Modelos de Linguagem Neurais

Com Transformers

“The true art of memory is the art of attention ” Samuel Johnson, Idler #74, September 1759

Profa Aline Paes
alinepaes@ic.uff.br

Adaptado de
https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

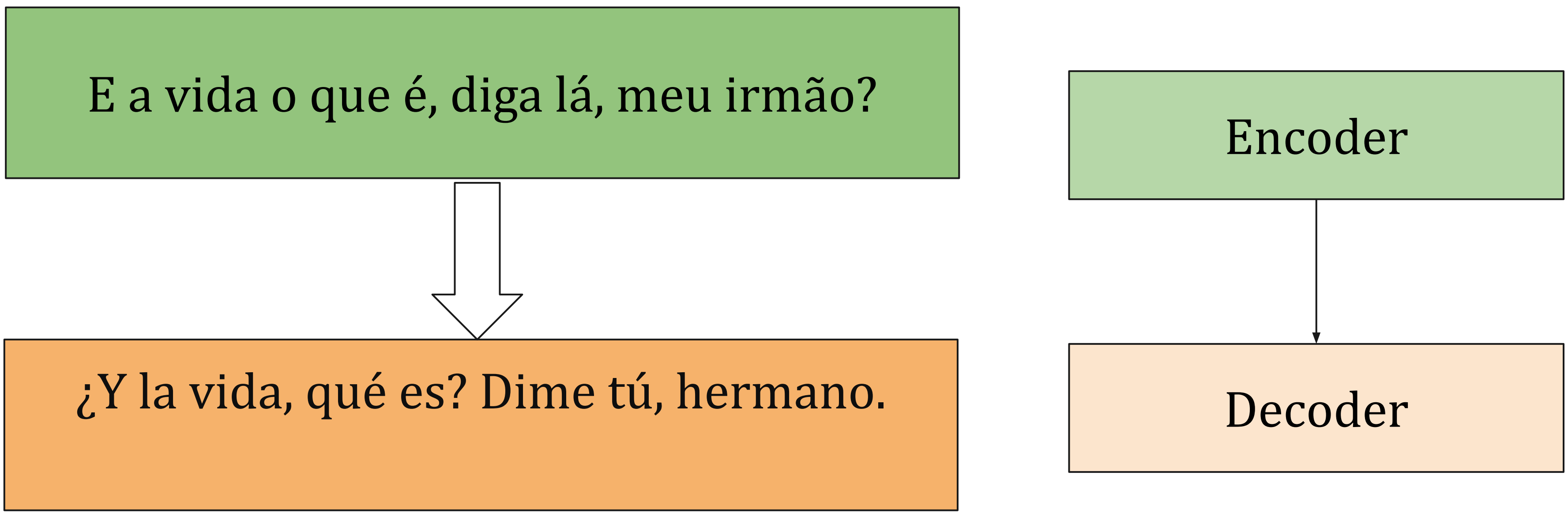
Tradução

E a vida o que é, diga lá, meu irmão?

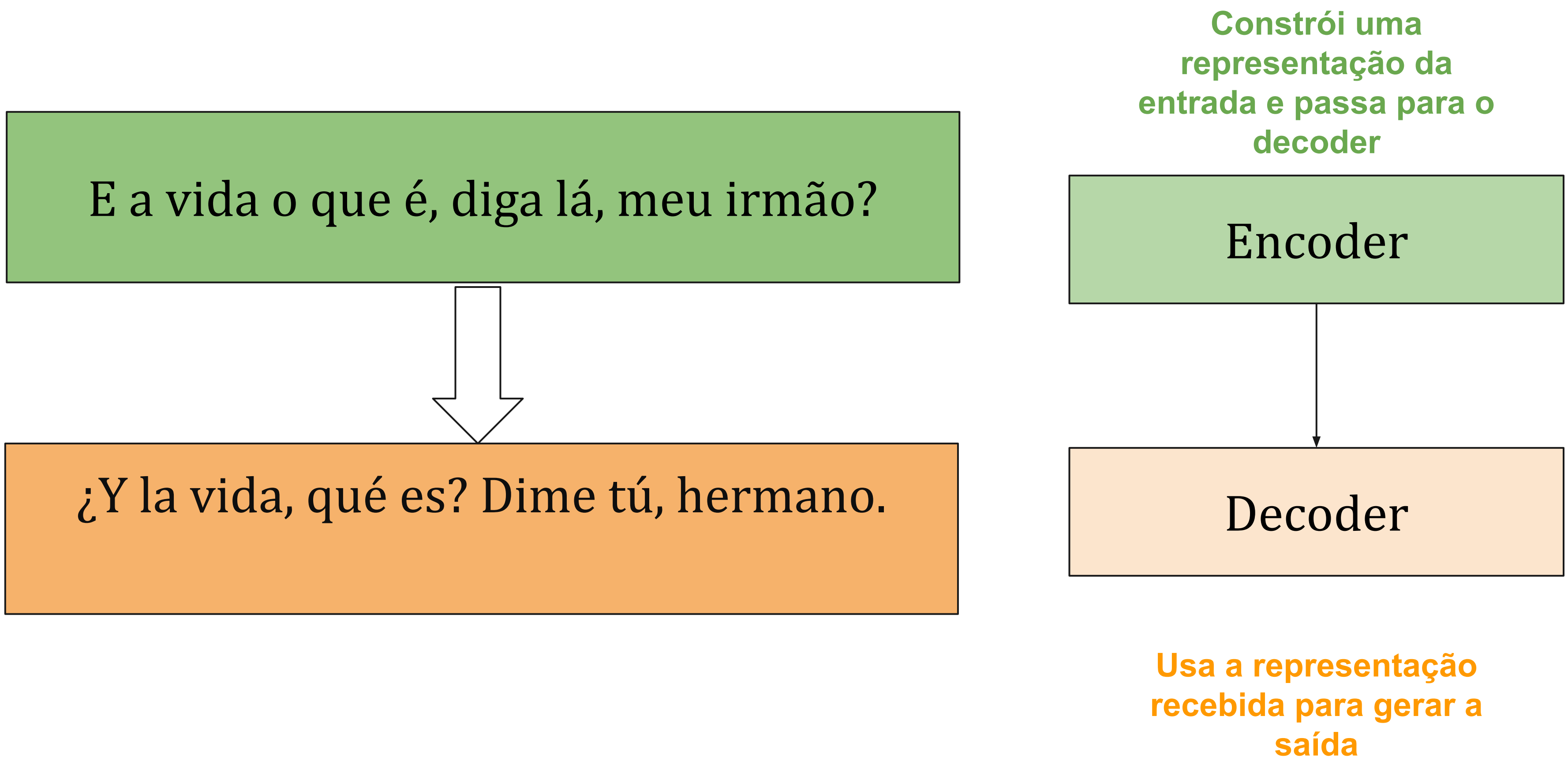
↓

¿Y la vida, qué es? Dime tú, hermano.

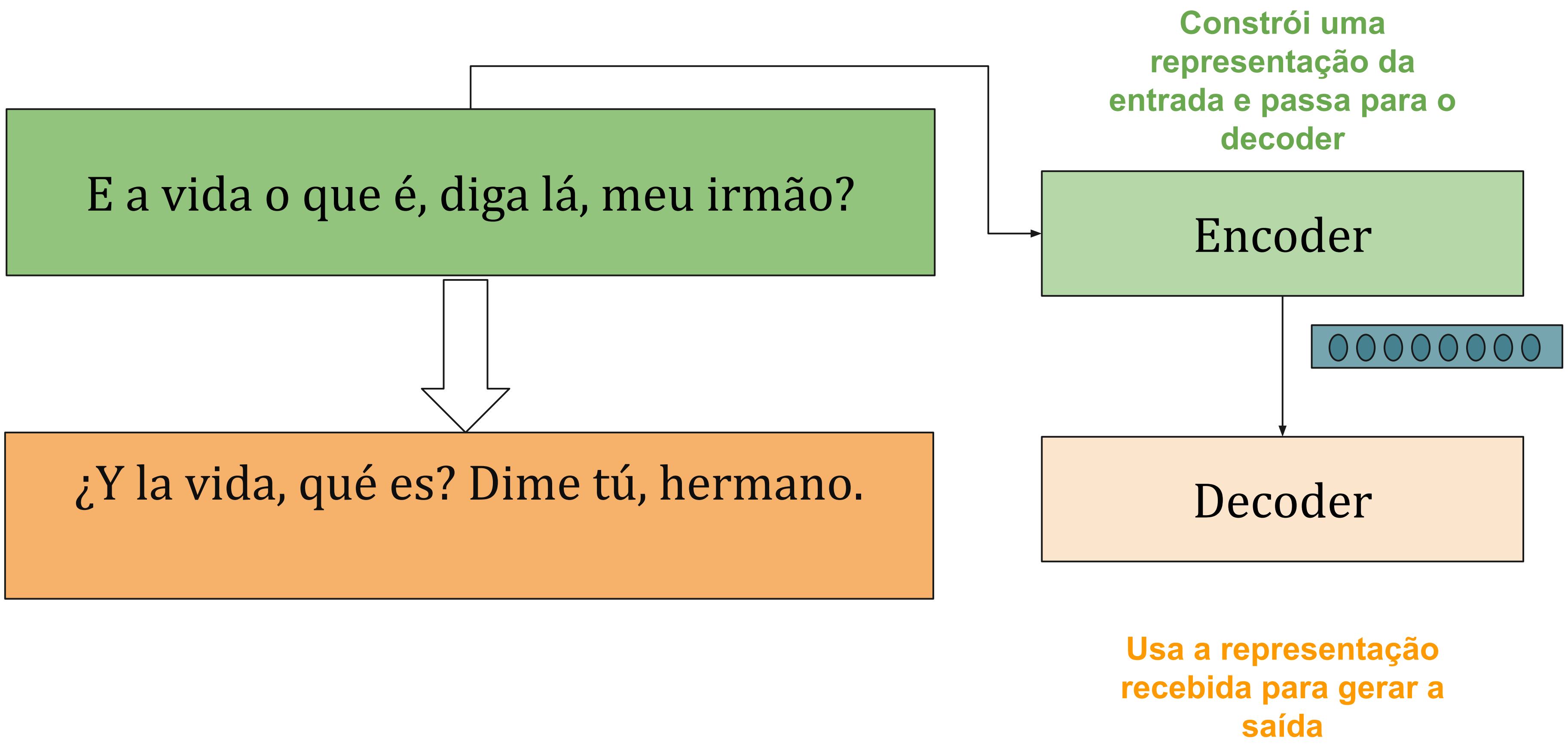
Encoder-Decoder



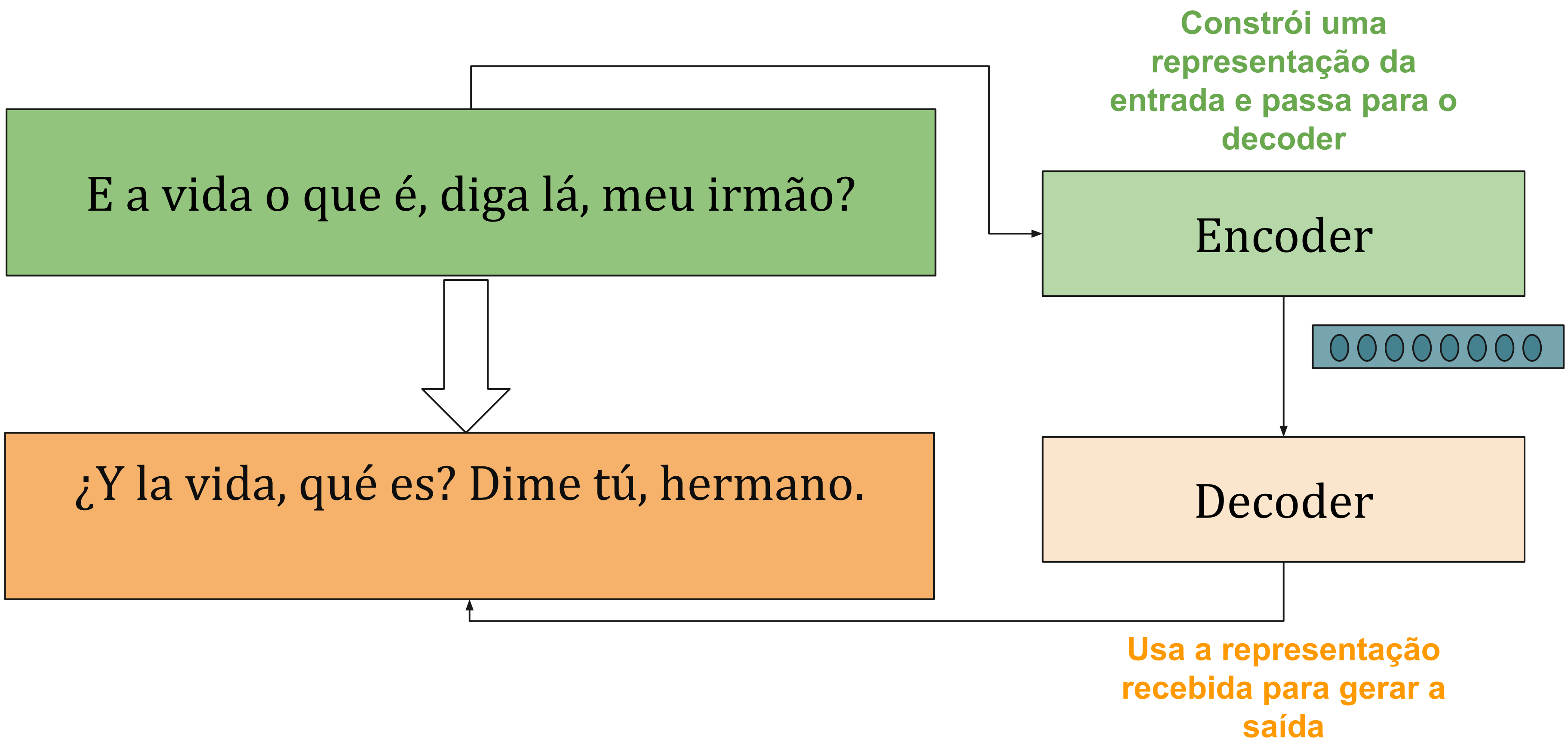
Encoder-Decoder



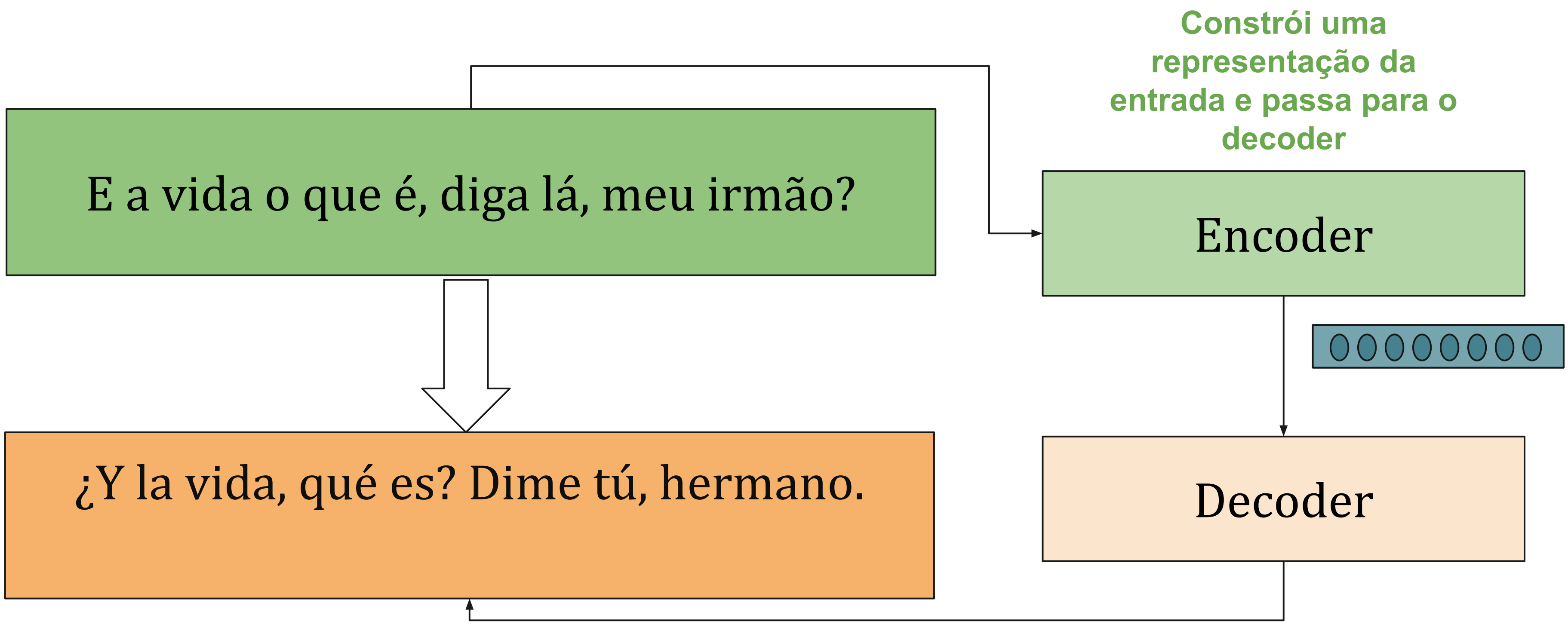
Encoder-Decoder



Encoder-Decoder



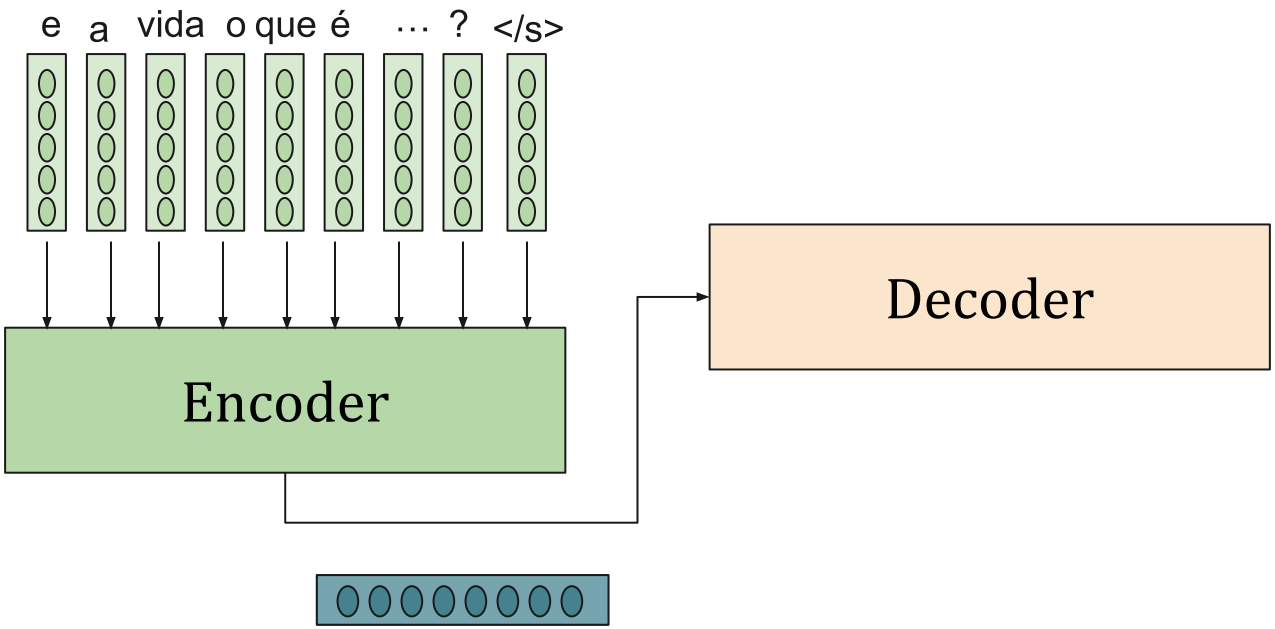
Encoder-Decoder



Usa a representação recebida para gerar a saída

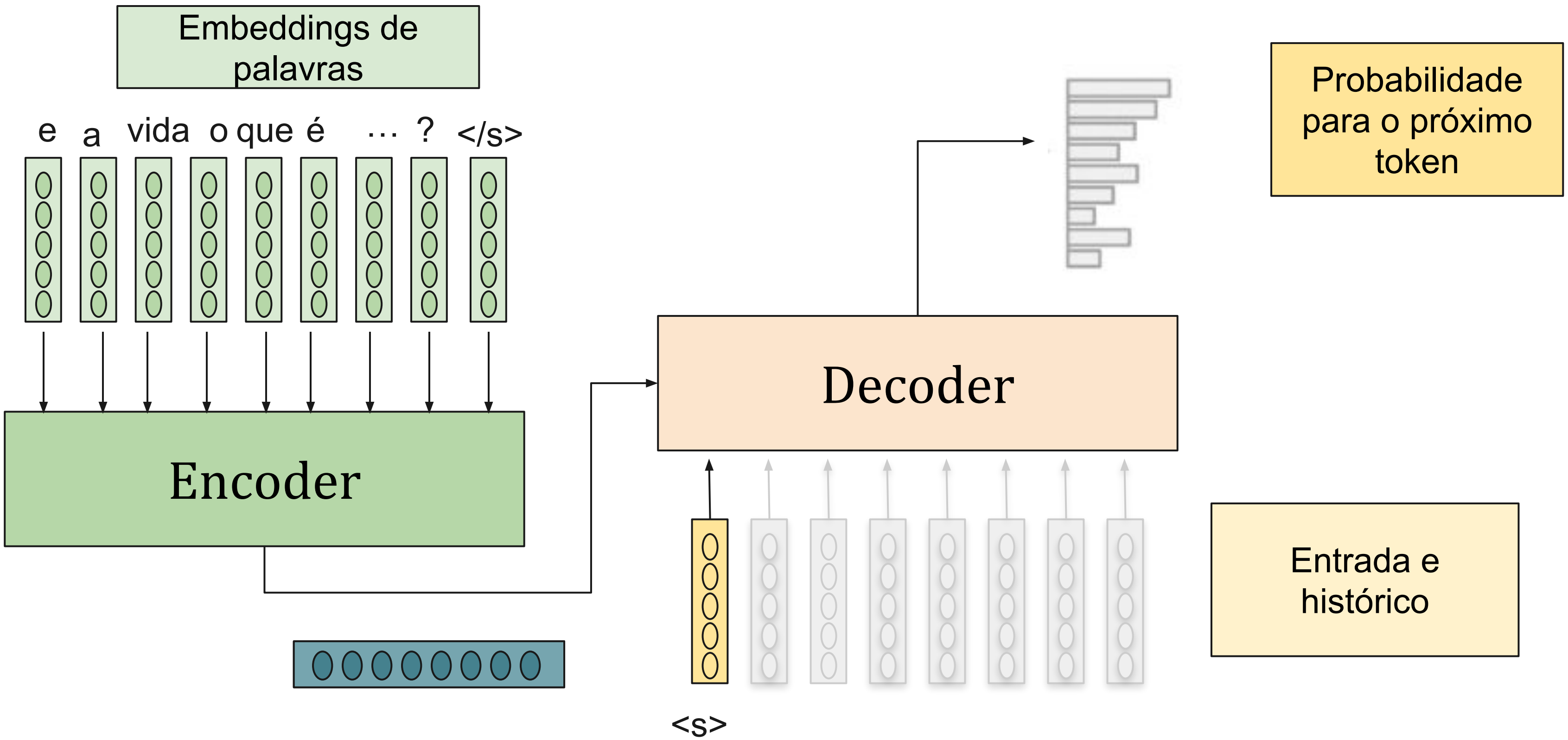
$$P(y_1, y_2, \ldots, y_n | X) = \prod_{t=1}^{n} p(y_t, y_{<t}, X)$$

Encoder-Decoder



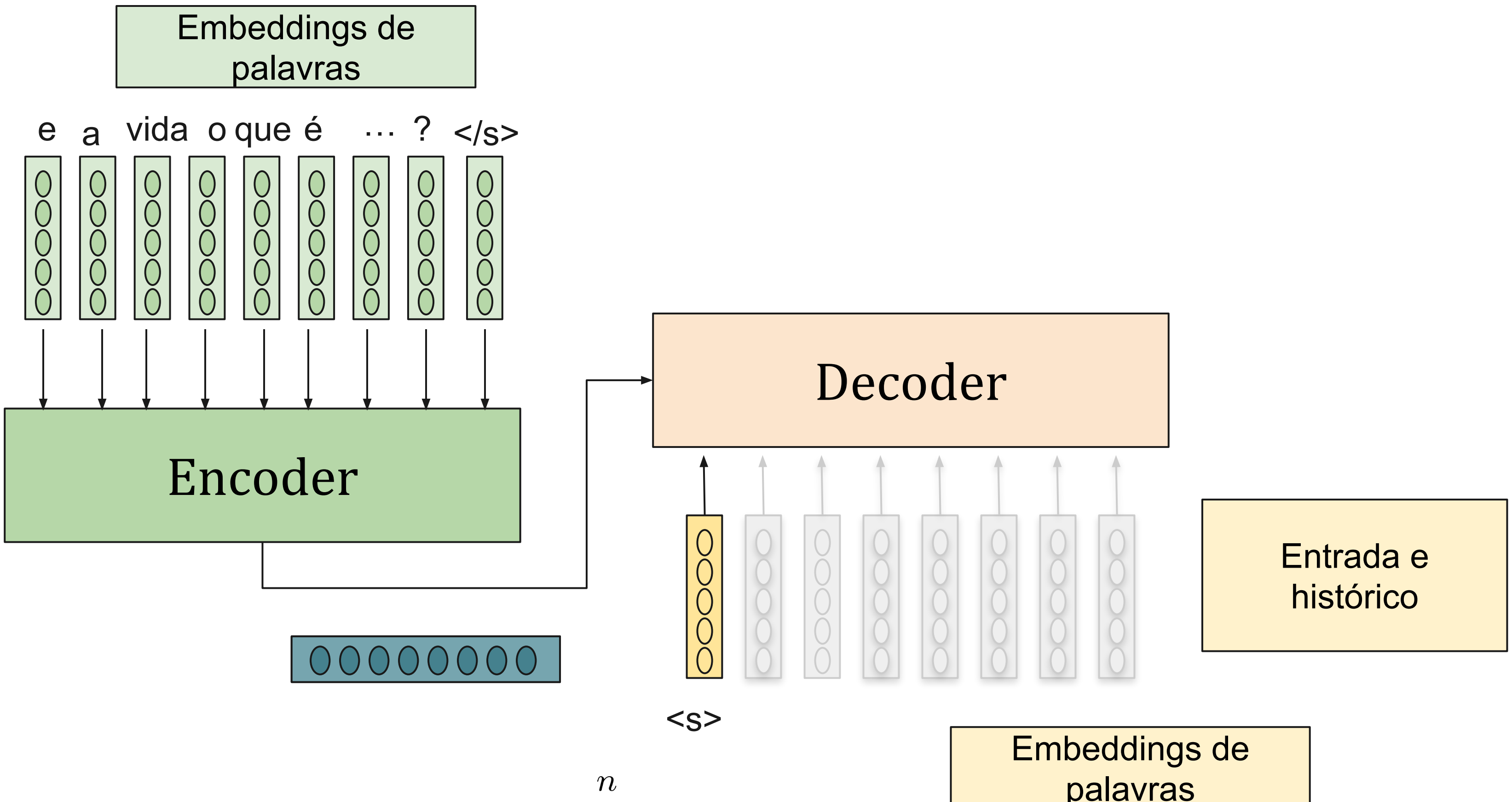
$$P(y_1, y_2, \ldots, y_n | X) = \prod_{t=1}^{n} p(y_t, y_{<t}, X)$$

Encoder-Decoder



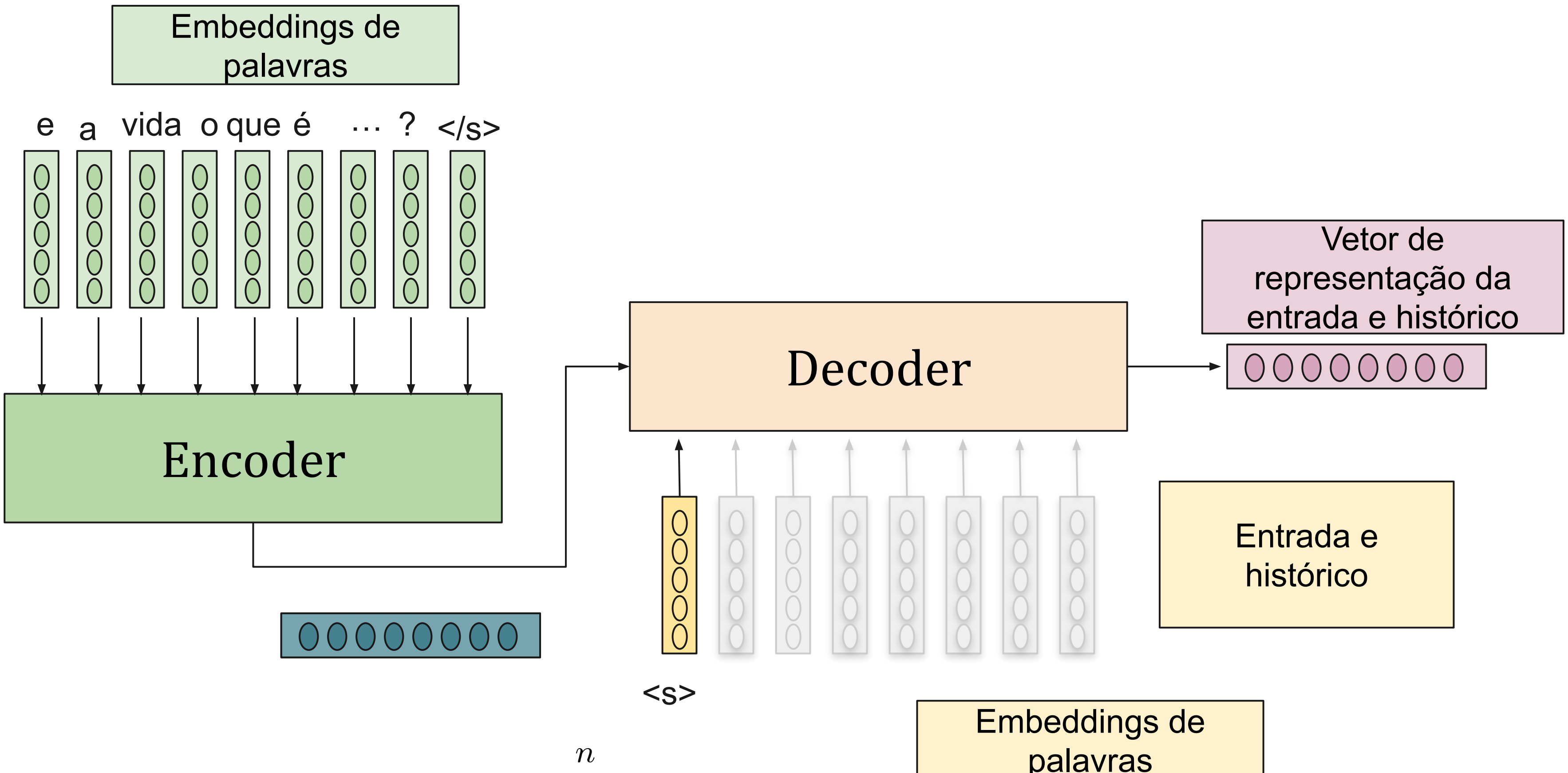
$$P(y_1, y_2, \ldots, y_n | X) = \prod_{t=1}^{n} p(y_t, y_{<t}, X)$$

Encoder-Decoder



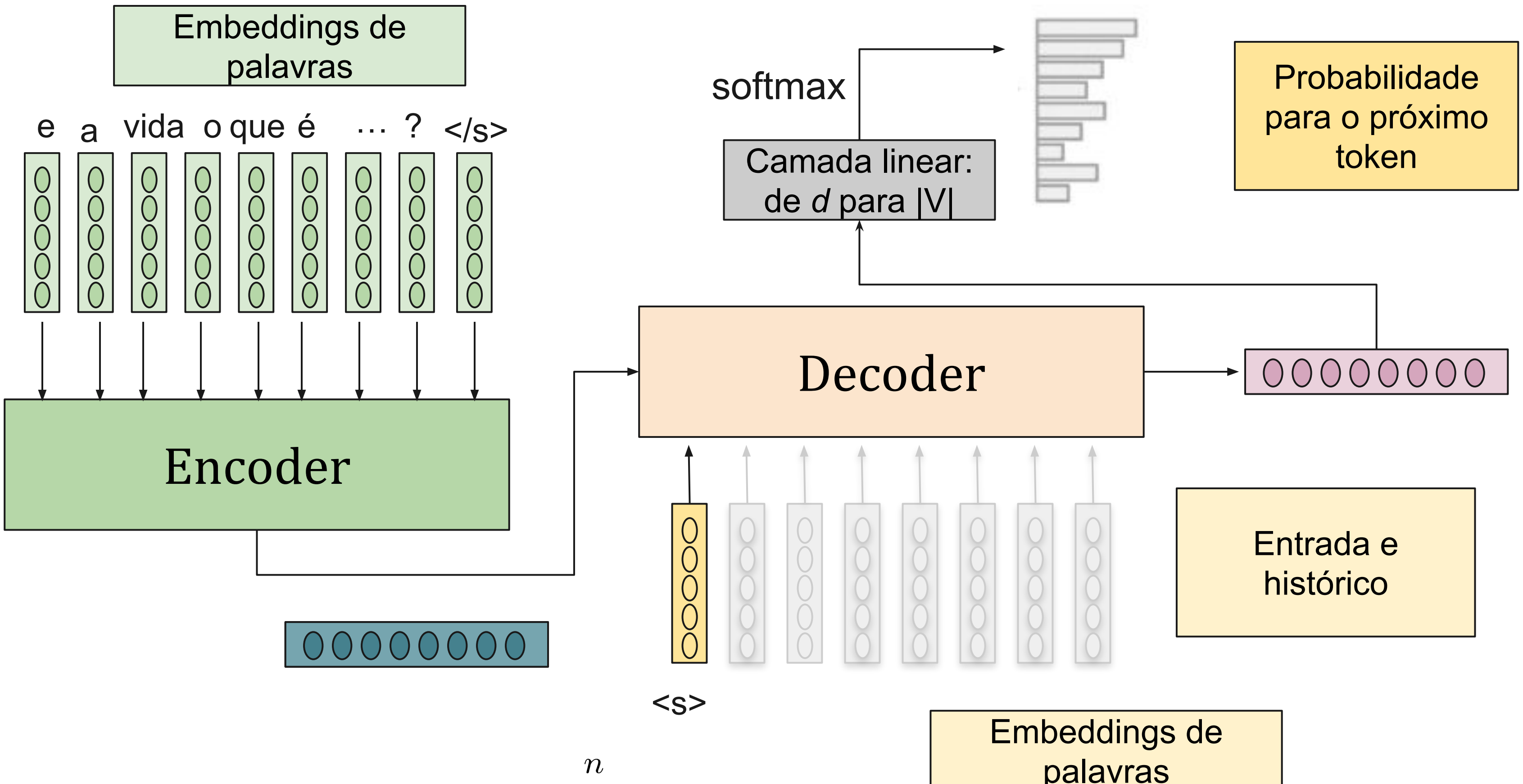
$$P(y_1, y_2, \ldots, y_n | X) = \prod_{t=1}^{n} p(y_t, y_{<t}, X)$$

Encoder-Decoder



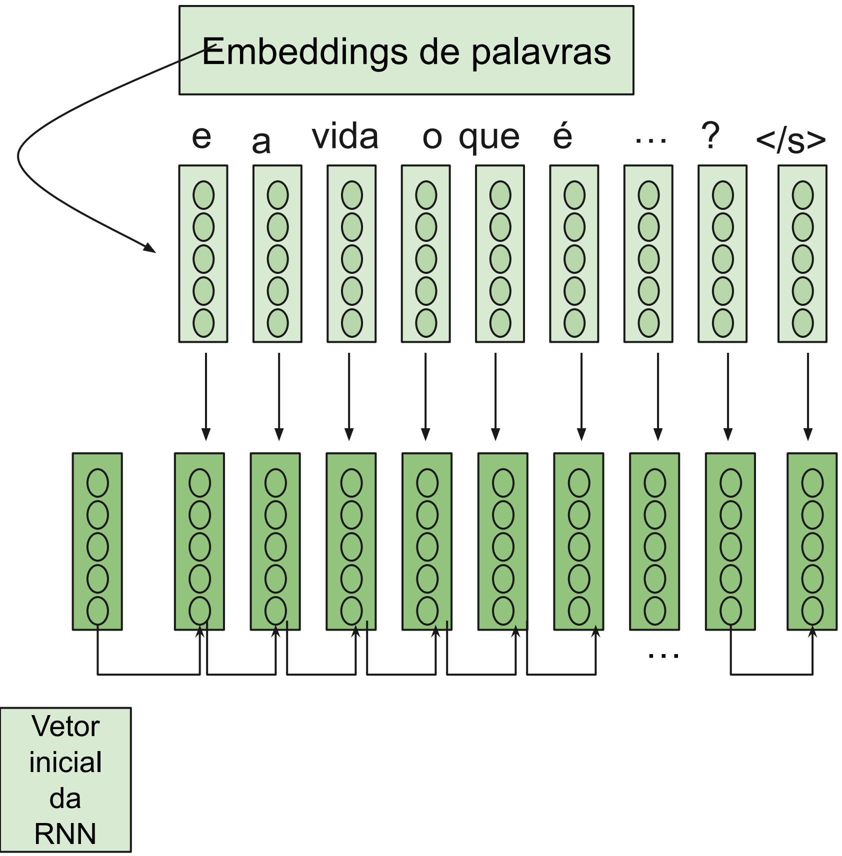
$$P(y_1, y_2, \ldots, y_n | X) = \prod_{t=1}^{n} p(y_t, y_{<t}, X)$$

