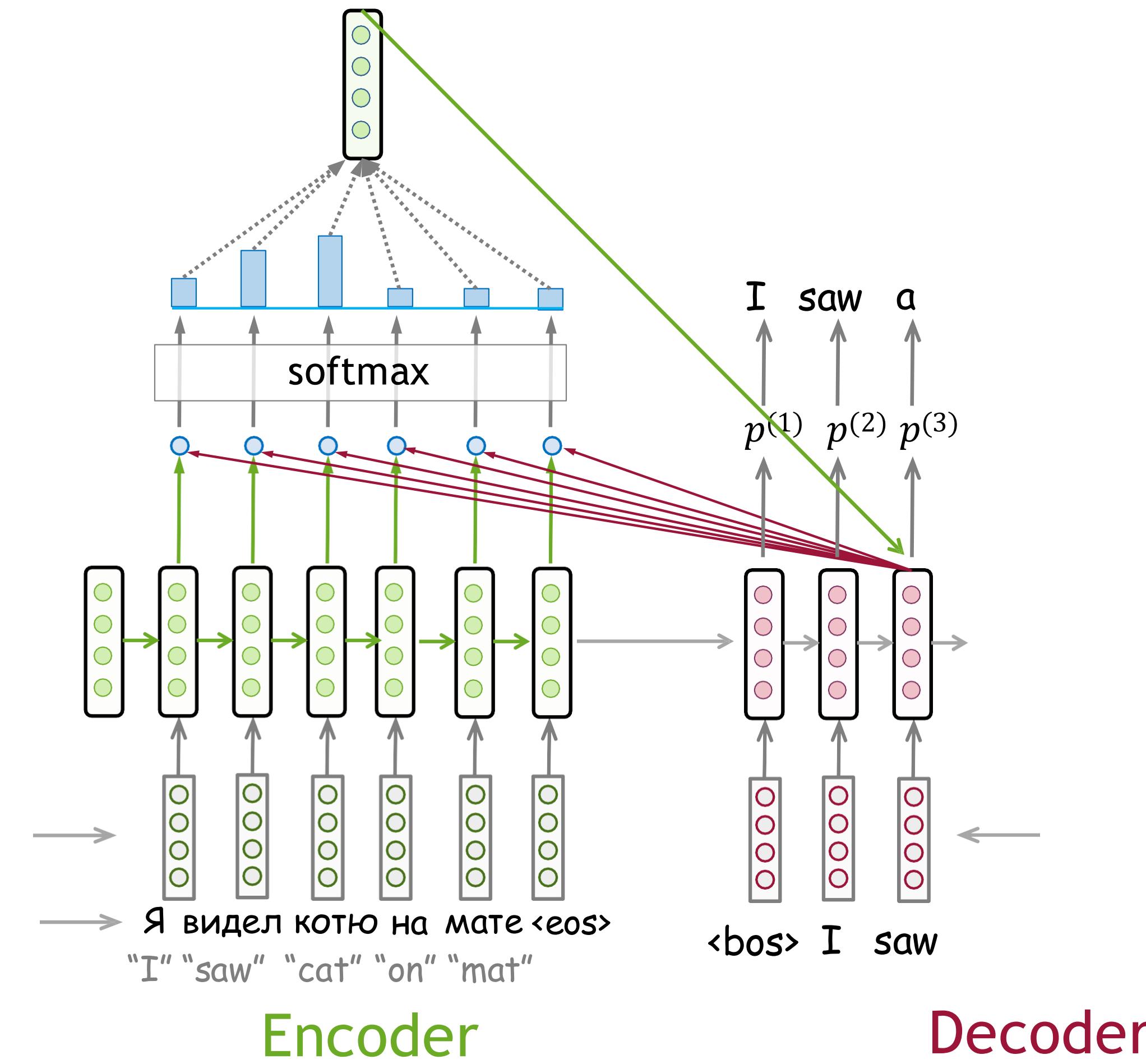
# Le modèle apprend à choisir des tokens pertinents



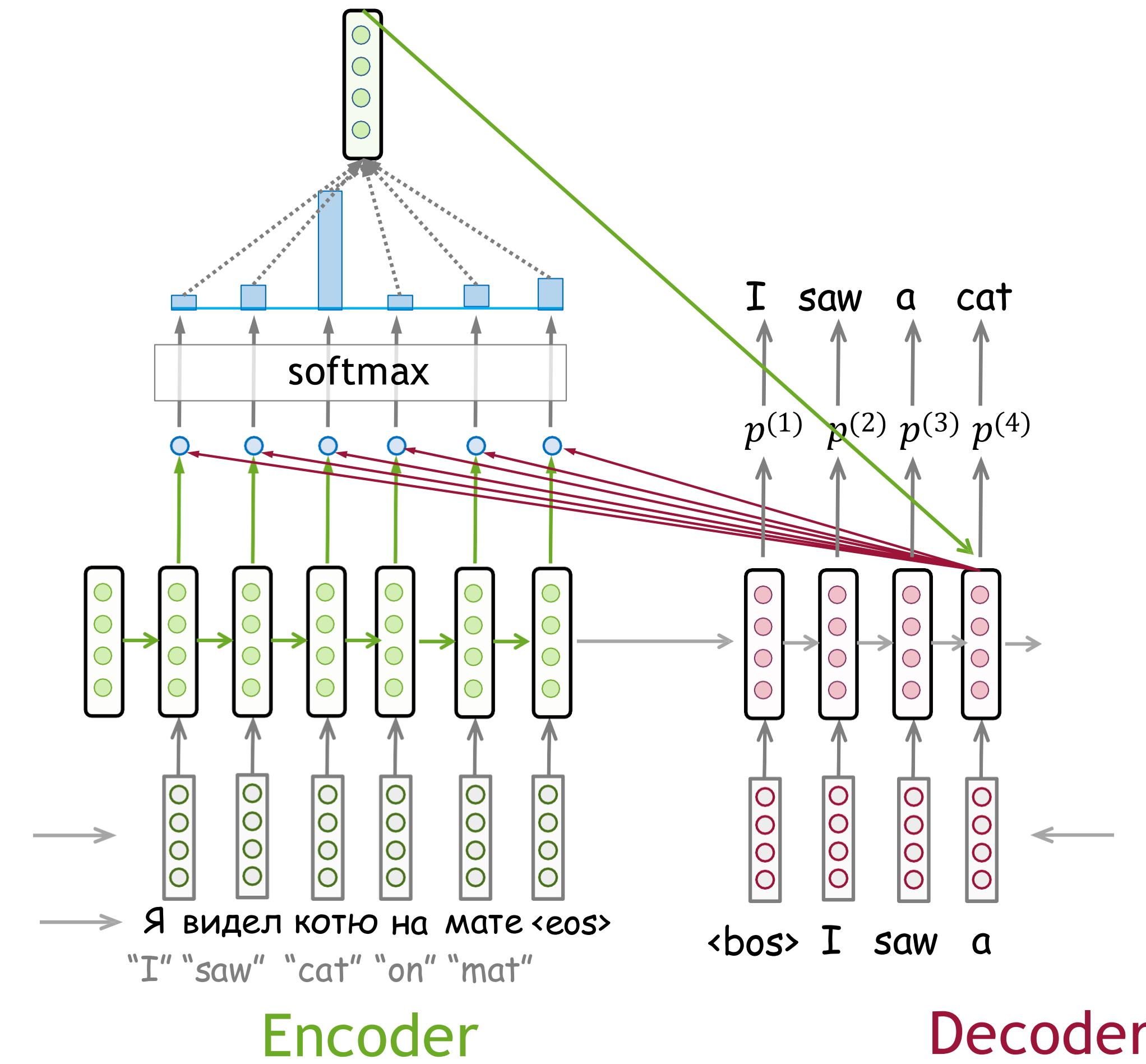
# Le modèle apprend à choisir des tokens pertinents



