Attention

Natural Language Processing — Lecture 5

Kenneth Enevoldsen | 2024

CENTER FOR HUMANITIES COMPUTING

Agenda

- Mid-term evaluation
- Recap
  - Document Embeddings
  - Neural Network
- Attention
  - History of Attention
    - Example using Neural Machine translation (NMT)
  - Intuitive understanding
  - In-depth example
- Next time
  - Looking digger deeper in the transformer architecture

Mid-term

- Link:
  - https://forms.gle/4zLAa2vhs35E4NFX7

Updated Lecture plan

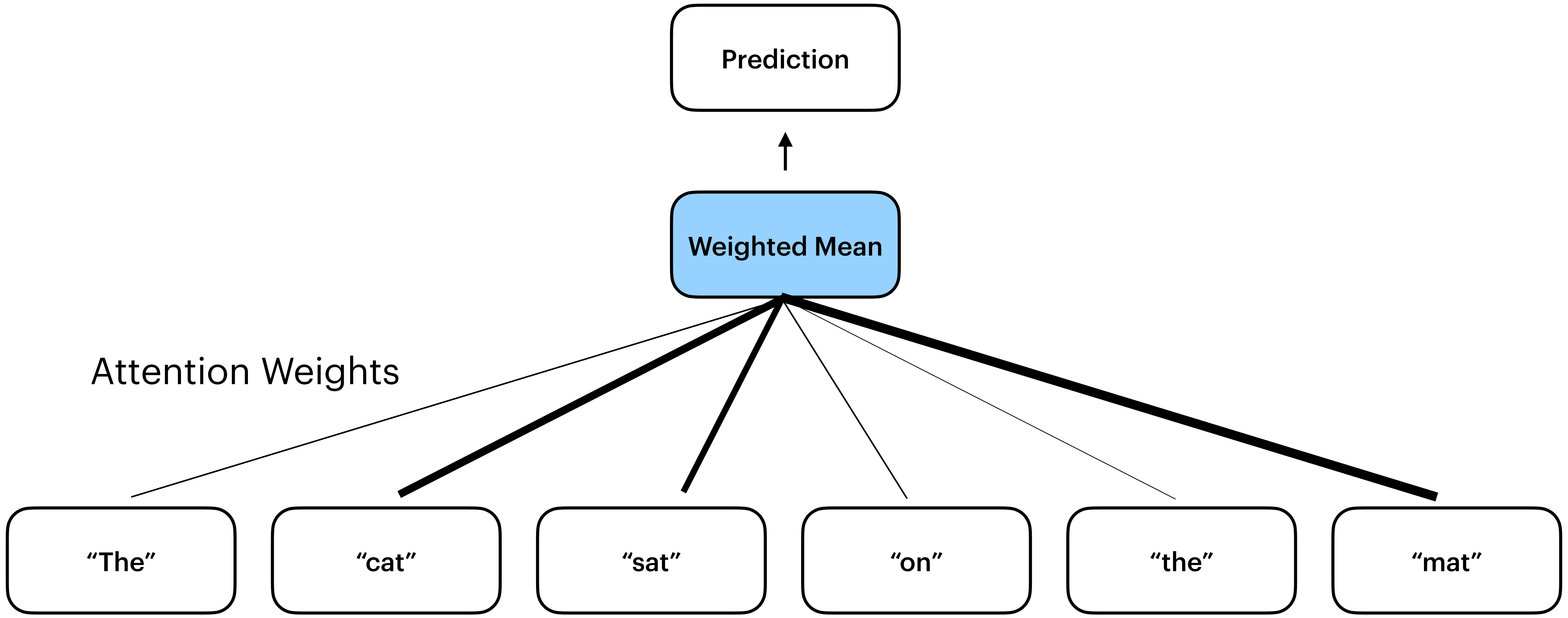
Goal: Create semantic document representations

- Uses:
  - Semantic Embeddings:
    - Classification
    - Bulk labelling
    - Duplicate detection
    - …
  - Semantic Search
    - Finding answer for target question
- Future: Generating coherent text (language modelling)*

* we will get to this next week.

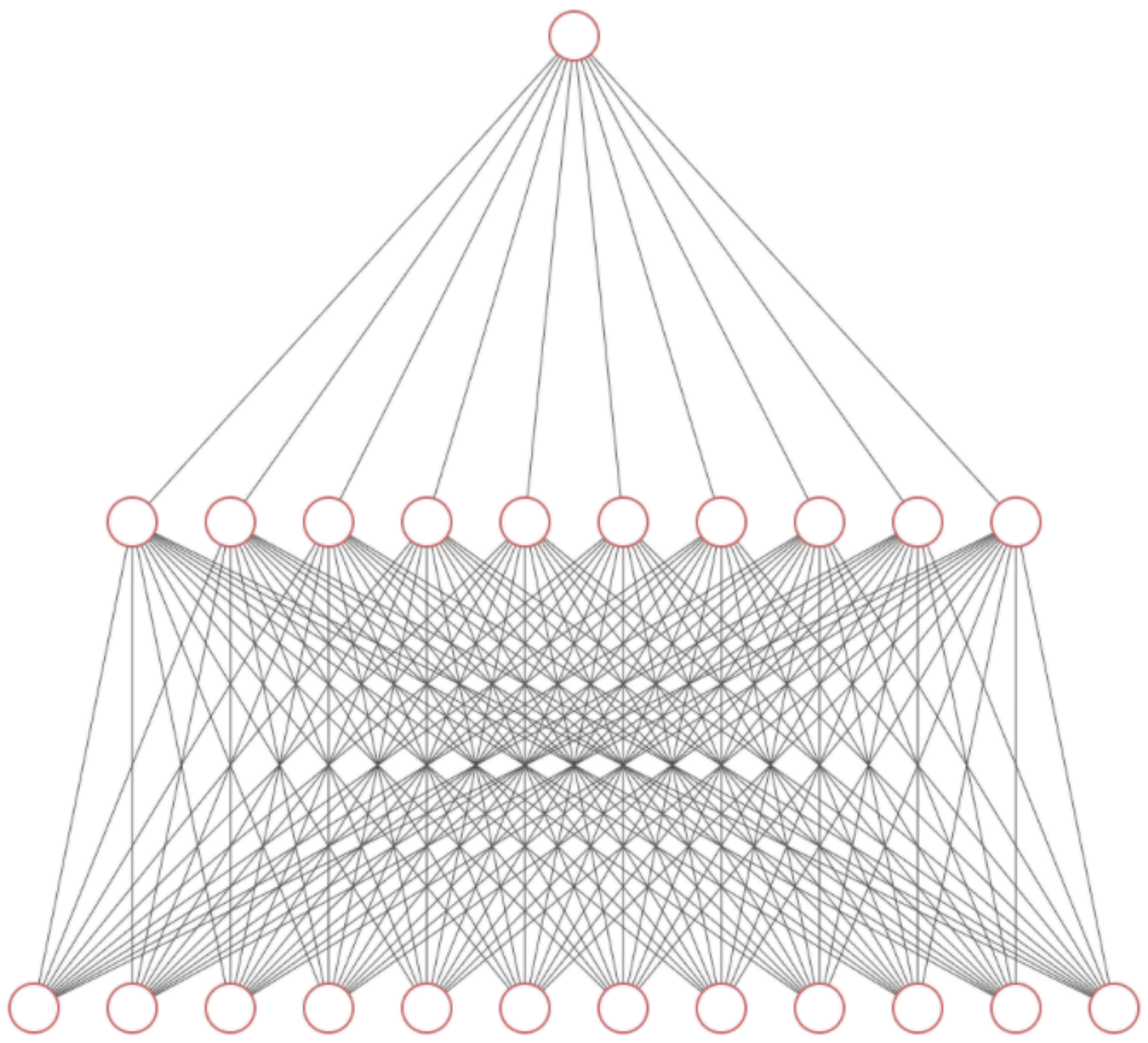
Recap: Aggregation Methods

- Simple Aggregations
  - Mean
  - Sum
  - ...
- **Model-based** Aggregation
  - Recurrent Neural Networks
  - Attention