Encoder-Decoder

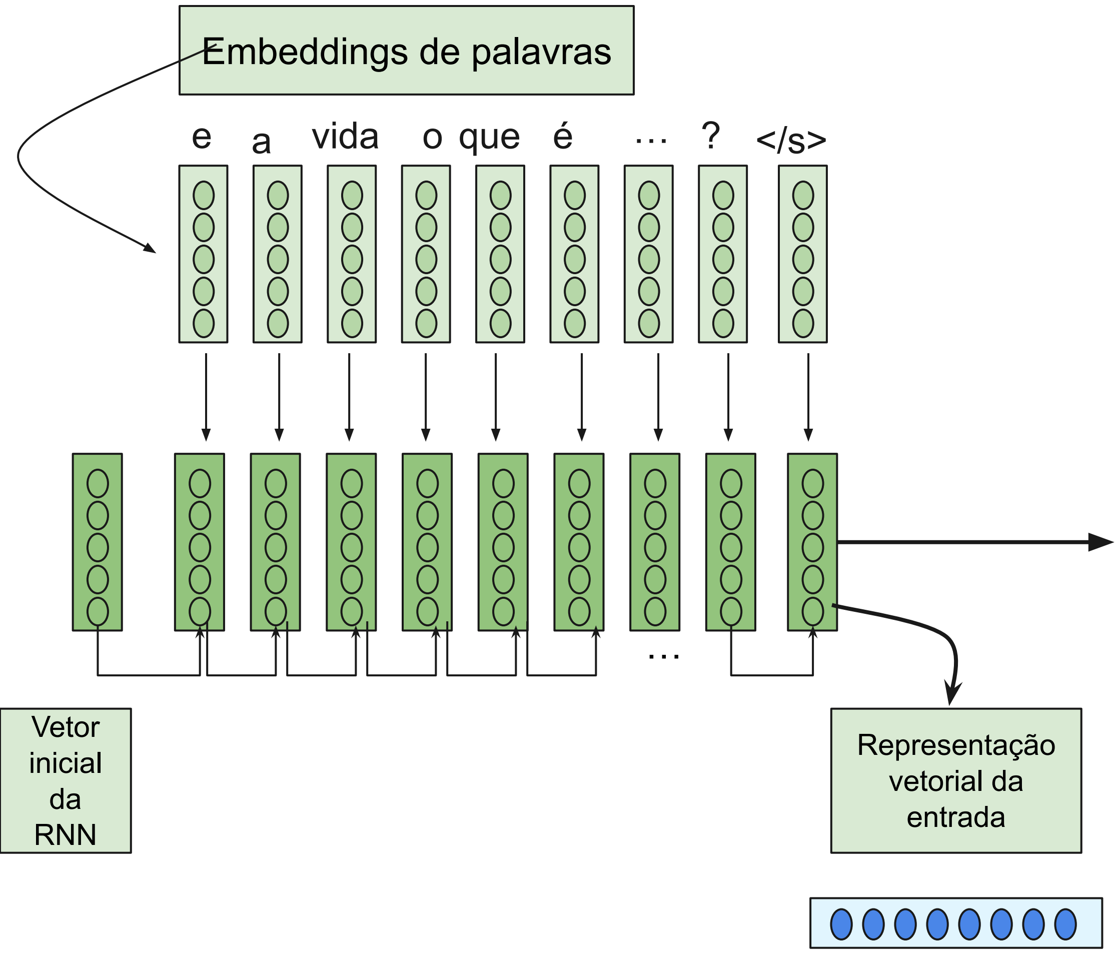


$$P(y_1, y_2, \ldots, y_n | X) = \prod_{t=1}^{n} p(y_t, y_{<t}, X)$$

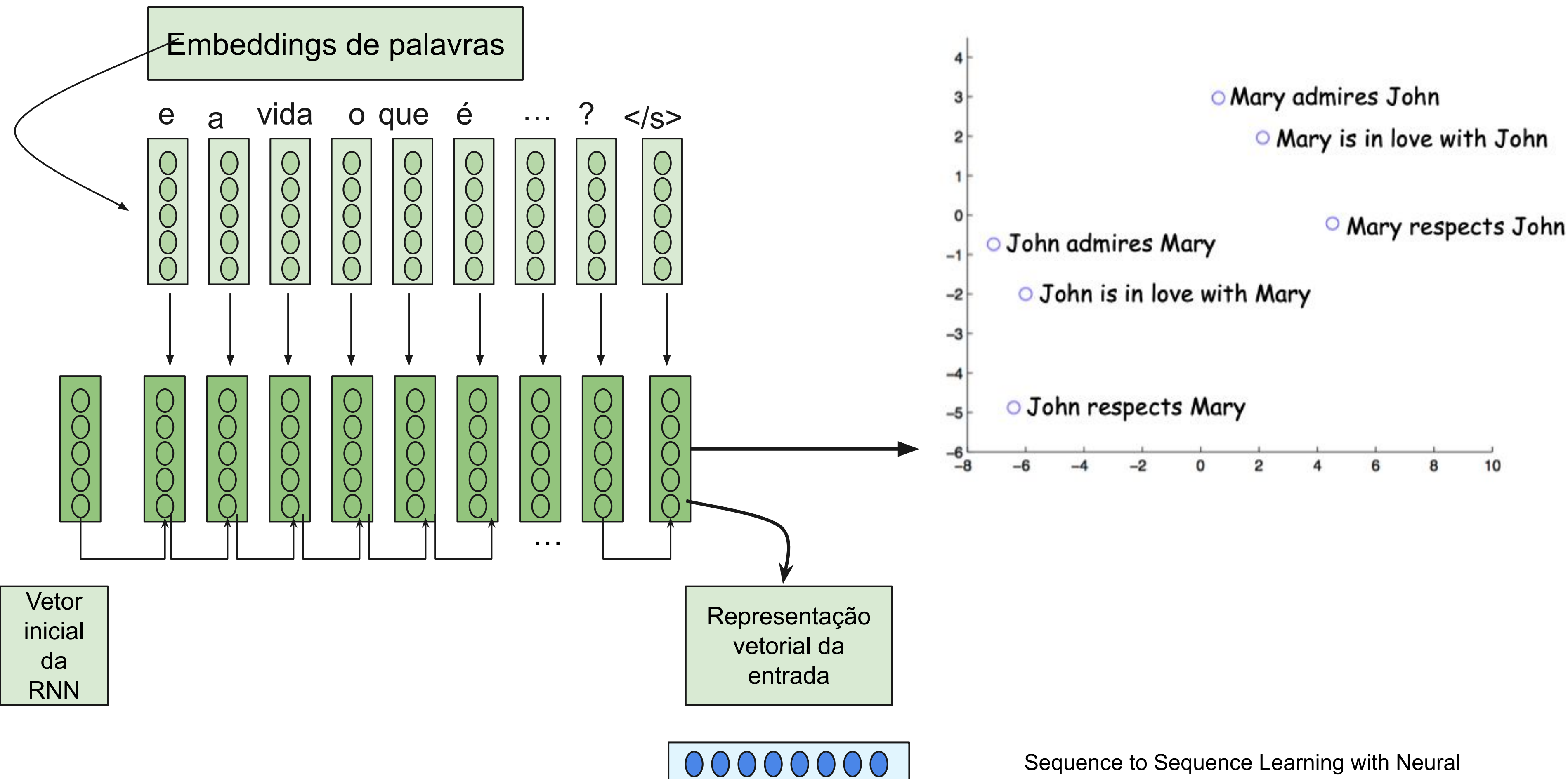
Encoder-Decoder com RNN



Encoder-Decoder com RNN

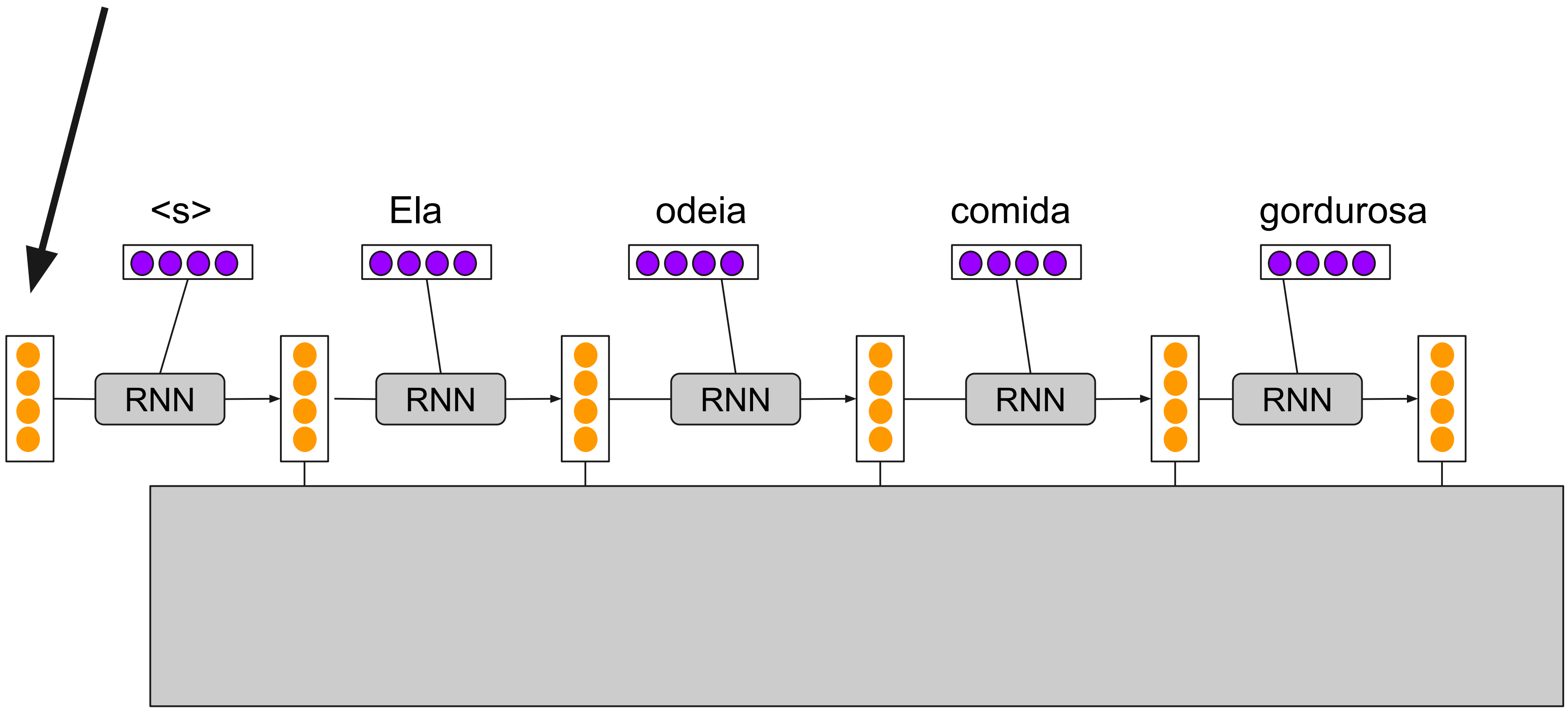


Encoder-Decoder com RNN



Sequence to Sequence Learning with Neural Networks , 2014.

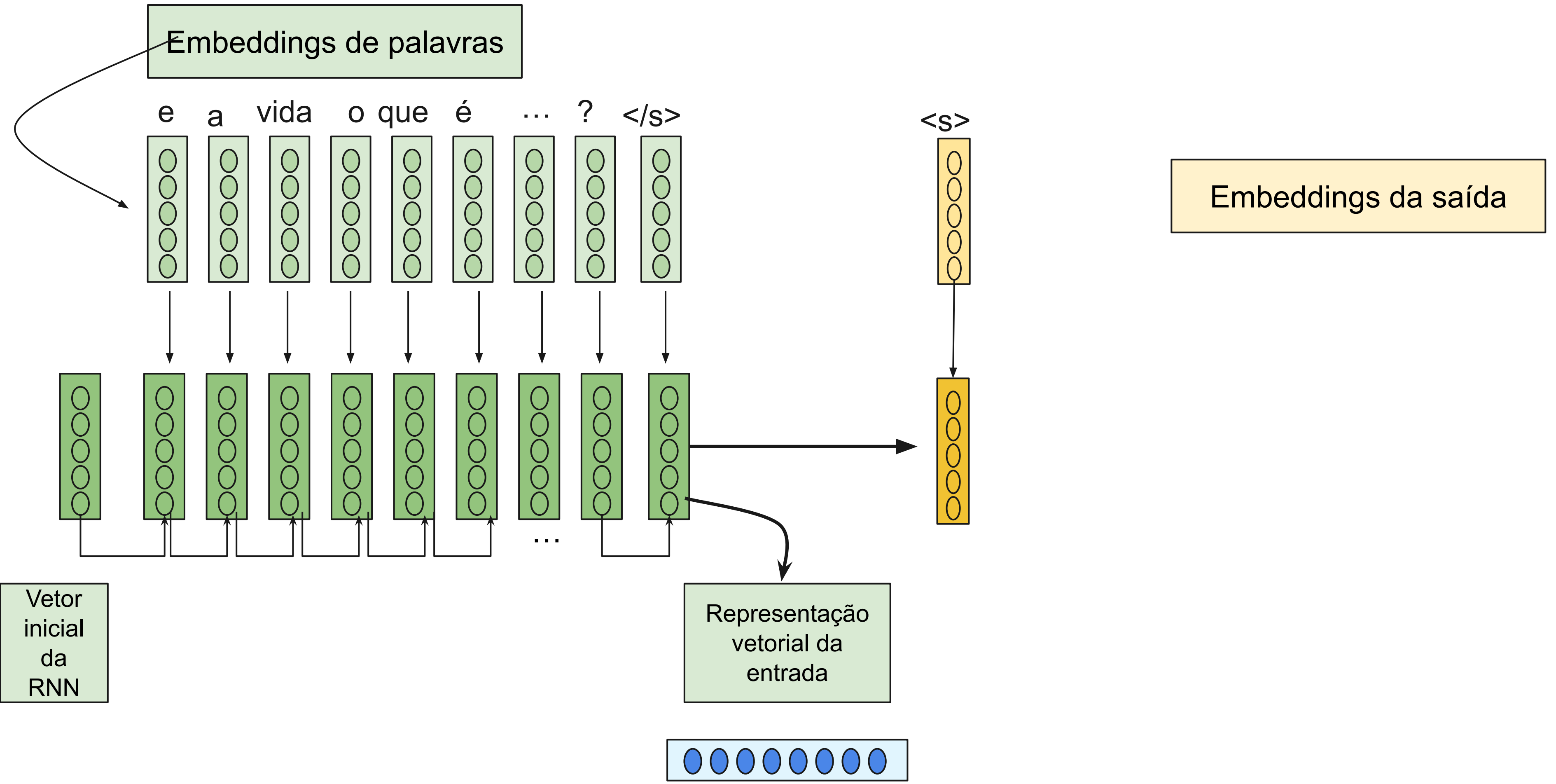
Inferência - geração de texto



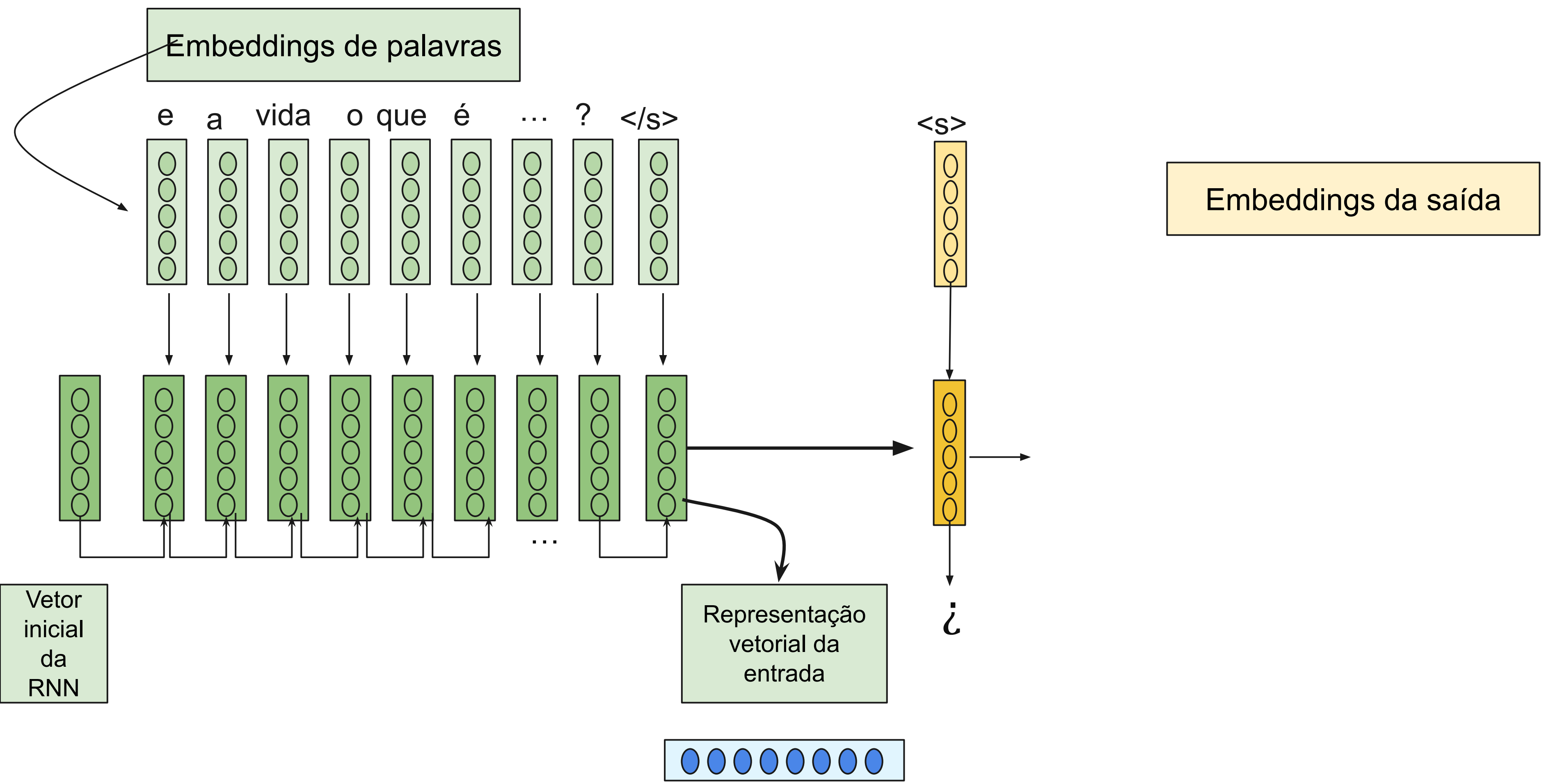
Geração autorregressiva

*CMU Neural nets for NLP

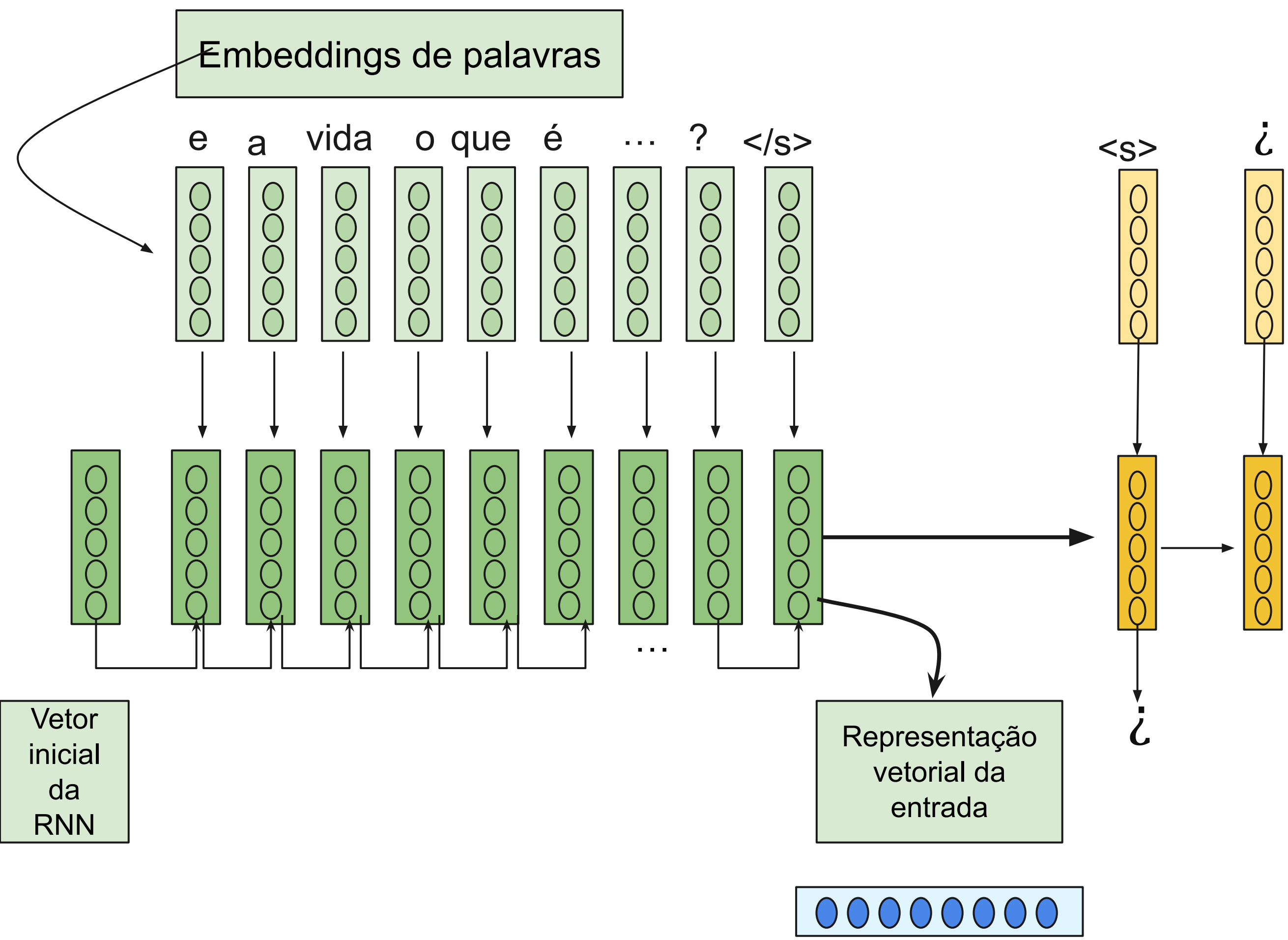
Encoder-Decoder com RNN



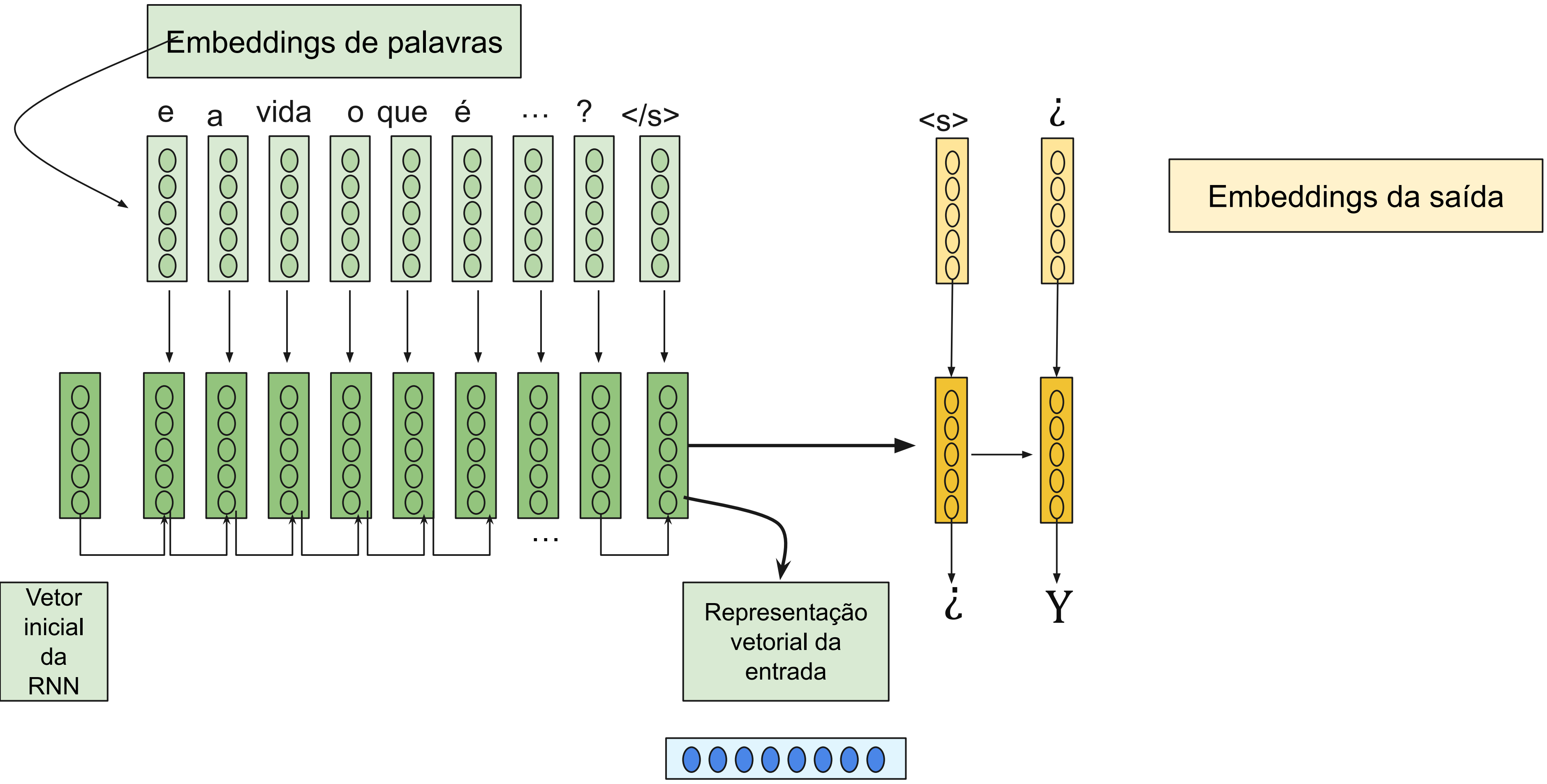
Encoder-Decoder com RNN



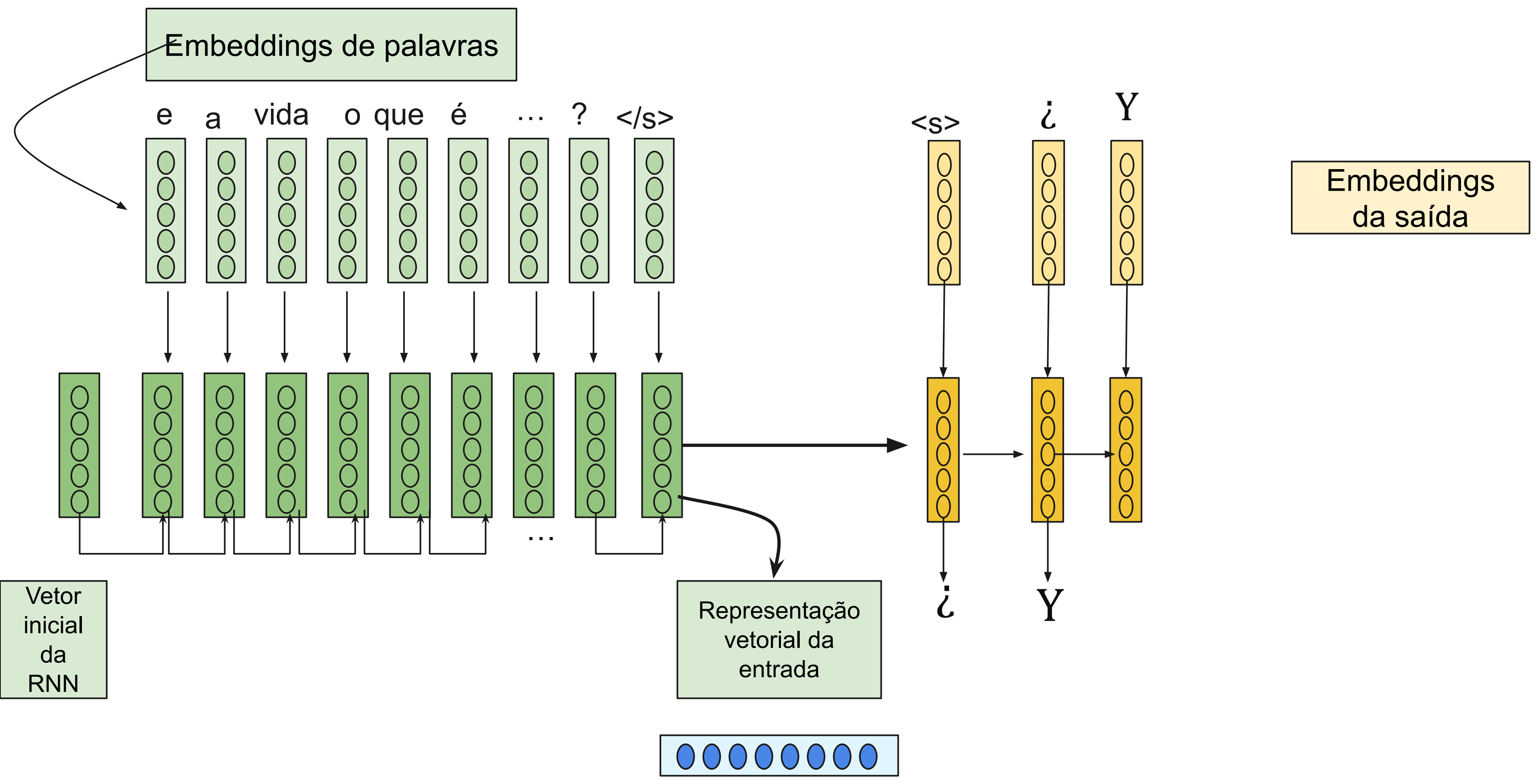
Encoder-Decoder com RNN



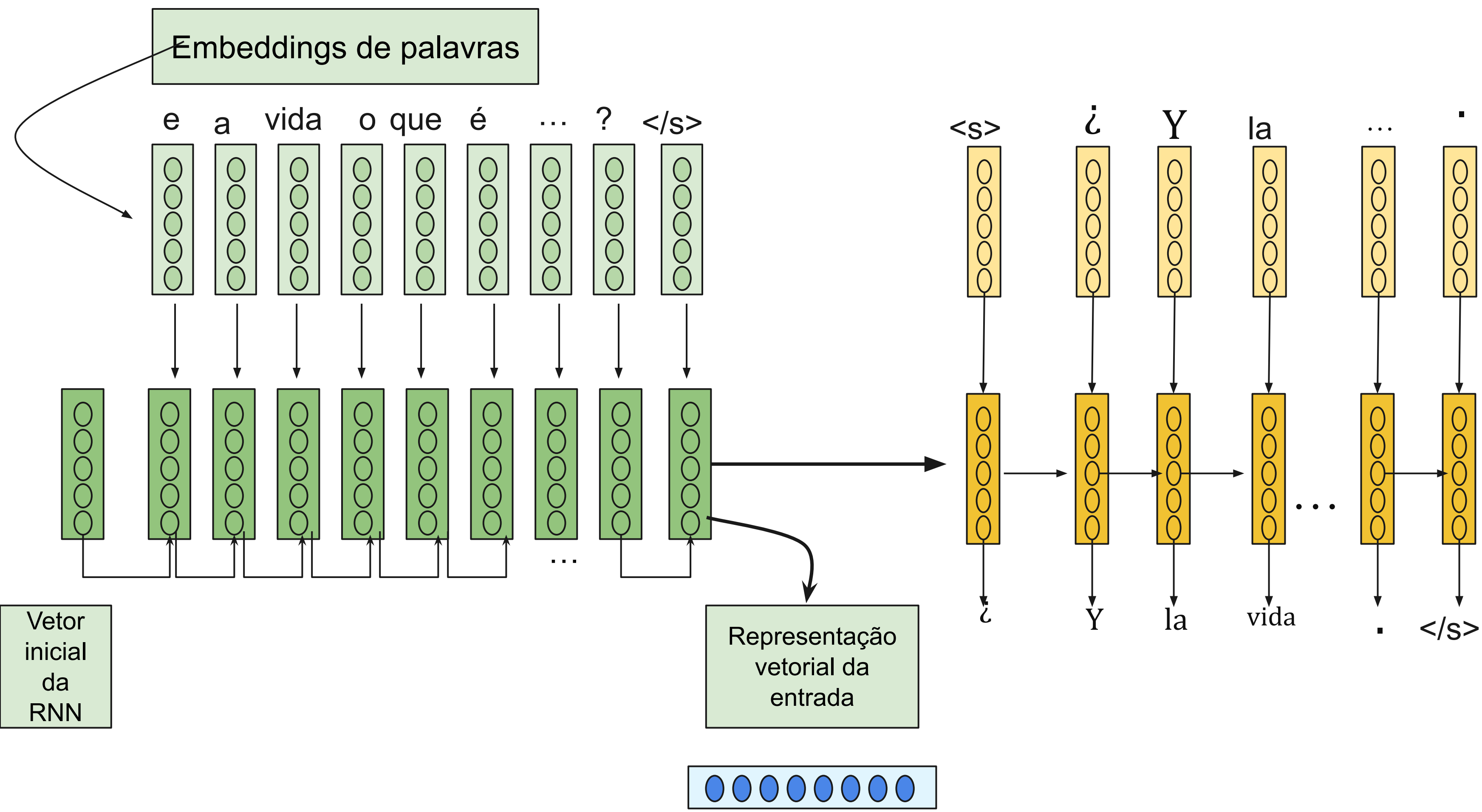
Encoder-Decoder com RNN



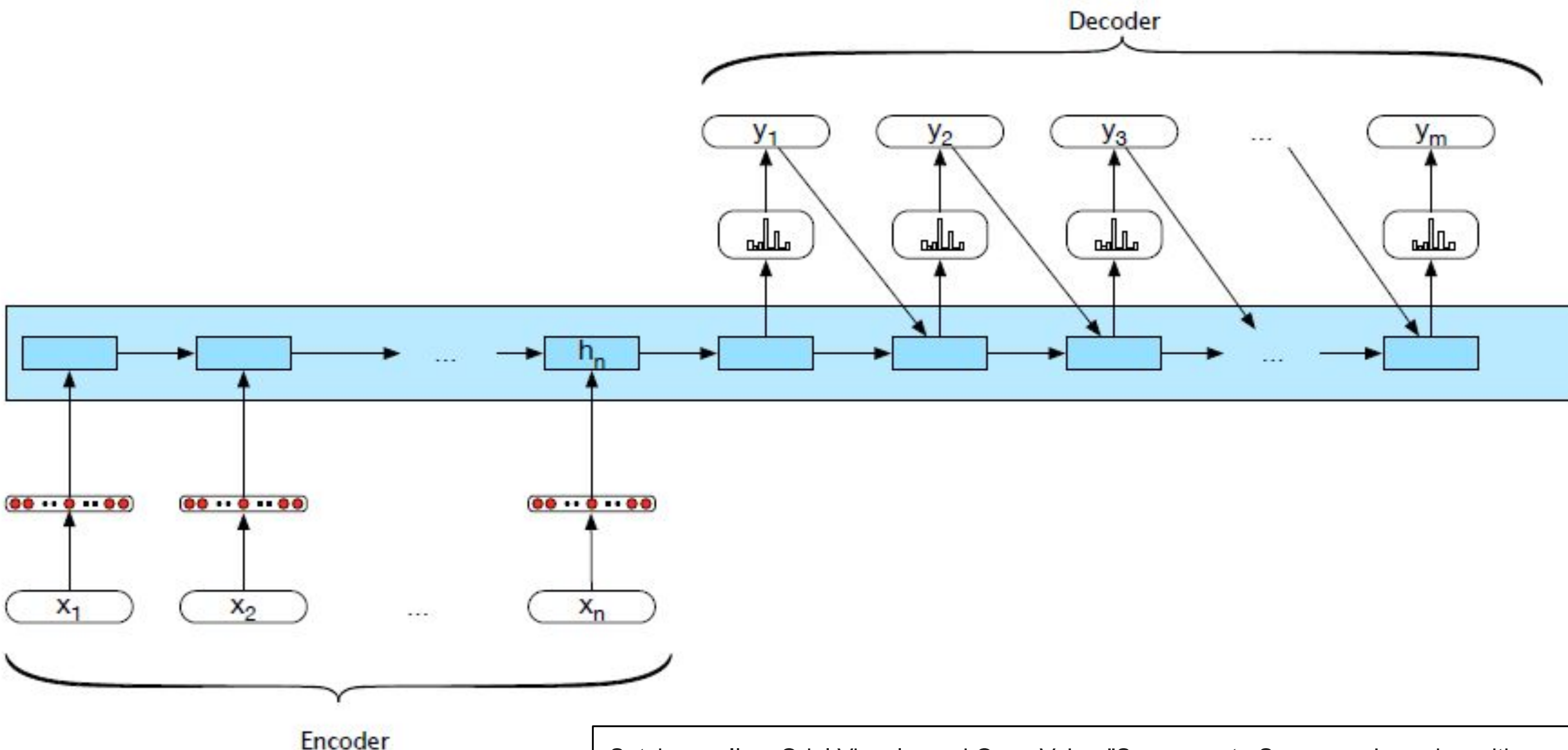
Encoder-Decoder com RNN



Encoder-Decoder com RNN



Encoder-decoder básico



Speech & language Processing

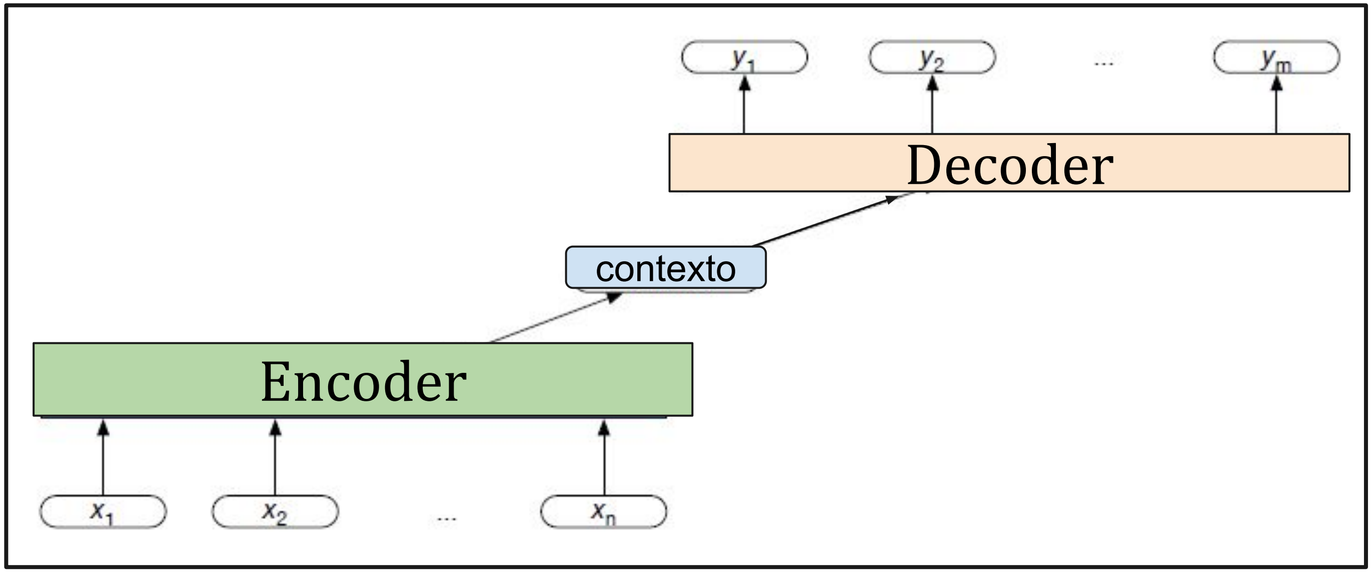
Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks." *Advances in Neural Information Processing Systems* 27 (2014): 3104-3112.