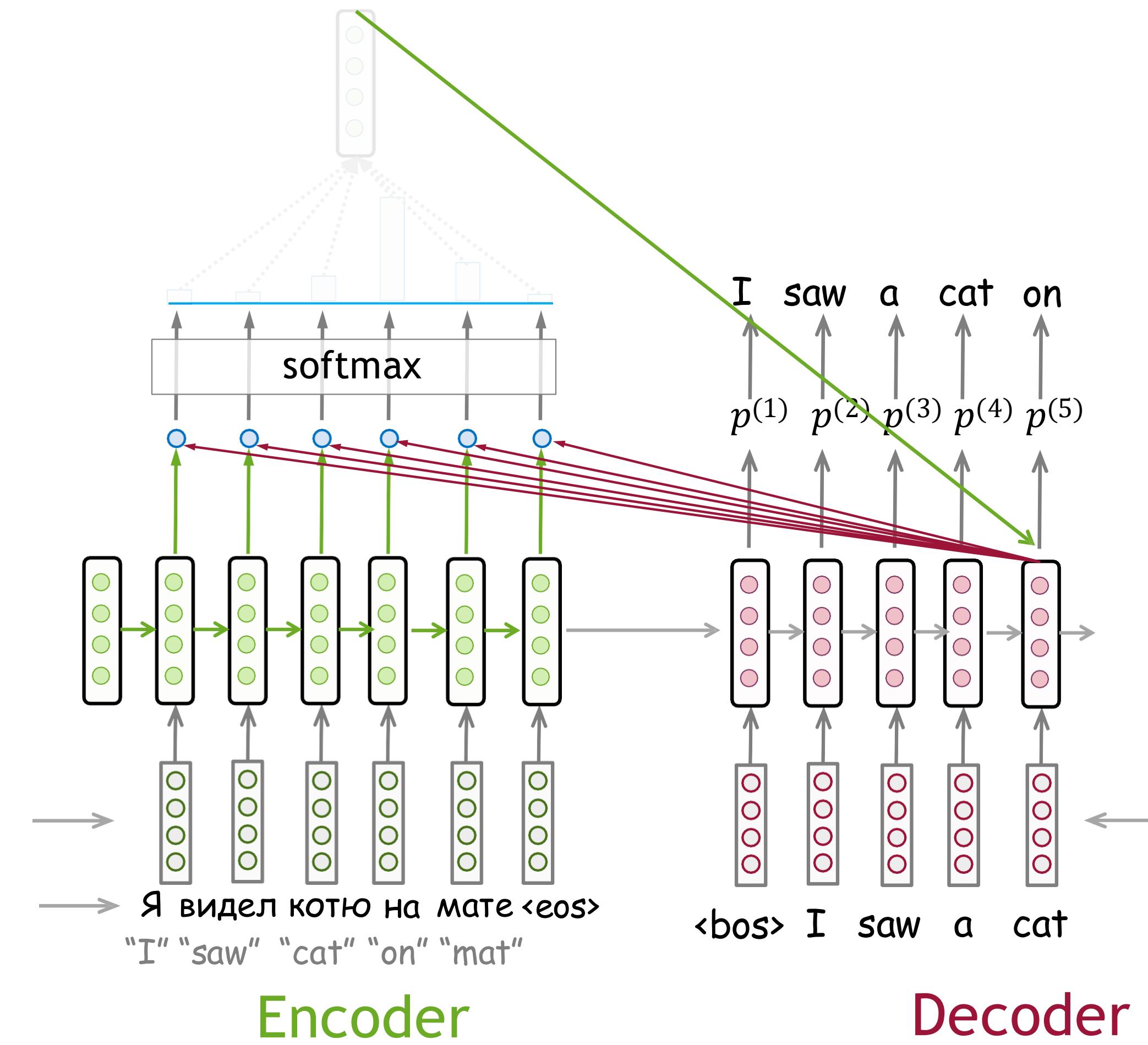
# Le modèle apprend à choisir des tokens pertinents



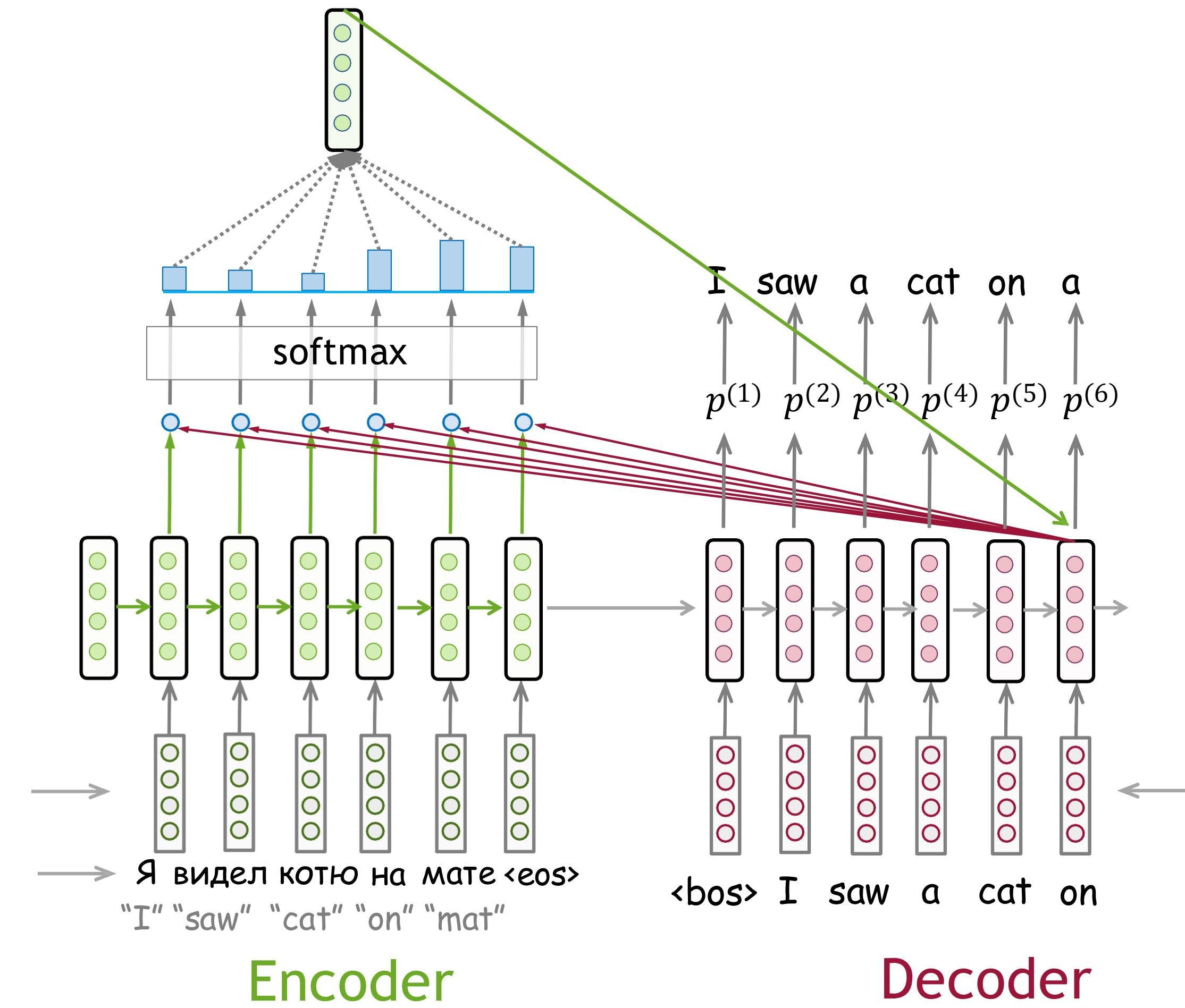
# Le modèle apprend à choisir des tokens pertinents