Recap: Fully connected feedforward neural networks

Output: sentiment: [0,1]



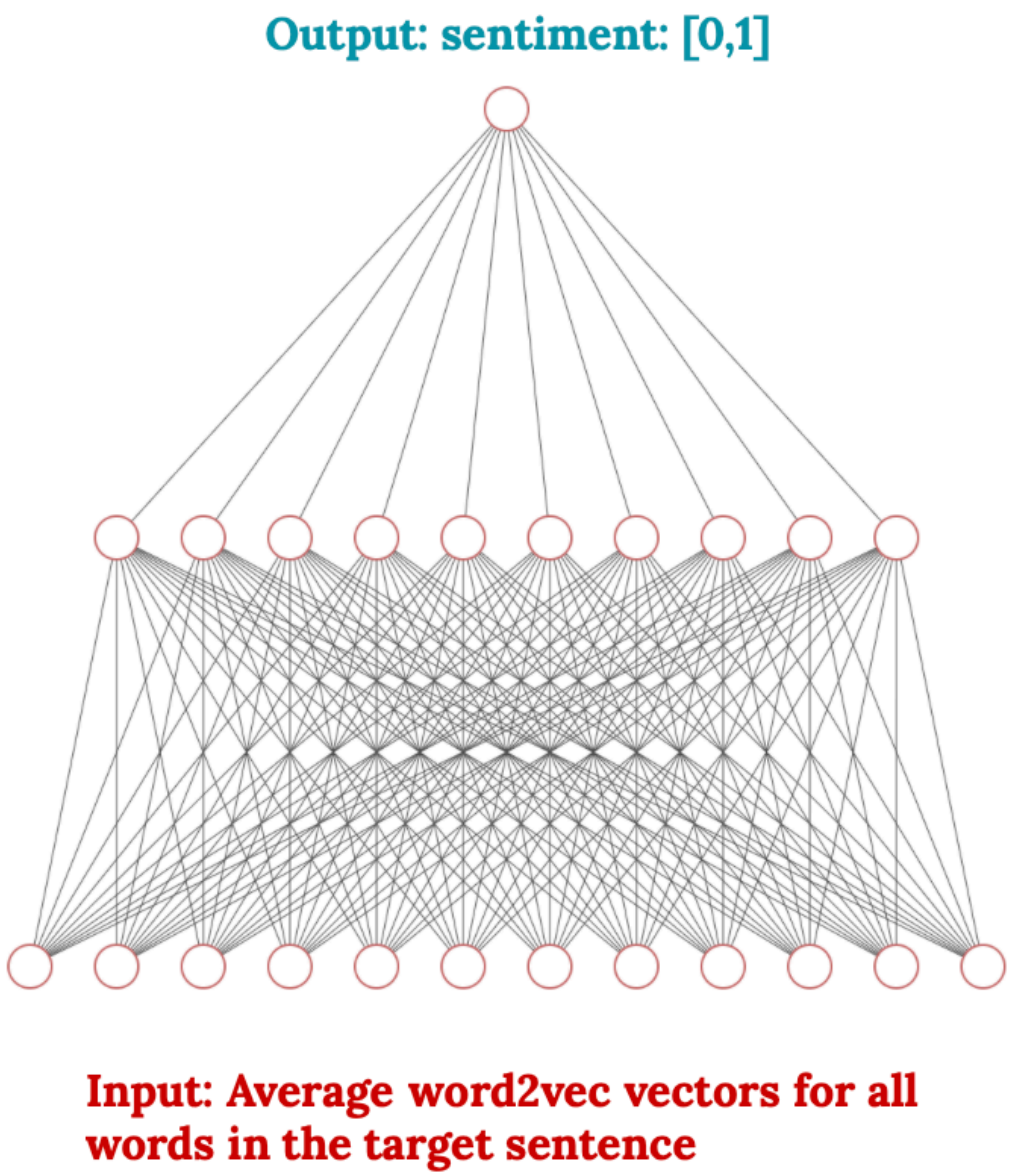
Input: Average word2vec vectors for all words in the target sentence