Pergunta

Como aprender os pesos do encoder?

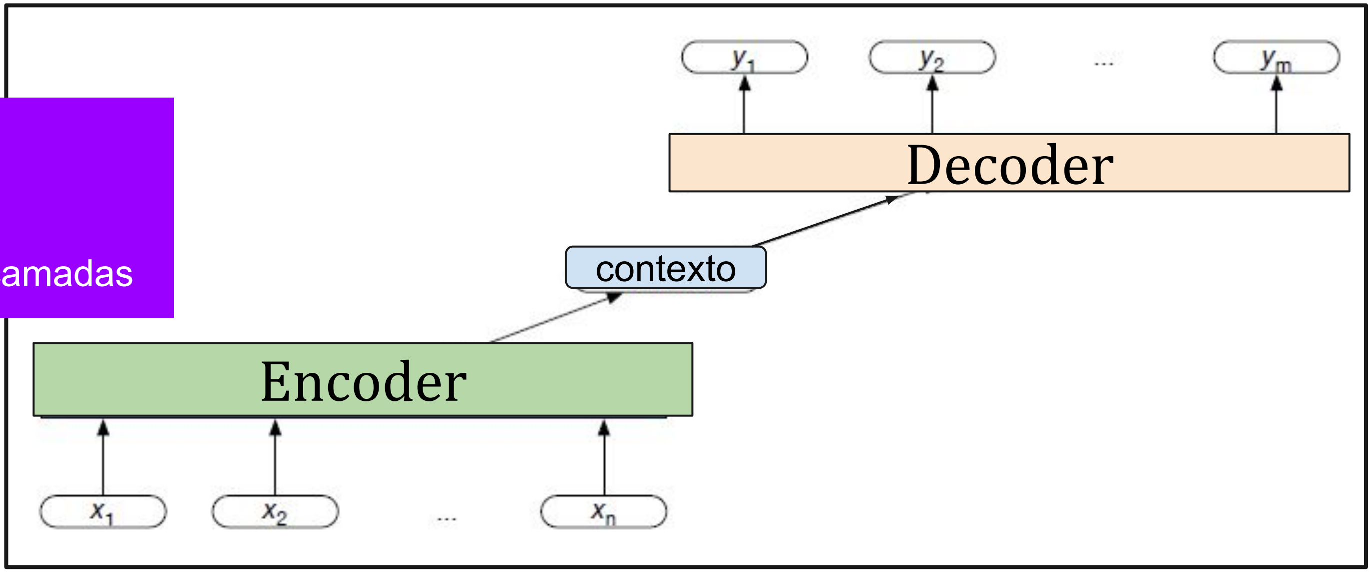
a) Usando uma função de custo para o contexto
b) A partir da função de custo calculada no fim do decoder
c) Usando uma função de custo para cada componente do encoder
d) Descobrindo numericamente o máximo da função

Encoder-decoder abstração



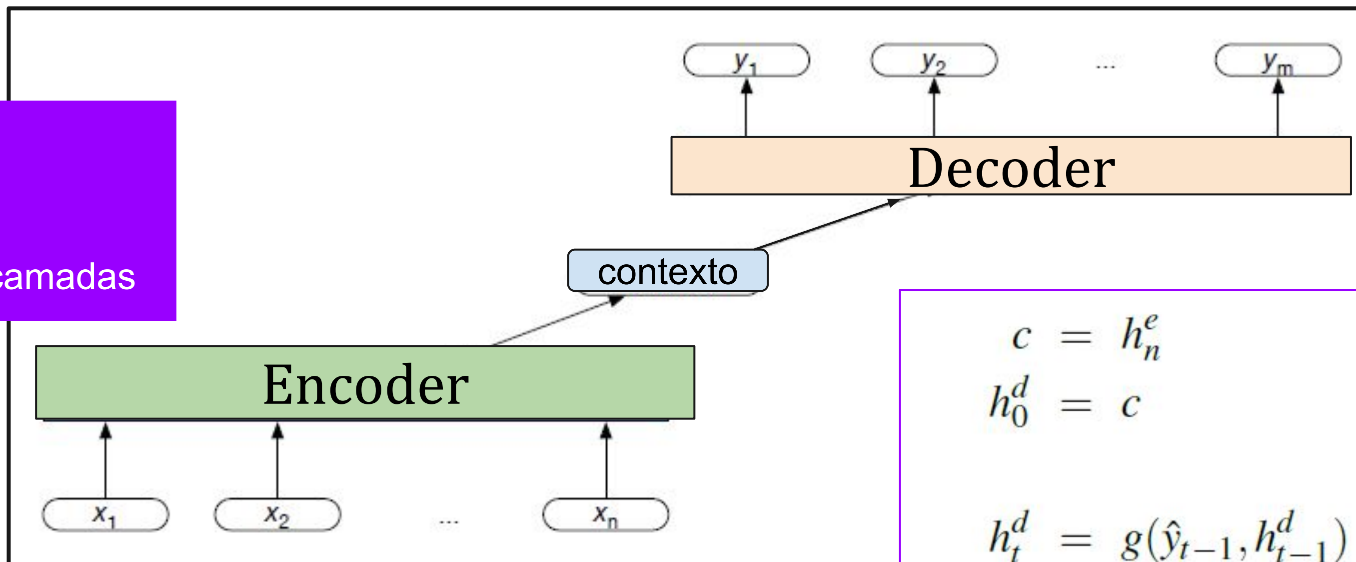
Encoder-decoder abstração

RNN
(Bi)LSTM
GRU
Múltiplas camadas



Encoder-decoder abstração

RNN
(Bi)LSTM
GRU
Múltiplas camadas

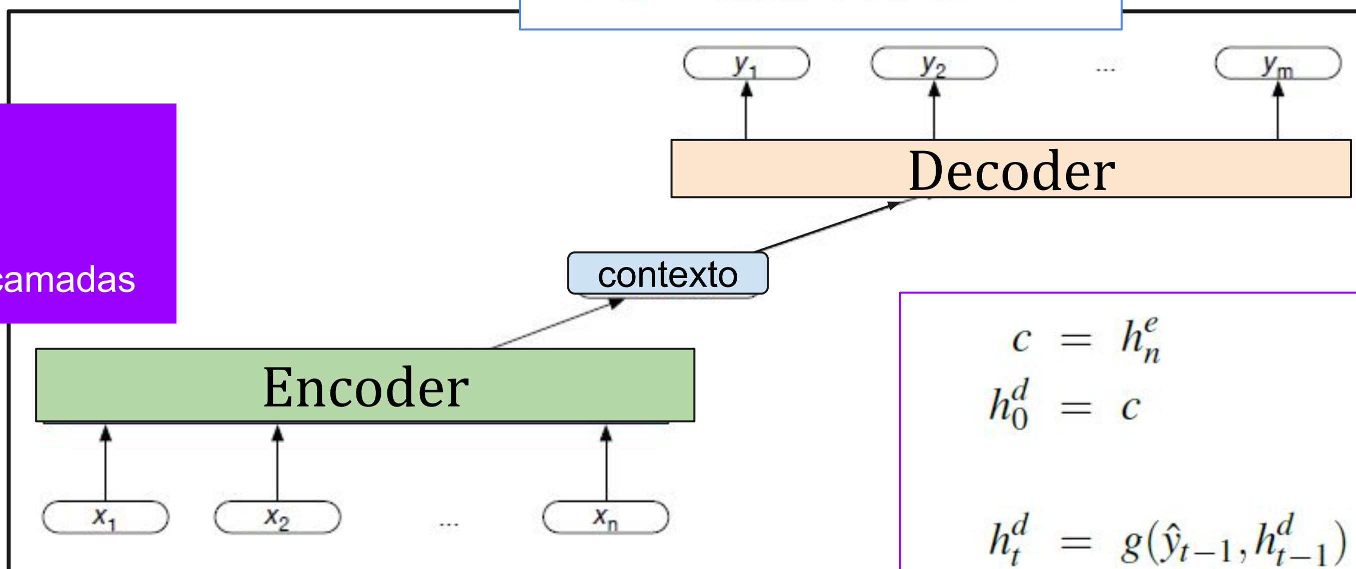


$$
\begin{aligned}
c &= h^e_n \\
h^d_0 &= c \\
\\
h^d_t &= g(\hat{y}_{t-1}, h^d_{t-1}) \\
z_t &= f(h^d_t) \\
y_t &= \text{softmax}(z_t)
\end{aligned}
$$

Encoder-decoder abstração

$$h_t^d = g(\hat{y}_{t-1}, h_{t-1}^d, c)$$

RNN
(Bi)LSTM
GRU
Múltiplas camadas

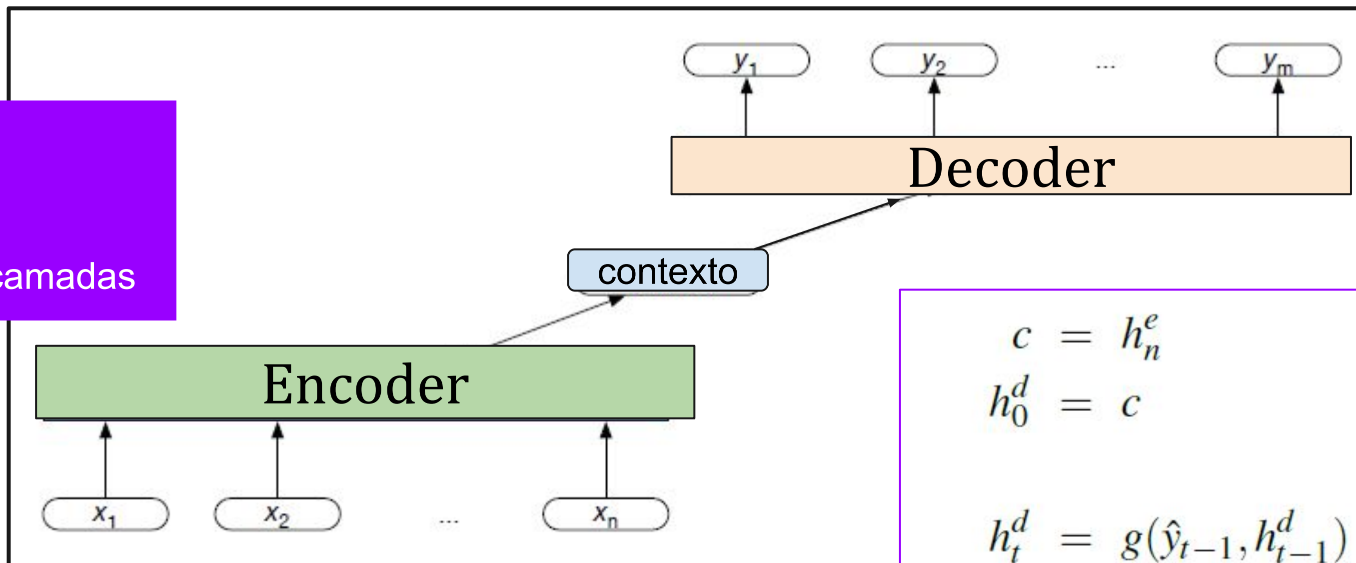


$$
\begin{aligned}
c &= h_n^e \\
h_0^d &= c \\
\\
h_t^d &= g(\hat{y}_{t-1}, h_{t-1}^d) \\
z_t &= f(h_t^d) \\
y_t &= \text{softmax}(z_t)
\end{aligned}
$$

Encoder-decoder abstração

Decoder olha apenas o estado escondido e a última saída

RNN
(Bi)LSTM
GRU
Múltiplas camadas



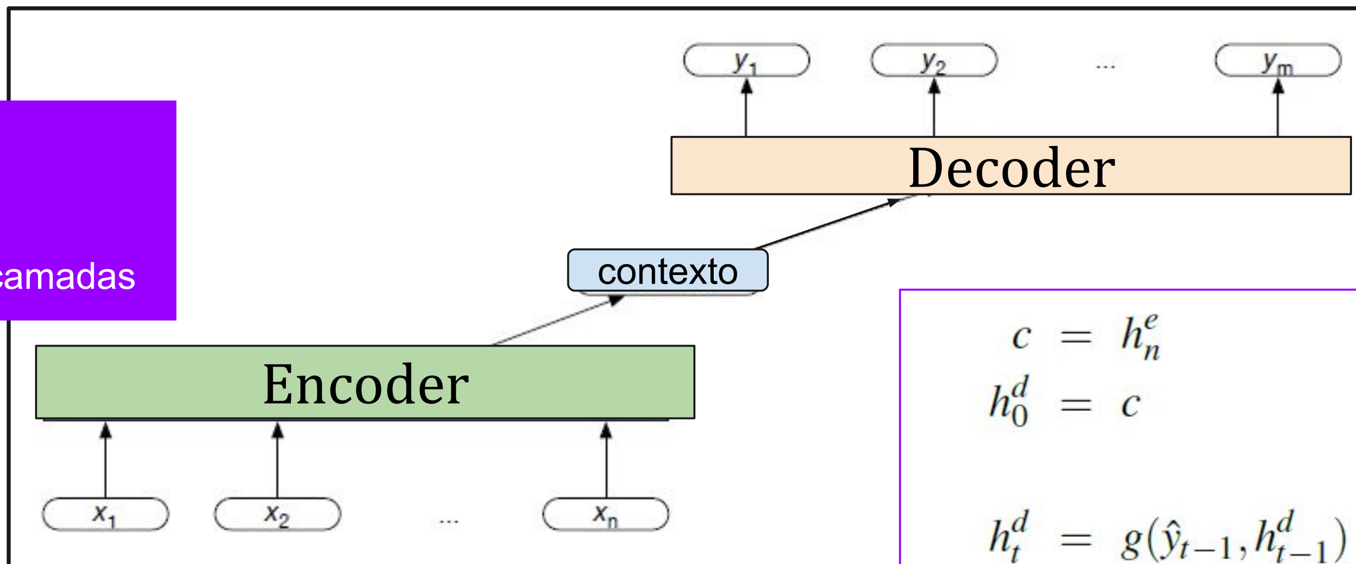
$$
\begin{aligned}
c &= h_n^e \\
h_0^d &= c \\
\\
h_t^d &= g(\hat{y}_{t-1}, h_{t-1}^d) \\
z_t &= f(h_t^d) \\
y_t &= \text{softmax}(z_t)
\end{aligned}
$$

Encoder-decoder abstração

Decoder olha apenas o estado escondido e a última saída

$$y_t = \text{softmax}(\hat{y}_{t-1}, z_t, c)$$

RNN
(Bi)LSTM
GRU
Múltiplas camadas



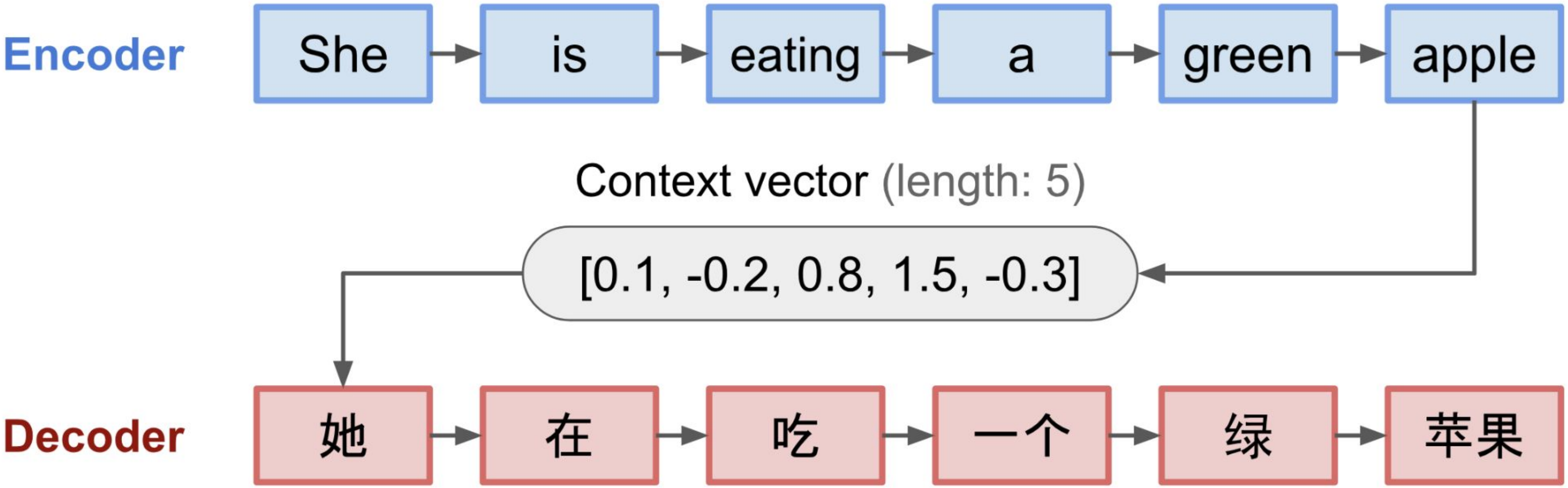
$$
\begin{aligned}
c &= h_n^e \\
h_0^d &= c \\
\\
h_t^d &= g(\hat{y}_{t-1}, h_{t-1}^d) \\
z_t &= f(h_t^d) \\
y_t &= \text{softmax}(z_t)
\end{aligned}
$$

Bottleneck

http://jalammar.github.io/visualizing-neural-machine-translation
-mechanics-of-seq2seq-models-with-attention/

Bottleneck

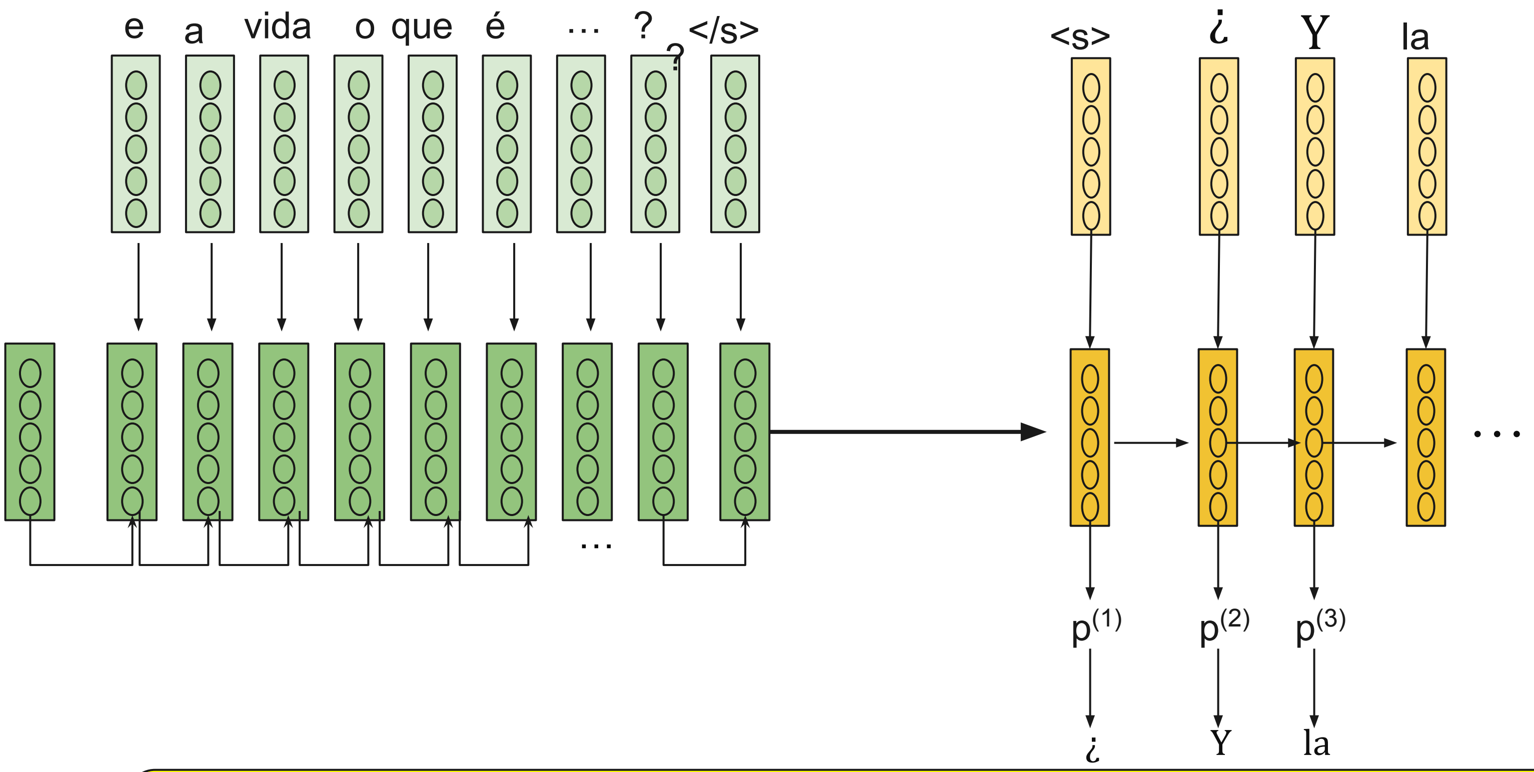
Processa a sequência de entrada e compacta as informações em um vetor de contexto de comprimento fixo. Espera-se que esta representação seja um bom resumo do significado de toda a sequência fonte.



Um decodificador é inicializado com o vetor de contexto para emitir a saída transformada. Os primeiros trabalhos usavam apenas o último estado da rede do codificador como estado inicial do decodificador : como lembrar de tudo aqui??.

https://lilianweng.github.io/posts/2018-06-24-attention/

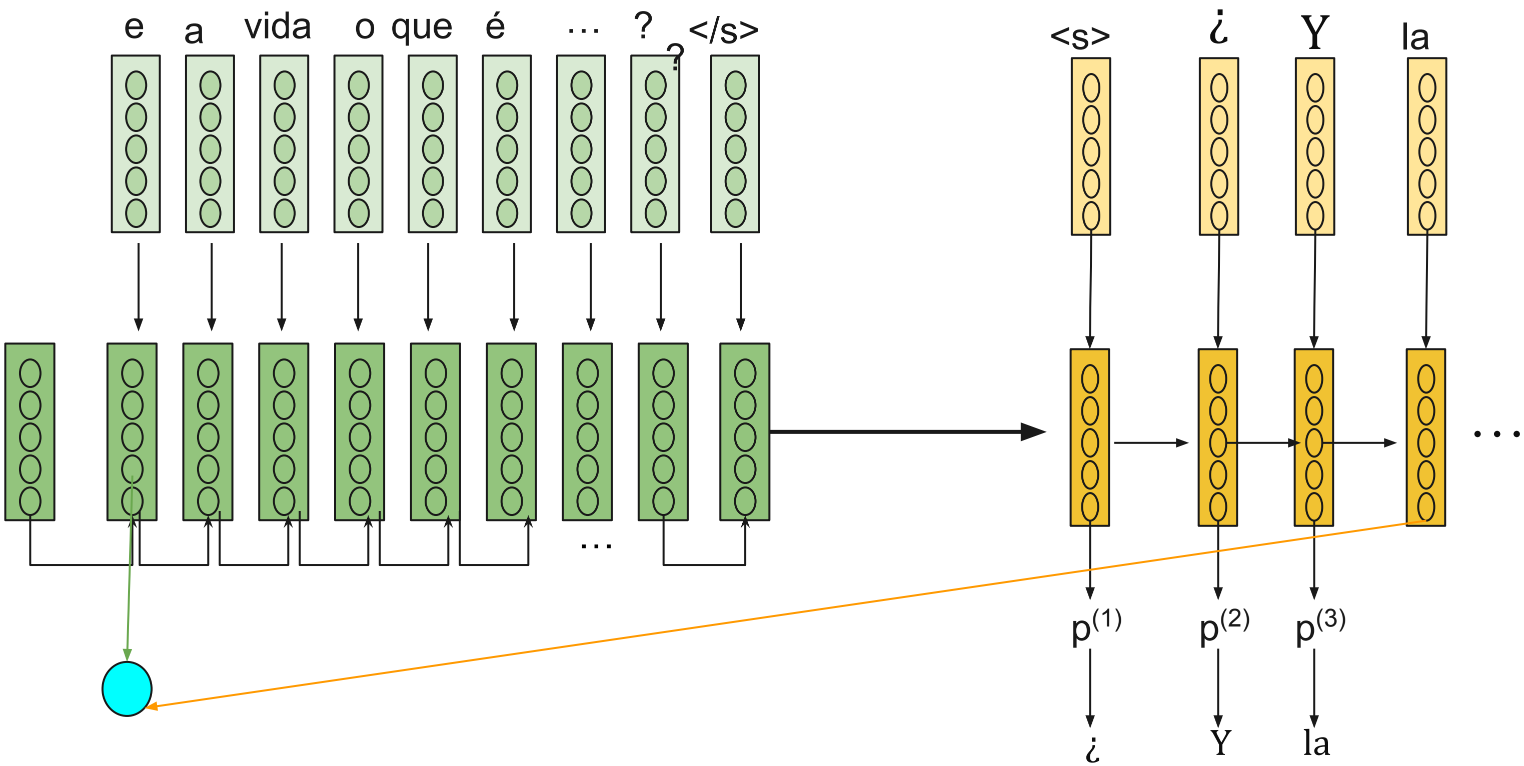
ATENÇÃO



FAÇA O MODELO FOCAR EM DIFERENTES PARTES DA ENTRADA A CADA PASSO

Generating sequences with recurrent neural networks.

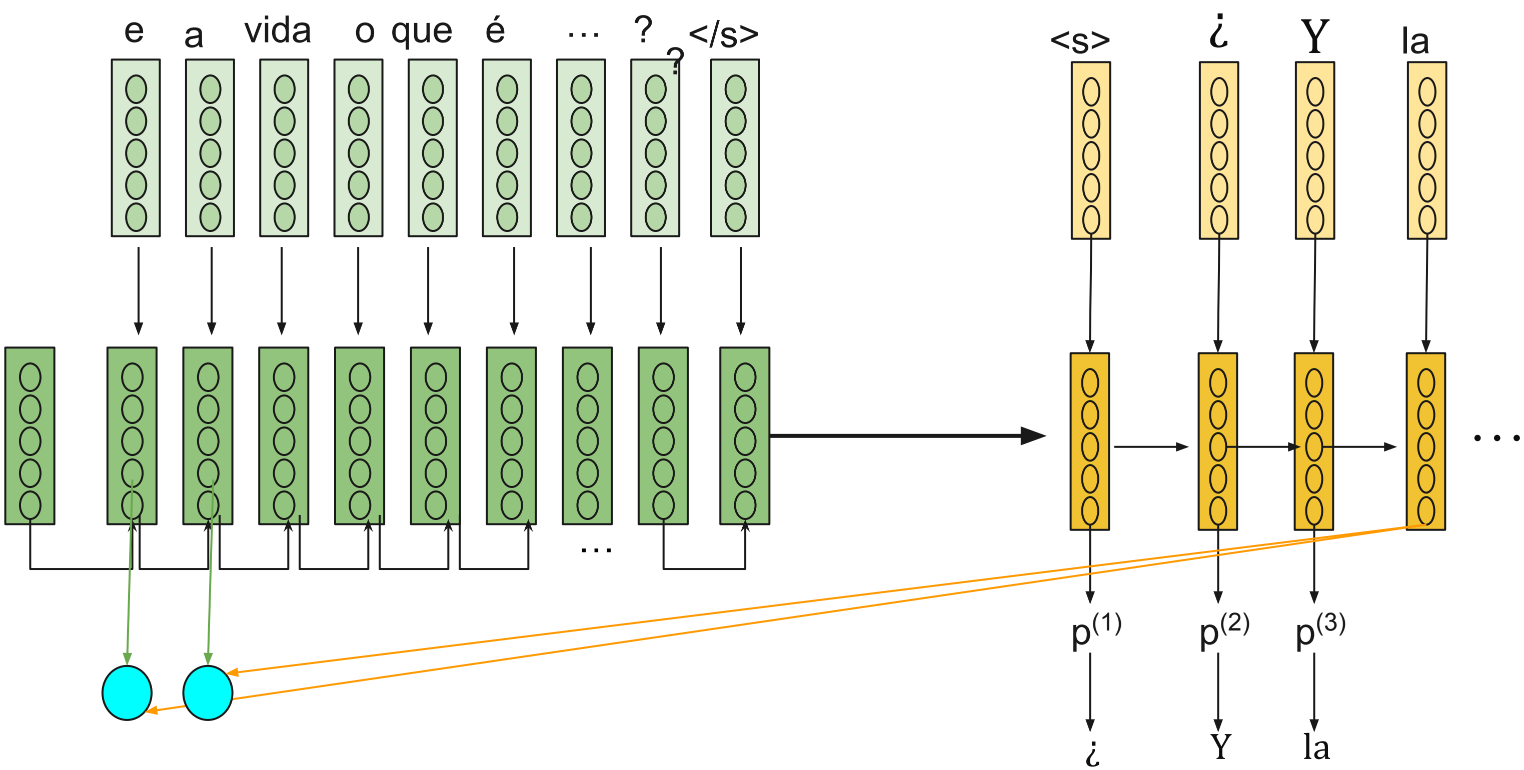
ATENÇÃO



FAÇA O MODELO FOCAR EM DIFERENTES PARTES DA ENTRADA **A CADA PASSO**

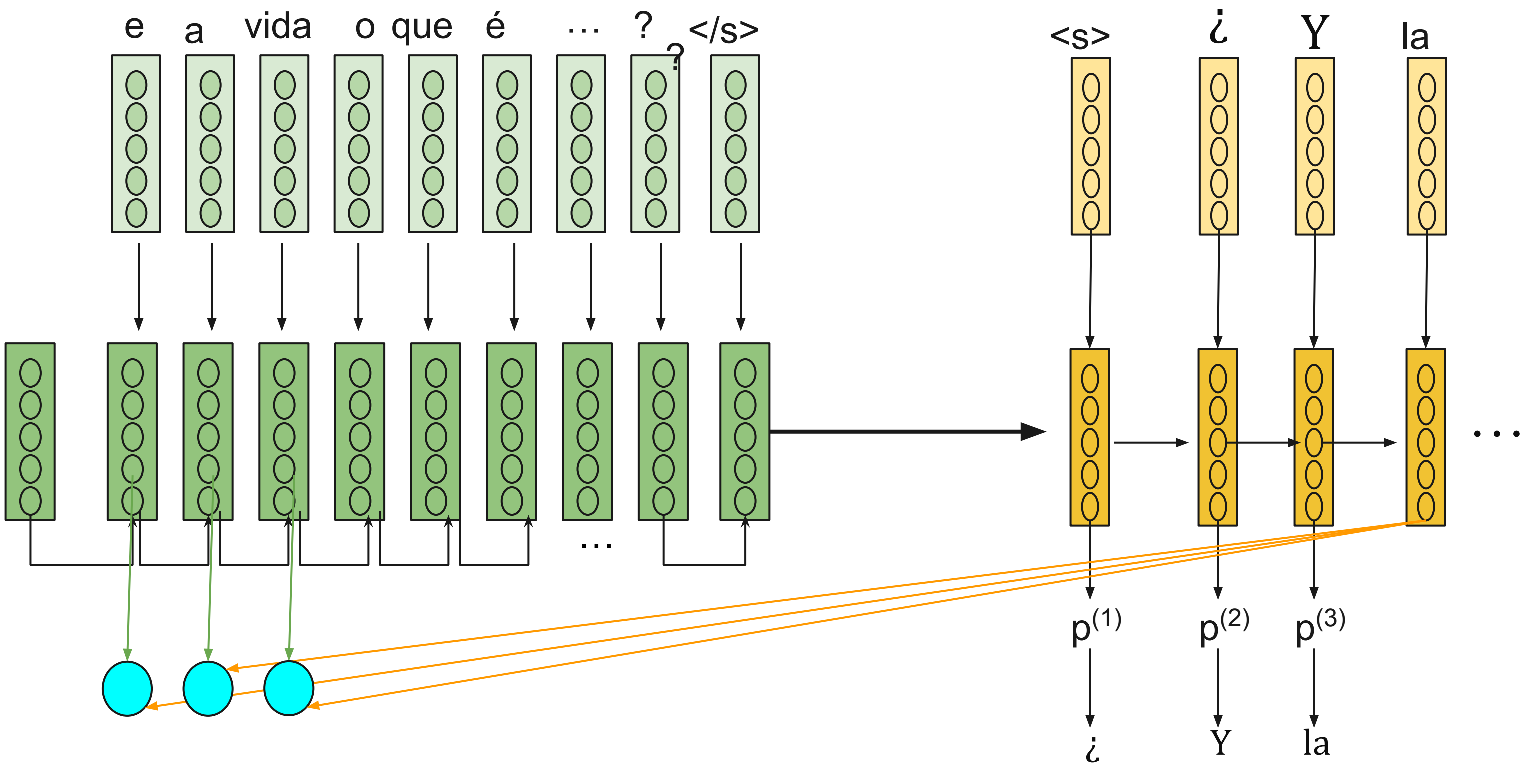
Generating sequences with recurrent neural networks.

ATENÇÃO



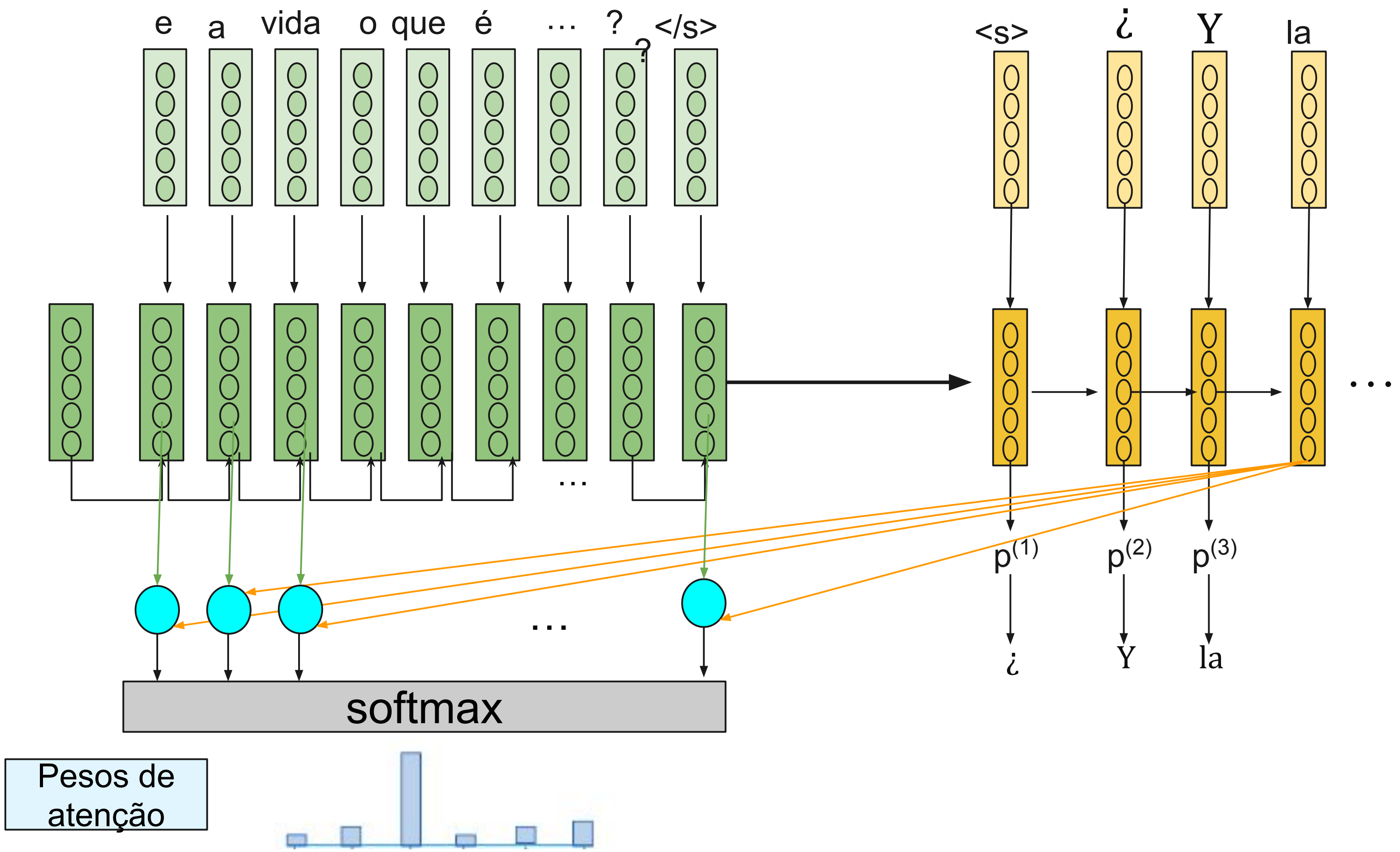
FAÇA O MODELO FOCAR EM DIFERENTES PARTES DA ENTRADA **A CADA PASSO**

ATENÇÃO



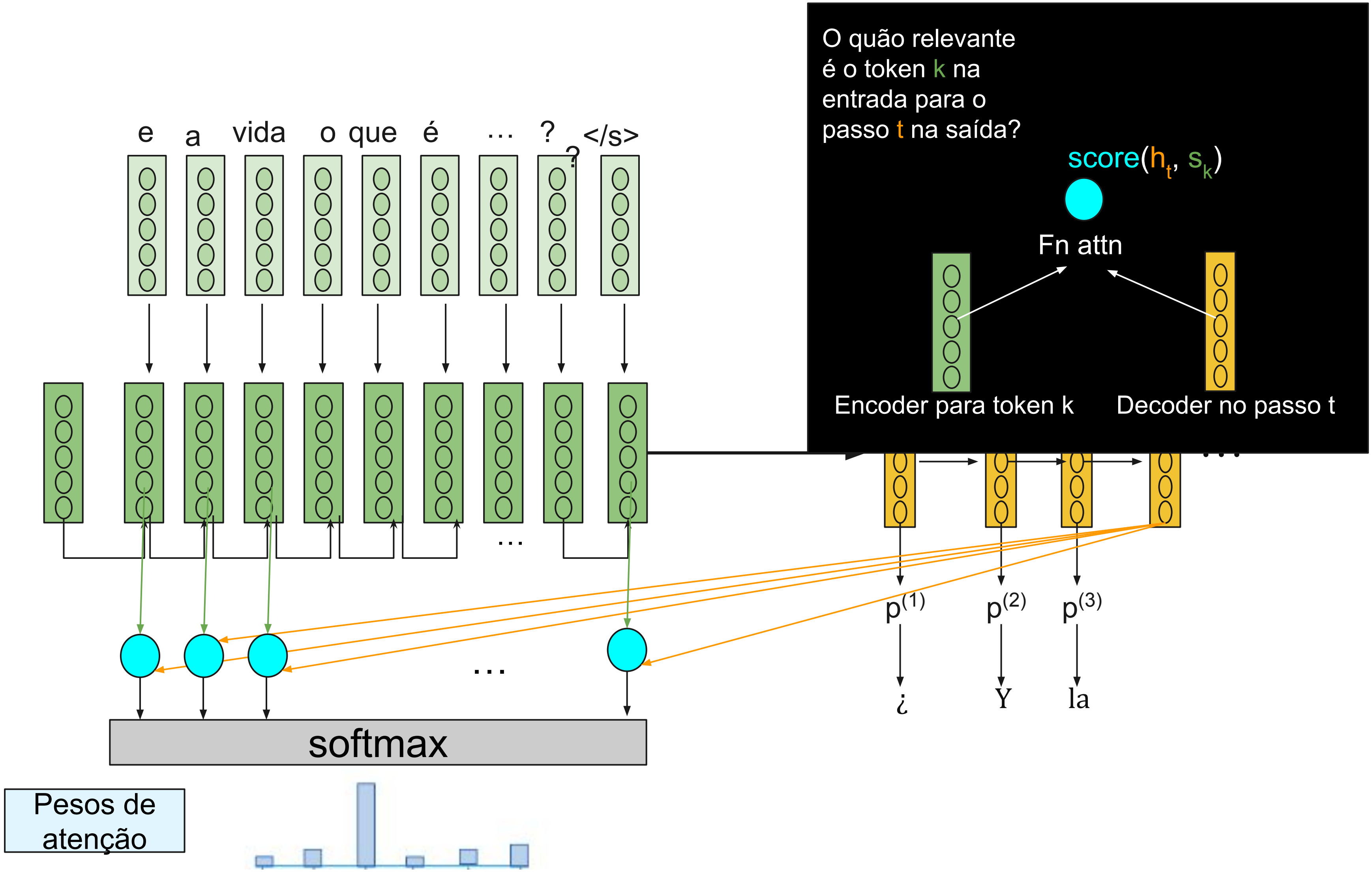
FAÇA O MODELO FOCAR EM DIFERENTES PARTES DA ENTRADA A CADA PASSO

ATENÇÃO



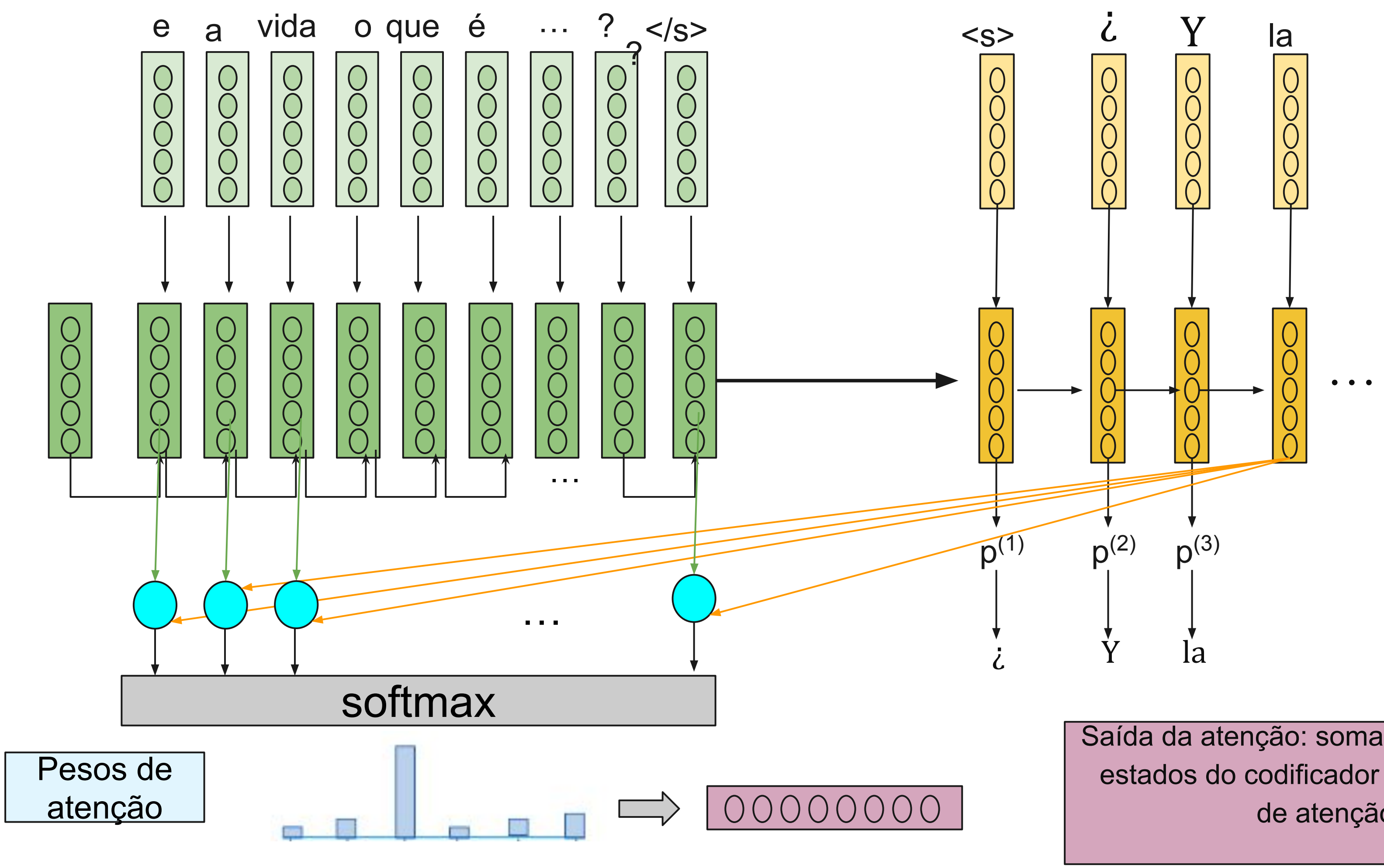
Saída da atenção: soma ponderada dos estados do codificador com pesos de atenção.