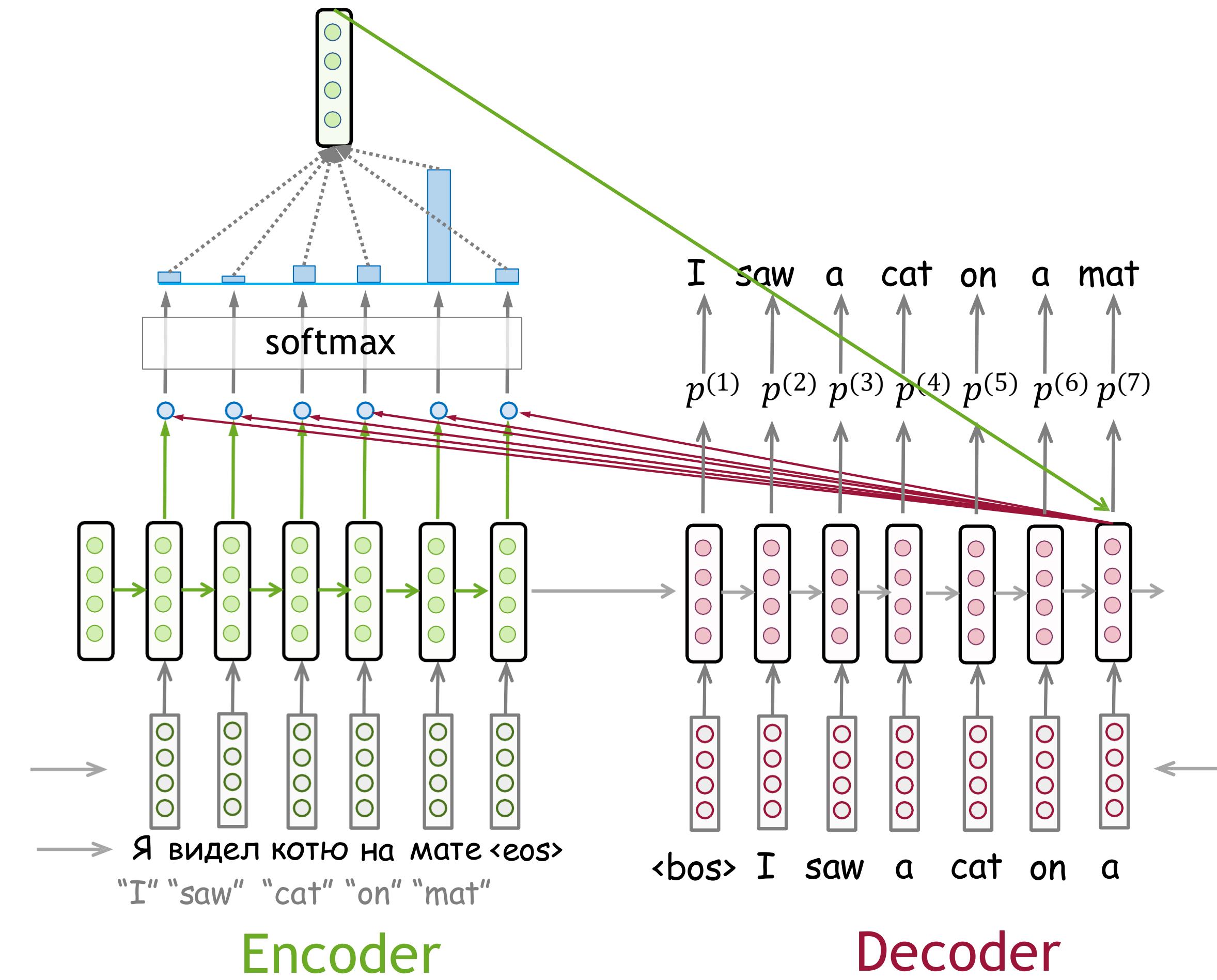
# Le modèle apprend à choisir des tokens pertinents



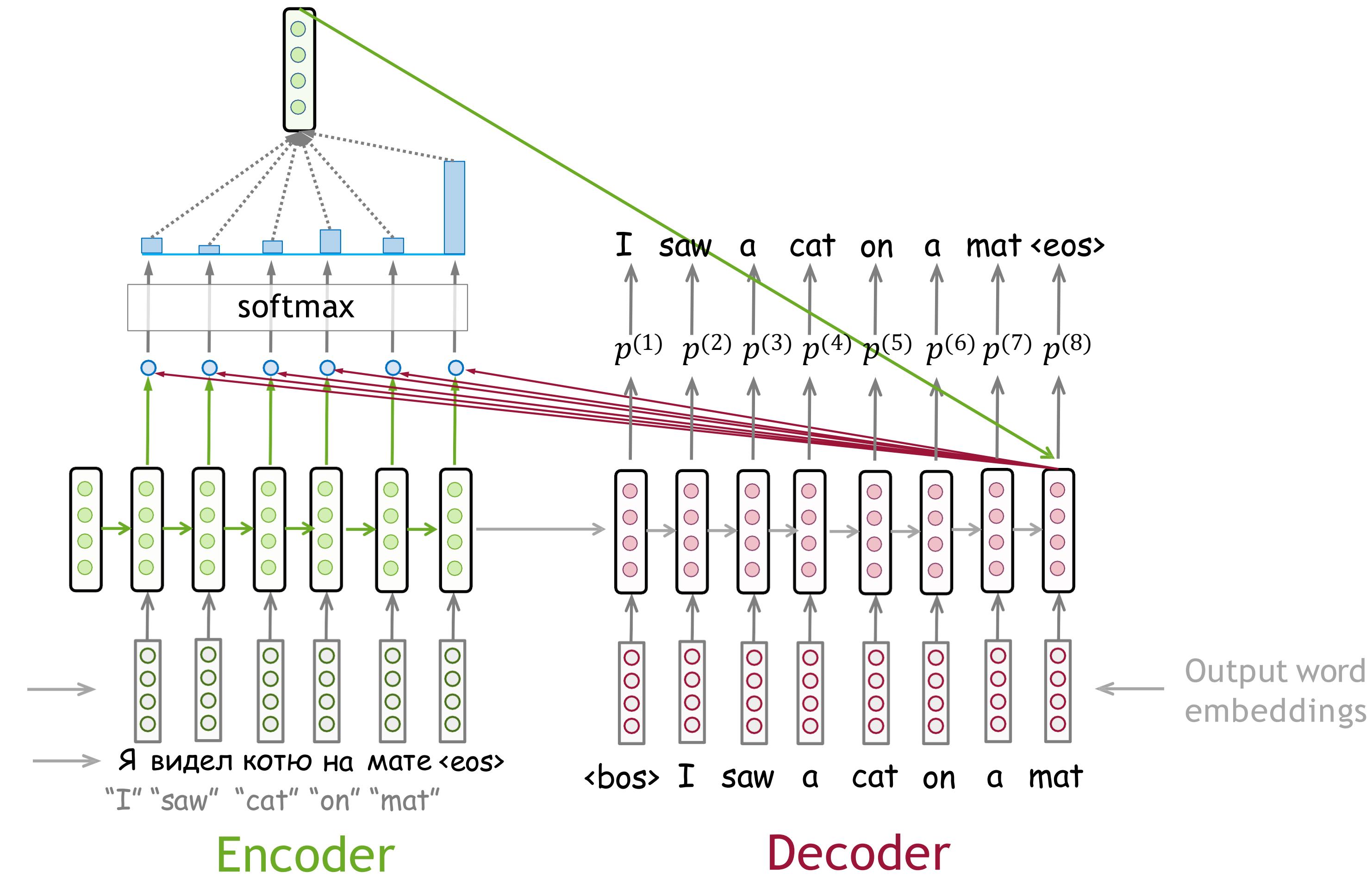
# Le modèle apprend à choisir des tokens pertinents