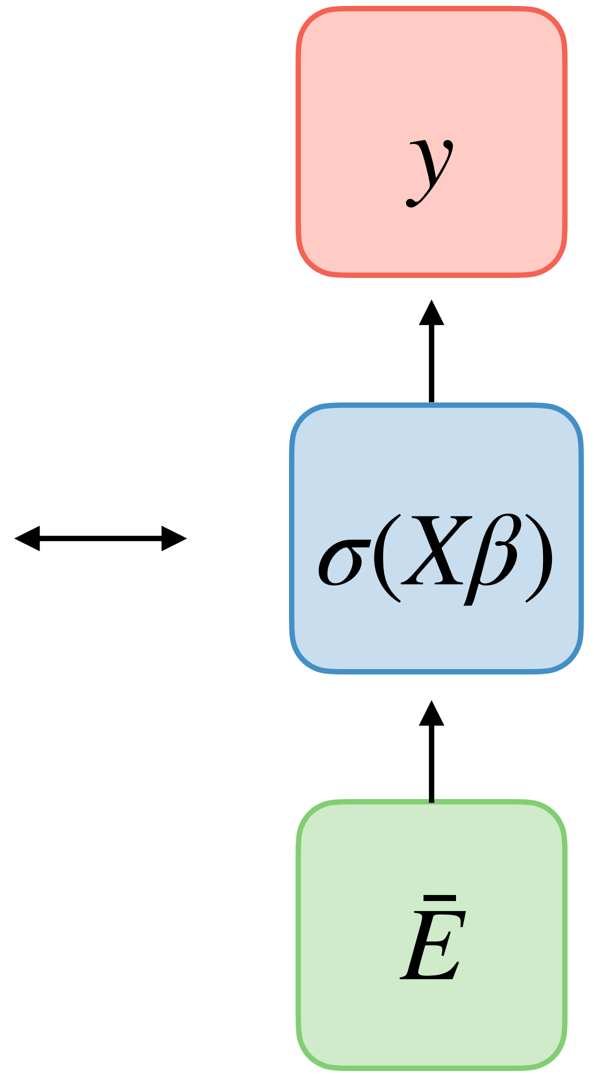
$\longleftrightarrow$

$$y$$
$$\uparrow$$
$$\sigma(X\beta)$$
$$\uparrow$$
$$\bar{E}$$

Question:
What is meant by this

Recap: Fully connected feedforward neural networks



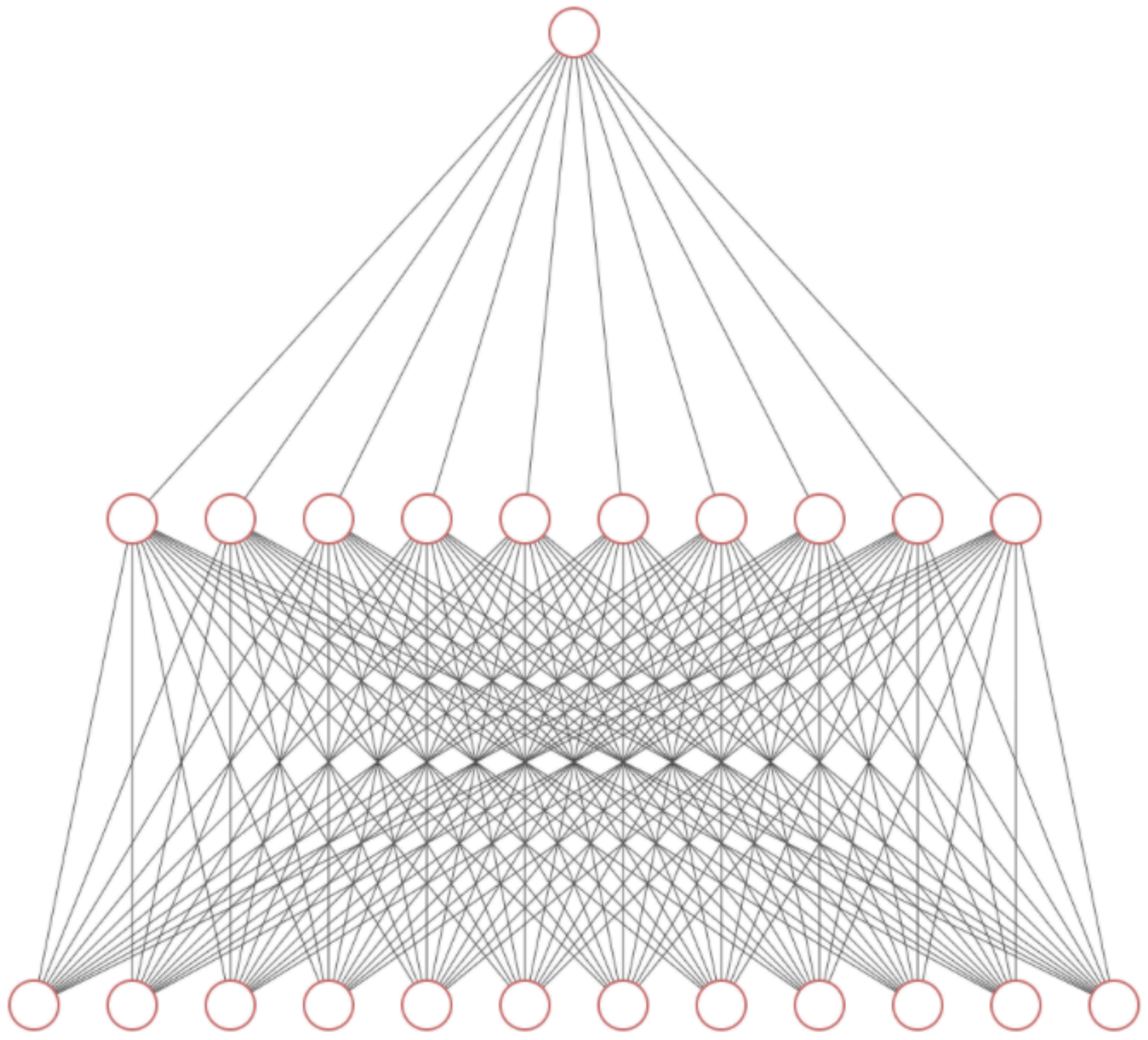


Question: What other things you could do this this architecture?

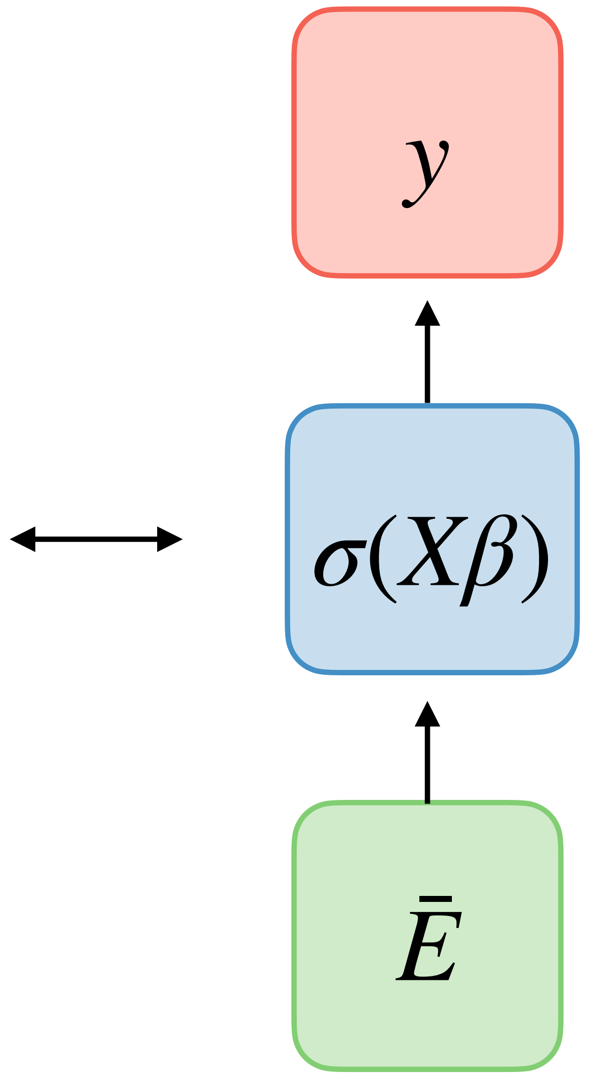
CENTER FOR
HUMANITIES
COMPUTING

Recap: Fully connected feedforward neural networks

Output: sentiment: [0,1]



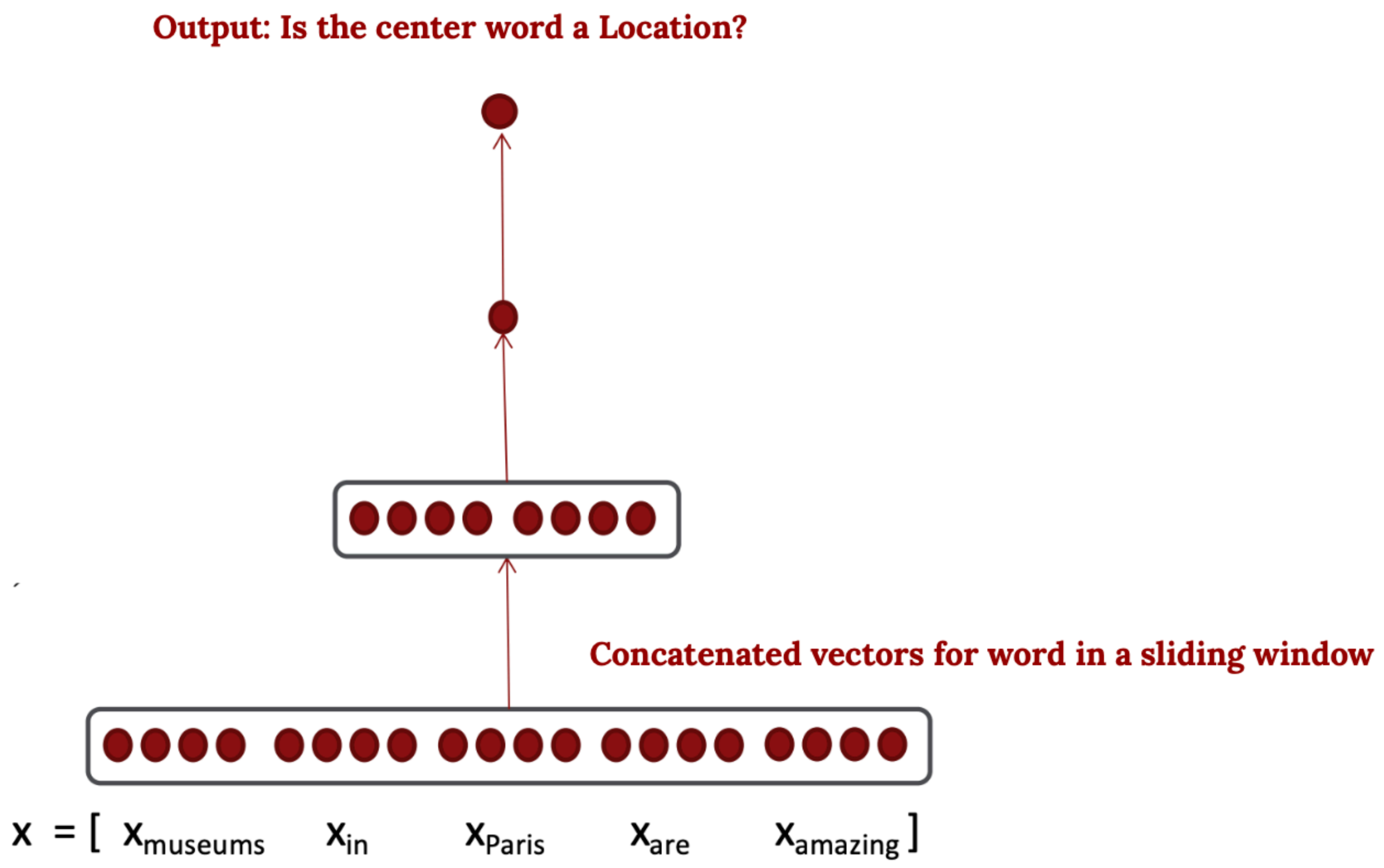
Input: Average word2vec vectors for all words in the target sentence