ATENÇÃO



[redacted]

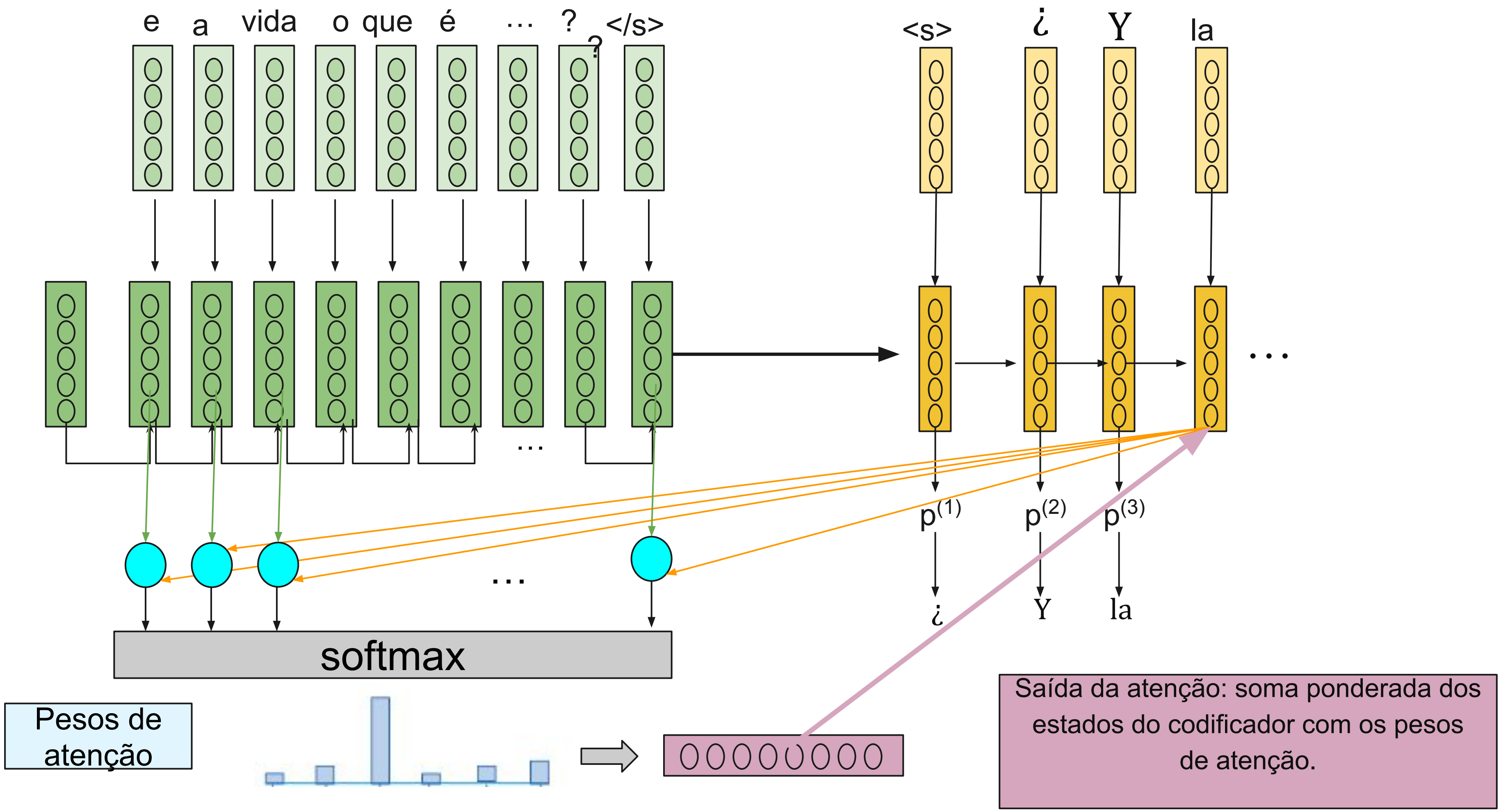
ATENÇÃO



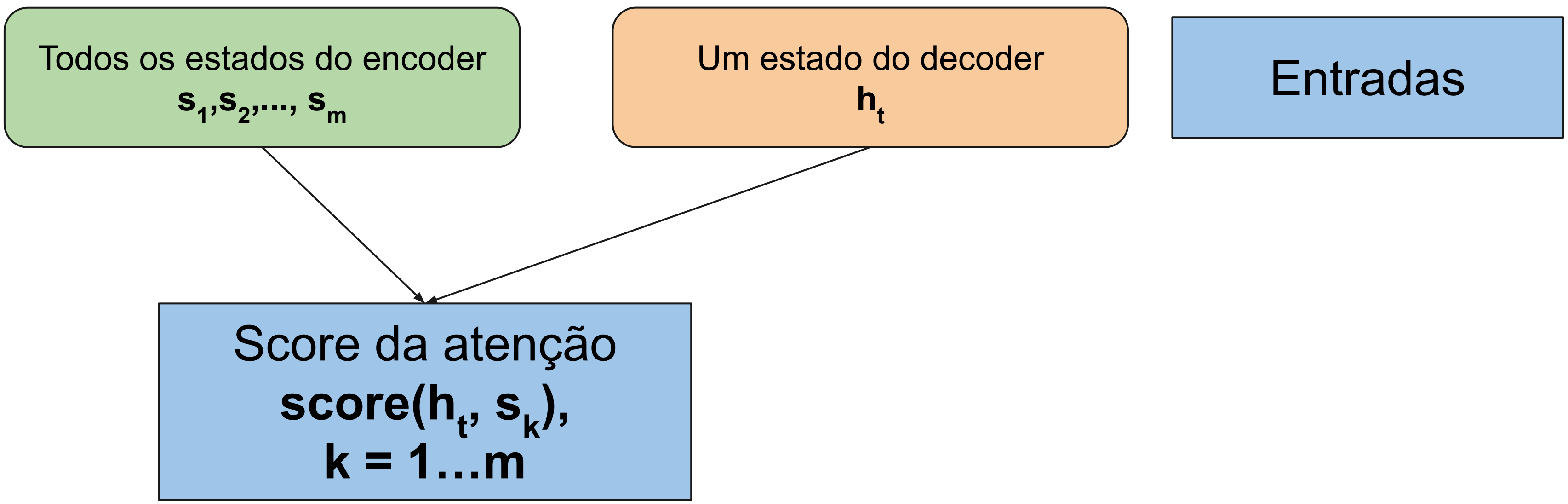
Saída da atenção: soma ponderada dos estados do codificador com os pesos de atenção.

Saída da atenção: soma ponderada dos
estados do codificador com pesos de atenção.

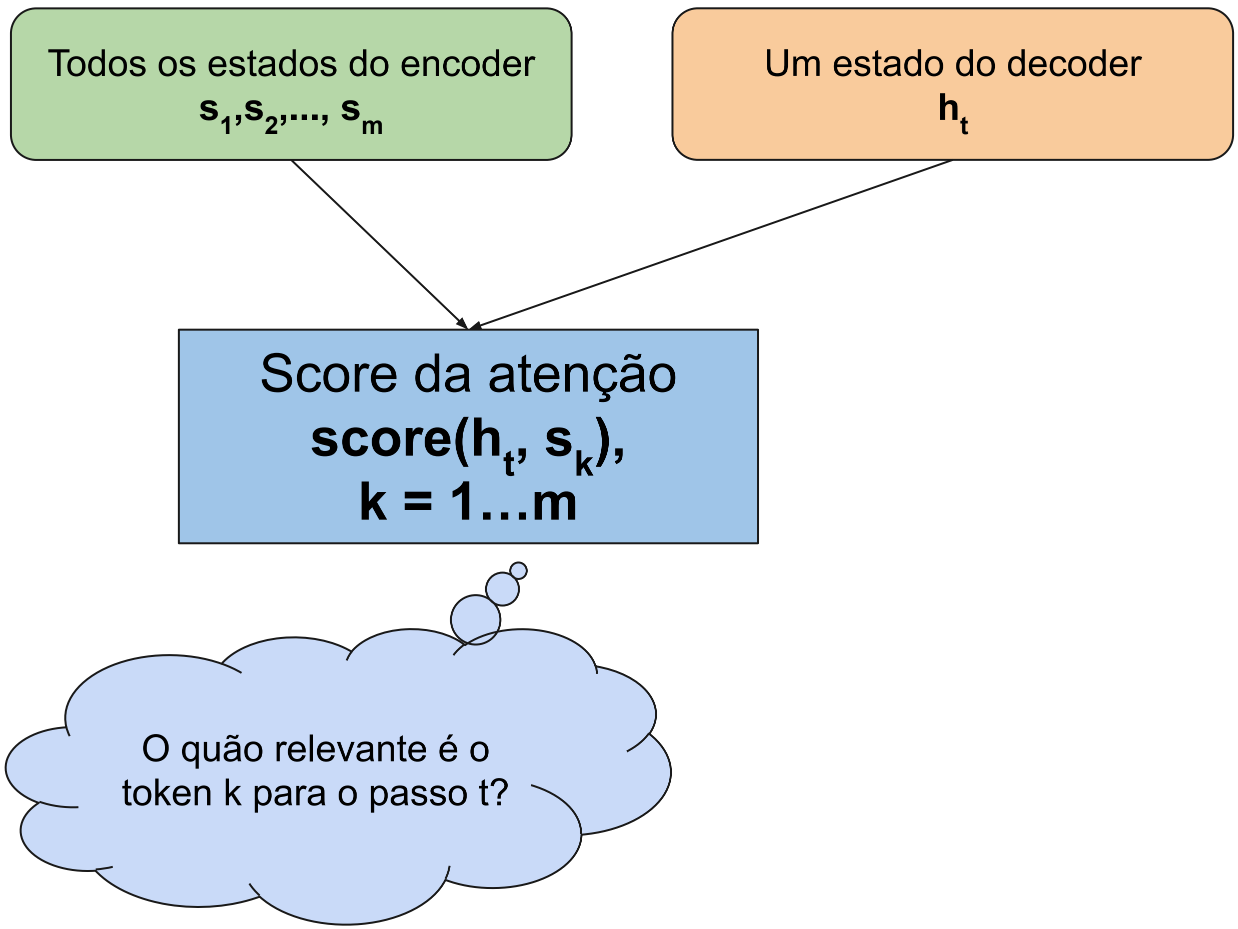
ATENÇÃO



Pipeline

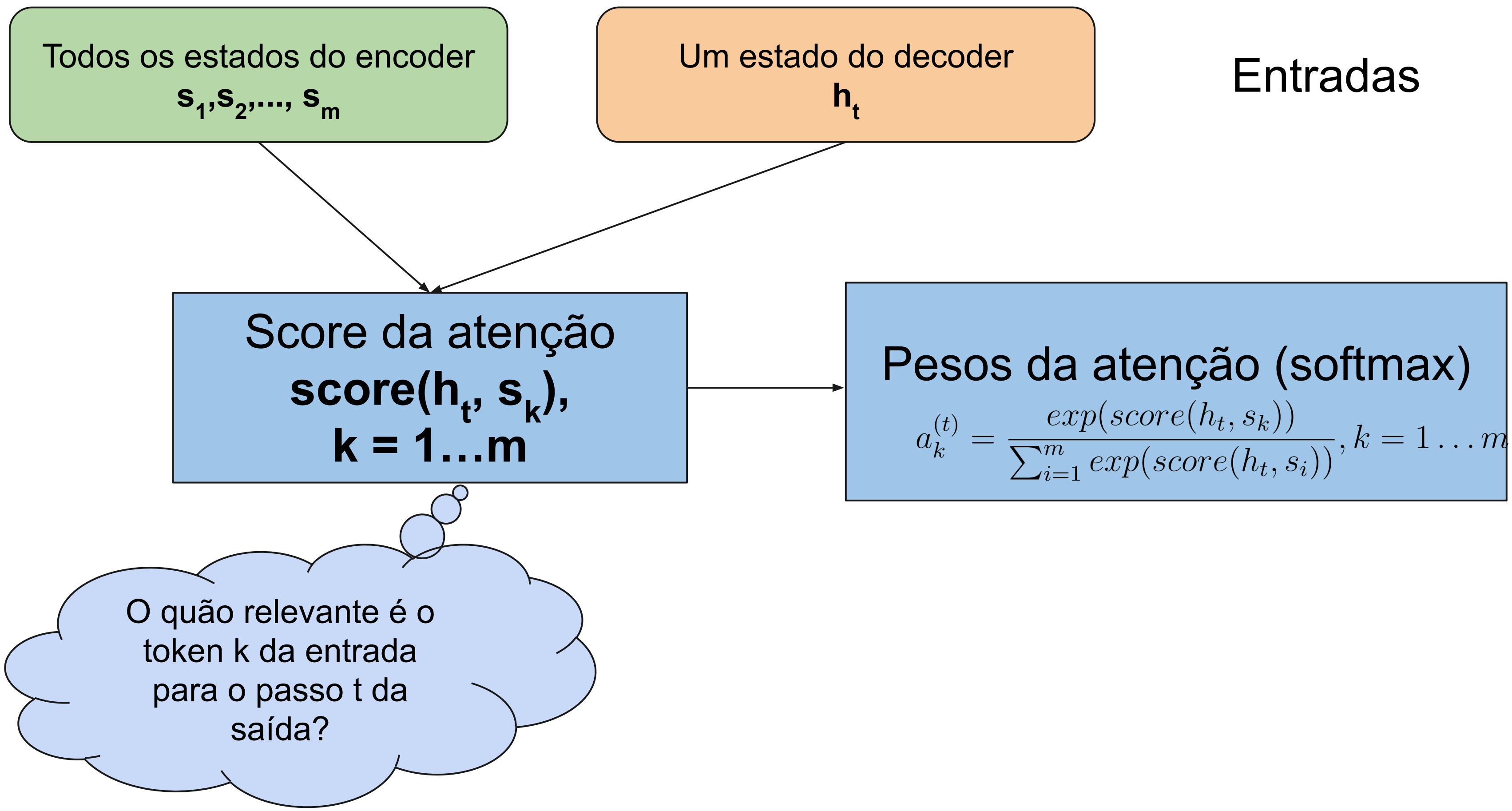

Todos os estados do encoder
$s_1, s_2, \ldots, s_m$
Um estado do decoder
h_t
Entradas
Score da atenção
$\mathrm{score}(h_t, s_k)$,
$k = 1 \ldots m$

Pipeline

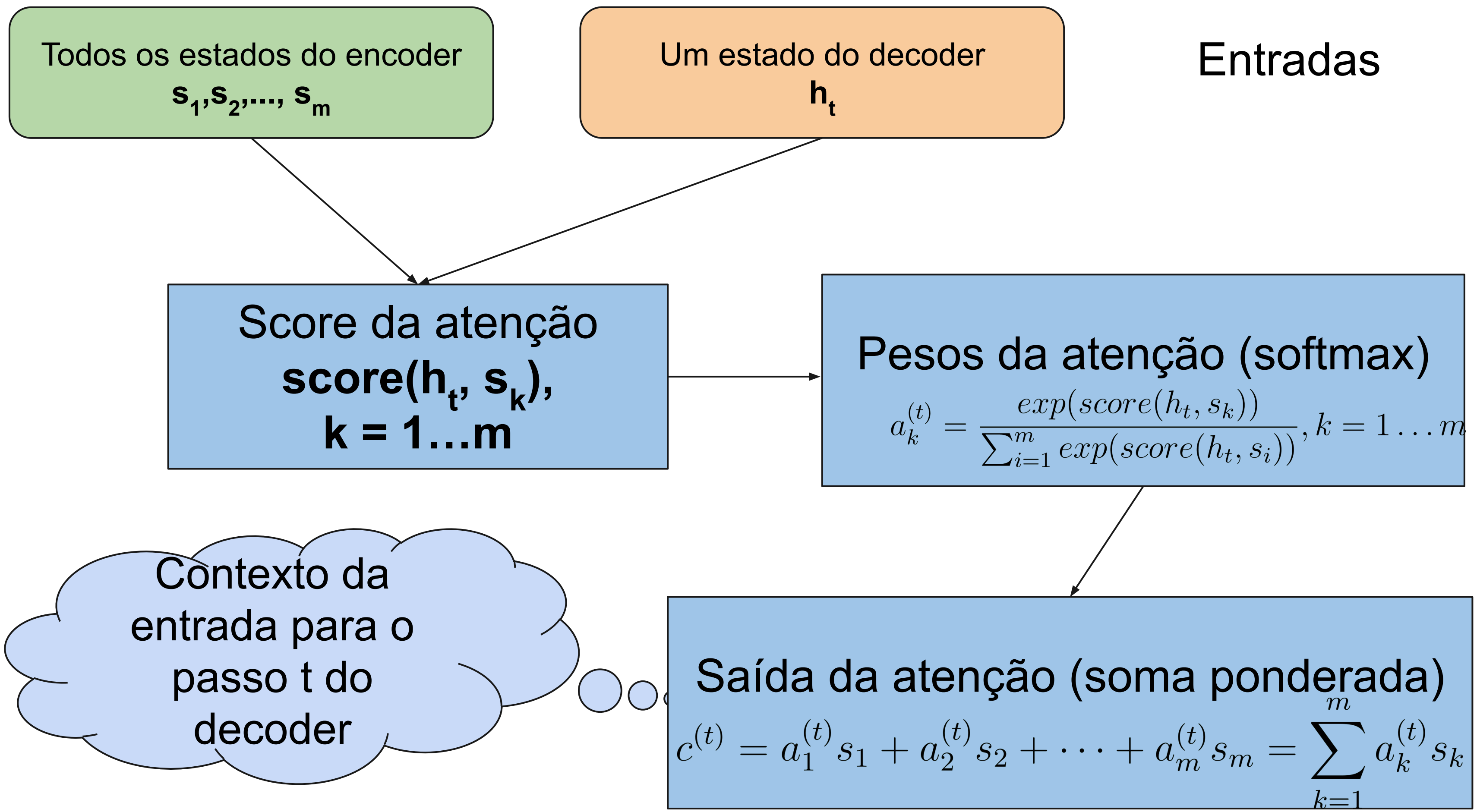


Entradas

Pipeline


Todos os estados do encoder
$s_1, s_2, \ldots, s_m$
Um estado do decoder
h_t
Entradas
Score da atenção
score(h_t, s_k),
k = 1…m
Pesos da atenção (softmax)
$a_k^{(t)} = \frac{exp(score(h_t, s_k))}{\sum_{i=1}^{m} exp(score(h_t, s_i))}, k = 1 \ldots m$
O quão relevante é o token k da entrada para o passo t da saída?

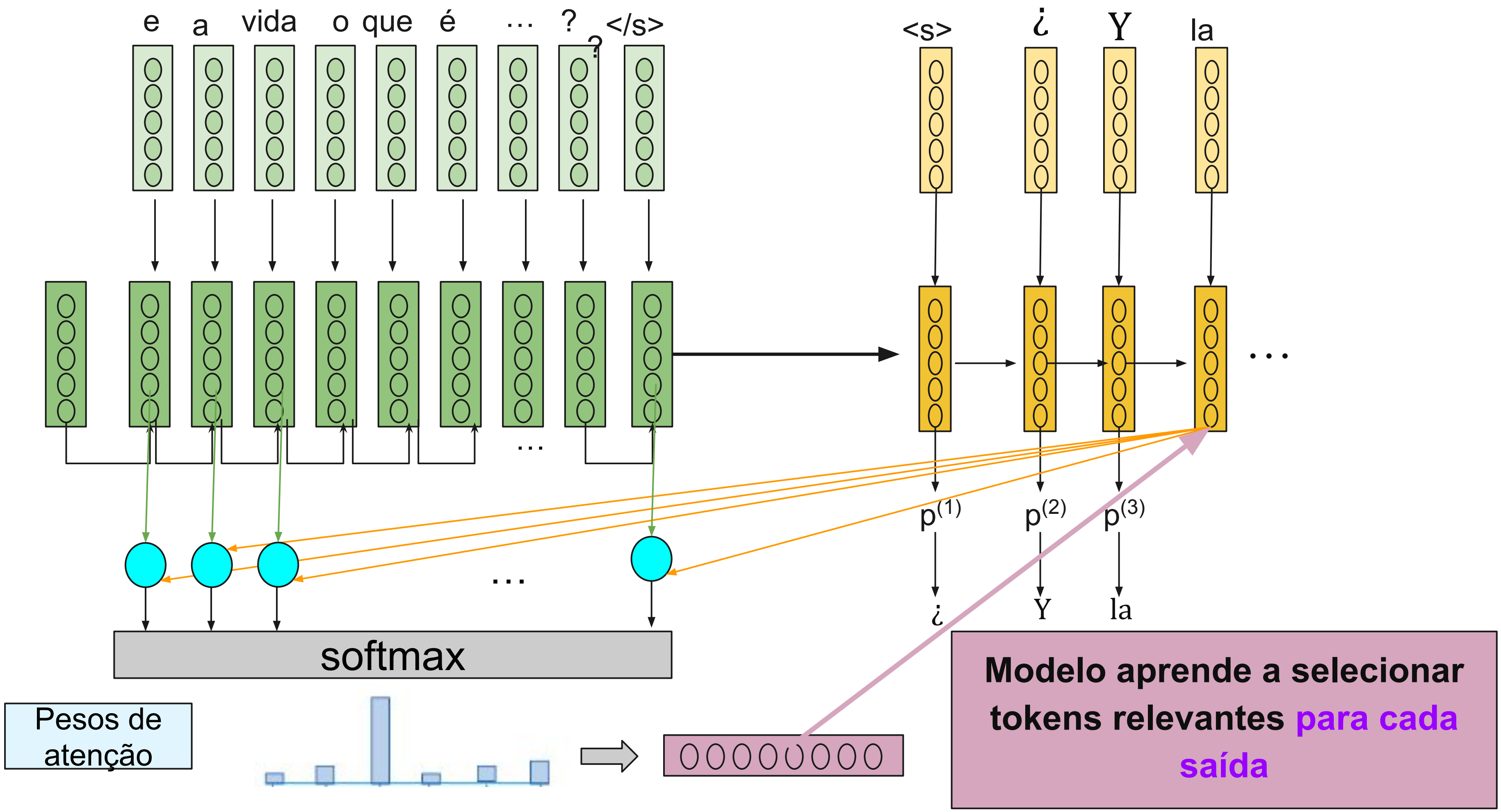
Pipeline



Todos os estados do encoder
$s_1, s_2, \ldots, s_m$
Um estado do decoder
h_t
Entradas
Score da atenção
score(h_t, s_k),
k = 1…m
Pesos da atenção (softmax)
$a_k^{(t)} = \frac{exp(score(h_t, s_k))}{\sum_{i=1}^{m} exp(score(h_t, s_i))}, k = 1 \ldots m$
Saída da atenção (soma ponderada)
$c^{(t)} = a_1^{(t)} s_1 + a_2^{(t)} s_2 + \cdots + a_m^{(t)} s_m = \sum_{k=1}^{m} a_k^{(t)} s_k$
Contexto da entrada para o passo t do decoder

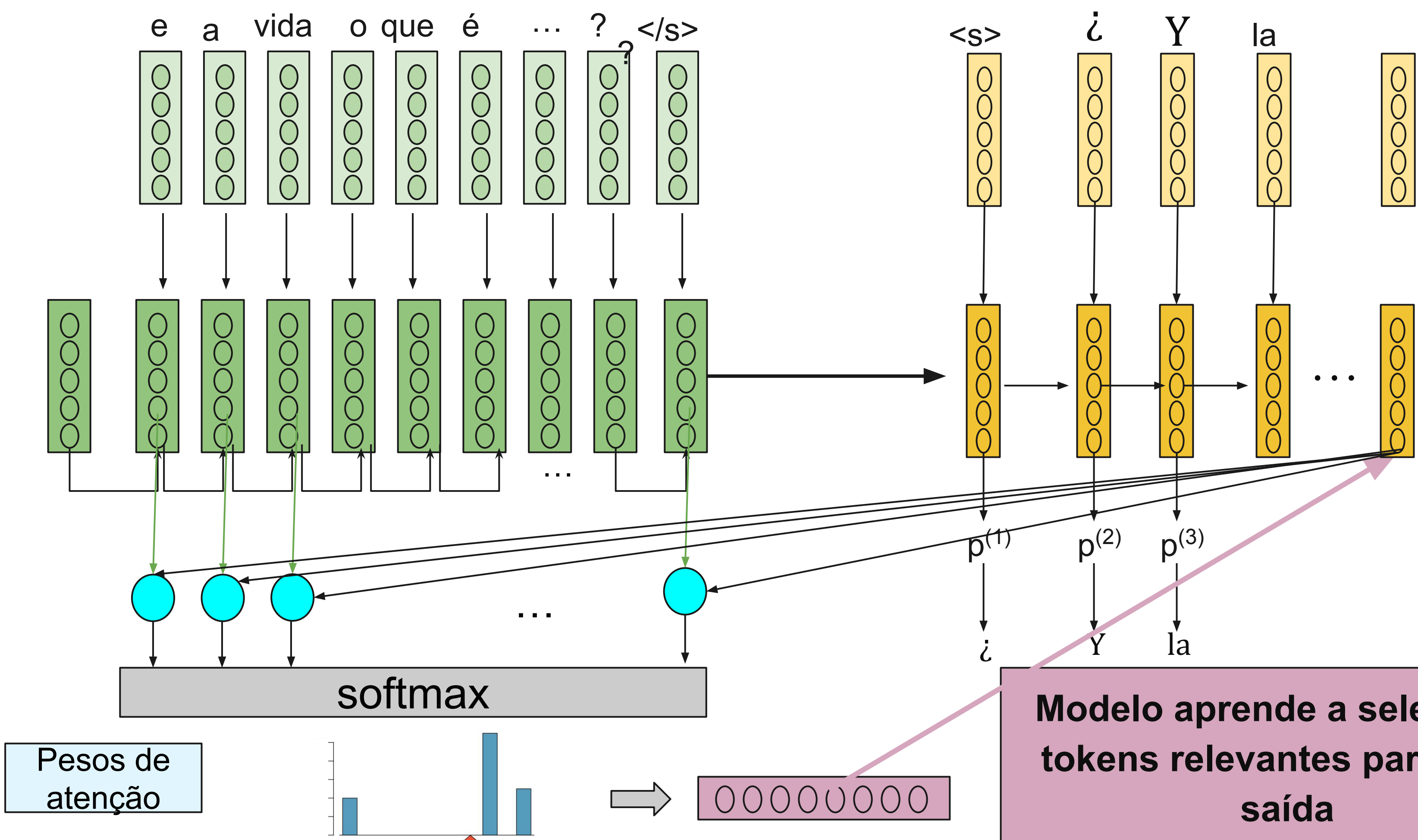
Saída da atenção: soma ponderada dos estados do codificador com pesos de atenção.

ATENÇÃO



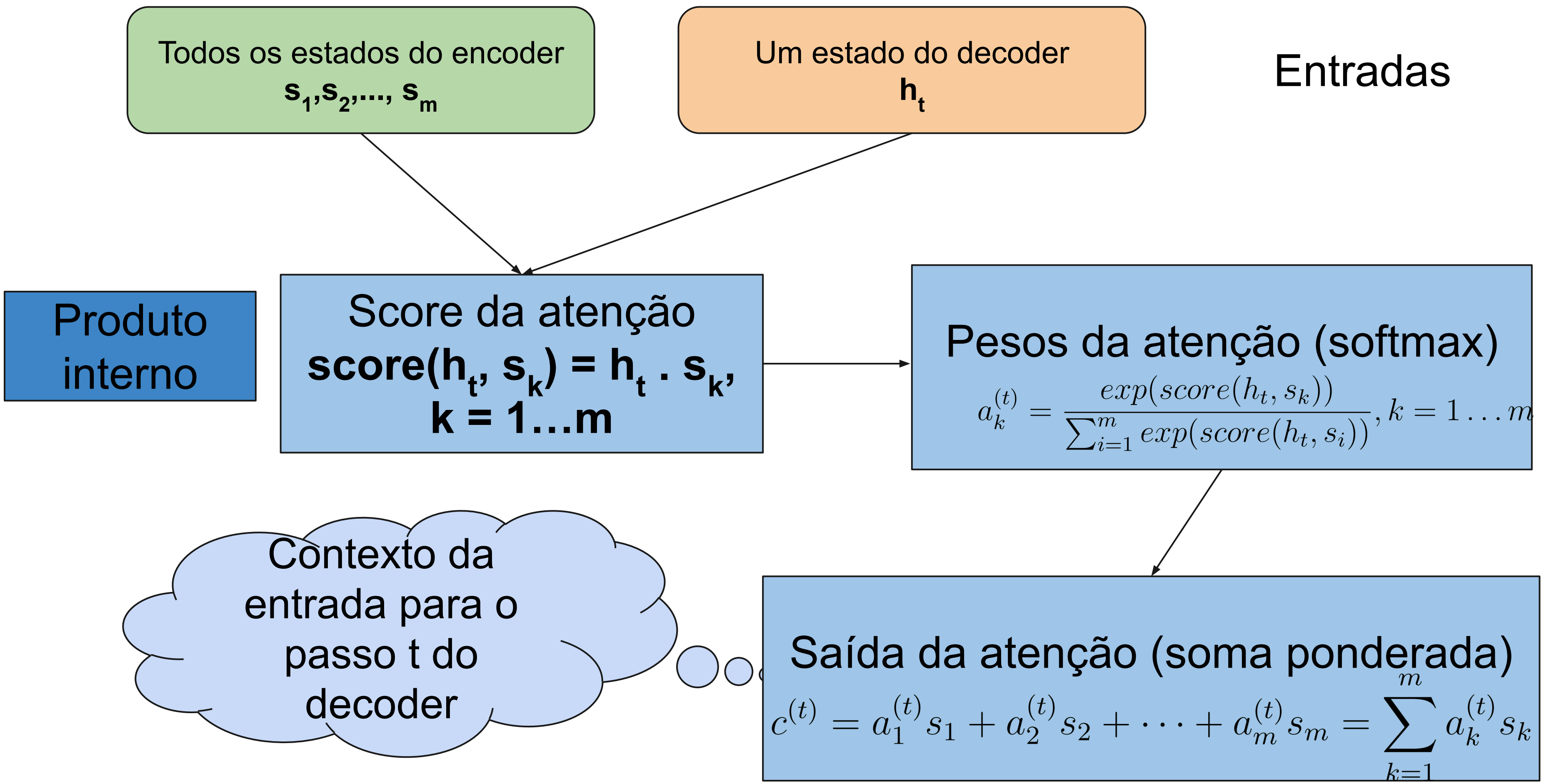
Saída da atenção: soma ponderada dos estados do codificador com pesos de atenção.