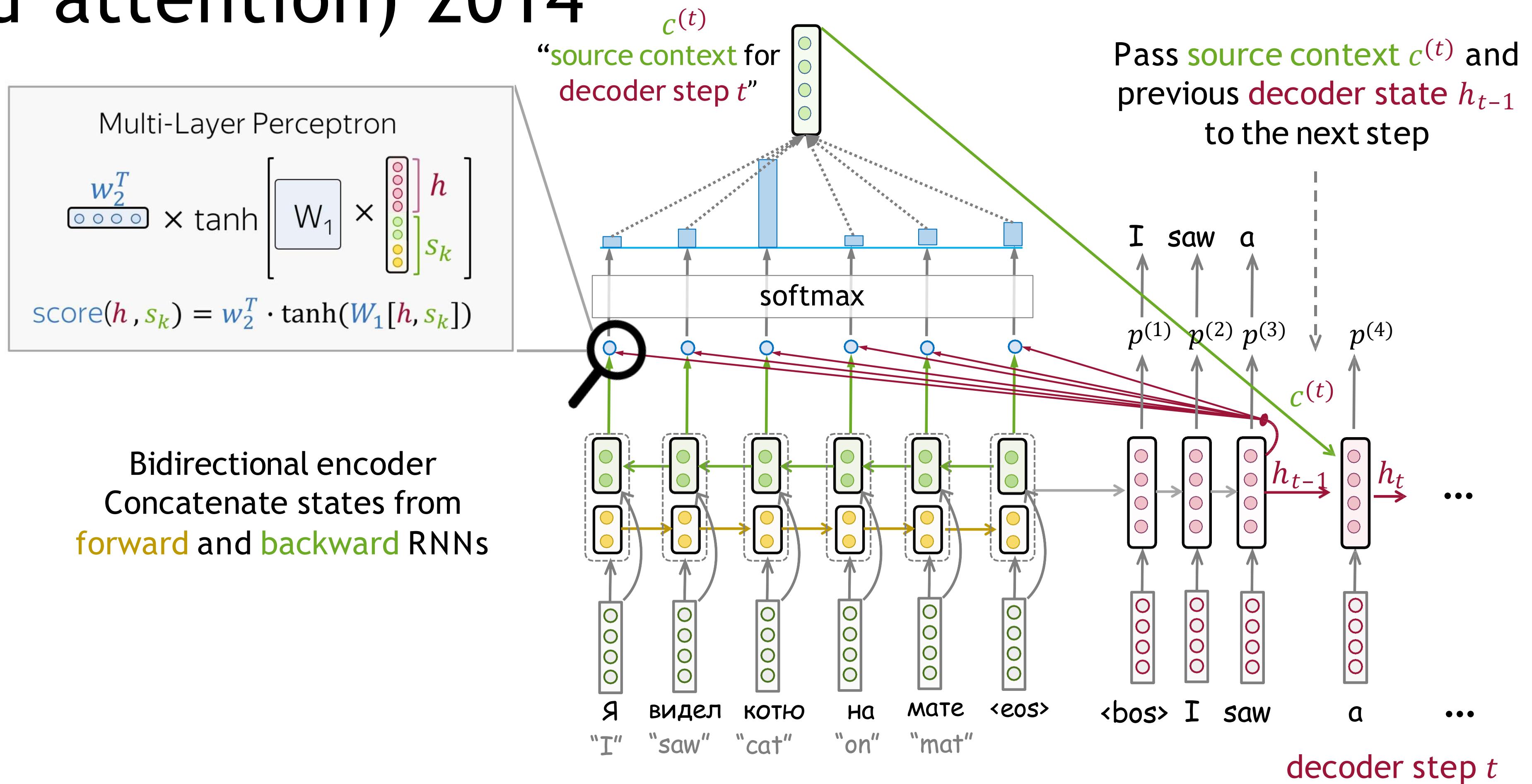
# Le modèle apprend à choisir des tokens pertinents



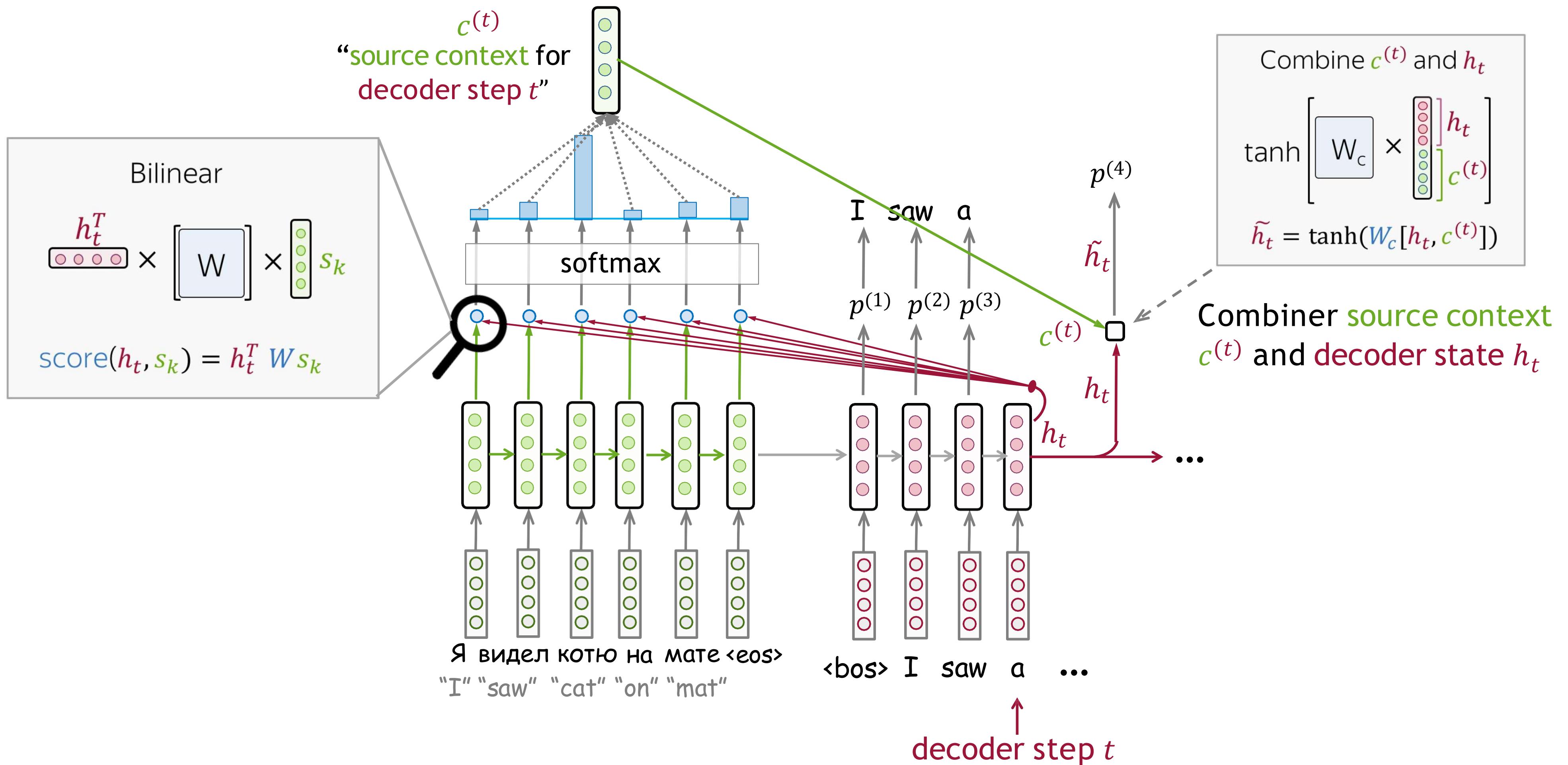
# Le modèle apprend à choisir des tokens pertinents