Question: What other things you could do this this architecture?

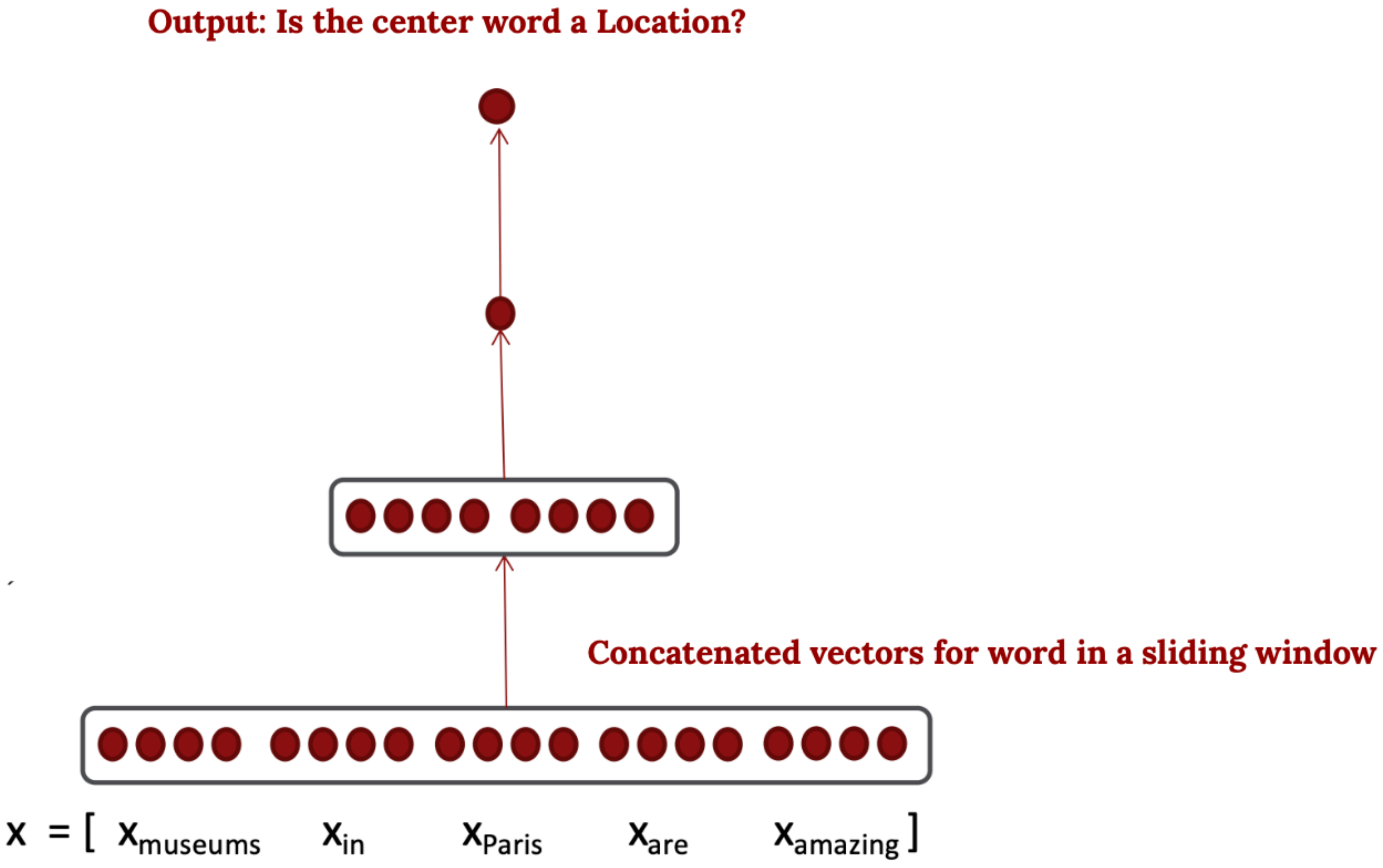
Classifcation:
- Hate-speech classification
- Unsafe question (e.g. on ChatGPT)
- Is this post about "our" product?
- ...

(Convolutional) Neural Network



Question: What other things you could do this this architecture?