ATENÇÃO



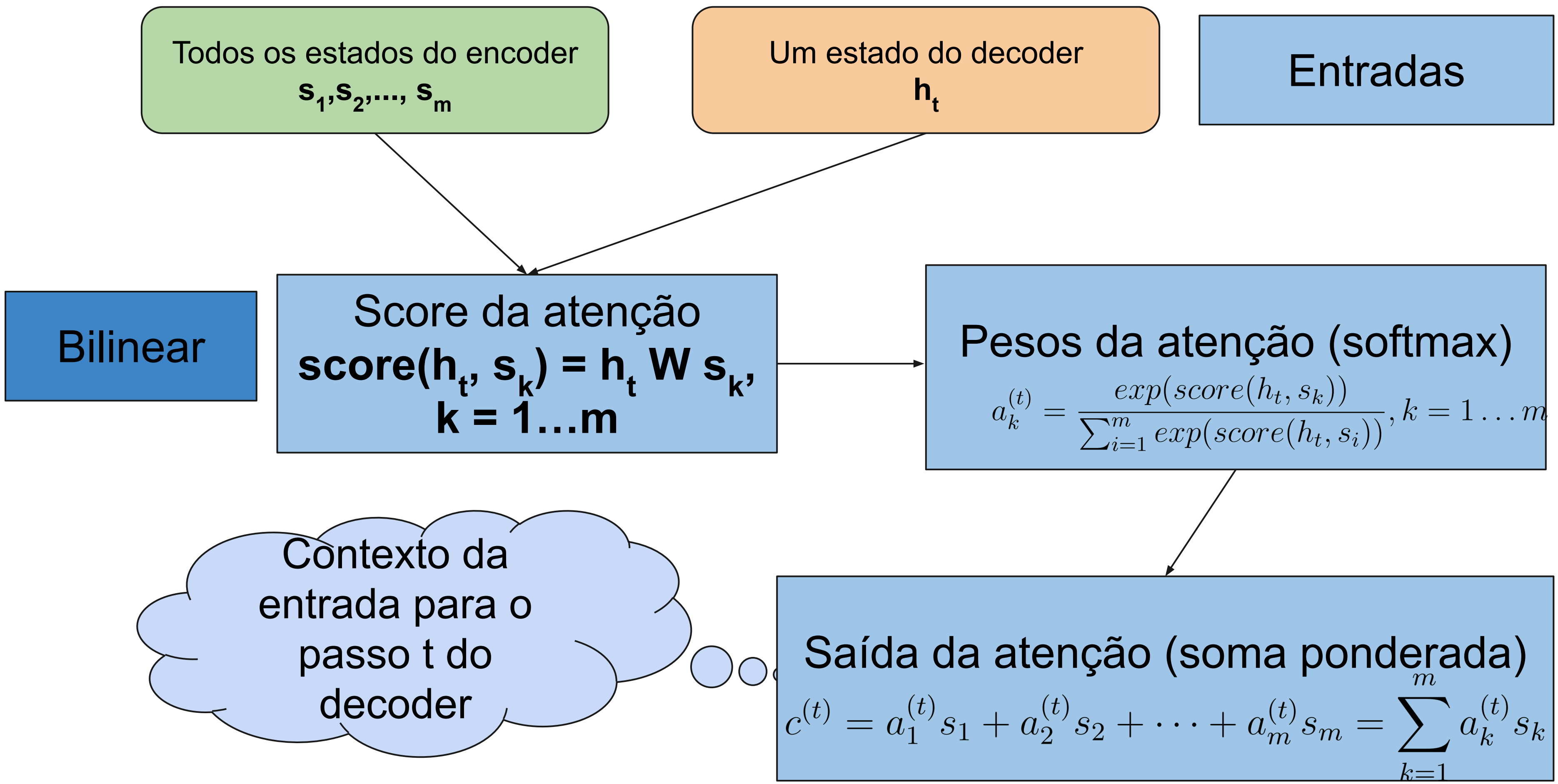
Modelo aprende a selecionar tokens relevantes para cada saída

Pipeline



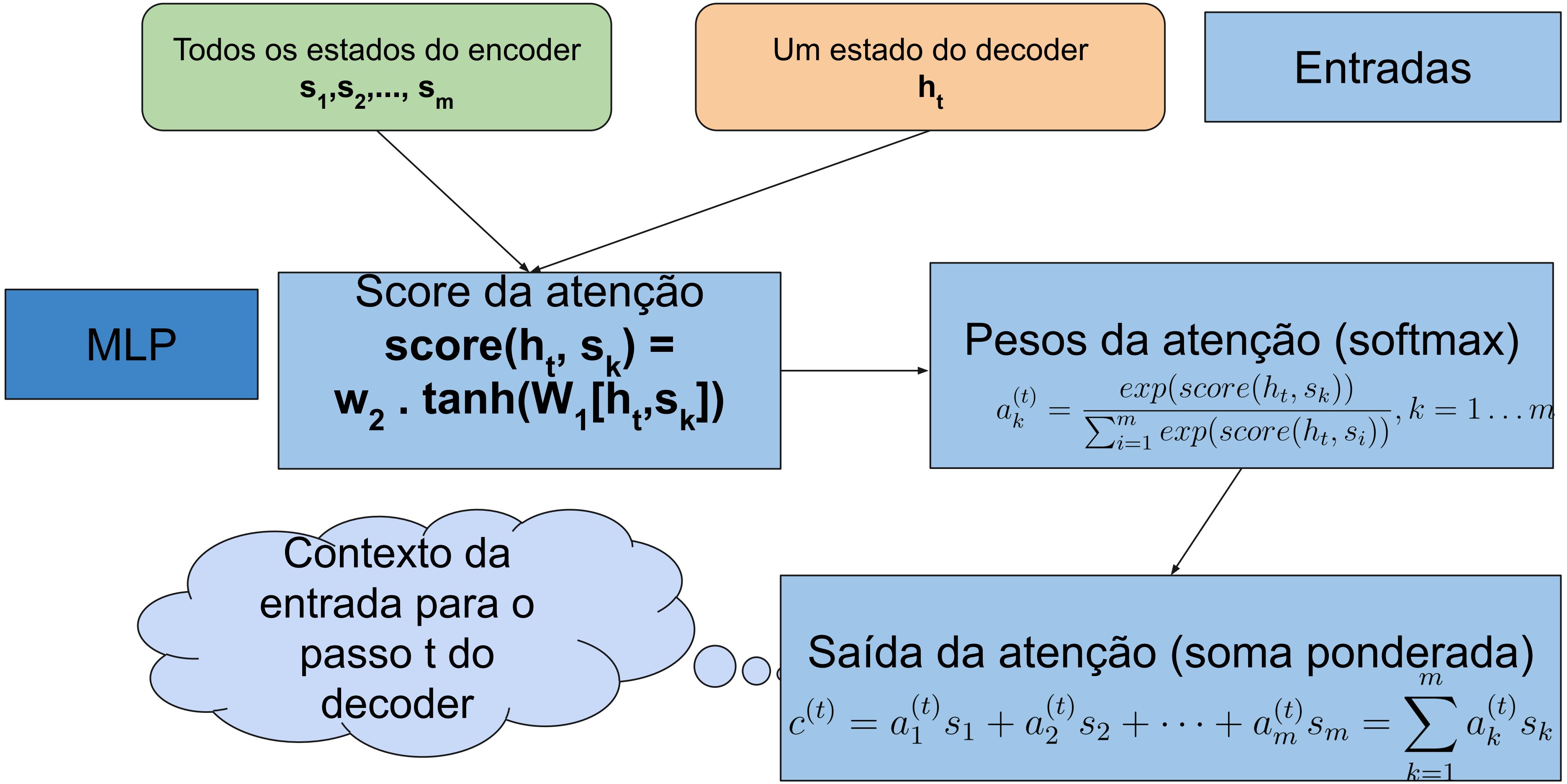
Todos os estados do encoder
$s_1, s_2, \ldots, s_m$
Um estado do decoder
h_t
Entradas
Produto interno
Score da atenção
$\mathrm{score}(h_t, s_k) = h_t \cdot s_k,\ k = 1 \ldots m$
Pesos da atenção (softmax)
$a_k^{(t)} = \frac{exp(score(h_t, s_k))}{\sum_{i=1}^{m} exp(score(h_t, s_i))}, k = 1 \ldots m$
Contexto da entrada para o passo t do decoder
Saída da atenção (soma ponderada)
$c^{(t)} = a_1^{(t)} s_1 + a_2^{(t)} s_2 + \cdots + a_m^{(t)} s_m = \sum_{k=1}^{m} a_k^{(t)} s_k$

Pipeline



Todos os estados do encoder
$s_1, s_2, \ldots, s_m$
Um estado do decoder
h_t
Entradas
Bilinear
Score da atenção
$score(h_t, s_k) = h_t W s_k,$
$k = 1 \ldots m$
Pesos da atenção (softmax)
$a_k^{(t)} = \frac{exp(score(h_t, s_k))}{\sum_{i=1}^{m} exp(score(h_t, s_i))}, k = 1 \ldots m$
Contexto da entrada para o passo t do decoder
Saída da atenção (soma ponderada)
$c^{(t)} = a_1^{(t)} s_1 + a_2^{(t)} s_2 + \cdots + a_m^{(t)} s_m = \sum_{k=1}^{m} a_k^{(t)} s_k$

Pipeline


Todos os estados do encoder
$s_1, s_2, \ldots, s_m$
Um estado do decoder
h_t
Entradas
MLP
Score da atenção
$\mathrm{score}(h_t, s_k) = w_2 \cdot \tanh(W_1[h_t, s_k])$
Pesos da atenção (softmax)
$a_k^{(t)} = \frac{exp(score(h_t, s_k))}{\sum_{i=1}^{m} exp(score(h_t, s_i))}, k = 1 \ldots m$
Contexto da entrada para o passo t do decoder
Saída da atenção (soma ponderada)
$c^{(t)} = a_1^{(t)} s_1 + a_2^{(t)} s_2 + \cdots + a_m^{(t)} s_m = \sum_{k=1}^{m} a_k^{(t)} s_k$

Pipeline

Todos os estados do encoder
$s_1, s_2, \ldots, s_m$

Um estado do decoder
h_t

Entradas

$$w_2^T \times \tanh \left[W_1 \times \begin{bmatrix} h_t \\ s_k \end{bmatrix} \right]$$

Pesos da atenção (softmax)

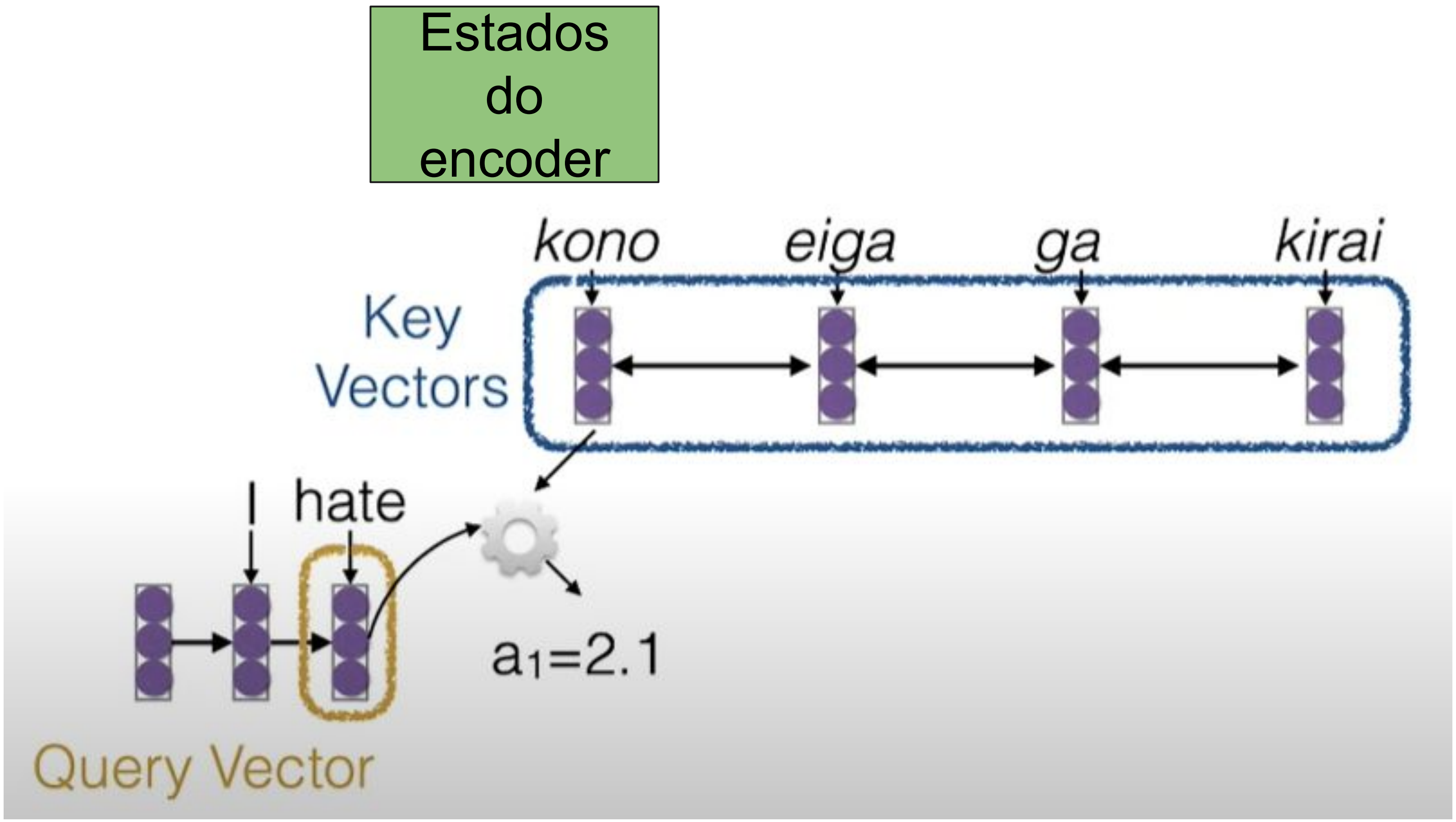
$$a_k^{(t)} = \frac{exp(score(h_t, s_k))}{\sum_{i=1}^{m} exp(score(h_t, s_i))}, k = 1 \ldots m$$

Saída da atenção (soma ponderada)

$$c^{(t)} = a_1^{(t)} s_1 + a_2^{(t)} s_2 + \cdots + a_m^{(t)} s_m = \sum_{k=1}^{m} a_k^{(t)} s_k$$

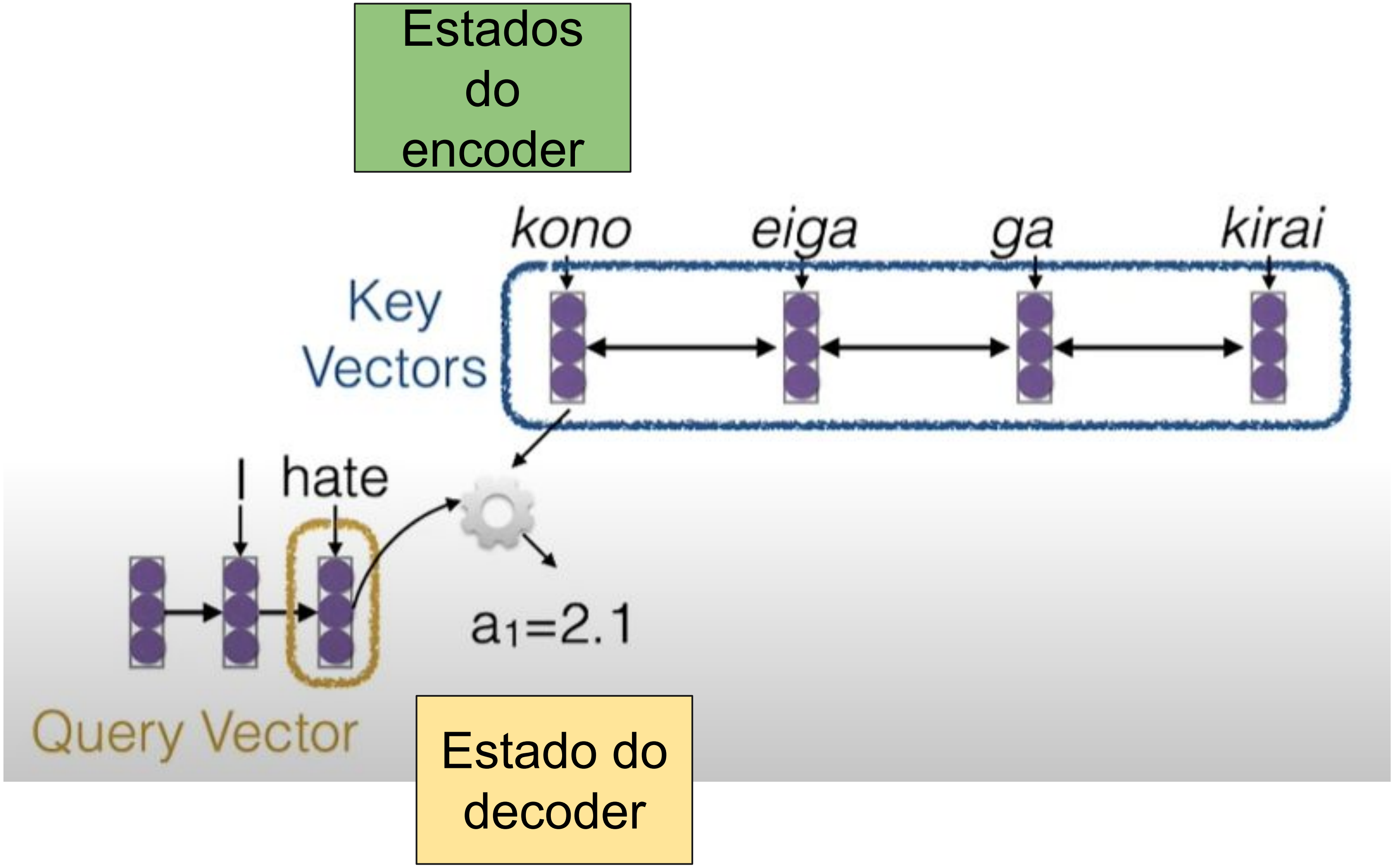
Contexto da entrada para o passo t do decoder

Attention



*from CMU Neural Nets for NLP

Attention

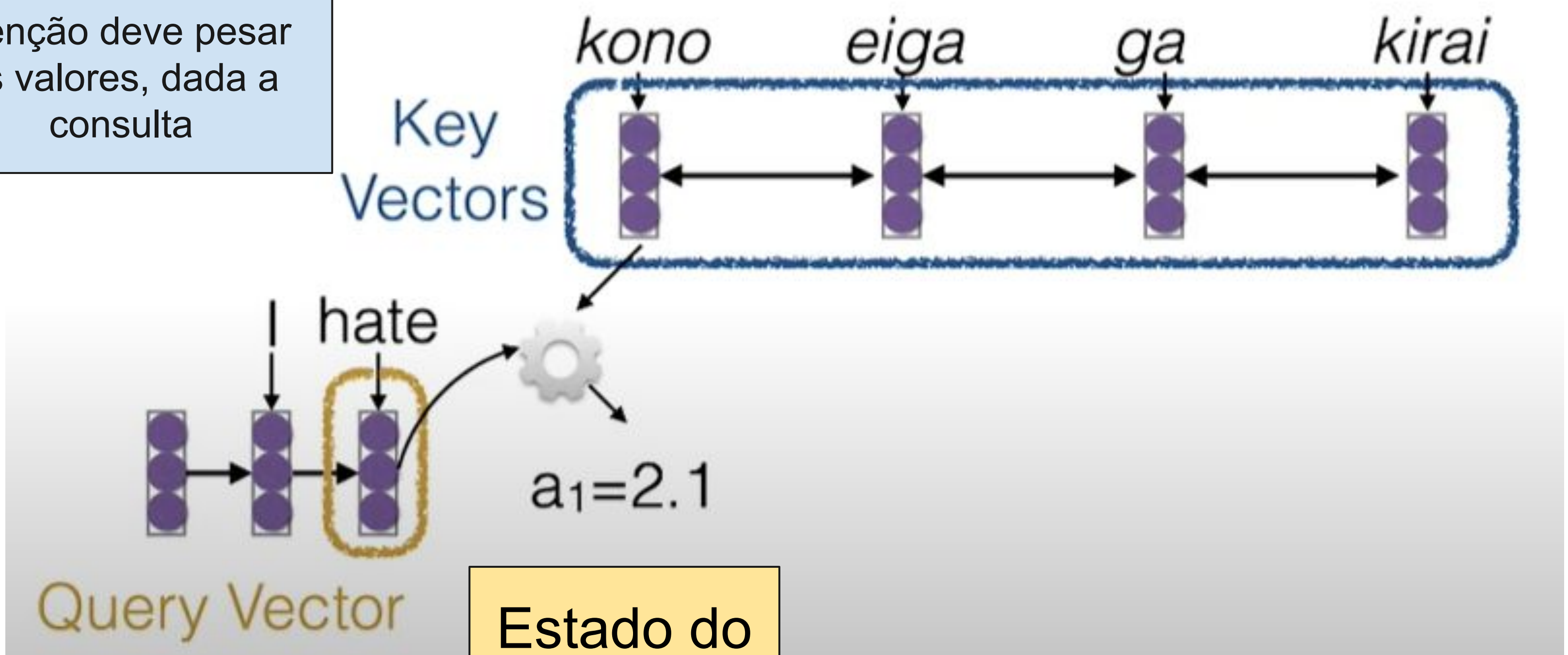


*from CMU Neural Nets for NLP

Attention

ponte entre as queries (o que procuramos) e o valor (o que queremos): elas vão apontar em como a atenção deve pesar os valores, dada a consulta

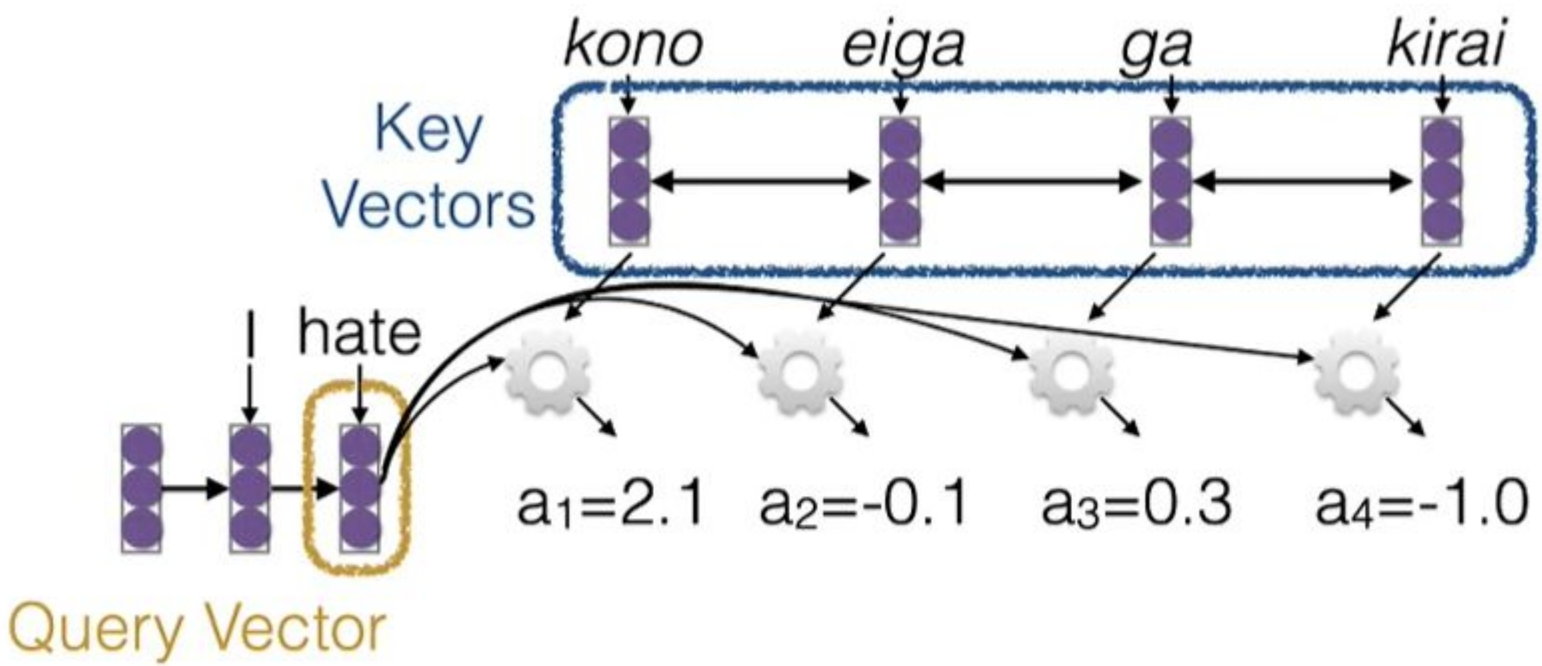
Estados do encoder



Estado do decoder

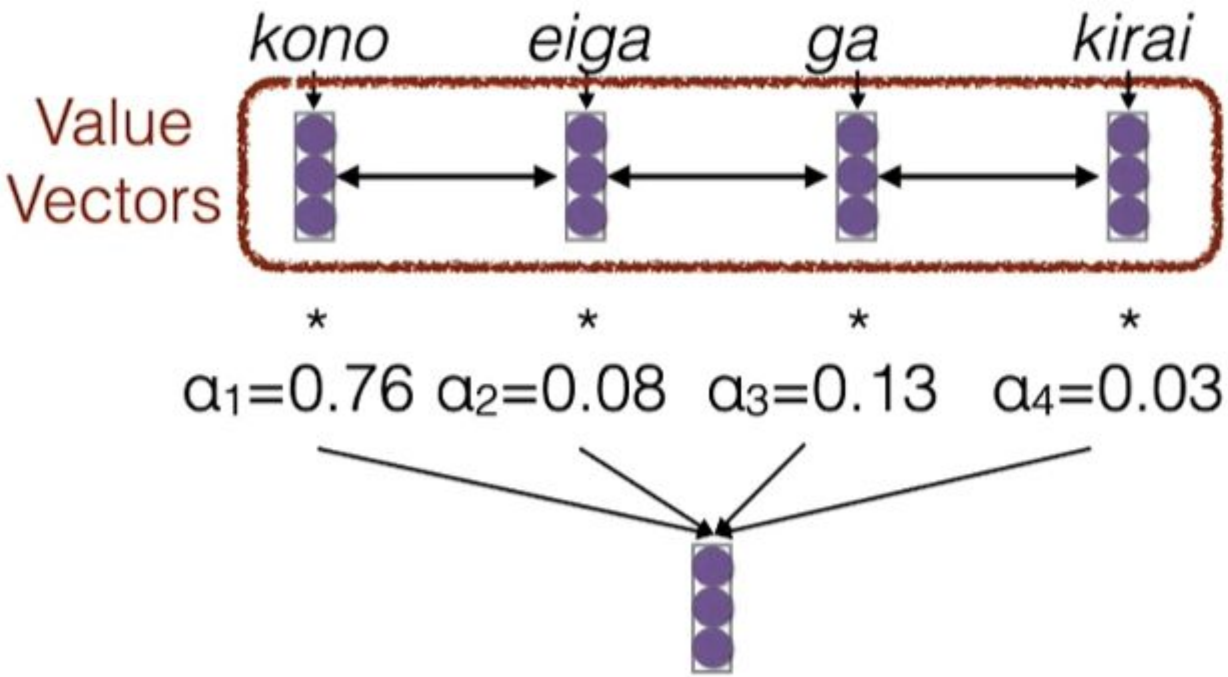
*from CMU Neural Nets for NLP

Attention



*from CMU Neural Nets for NLP

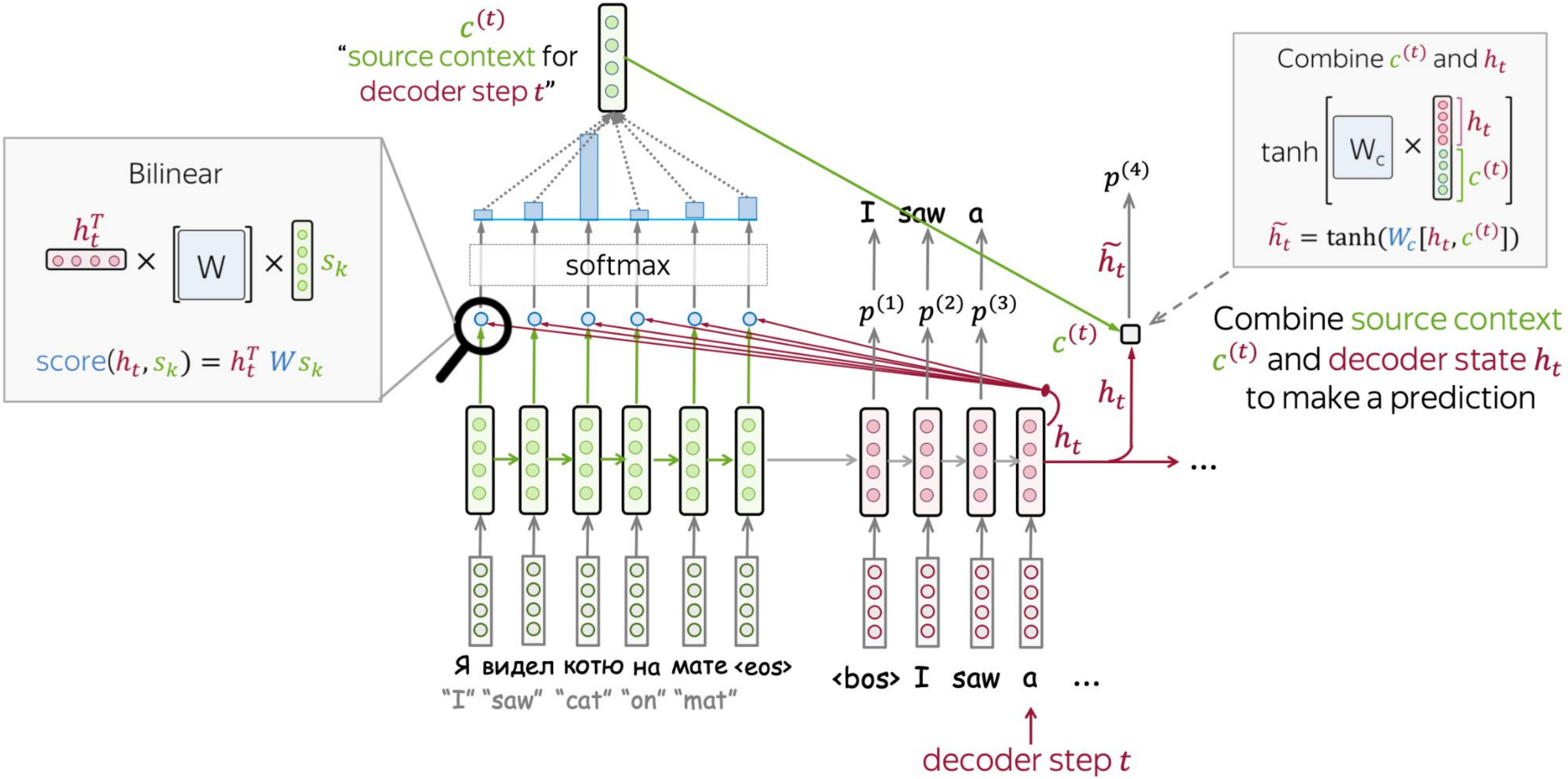
Attention



Atenção

http://jalammar.github.io/visualizing-neural-machine-translati
on-mechanics-of-seq2seq-models-with-attention/

Modelo de Luong



Effective Approaches to Attention-based Neural Machine Translation , 2015

Alinhamento

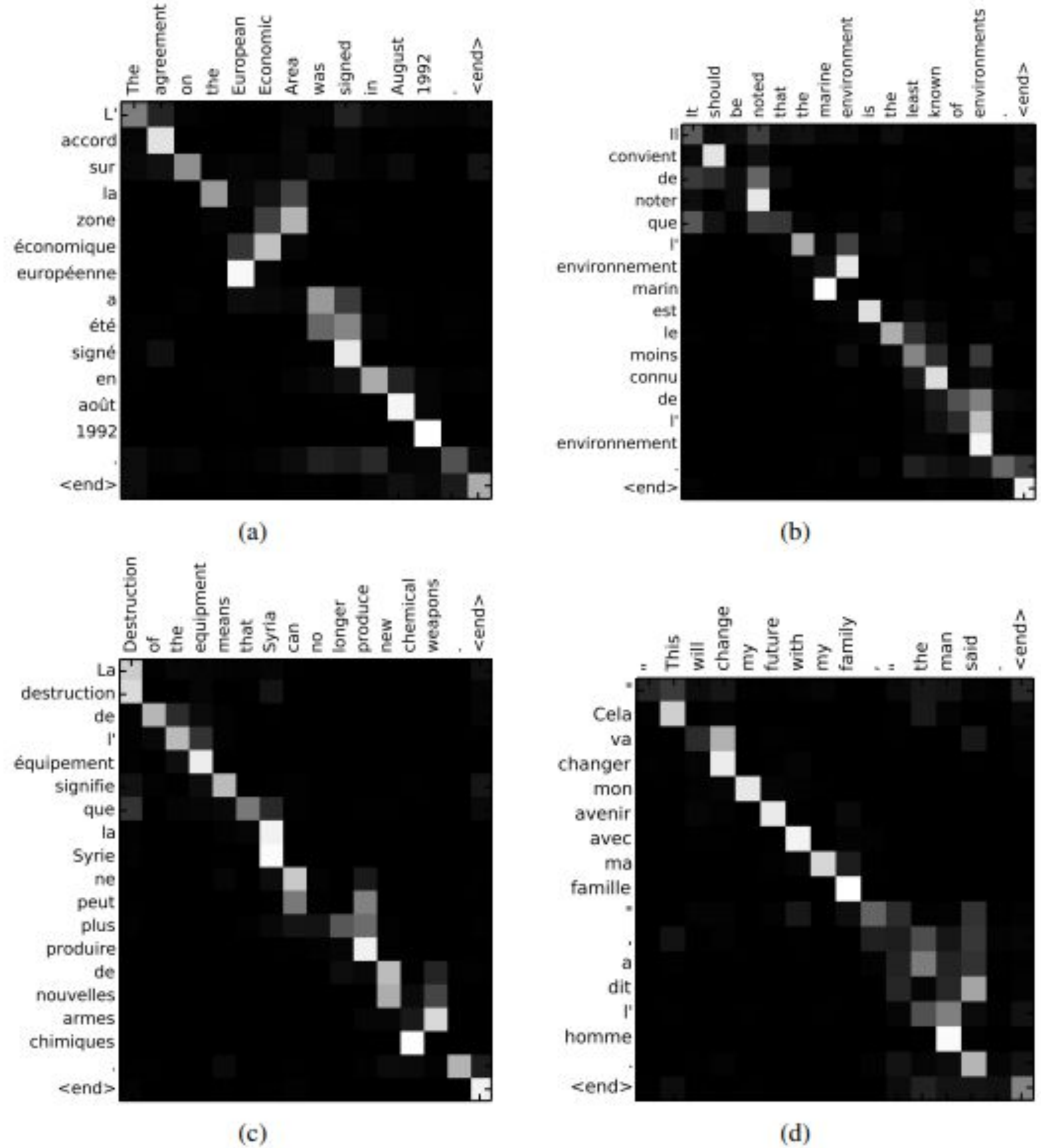
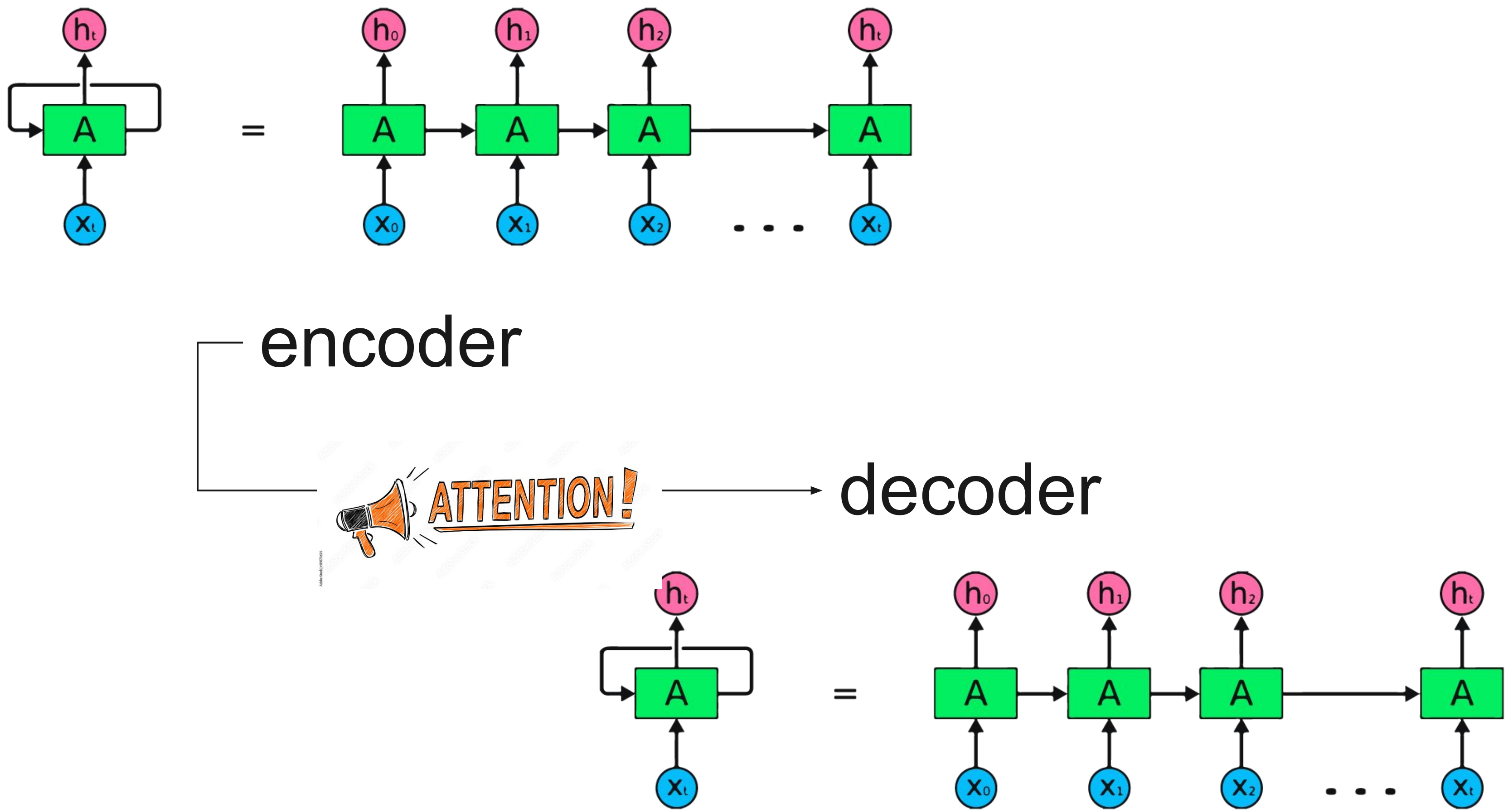


Figure 3: Four sample alignments found by RNNsearch-50. The x-axis and y-axis of each plot correspond to the words in the source sentence (English) and the generated translation (French), respectively. Each pixel shows the weight α_{ij} of the annotation of the j-th source word for the i-th target word (see Eq. (6)), in grayscale (0: black, 1: white). (a) an arbitrary sentence. (b–d) three randomly selected samples among the sentences without any unknown words and of length between 10 and 20 words from the test set.