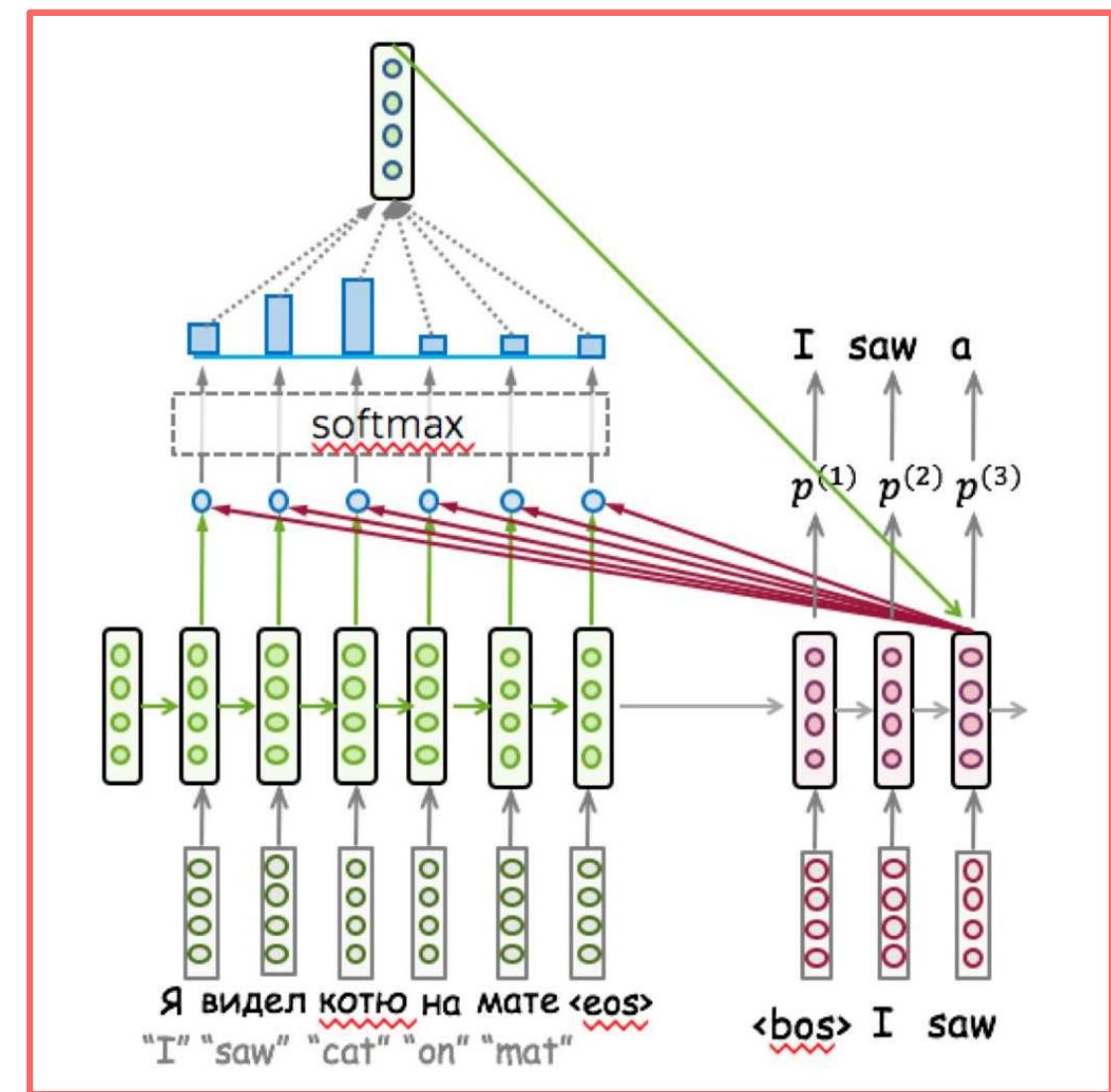
# Modèle Bahdanau (le modèle original d'attention) 2014



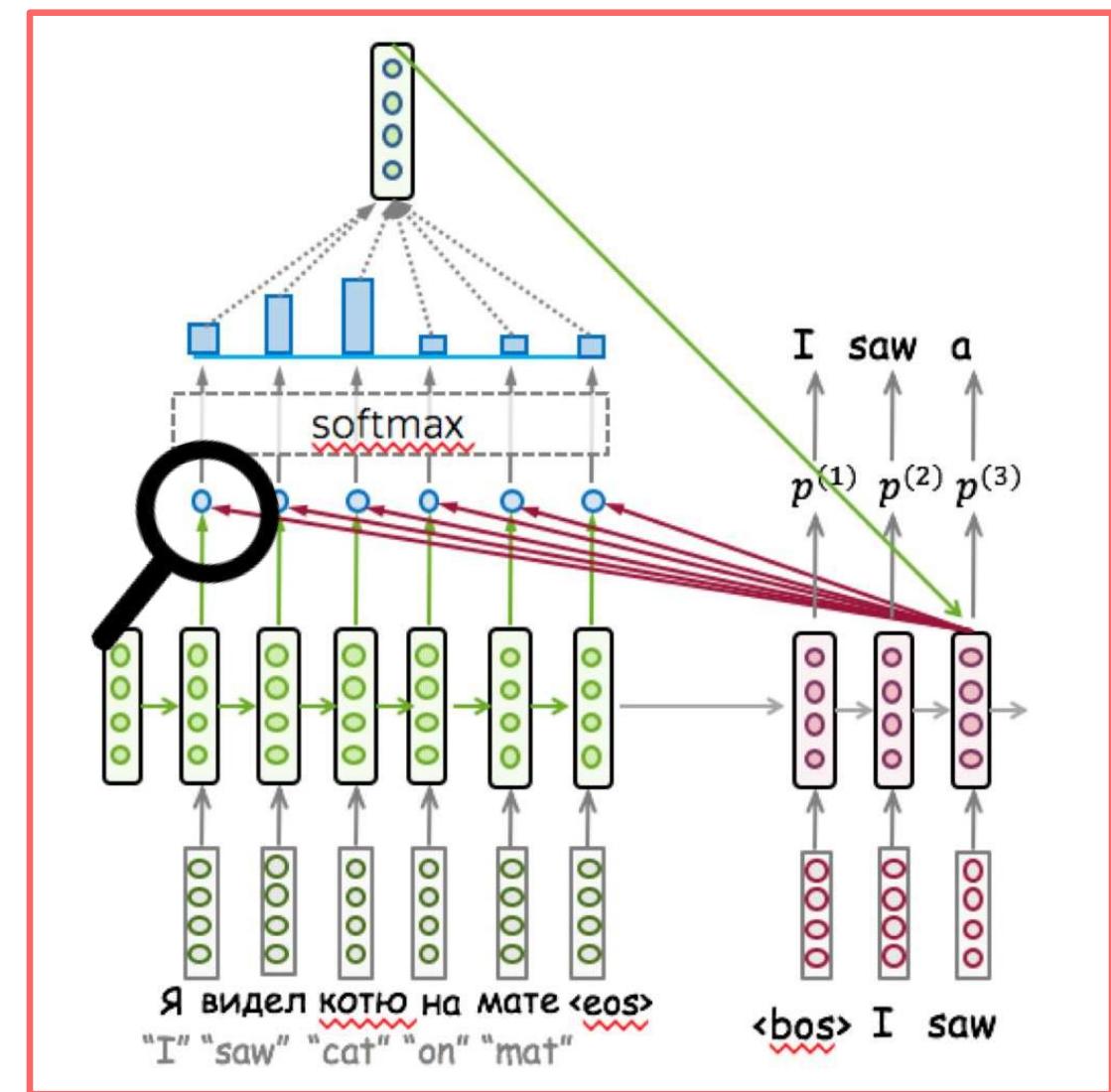
# Modèle Luong (2015)