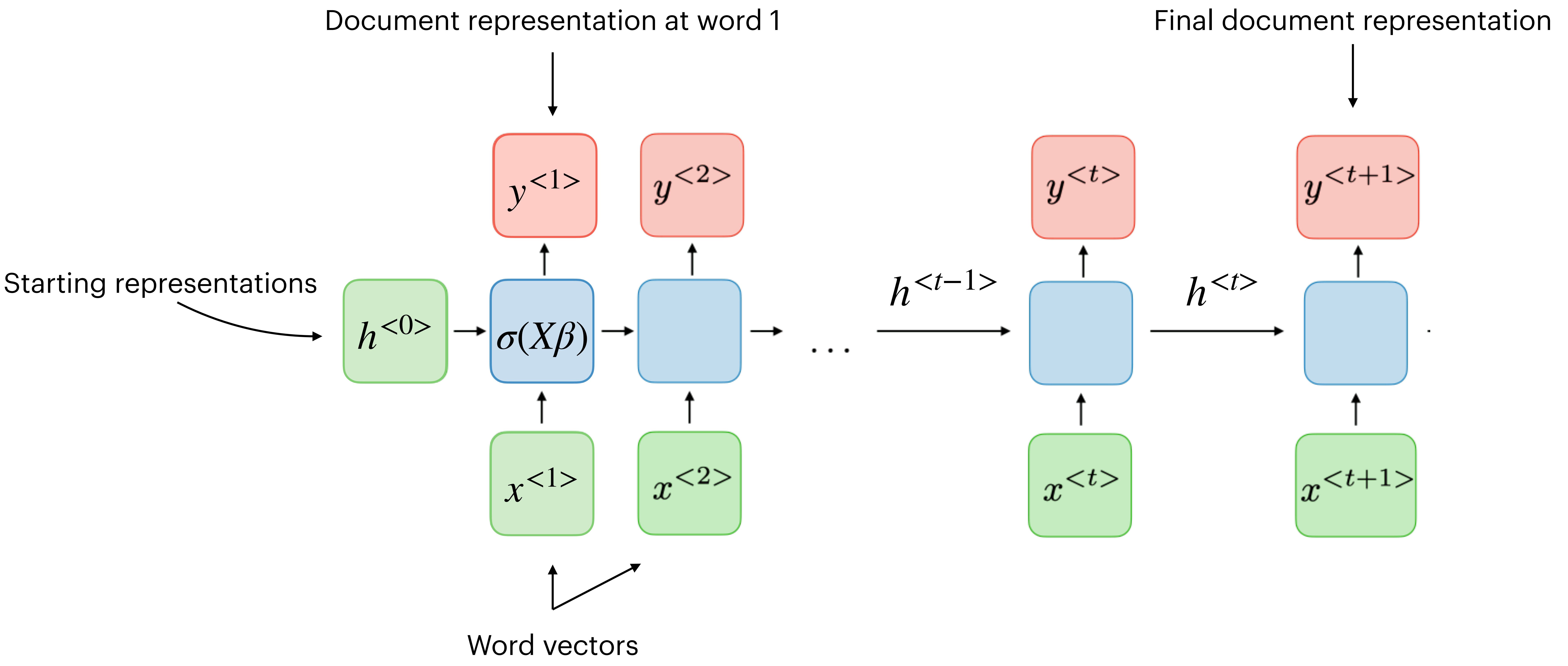
(Convolutional) Neural Network



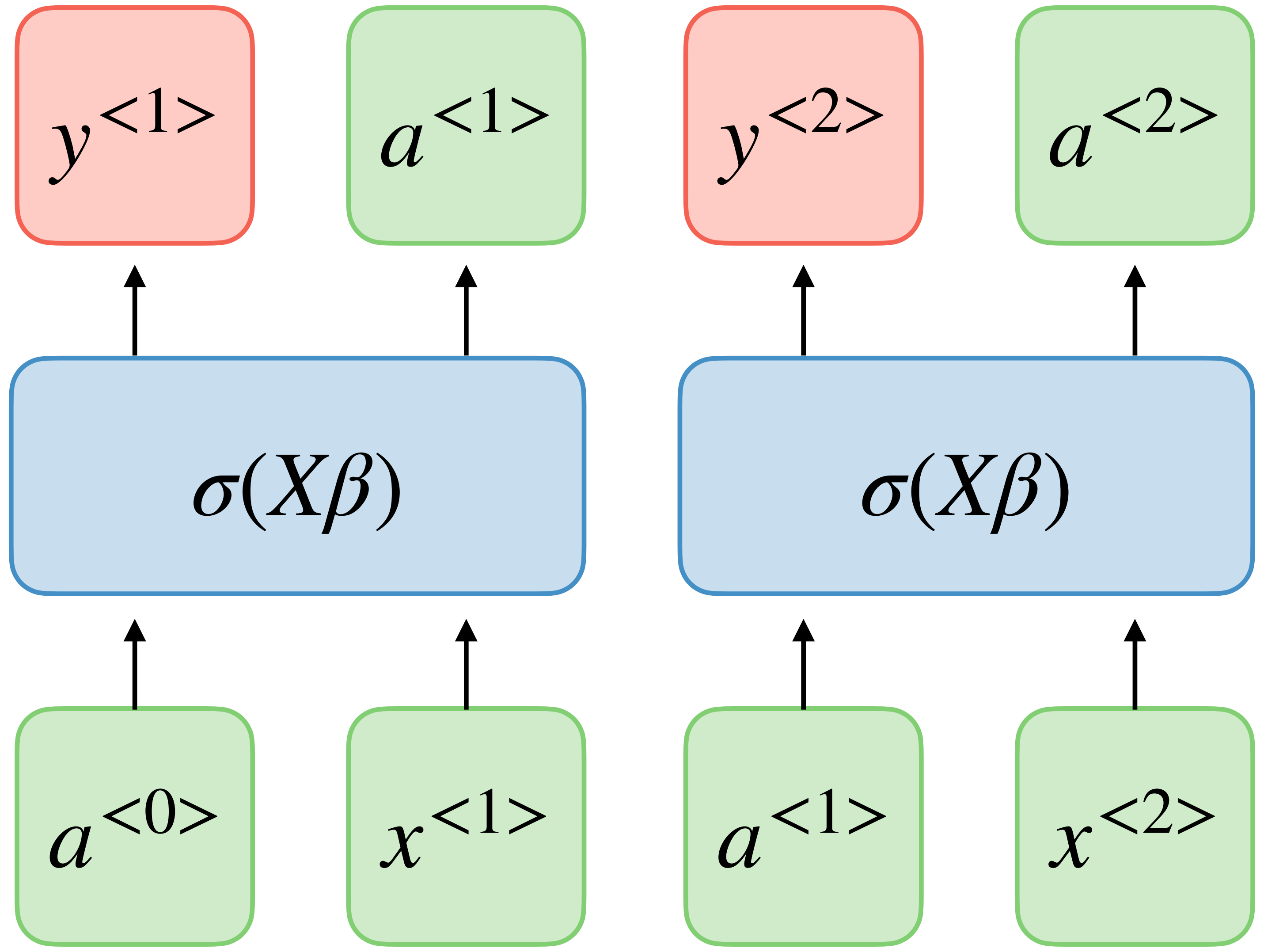
Question: What other things you could do this this architecture?

Span classifcation tasks:
- Highlight positive and negative elements of a sentence
- Drugs or symptoms in health records
- ...