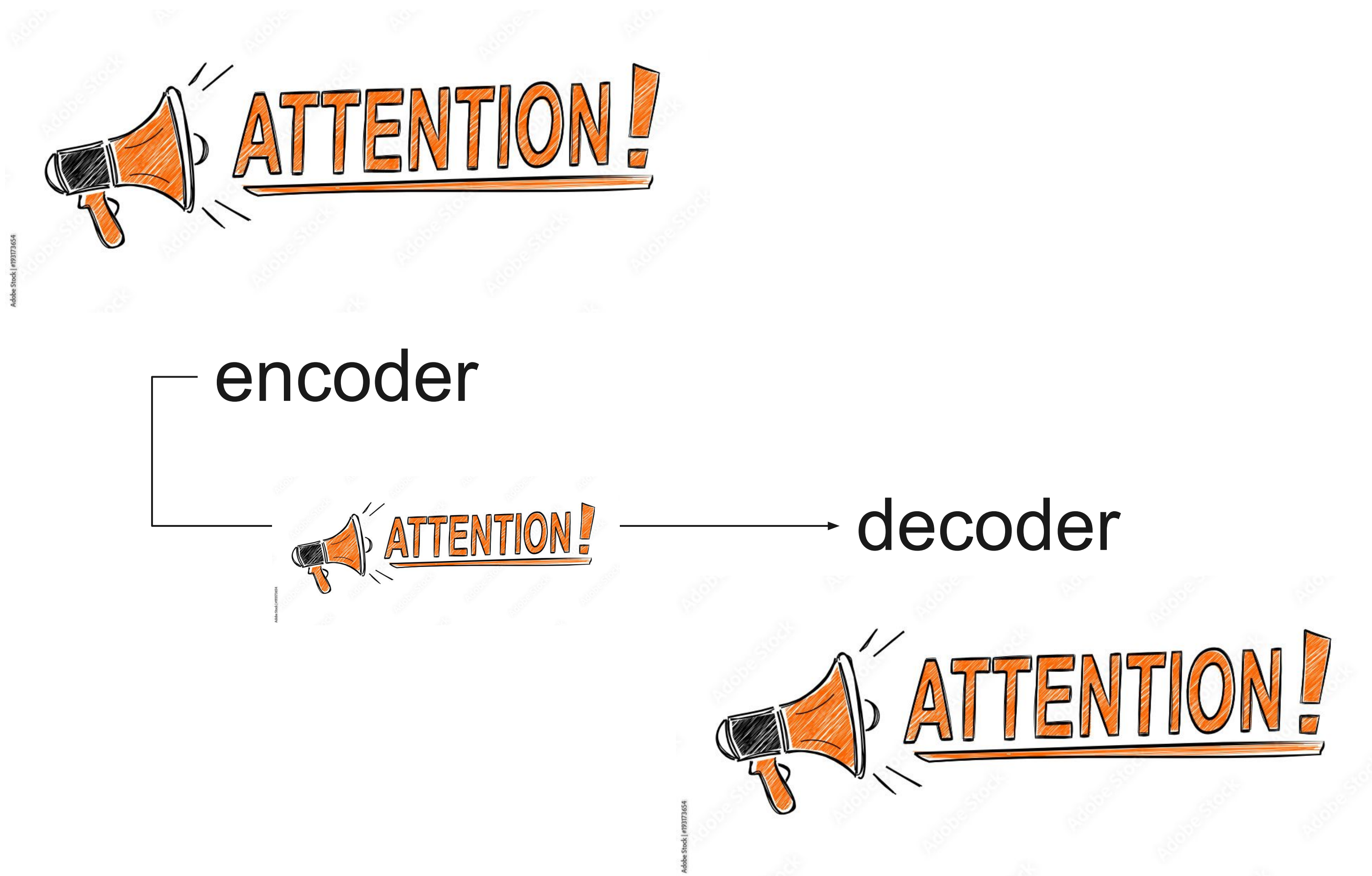
Bahdahanu et al. (2015)]

Atenção

All you need is attention



All you need is attention



All you need is attention

Todos os tokens
Olham uns para os outros
Atualizam representações

N X

encoder

decoder

self-attention

All you need is attention

SELF

Todos os tokens
Olham uns para os outros
Atualizam representações

N X

encoder

decoder

Token na saída corrente
Olha para os tokens anteriores
Olha para as representações da entrada
Atualiza representações

N X

Google animation

Self-attention

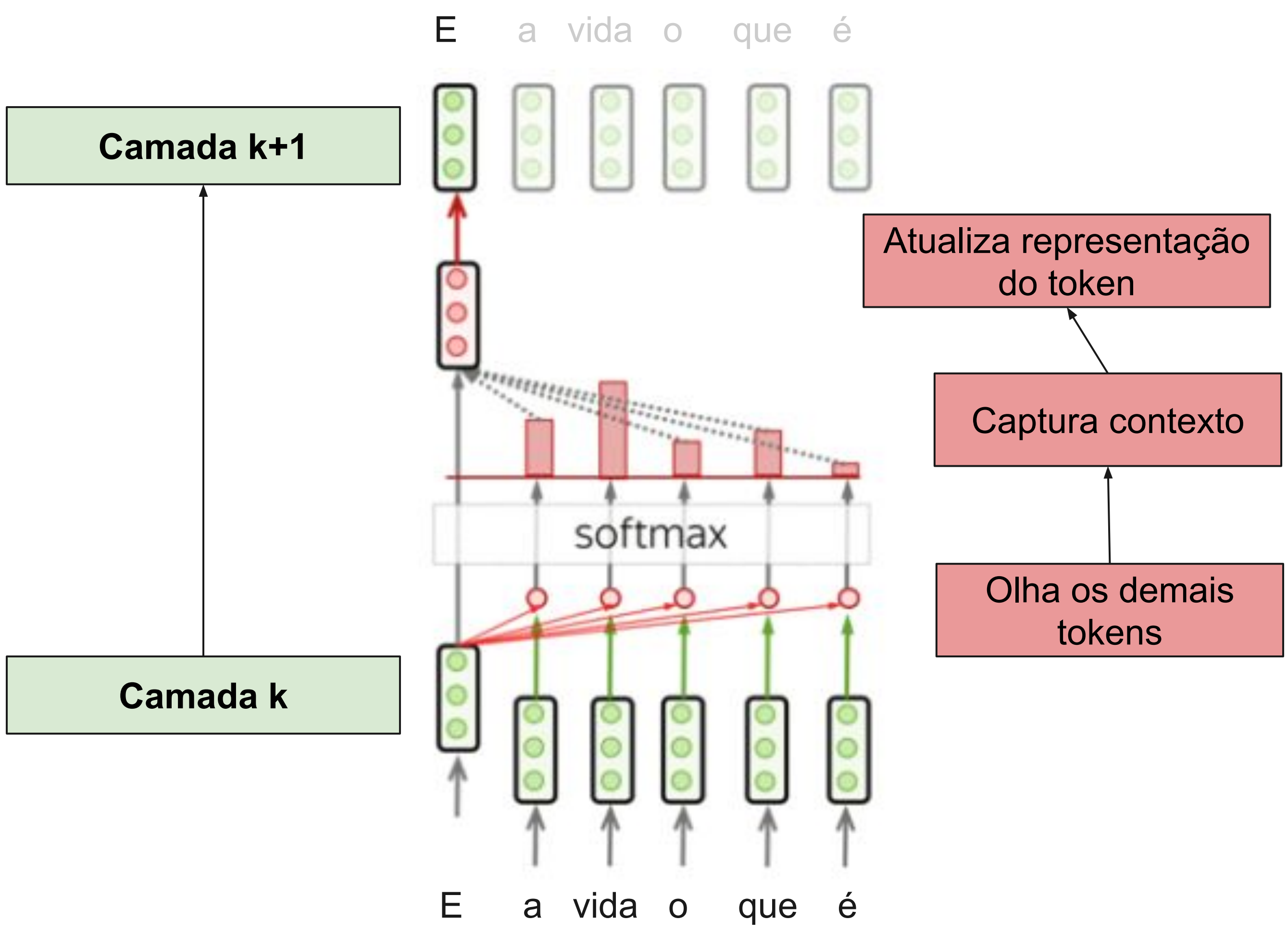


Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Self-attention,Q,K, V

Três "representações" para cada token

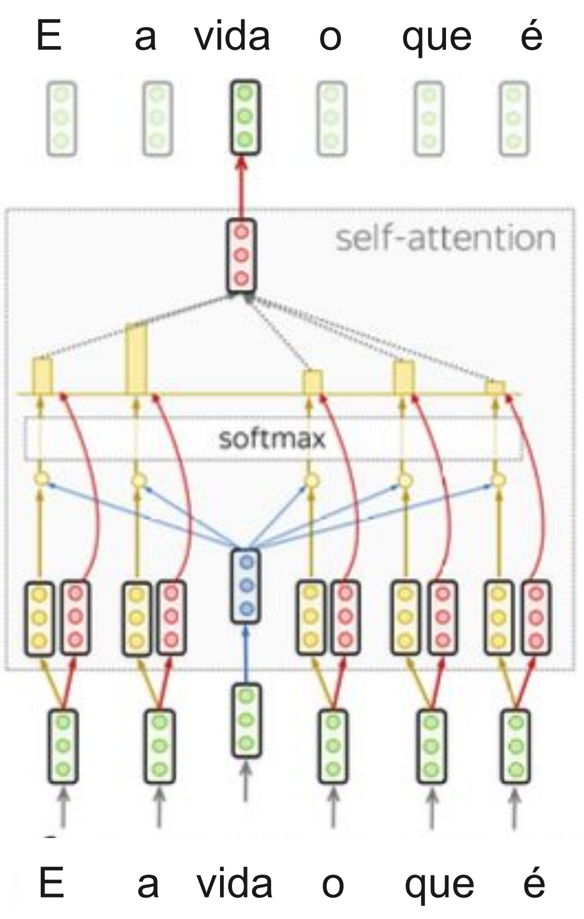
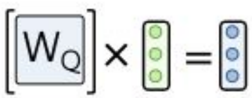


Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Self-attention,Q,K, V

Três "representações" para cada token



Query: "*Você tem essa informação?*"

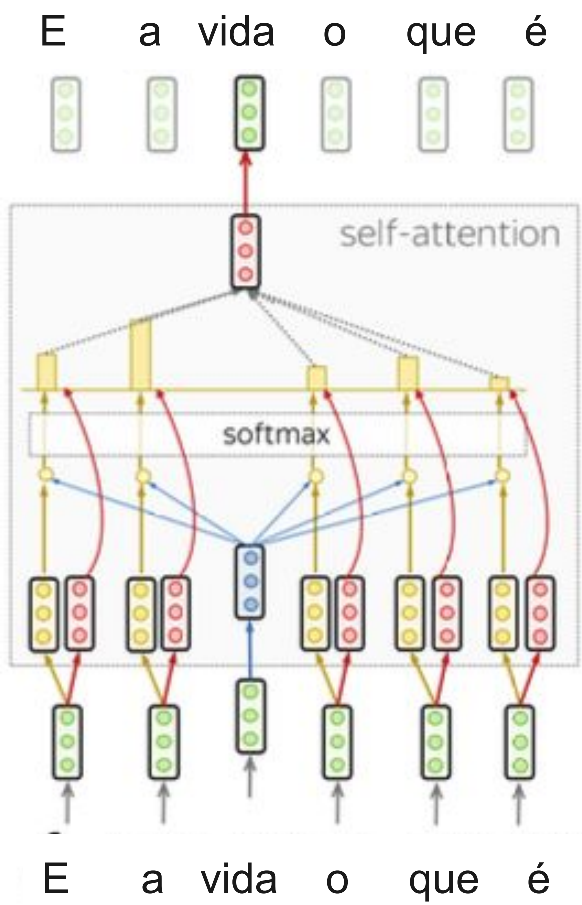
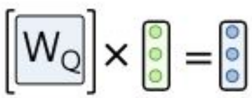


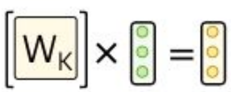
Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Self-attention,Q,K, V

Três "representações" para cada token



Query: "*Você tem essa informação?*"



Key: "Opa, eu tenho *a informação que você quer. Me dê relevância (mais peso)"*

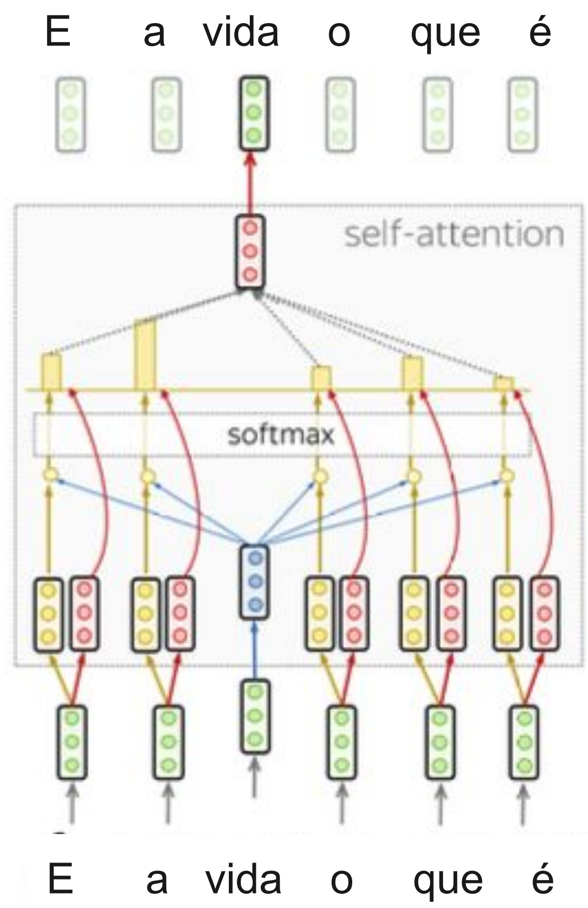
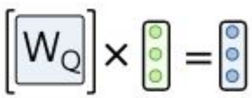


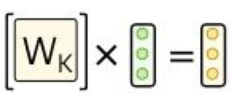
Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Self-attention,Q,K, V

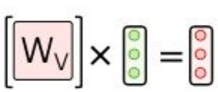
Três "representações" para cada token



Query: "*Você tem essa informação?"*



Key: "Opa, eu tenho *a informação que você quer. Me dê relevância (mais peso)"*



Value: "*Aqui está a informação."*

Soma ponderada

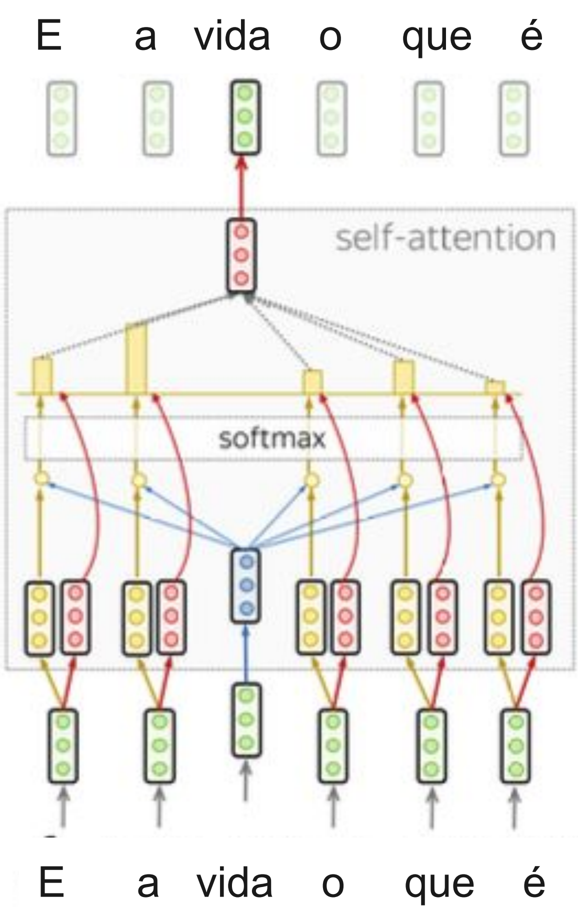
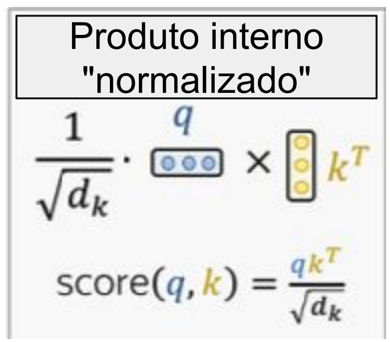


Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Self-attention,Q,K, V

Três "representações" para cada token



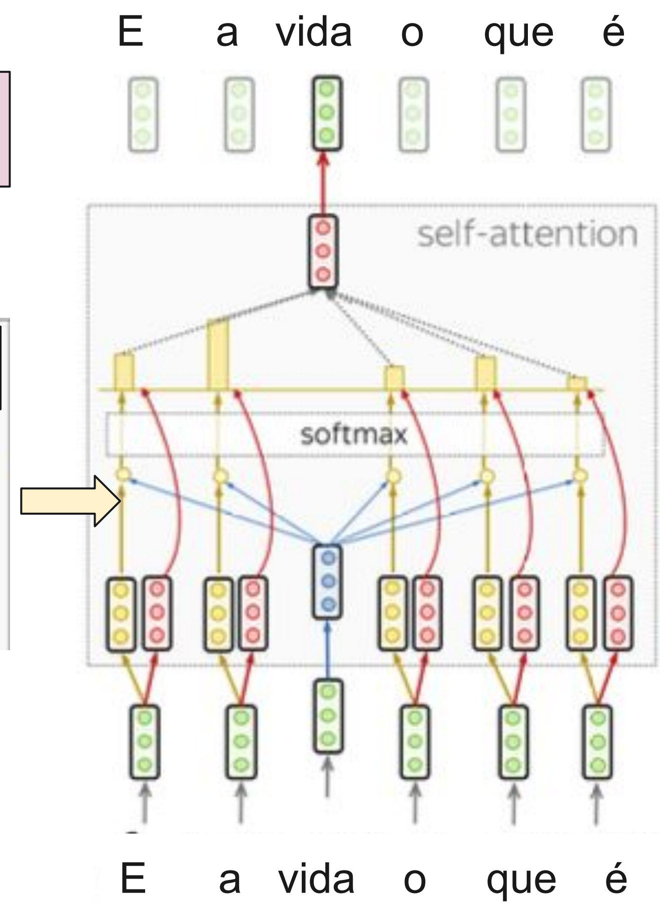


Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Self-attention,Q,K, V

Três "representações" para cada token

$$Attention(q,k,v) = \overbrace{softmax\left(\frac{qk^T}{\sqrt{d_k}}\right)}^{\text{Pesos da atenção}} v$$

de

para

Dimensão de K,V

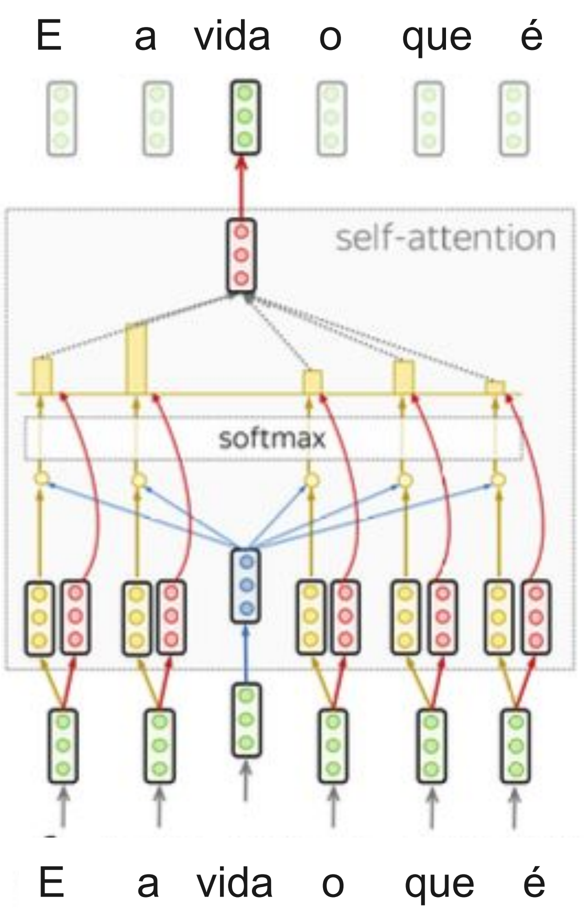


Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Atenção mascarada do decoder

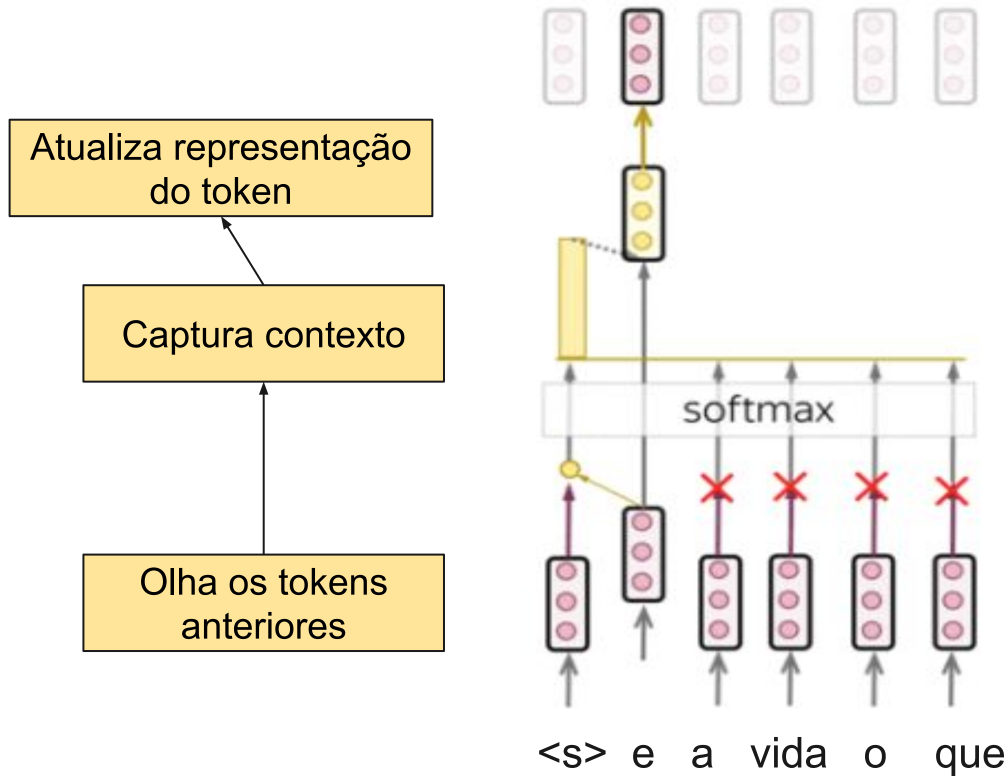
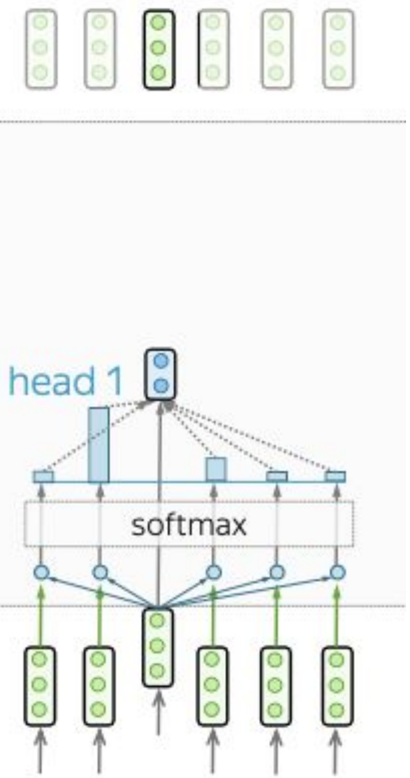
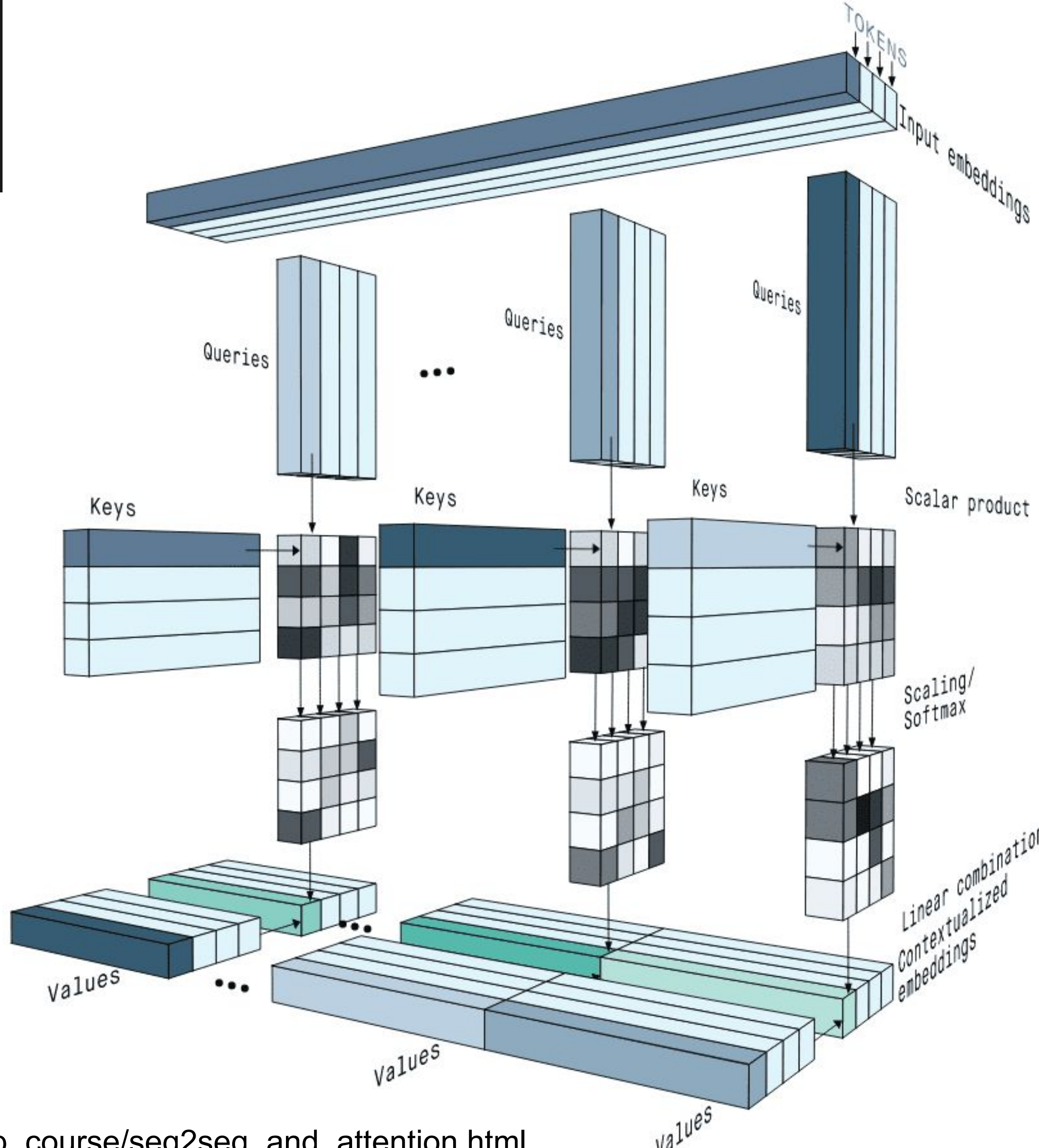


Figura de https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Multi-Head attention

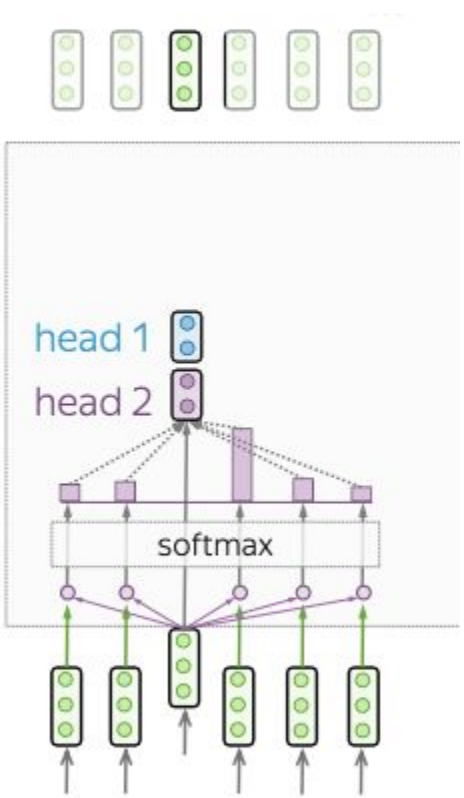


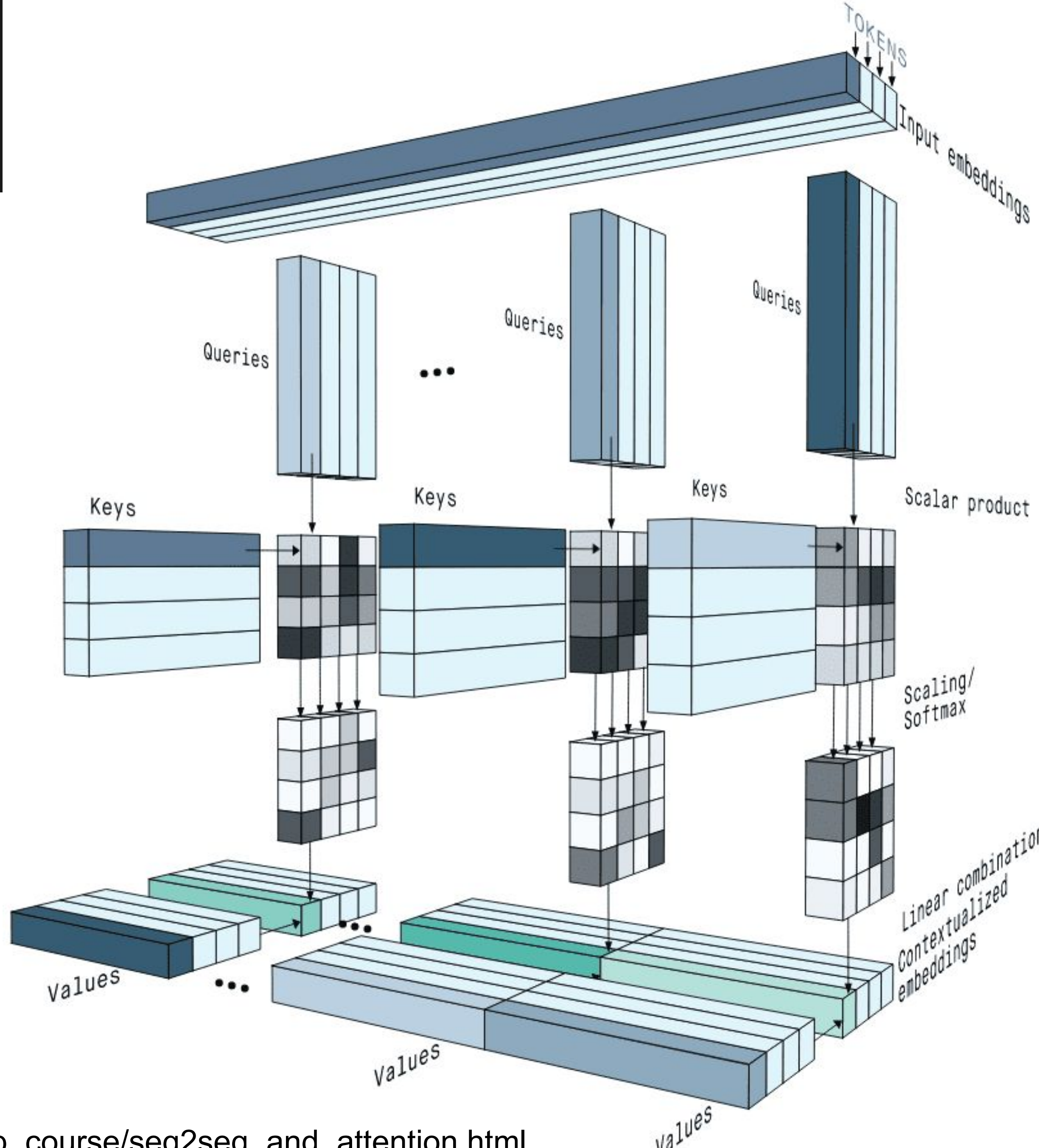


Fonte:https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Fonte: https://peltarion.com/blog/data-science/self-attention-video

Multi-Head attention

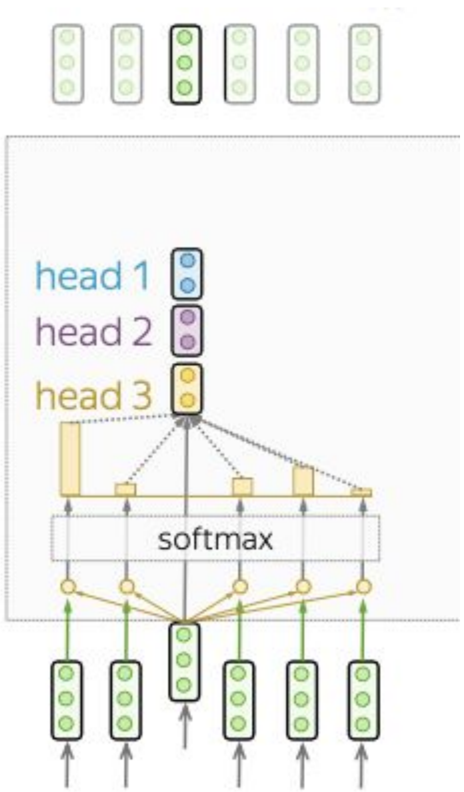


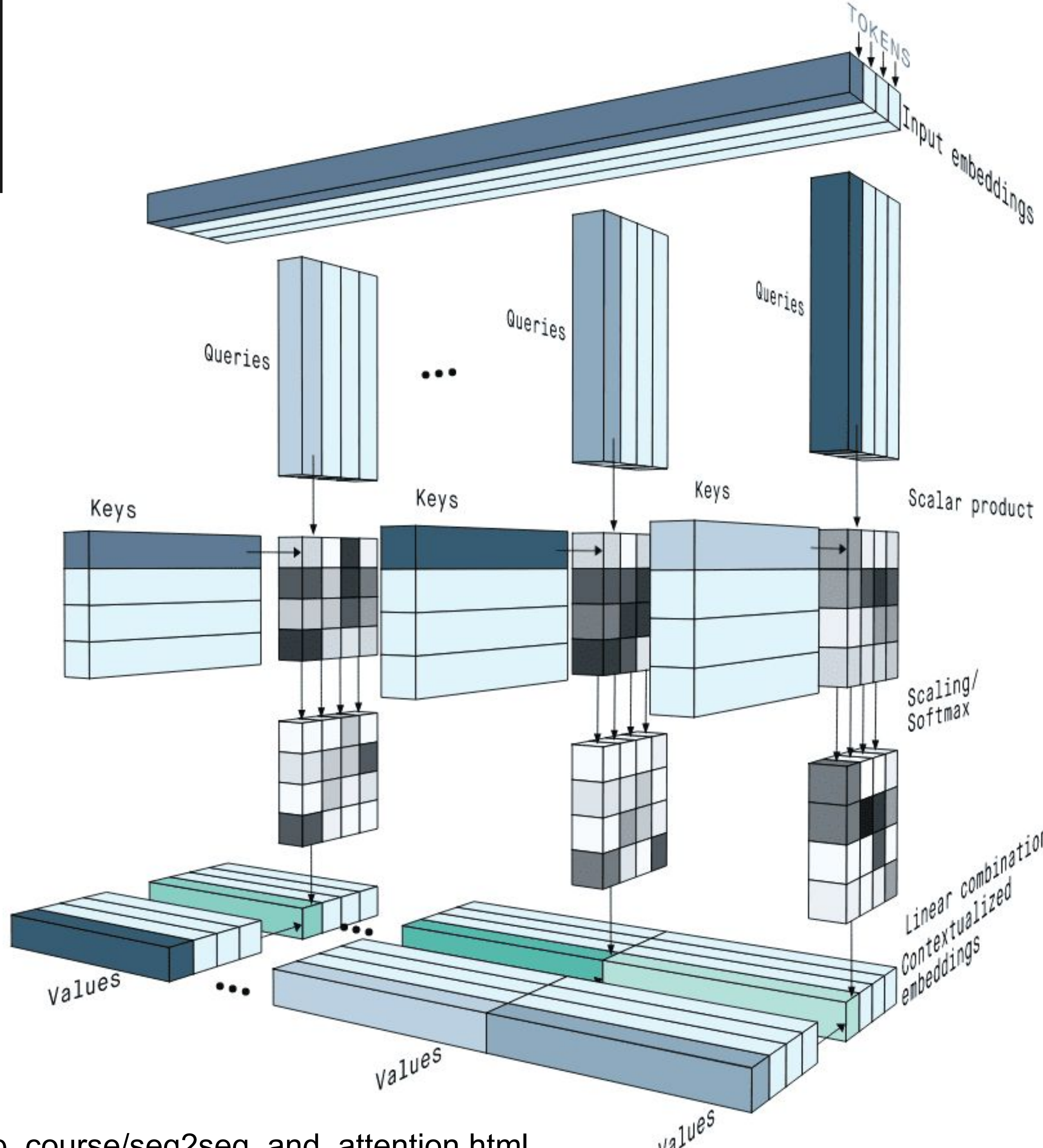


Fonte:https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Fonte: https://peltarion.com/blog/data-science/self-attention-video

Multi-Head attention

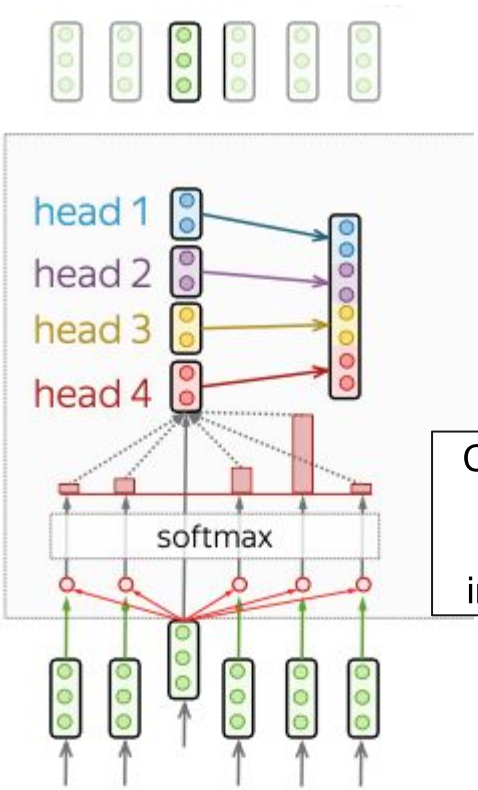




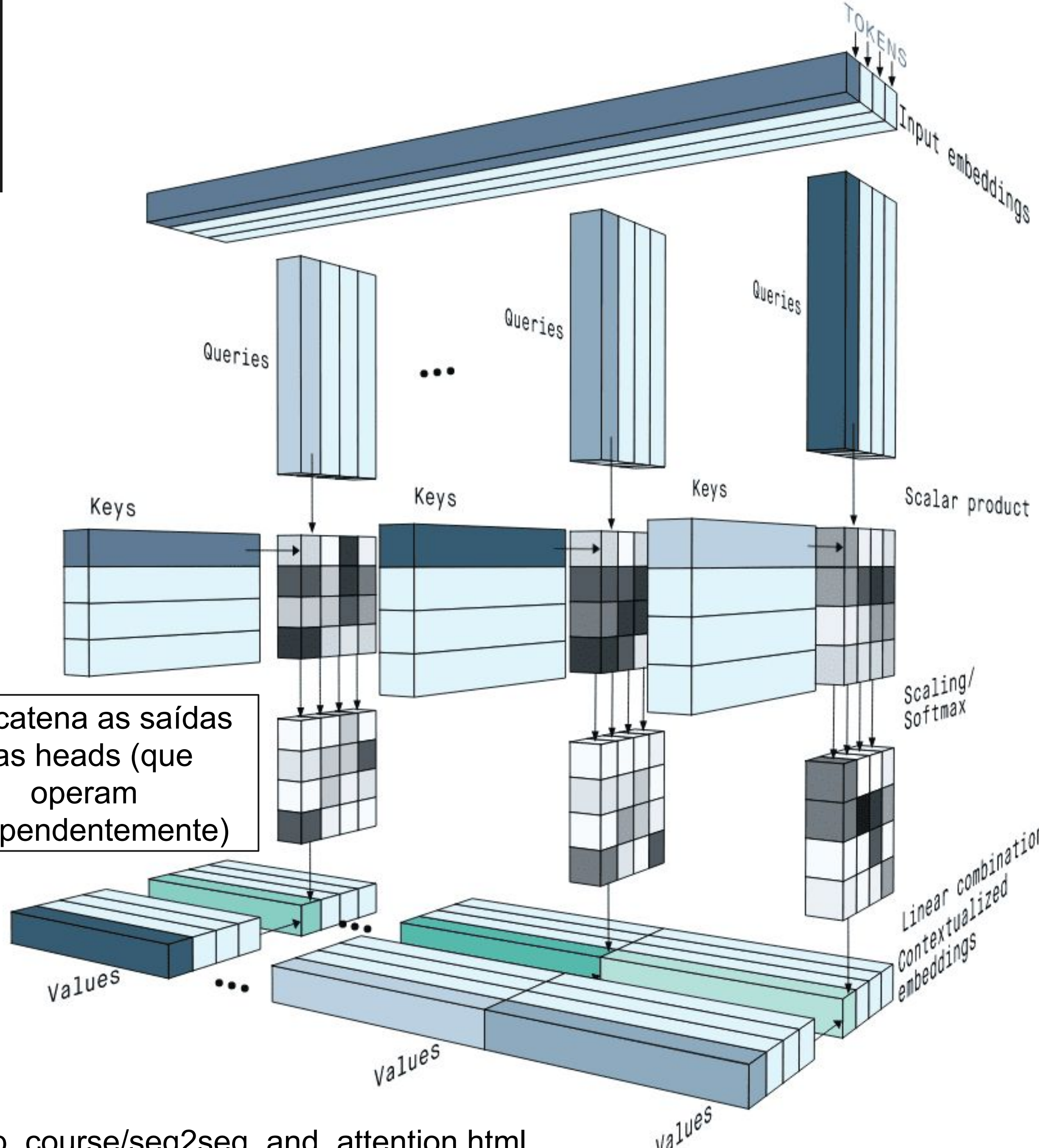
Fonte:https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Fonte: https://peltarion.com/blog/data-science/self-attention-video