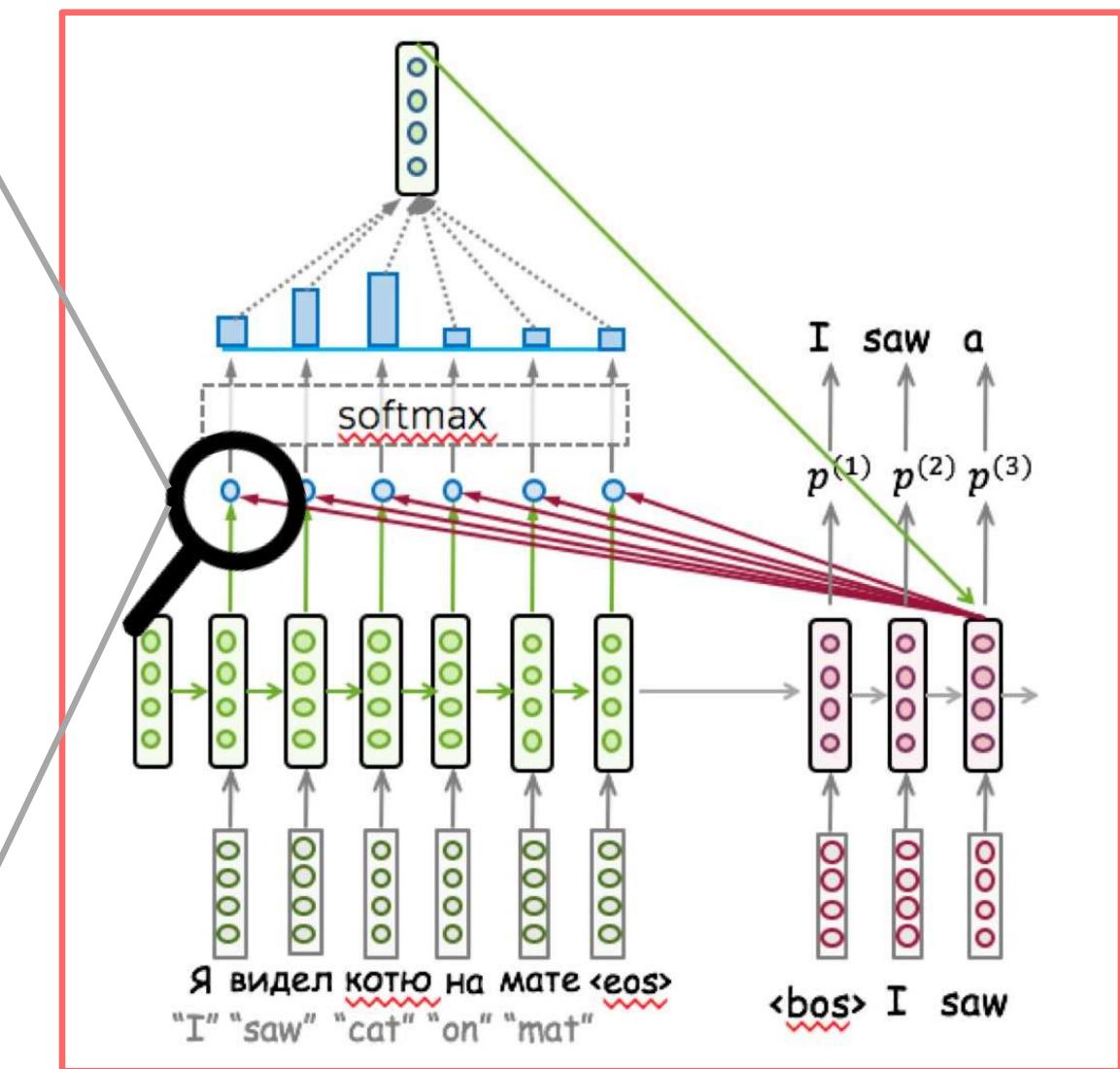
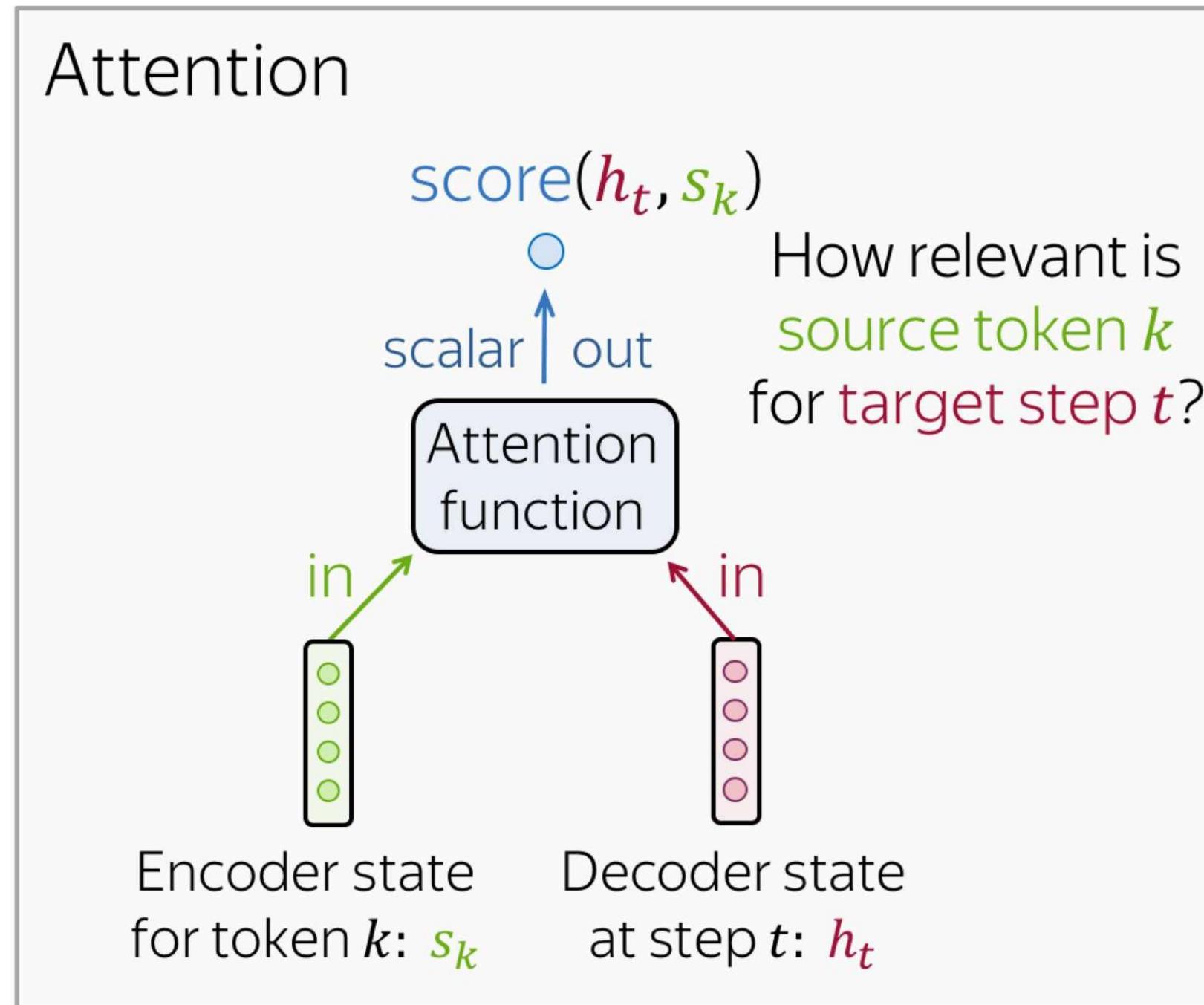
# Fonctions du score d'attention