Multi-Head attention



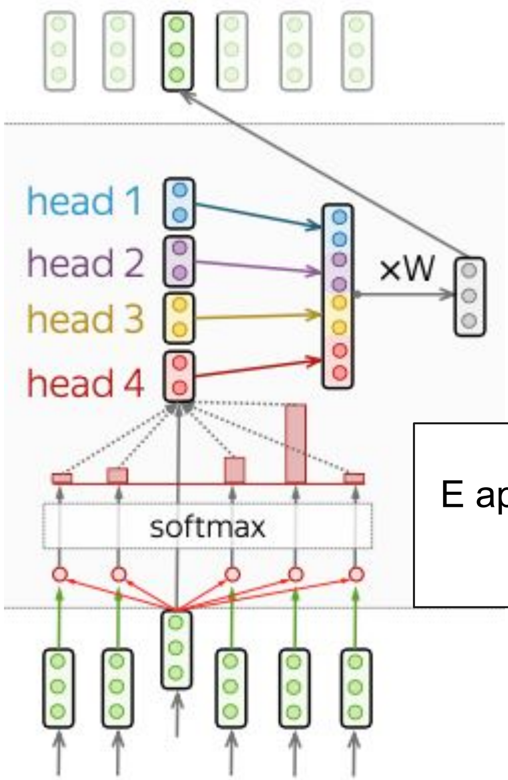
Concatena as saídas das heads (que operam independentemente)



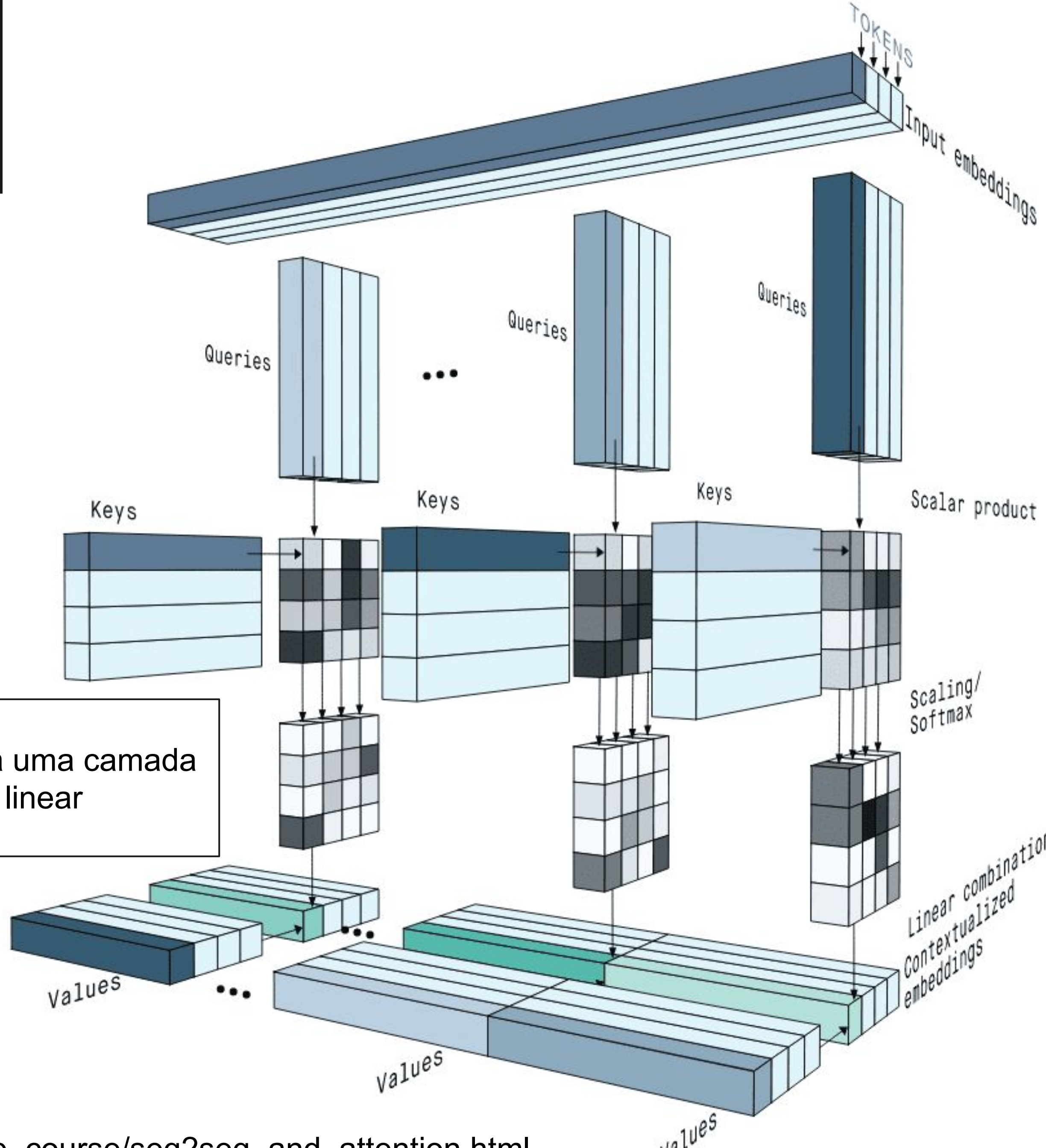
Fonte:https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Fonte: https://peltarion.com/blog/data-science/self-attention-video

Multi-Head attention



E aplica uma camada linear



Fonte:https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Fonte: https://peltarion.com/blog/data-science/self-attention-video

All you need is attention

encoder

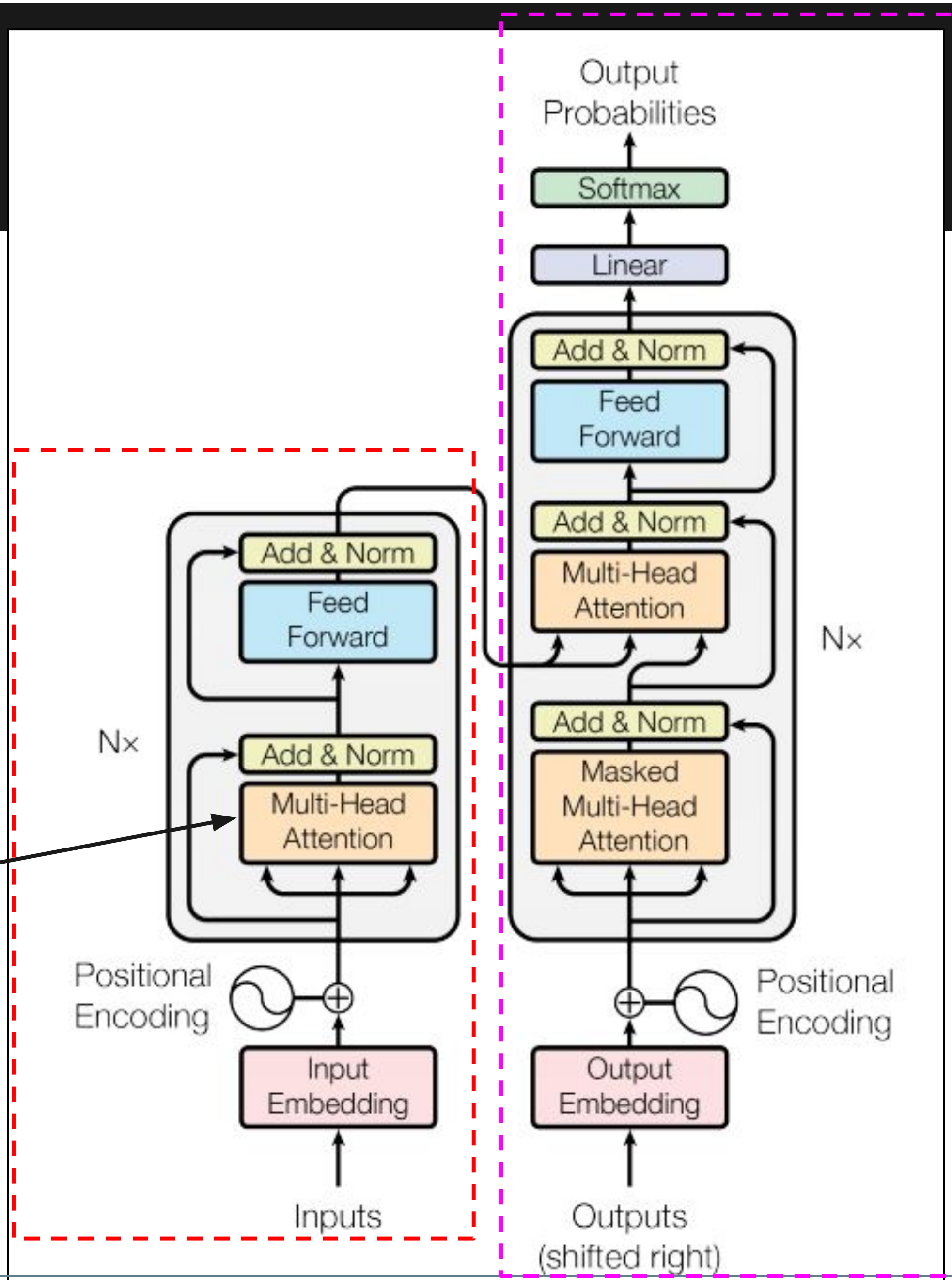
decoder

Só vantagens?

a) Ver todos os tokens "de uma vez".
b) Paralelizar por tokens.
c) Ordem dos tokens.
d) Tamanho da sequência de entrada.

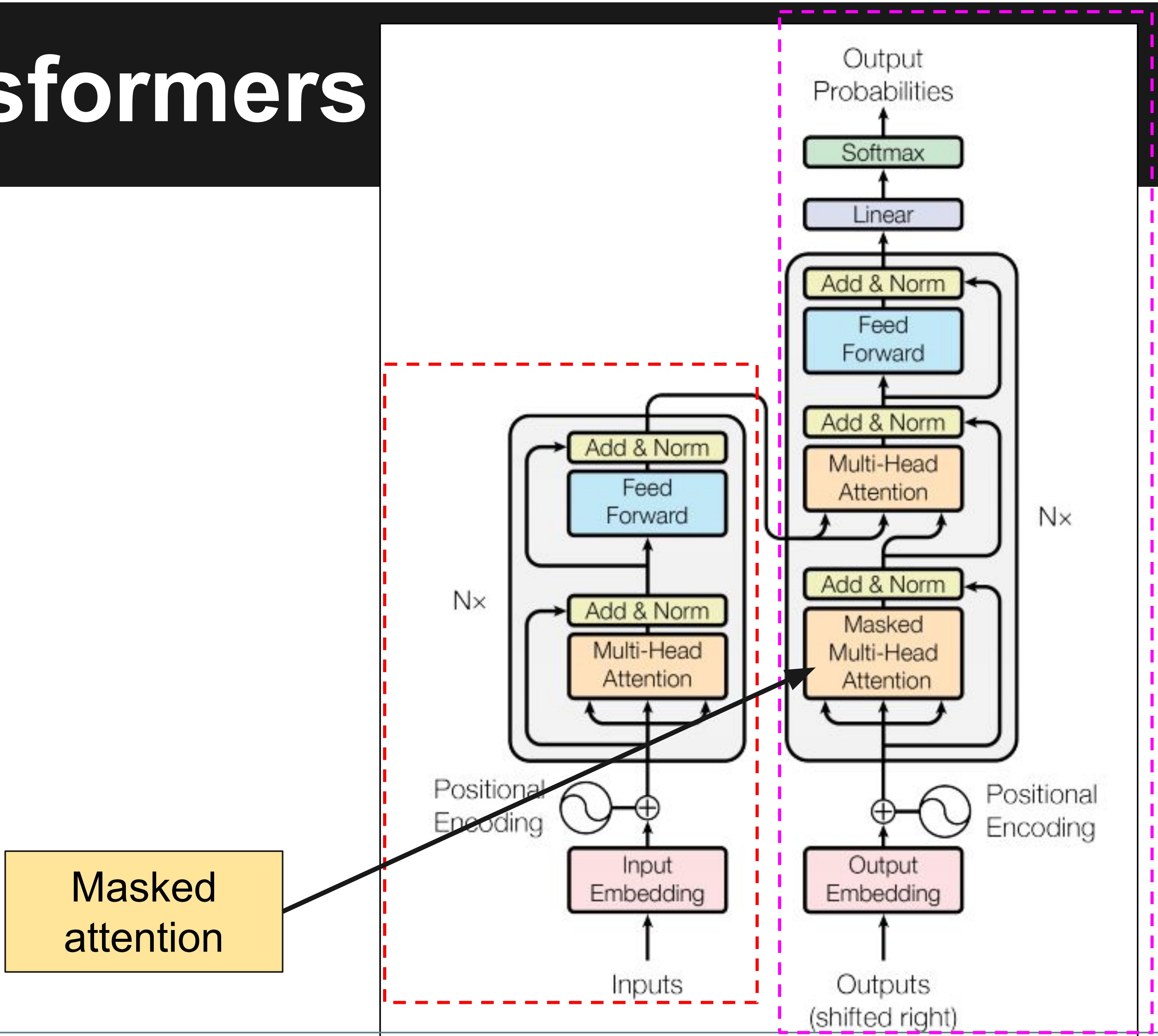
Transformers

Self attention



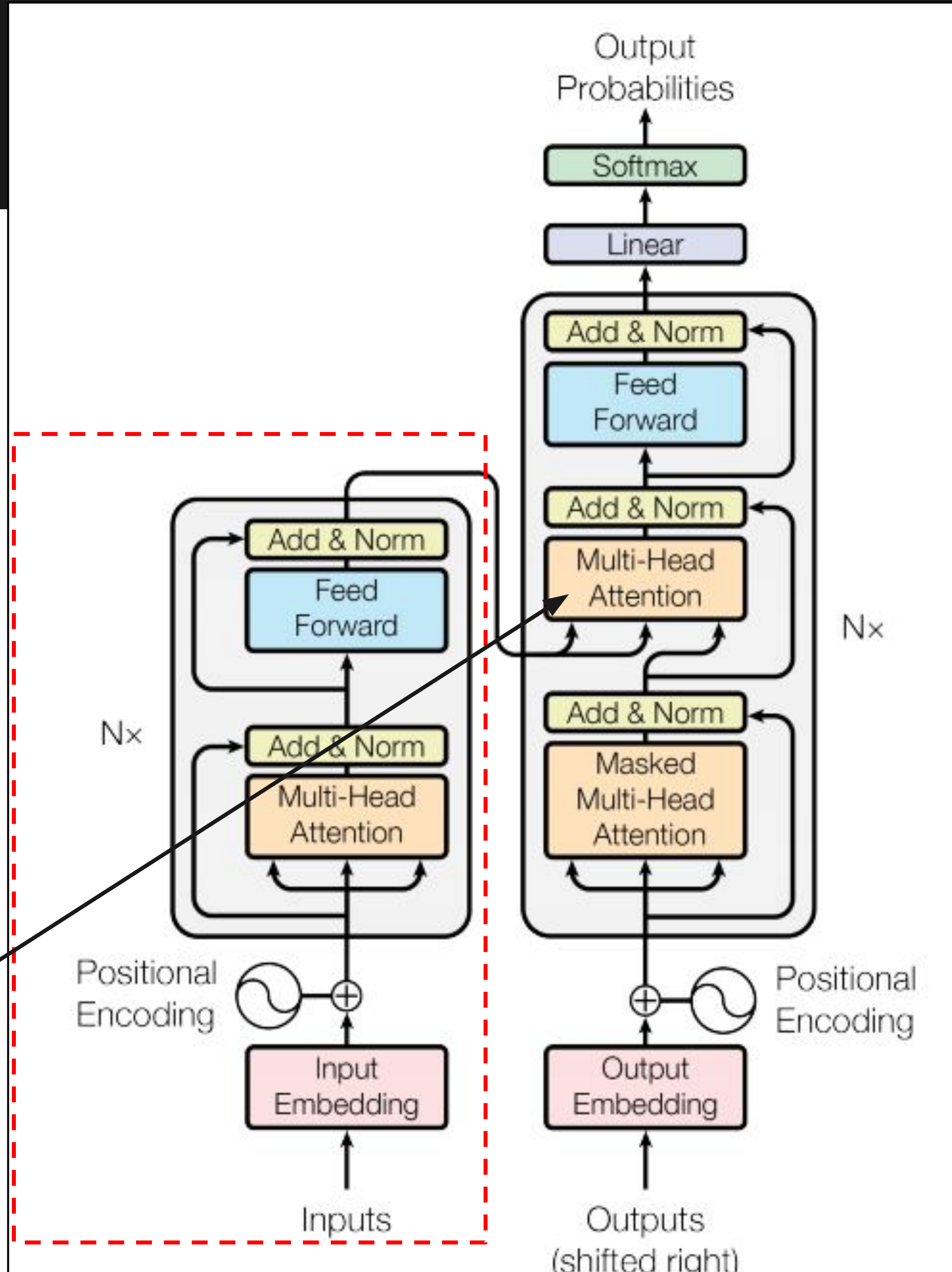
*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). **Attention is all you need**. In *Advances in neural information processing systems* (pp. 5998-6008).

Transformers



*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). **Attention is all you need**. In *Advances in neural information processing systems* (pp. 5998-6008).

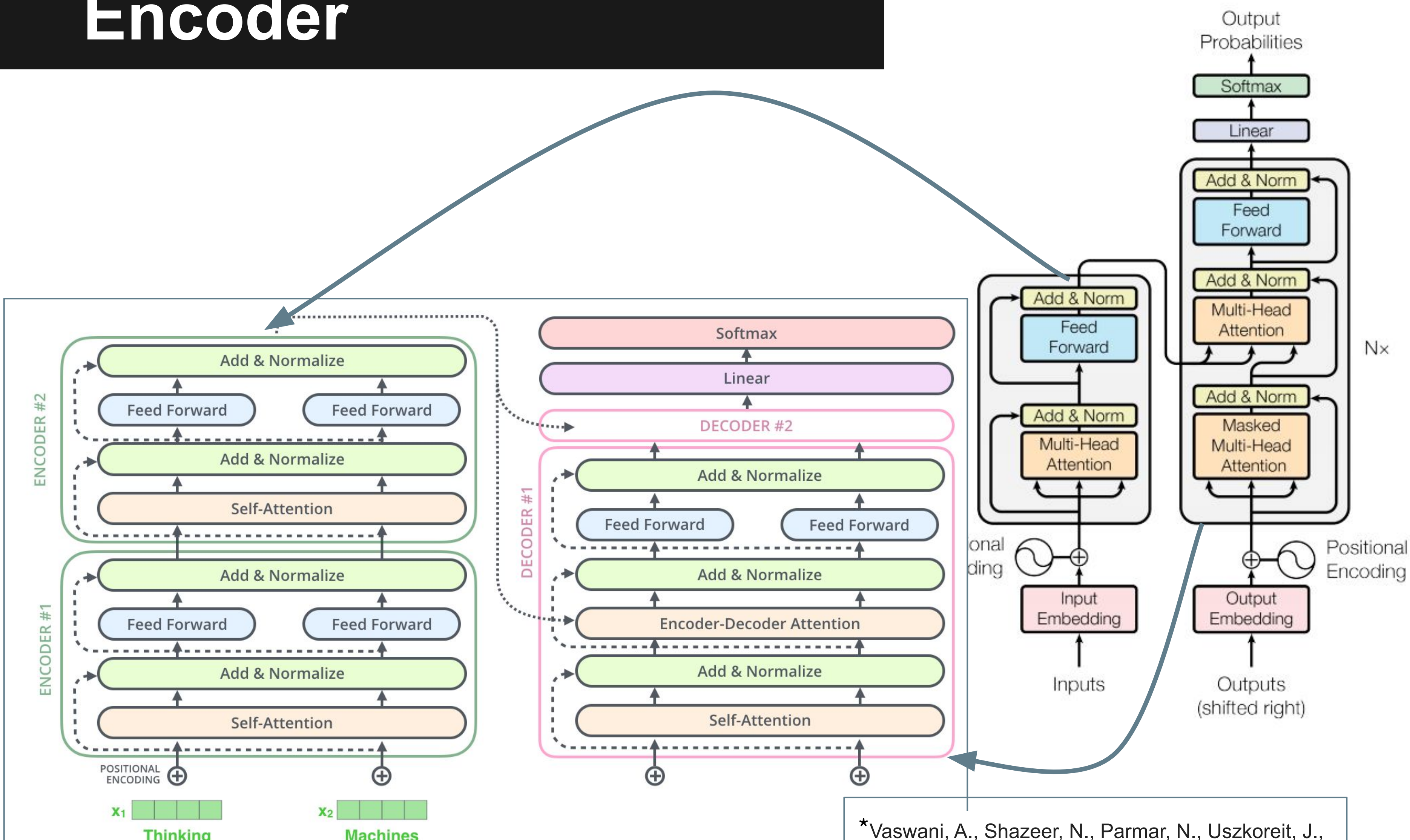
Transformers



Decoder-encoder attention

*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). **Attention is all you need**. In *Advances in neural information processing systems* (pp. 5998-6008).

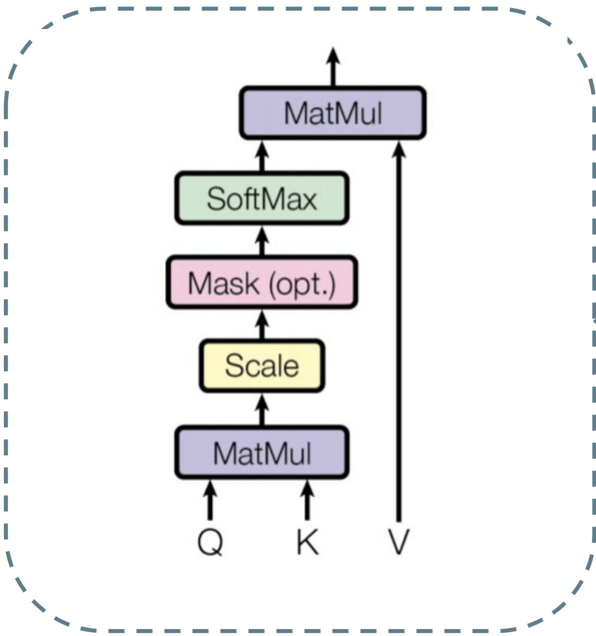
Encoder



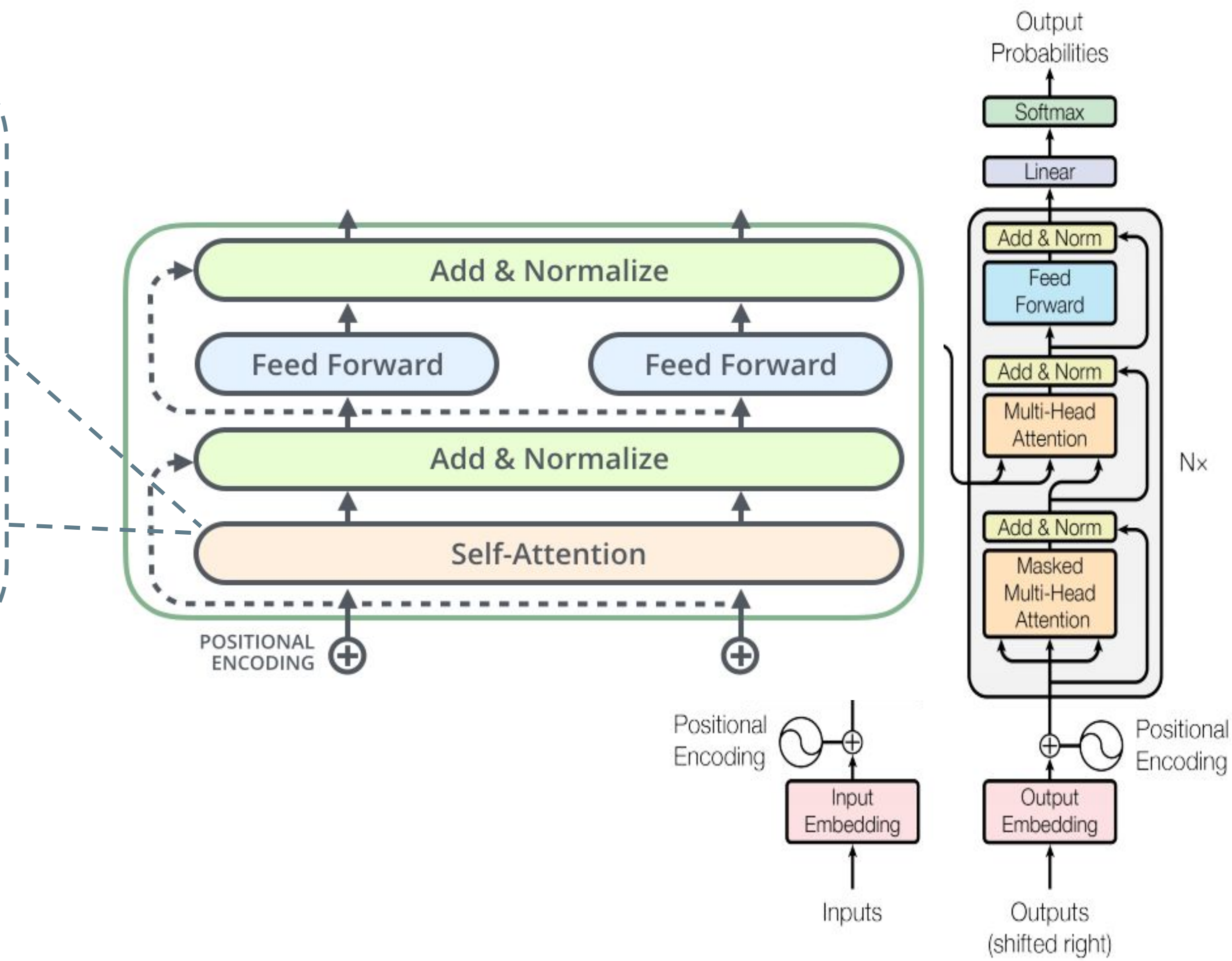
*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). **Attention is all you need**. In *Advances in neural information processing systems* (pp. 5998-6008).

* http://jalammar.github.io/illustrated-transformer/

Self-attention

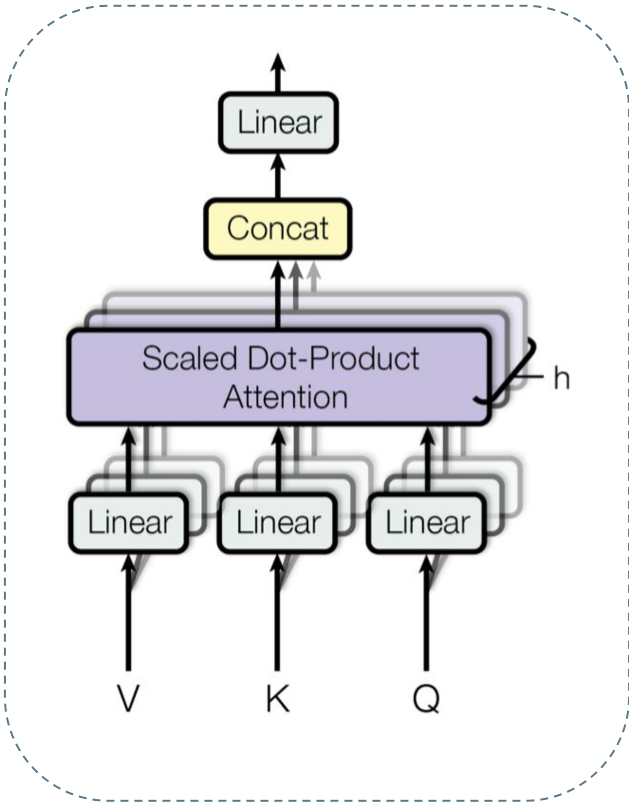


Scaled Dot-Product attention

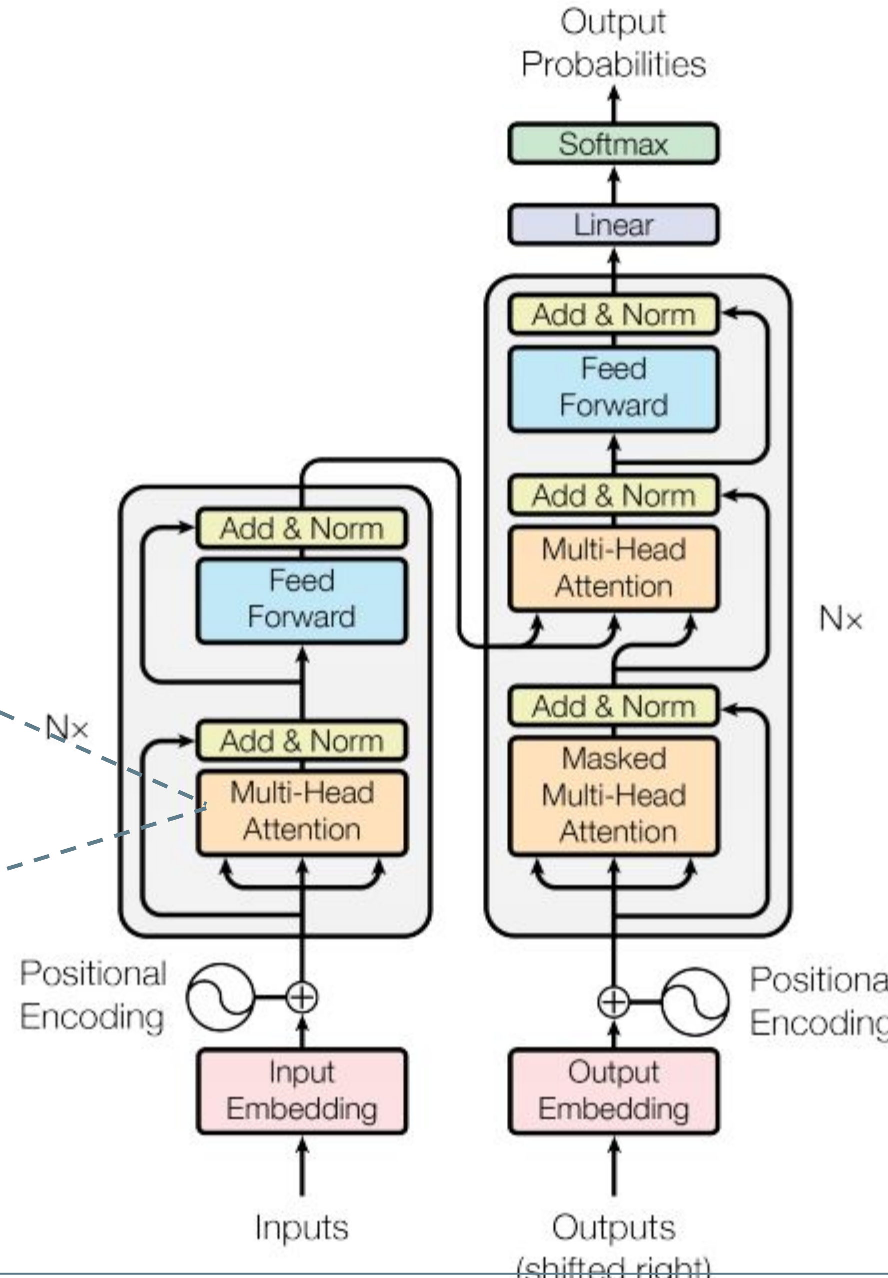


*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

Multi-head attention

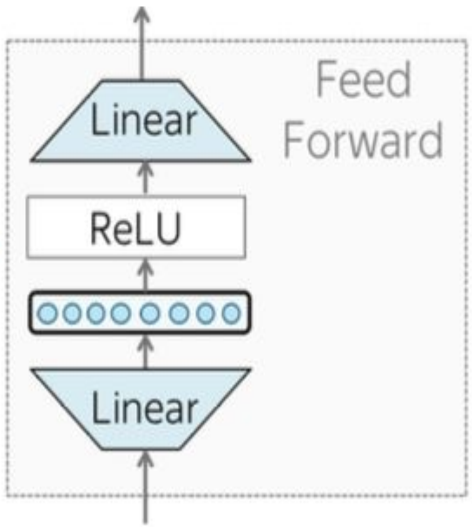


Multi-head attention



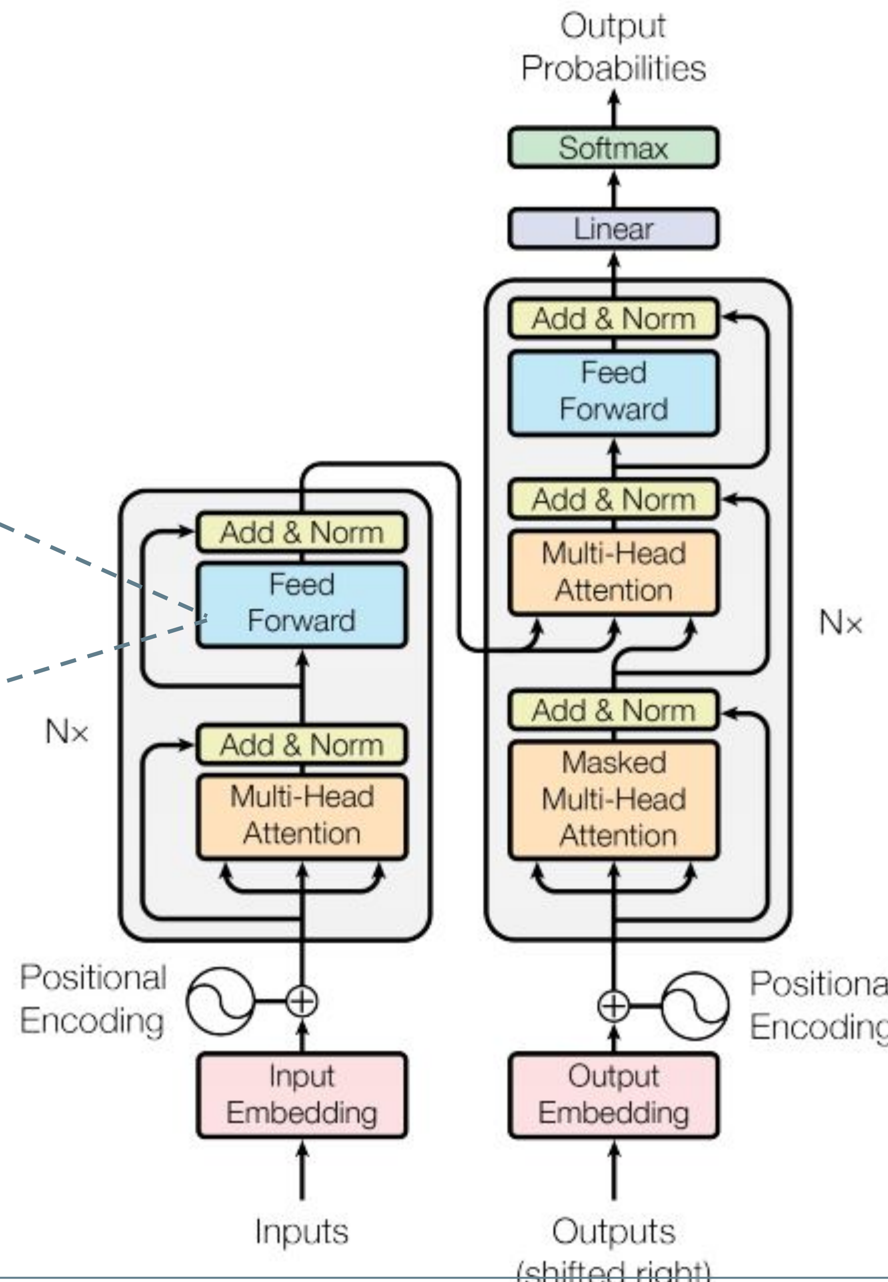
*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

Rede Neural



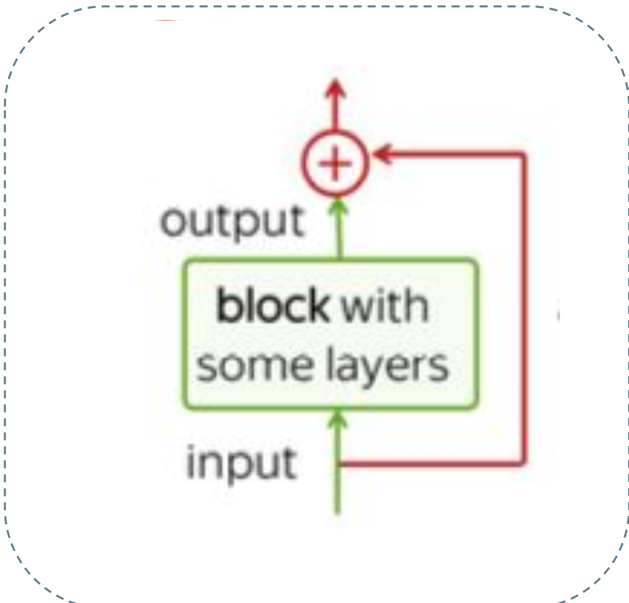
$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Camada de rede neural

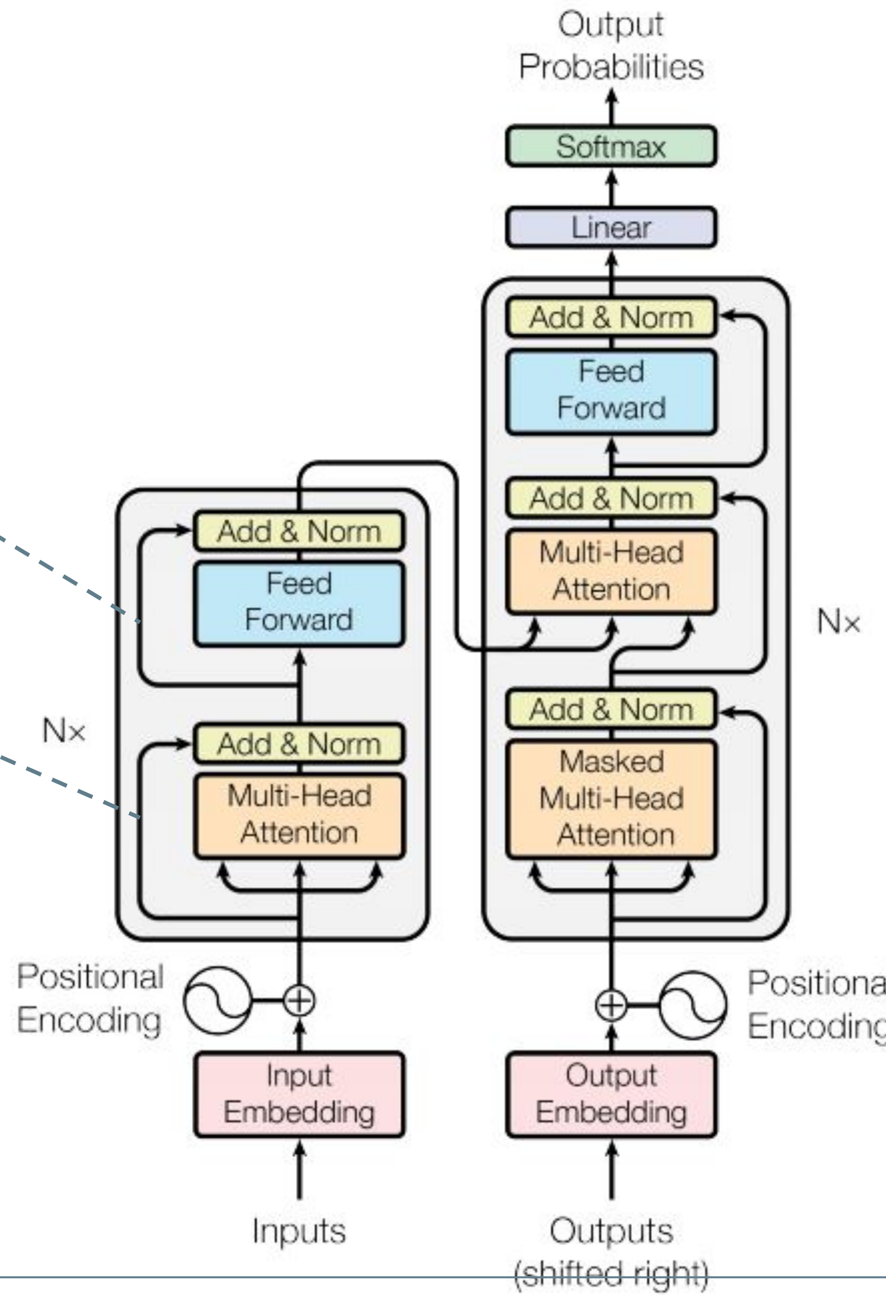


*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

Residual



Conexões residuais



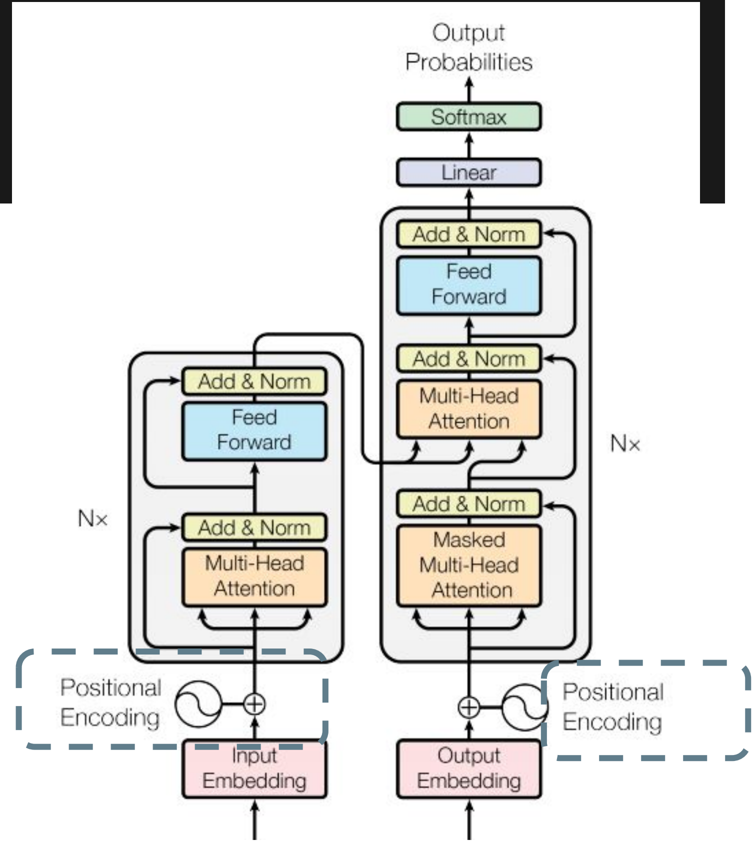
*Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

Positional encoding

Posição normalizada

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$

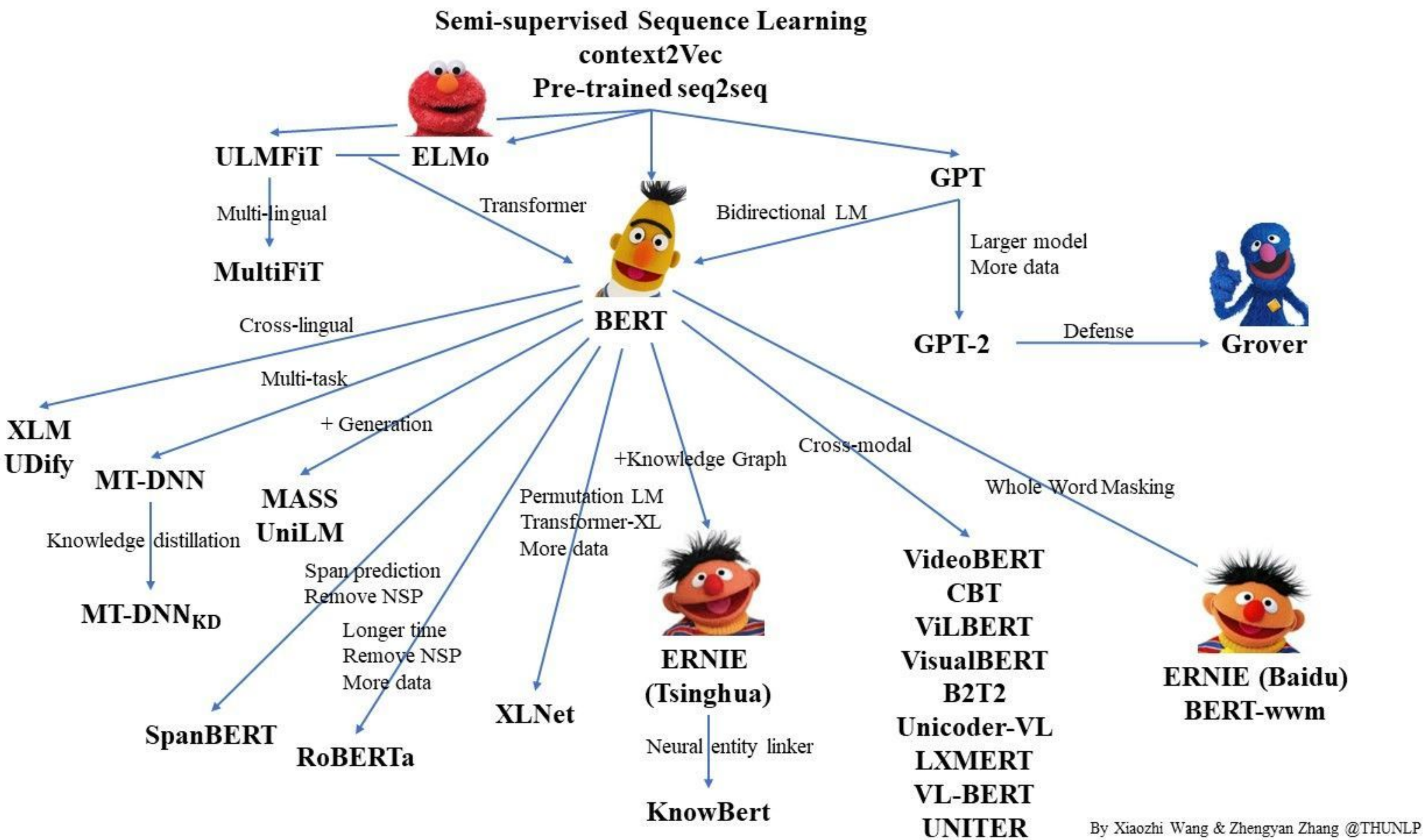
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$



Attention is all that you need?

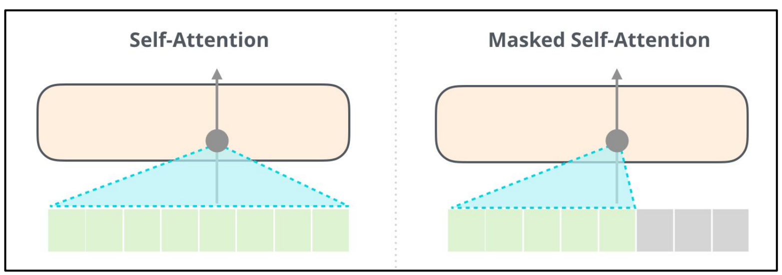
- Inferência lenta
- Hardware pesado
- Não necessariamente melhores que RNN
- Não convergem com poucos dados

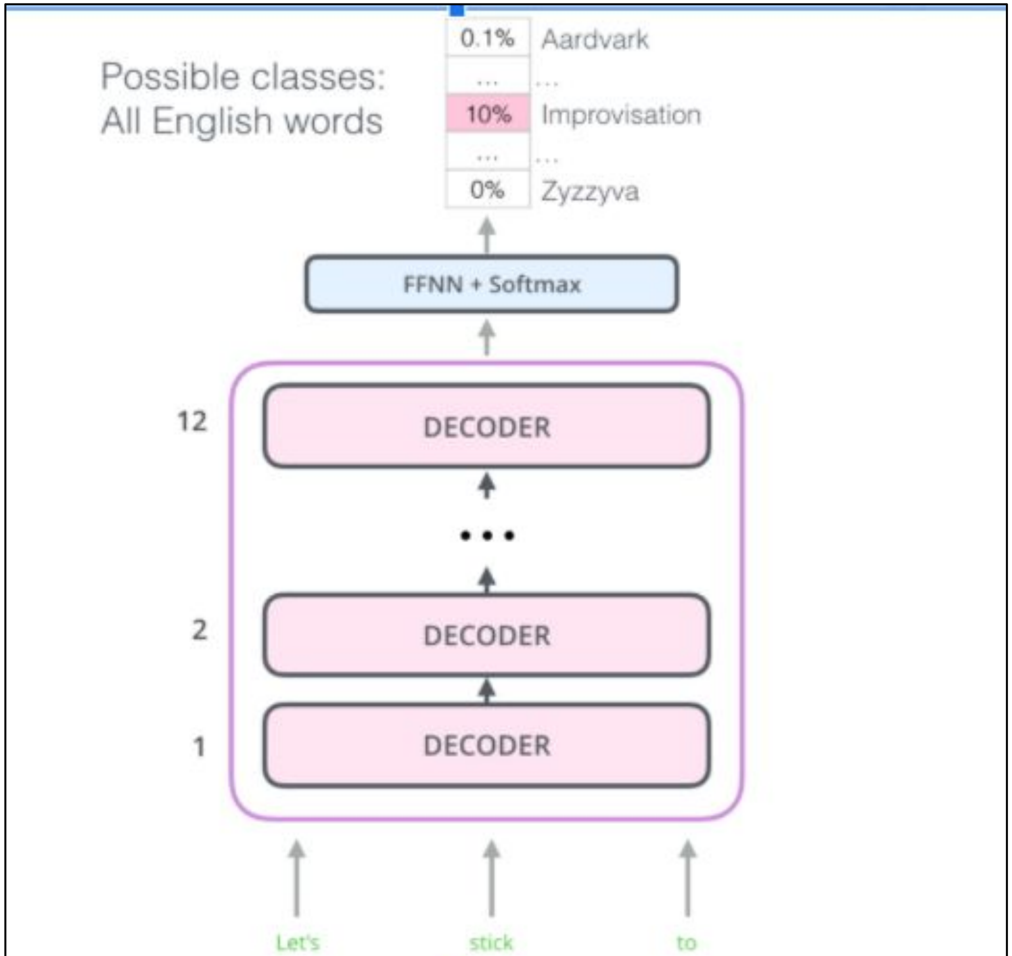
BERT e seus amigos



GPT

- Transformer-decoder
- Unidirecional



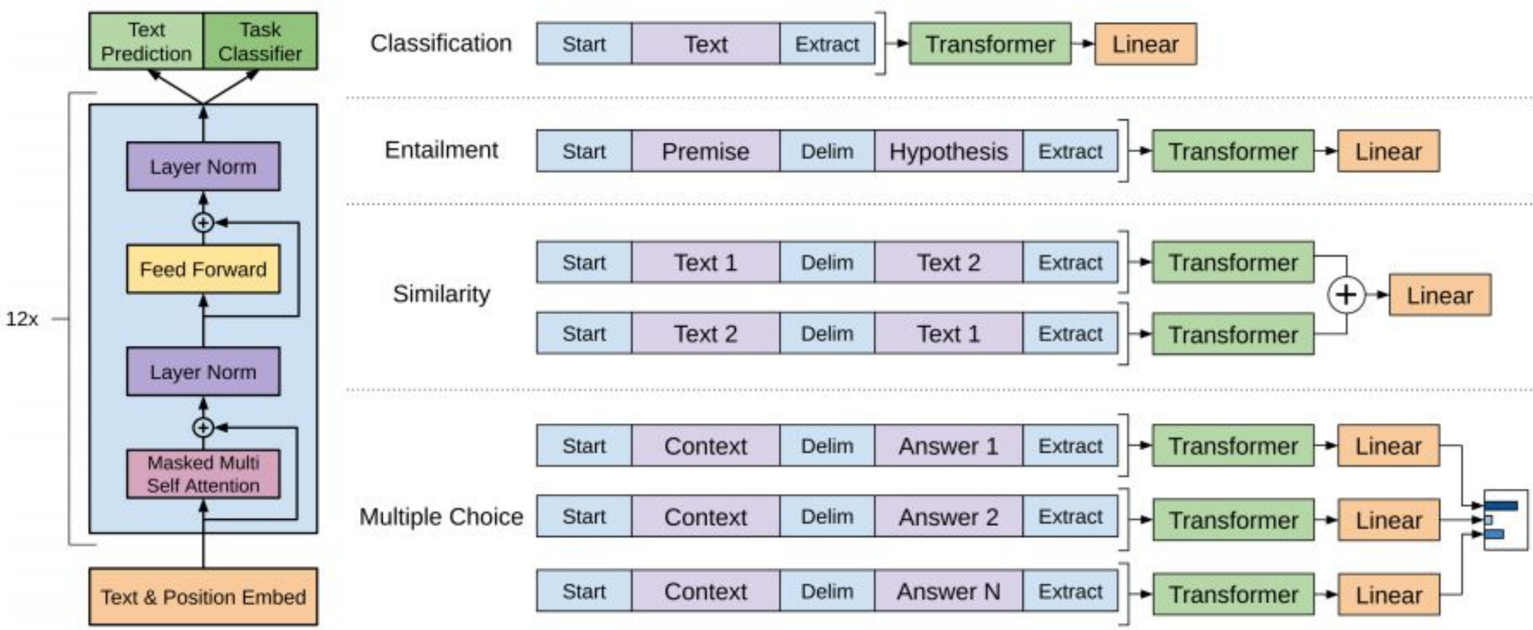


Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018) Improving Language Understanding by Generative Pre-Training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019) Language Models are Unsupervised Multitask Learners.

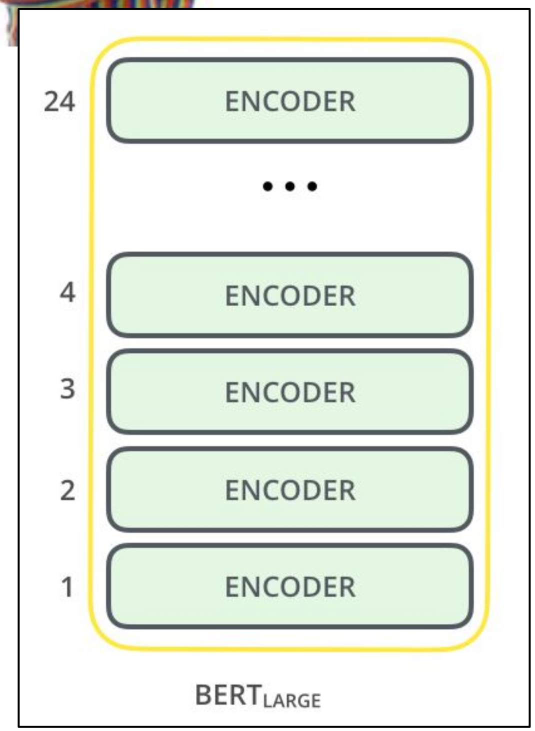
GPT

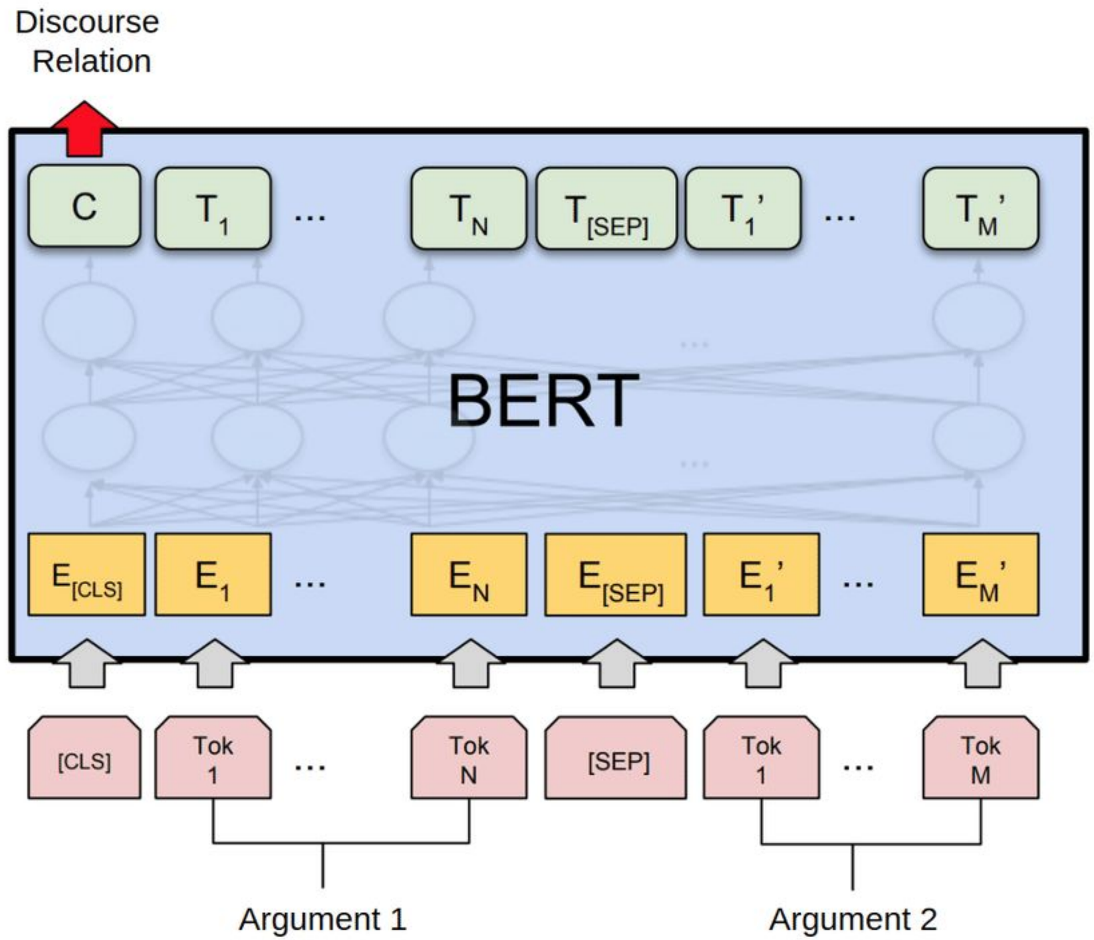
Auto-regressive language model



* P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer. Generating wikipedia by summarizing long sequences. ICLR, 2018
* Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.

BERT





Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT (1)*.

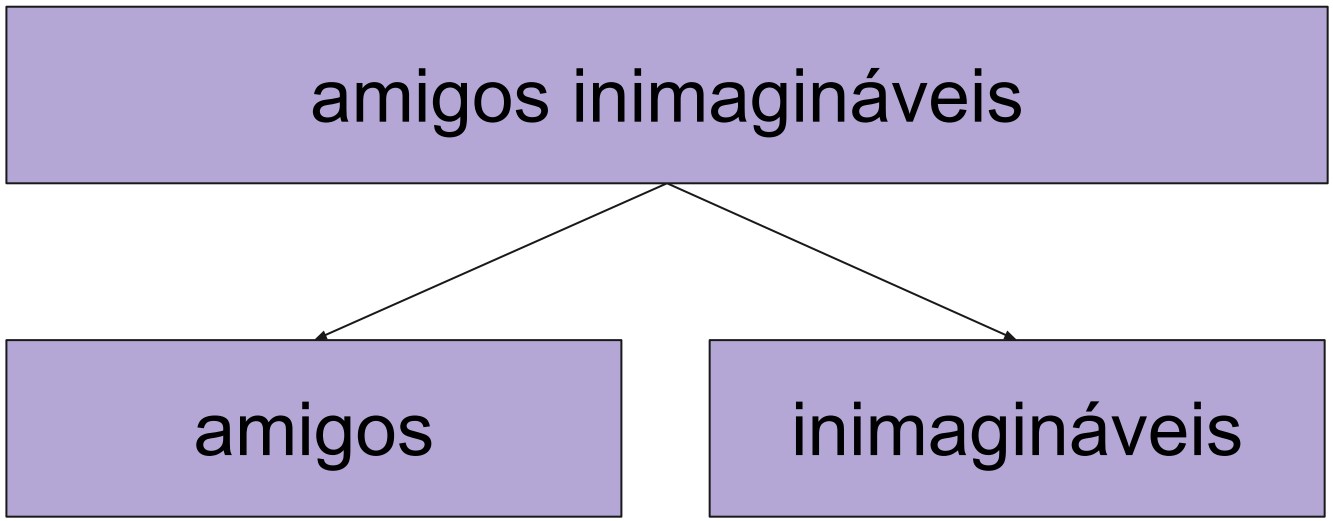
Lembram das palavras desconhecidas?

a) Assume que todas elas são um símbolo UNK
b) Nunca teremos palavras desconhecidas, pois todos os modelos conseguem lidar com o vocabulário inteiro
c) Ignora e segue em frente

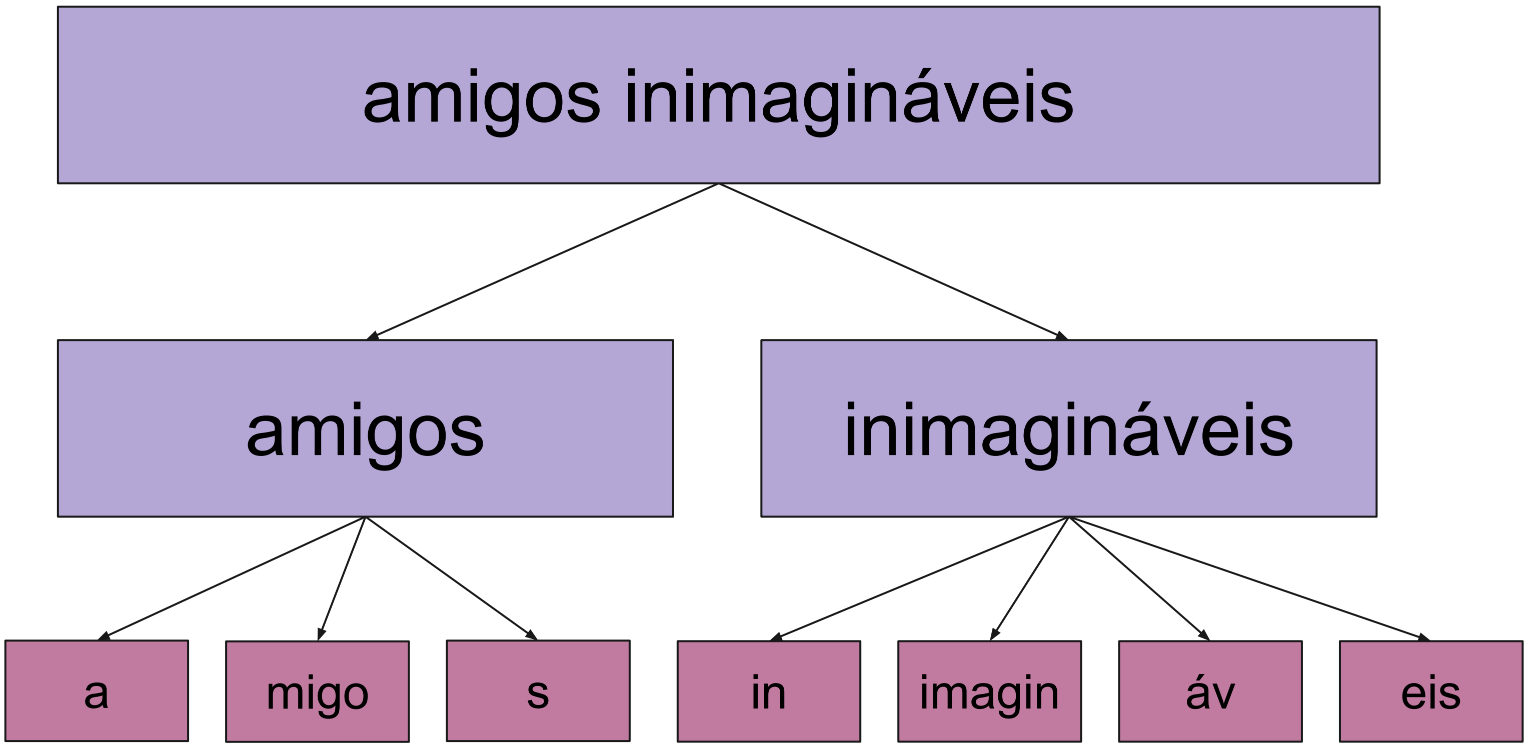
O que modelos de linguagem neurais fazem para diminuir UNKs?

amigos inimagináveis

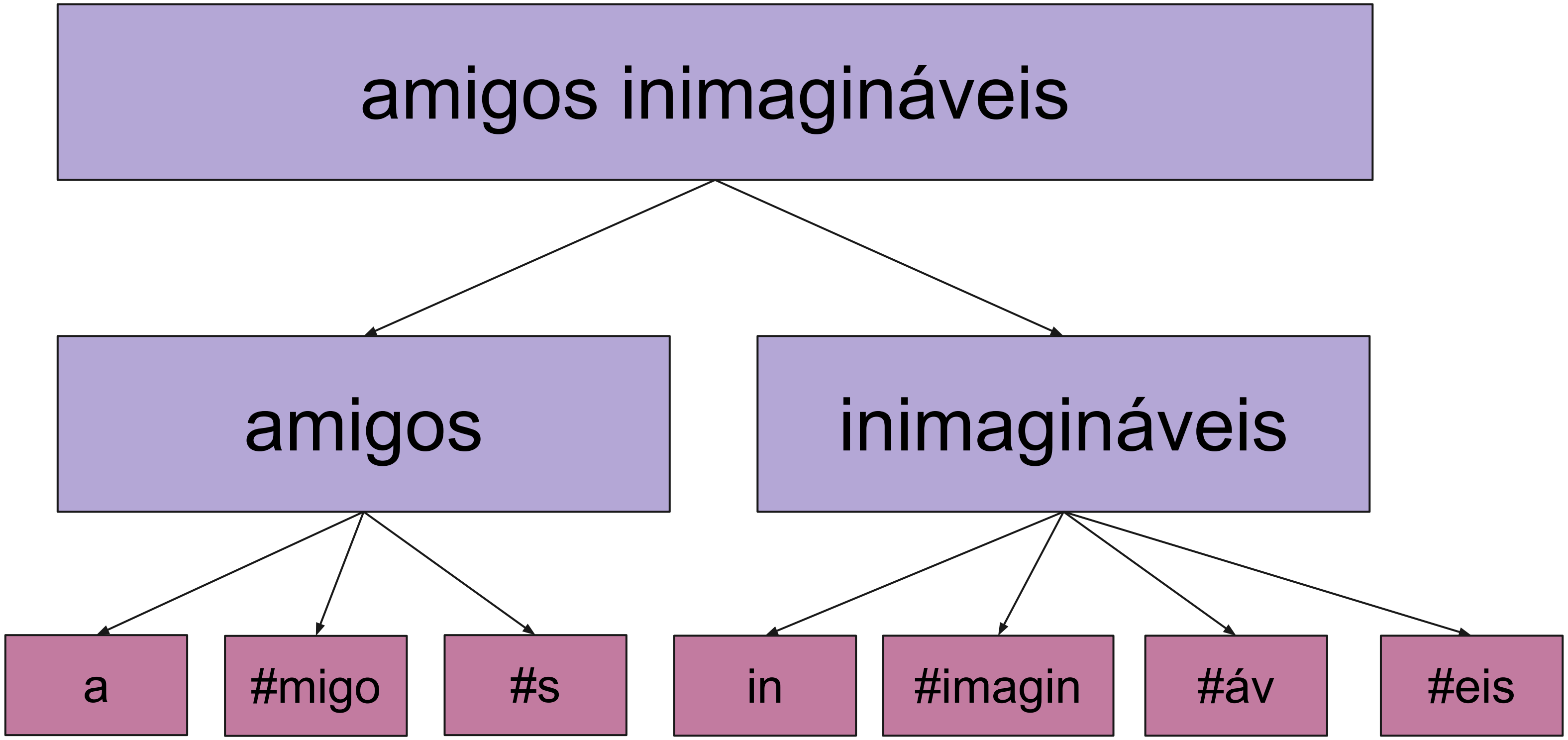
O que modelos de linguagem neurais fazem para diminuir UNKs?



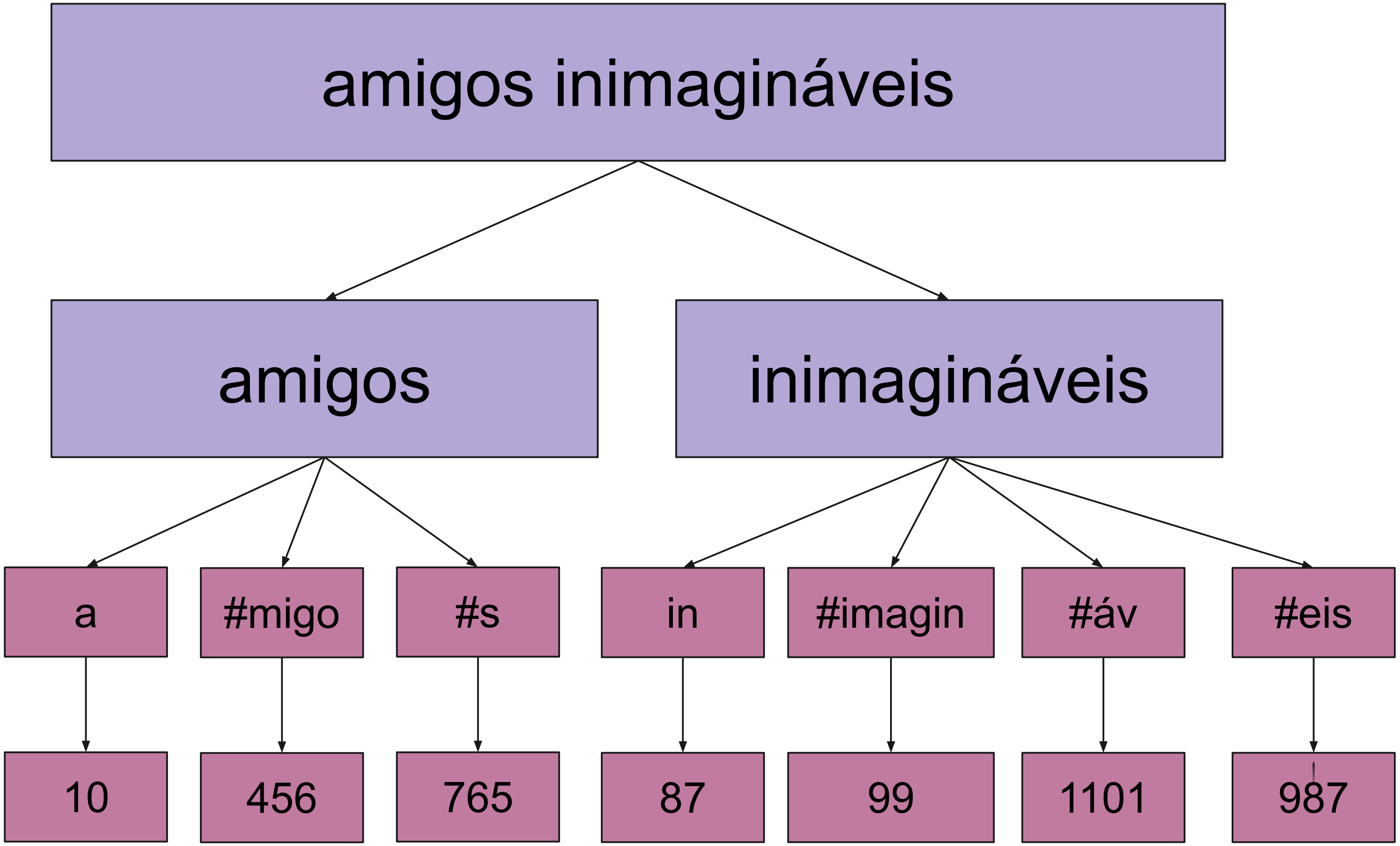
O que modelos de linguagem neurais fazem para diminuir UNKs?



O que modelos de linguagem neurais fazem para diminuir UNKs?



O que modelos de linguagem neurais fazem para diminuir UNKs?



Tokenizador

Tiktokenizer

microsoft/phi-2

amigos inimagináveis

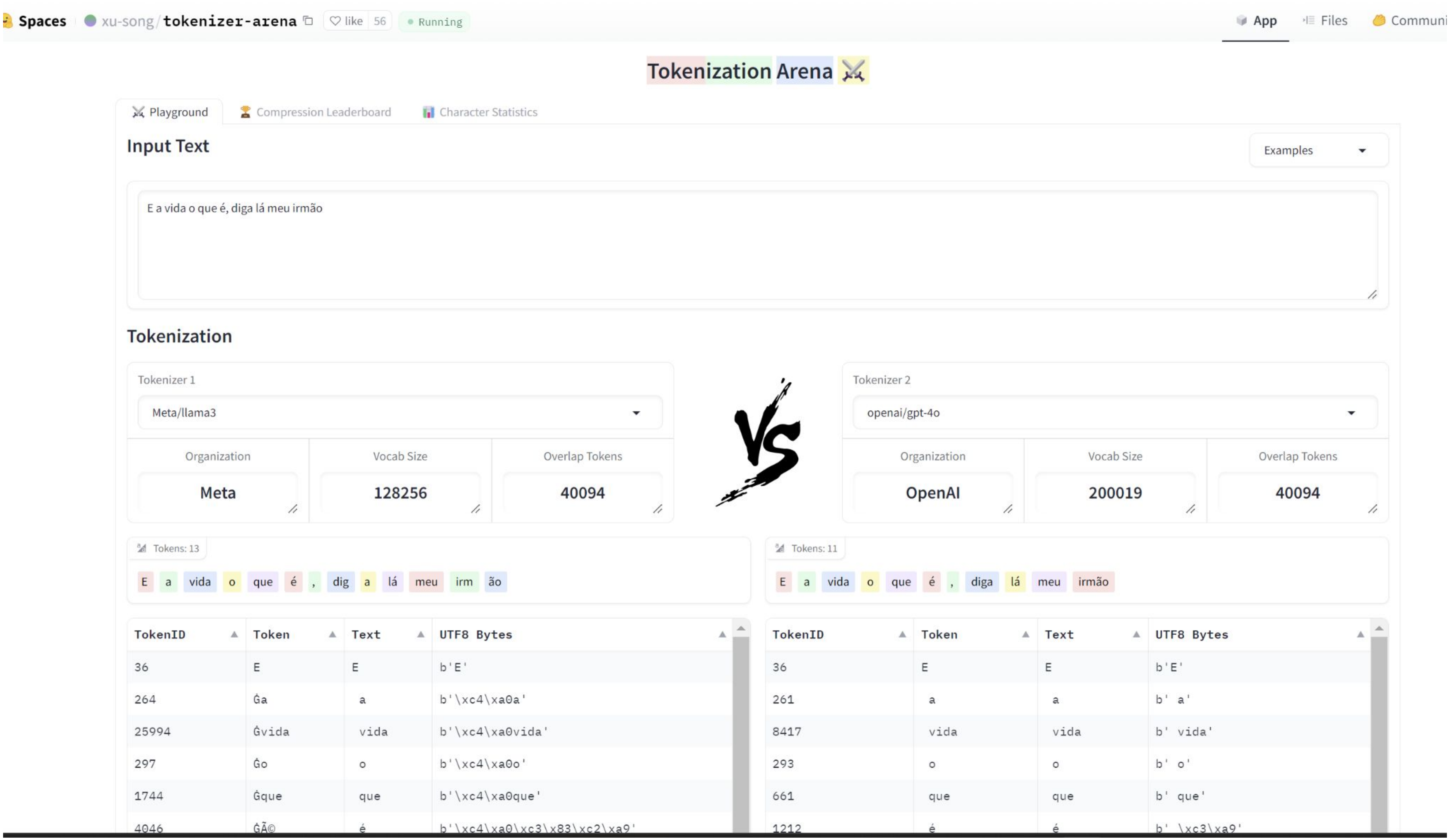
Token count
9

amigos inimagináveis

321, 328, 418, 287, 320, 23183, 6557, 303, 271

Show whitespace

Tokenizador



Notebook embeddings