# Fonctions du score d'attention



# Fonctions du score d'attention

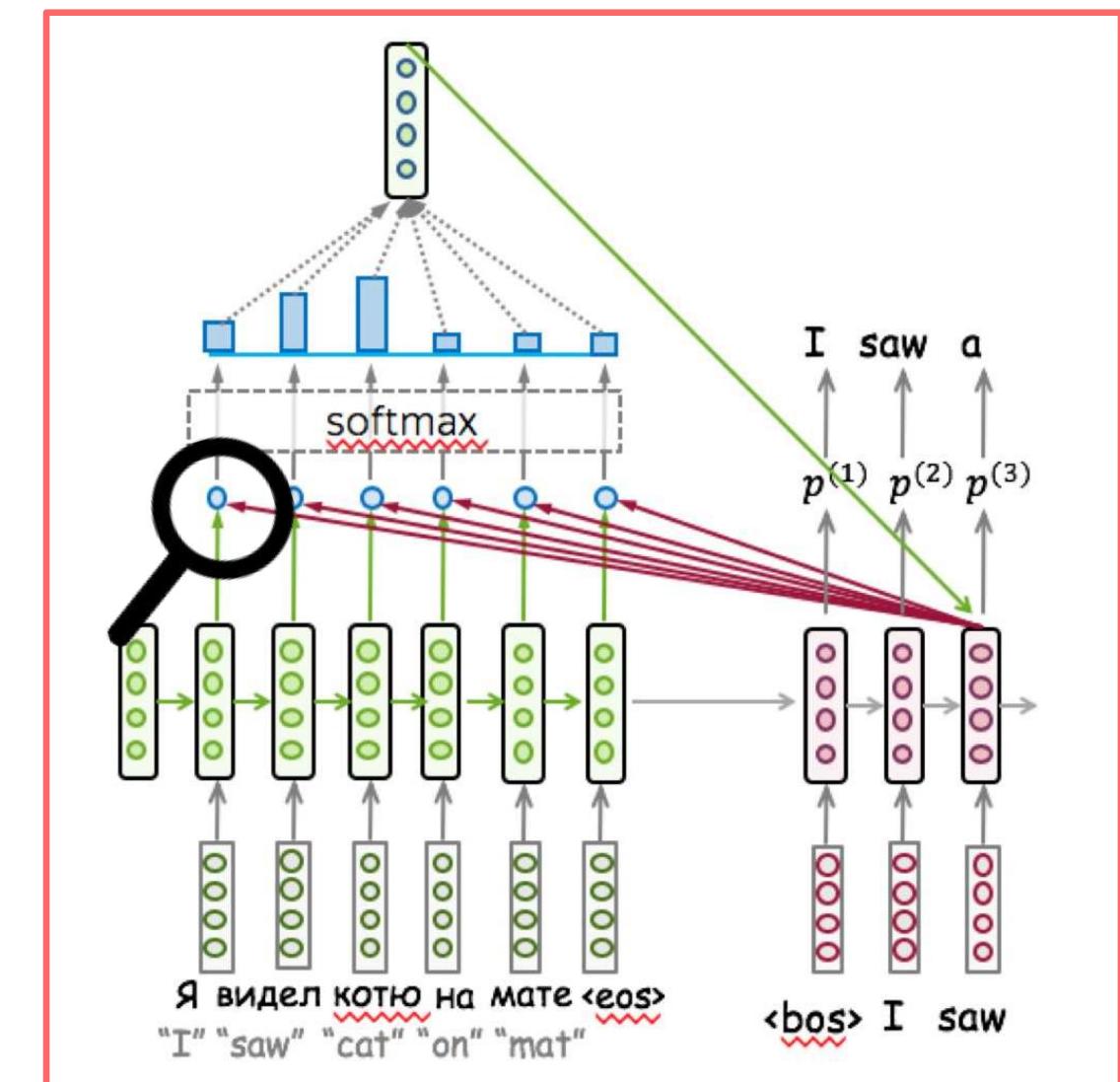


# Fonctions du score d'attention

- Dot-product:  $\text{score}(h_t, s_k) = h_t^T s_k$

$$h_t^T \times \begin{array}{c} s_k \\ \times \end{array}$$

The diagram shows a horizontal vector  $h_t^T$  with four pink circles, followed by a multiplication symbol  $\times$ , and then a vertical vector  $s_k$  with four green circles. This visualizes the dot product operation.



# Fonctions du score d'attention

- Dot-product:  $\text{score}(h_t, s_k) = h_t^T s_k$

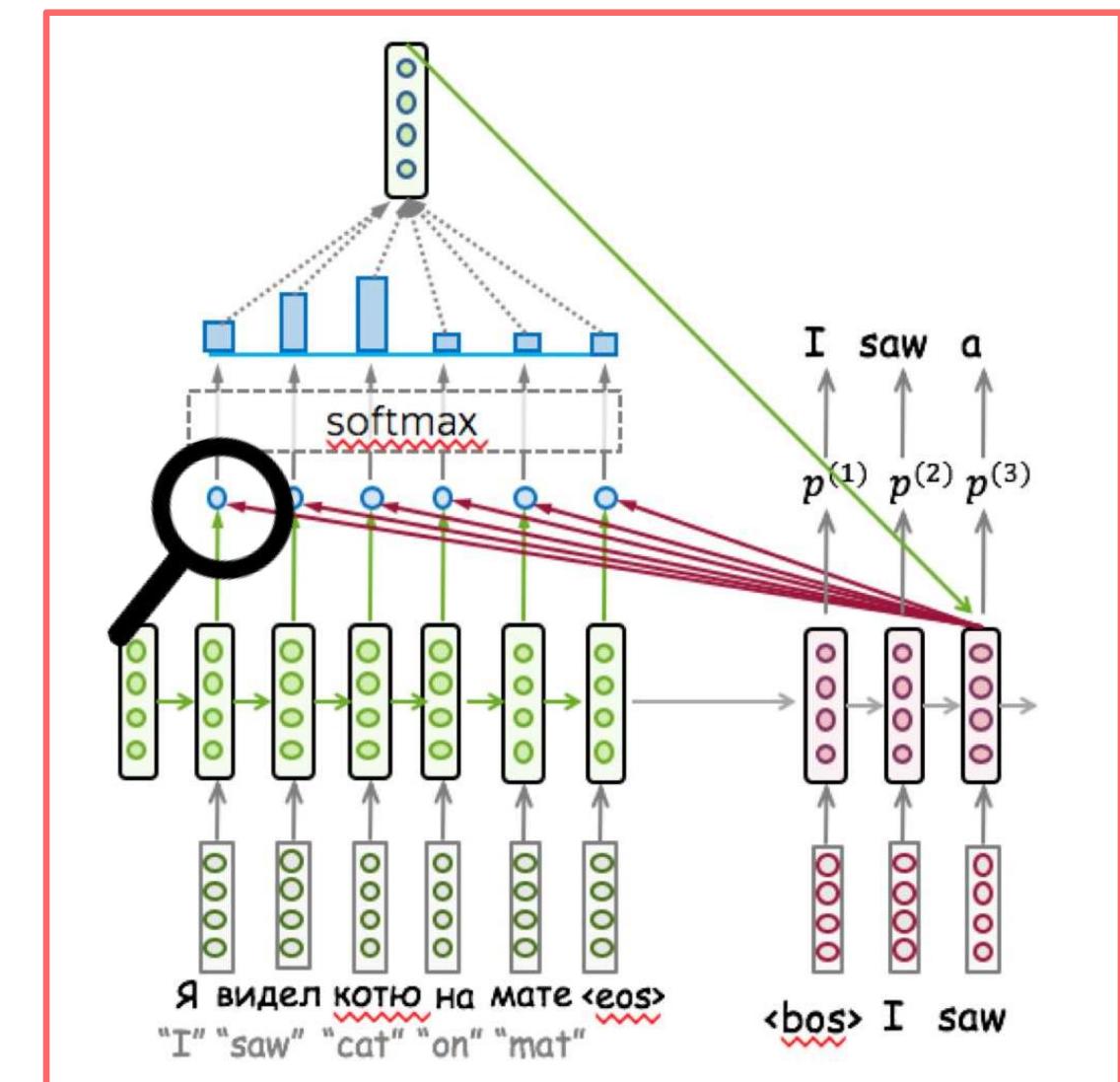
$$h_t^T \times \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} s_k$$

A diagram showing a horizontal vector  $h_t^T$  (represented by four pink circles) multiplied by a vertical vector  $s_k$  (represented by four green circles).

- Bilinear:  $\text{score}(h_t, s_k) = h_t^T w s_k$

$$h_t^T \times \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} W \times \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} s_k$$

A diagram showing a horizontal vector  $h_t^T$  (four pink circles) multiplied by a weight matrix  $W$  (represented by a light blue rectangle), which is then multiplied by a vertical vector  $s_k$  (four green circles).



# Fonctions du score d'attention

- Dot-product:  $\text{score}(h_t, s_k) = h_t^T s_k$

$$h_t^T \times \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} s_k$$

A diagram showing a horizontal vector  $h_t^T$  (represented by four pink circles) multiplied by a vertical vector  $s_k$  (represented by four green circles).

- Bilinear:  $\text{score}(h_t, s_k) = h_t^T w s_k$

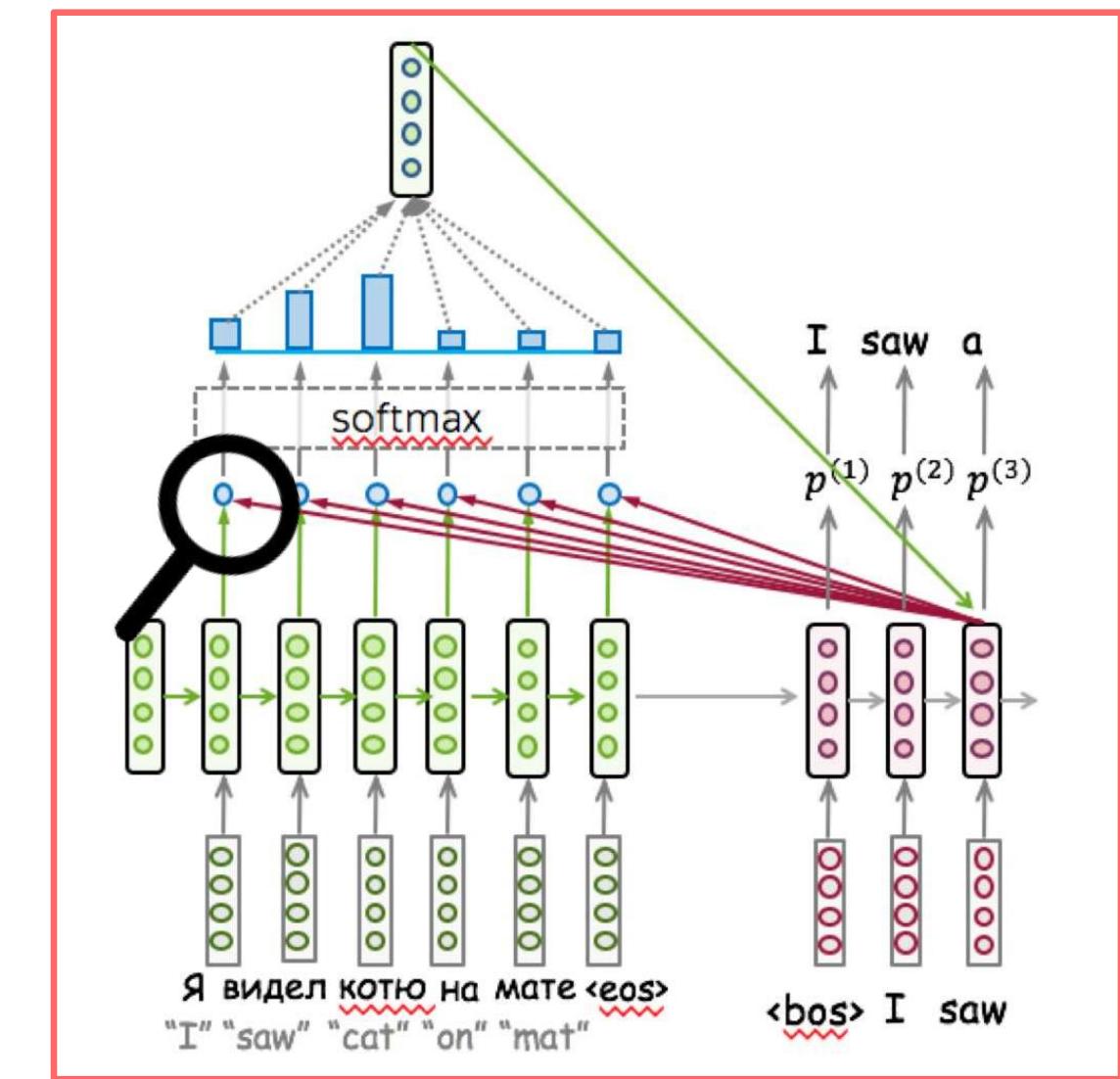
$$h_t^T \times \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} W \times \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} s_k$$

A diagram showing a horizontal vector  $h_t^T$  (four pink circles) multiplied by a weight matrix  $W$  (represented by a light blue rectangle), which is then multiplied by a vertical vector  $s_k$  (four green circles).

- Multi-Layer Perceptron:  $\text{score}(h_t, s_k) = w_2^T \cdot \tanh(w_1[h_t, s_k])$

$$w_2^T \times \tanh \left[ W_1 \times \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} h_t \right] \begin{array}{c} \text{---} \\ | \\ \text{---} \end{array} s_k$$

A diagram showing a horizontal vector  $w_2^T$  (four blue circles) multiplied by the tanh function of the product of a weight matrix  $W_1$  (light blue rectangle) and a vector  $[h_t, s_k]$ .



## A retenir

On peut utiliser des RNNs pour modéliser des séquences

L'architecture de référence est l'encodeur décodeur

Le mécanisme d'attention permet d'échanger des informations entre tokens

Le mécanisme d'attention permet d'améliorer ses performances en choisissant selectivement les entrées

Il existe plusieurs manières de définir l'attention

Prochain  
cours  
  
Découvrir le  
transformeur



# Merci !

Mohamed Abbas KONATE



[mohamed-abbas.konate@michelin.com](mailto:mohamed-abbas.konate@michelin.com)