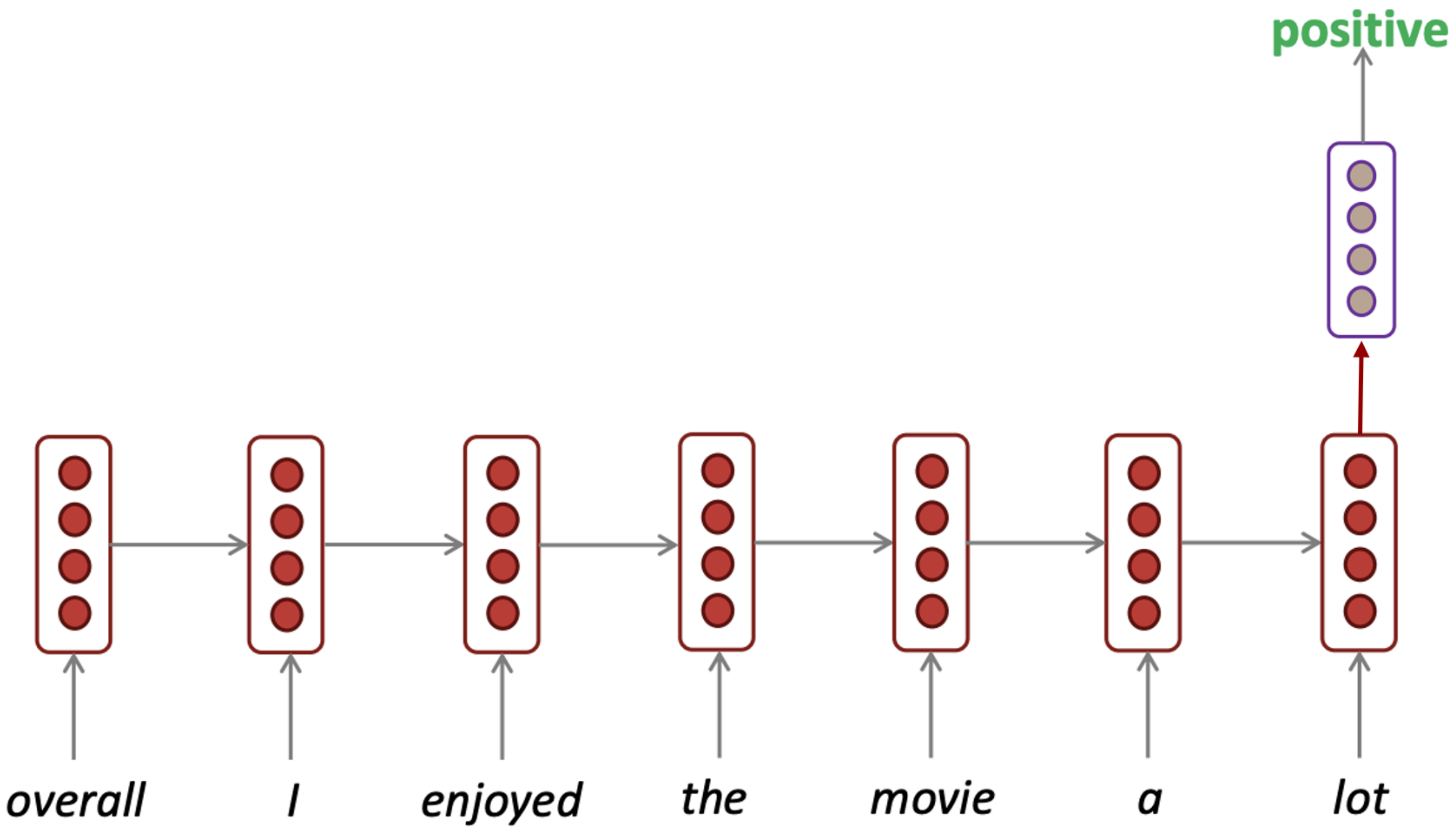
Recurrent Neural Network



Recurrent Neural Network



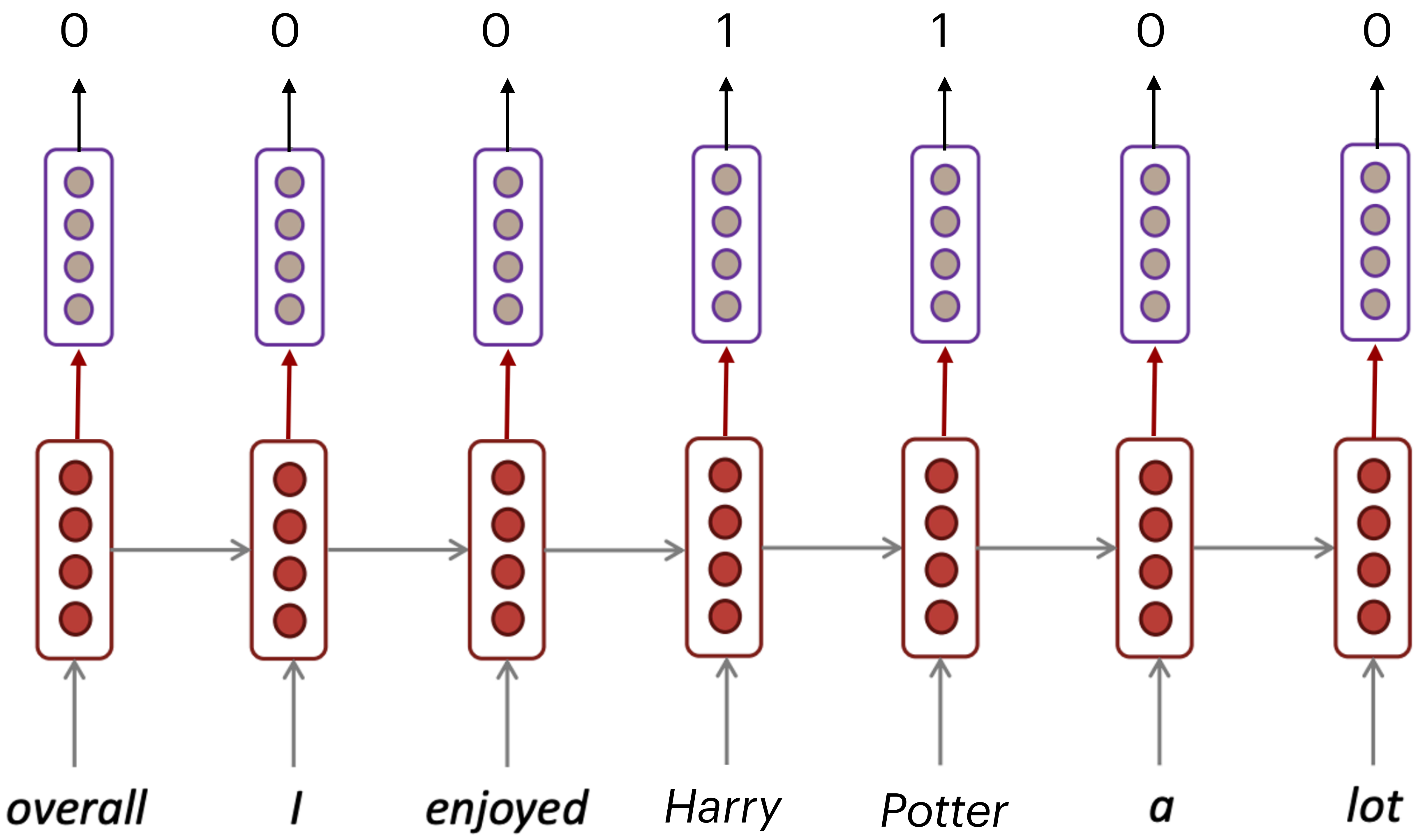
Recurrent neural Networks for classifcation



Recurrent neural Networks for NER

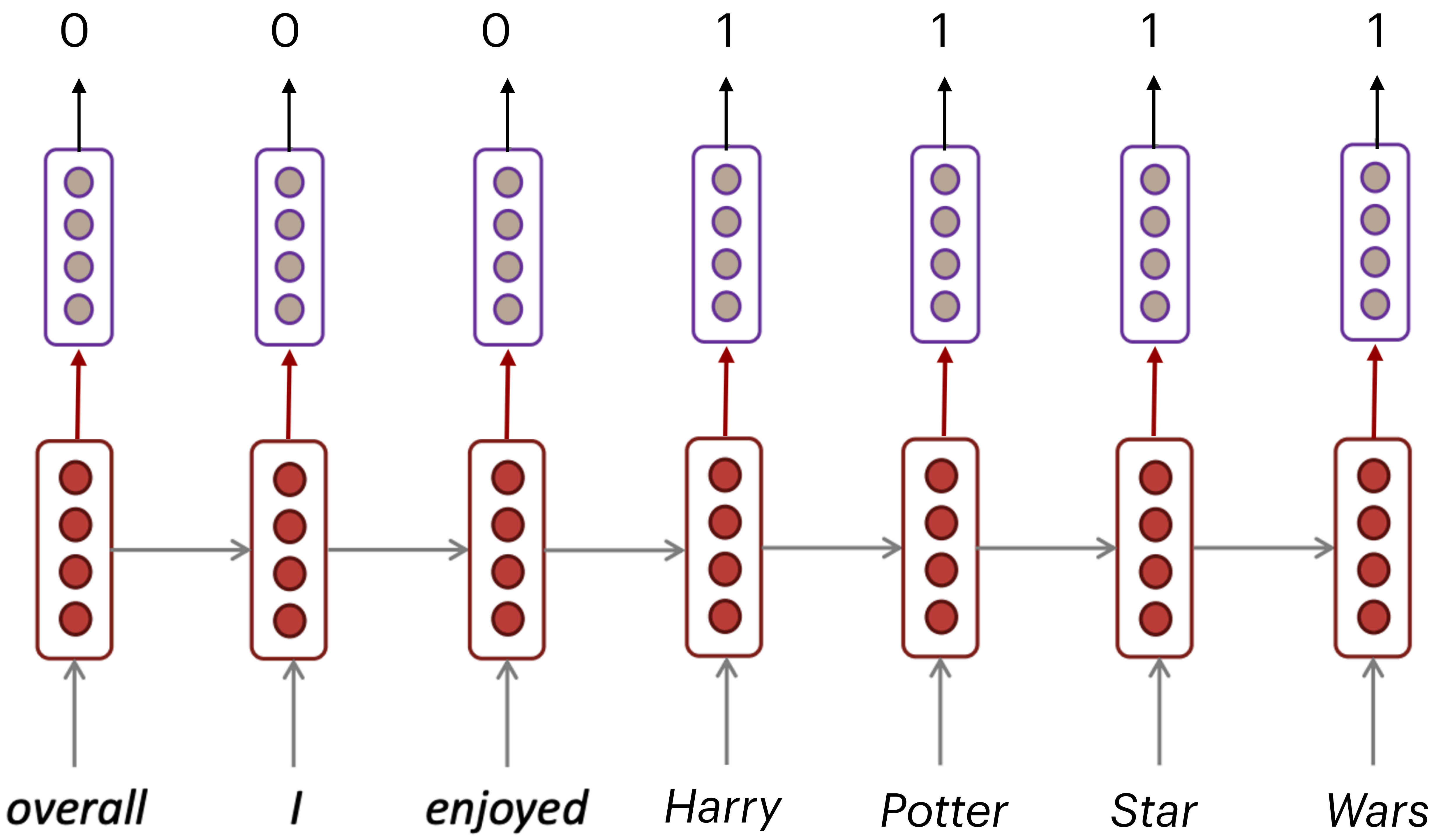
Is output a movie title:

Contextualized
word embeddings



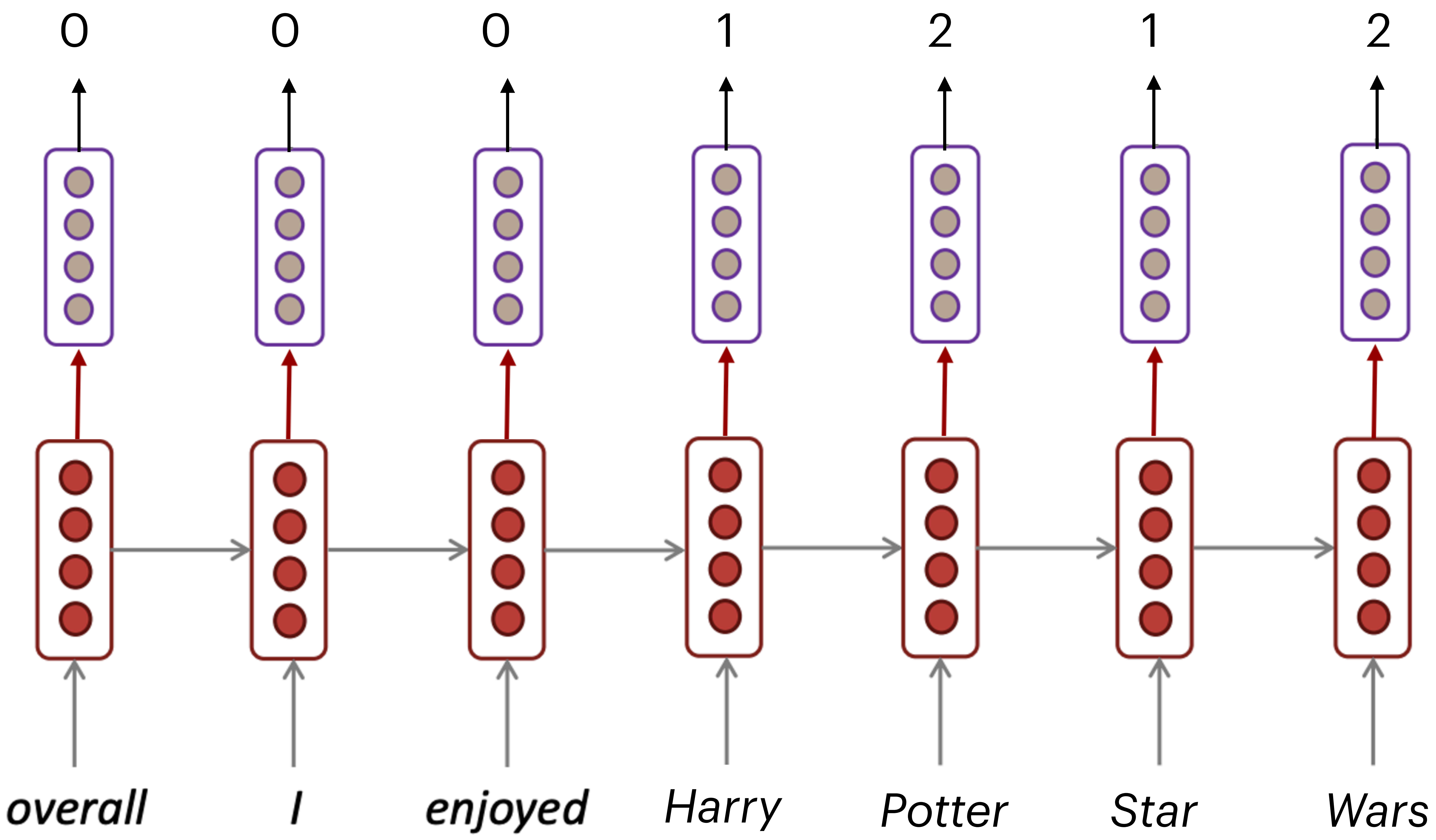
Problems?

Is output a movie title:



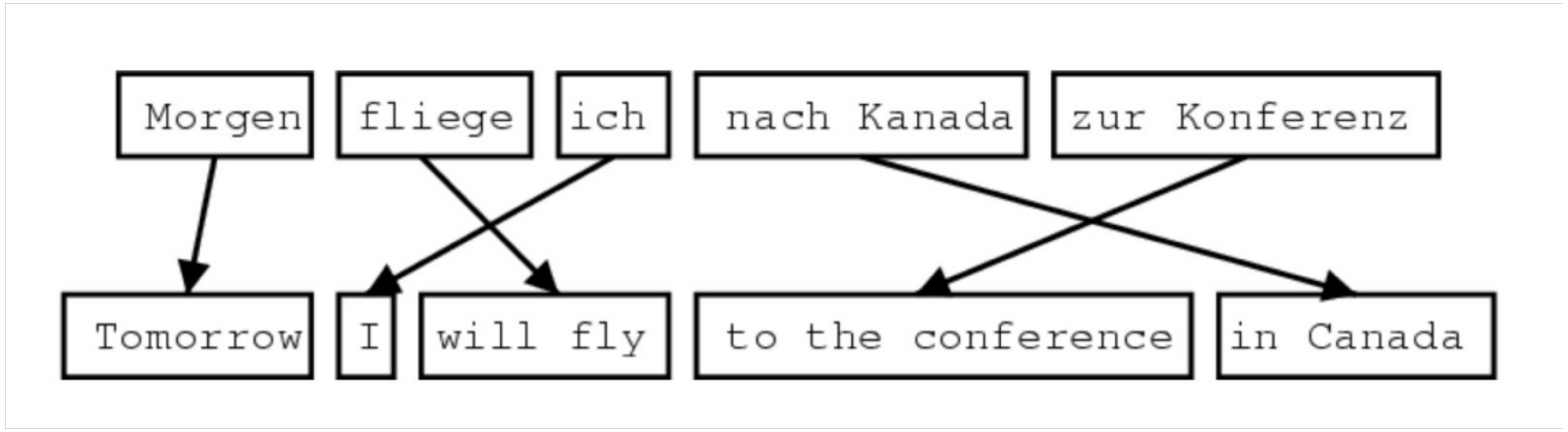
Label Schemes for NER

Is output a movie title:

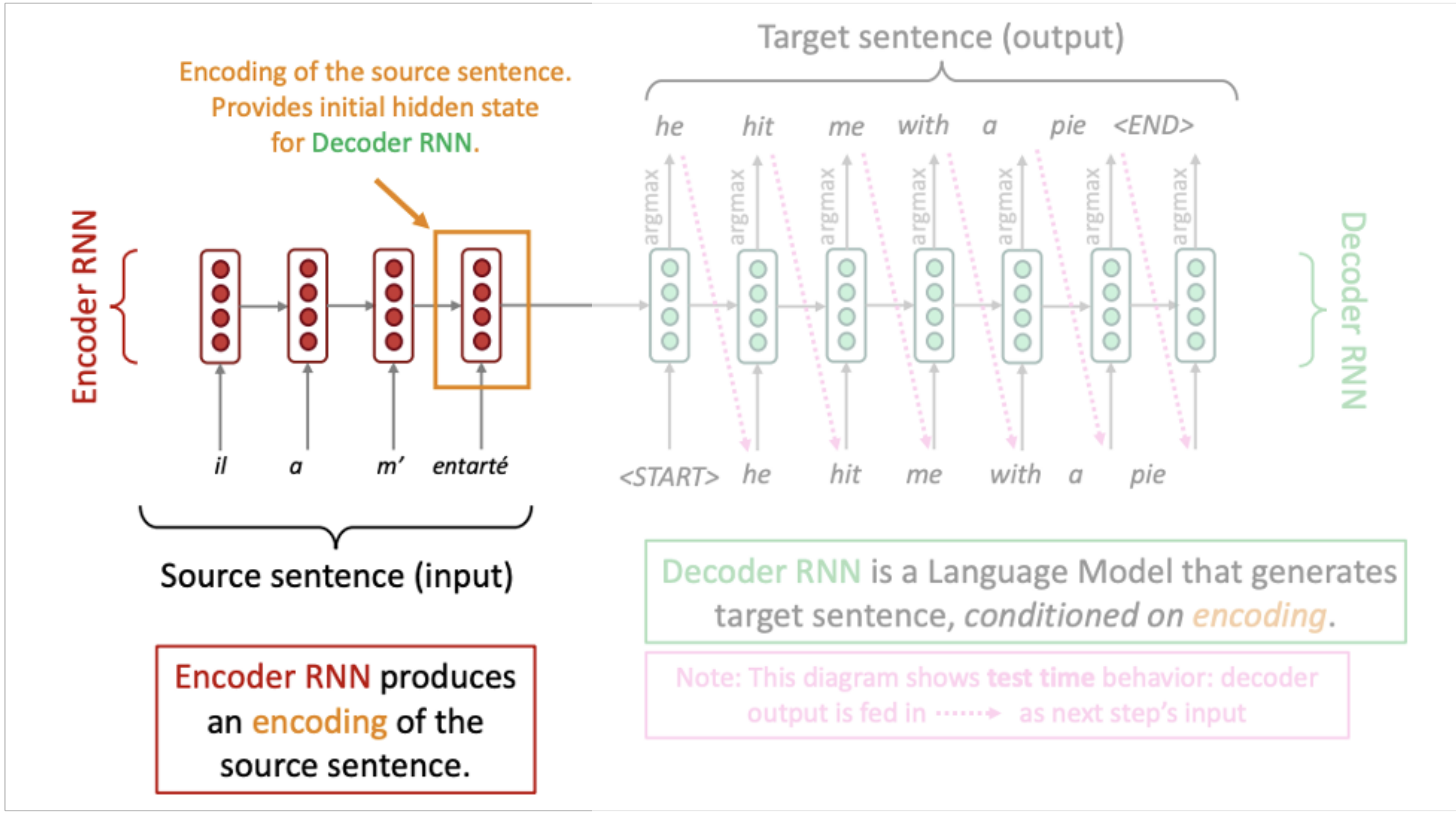


O: No entity
B-MOVIE: Beggining of MOVIE
I-MOVIE: “in” MOVIE
...
B-PERSON
I-PERSON

Brief history of Attention: Machine translation



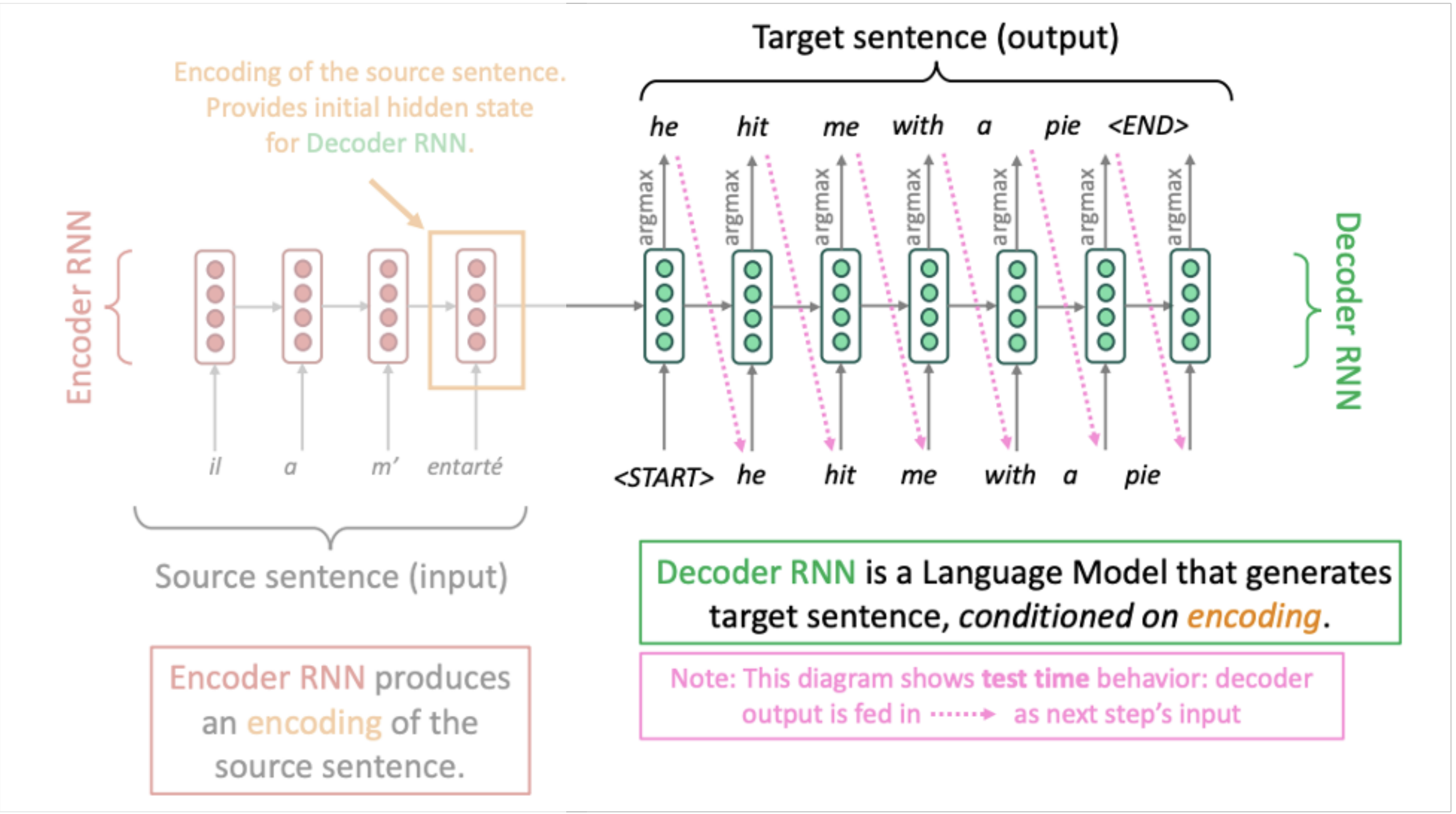
Standard Approach*



Sources & Notes

*at the time

Standard Approach*

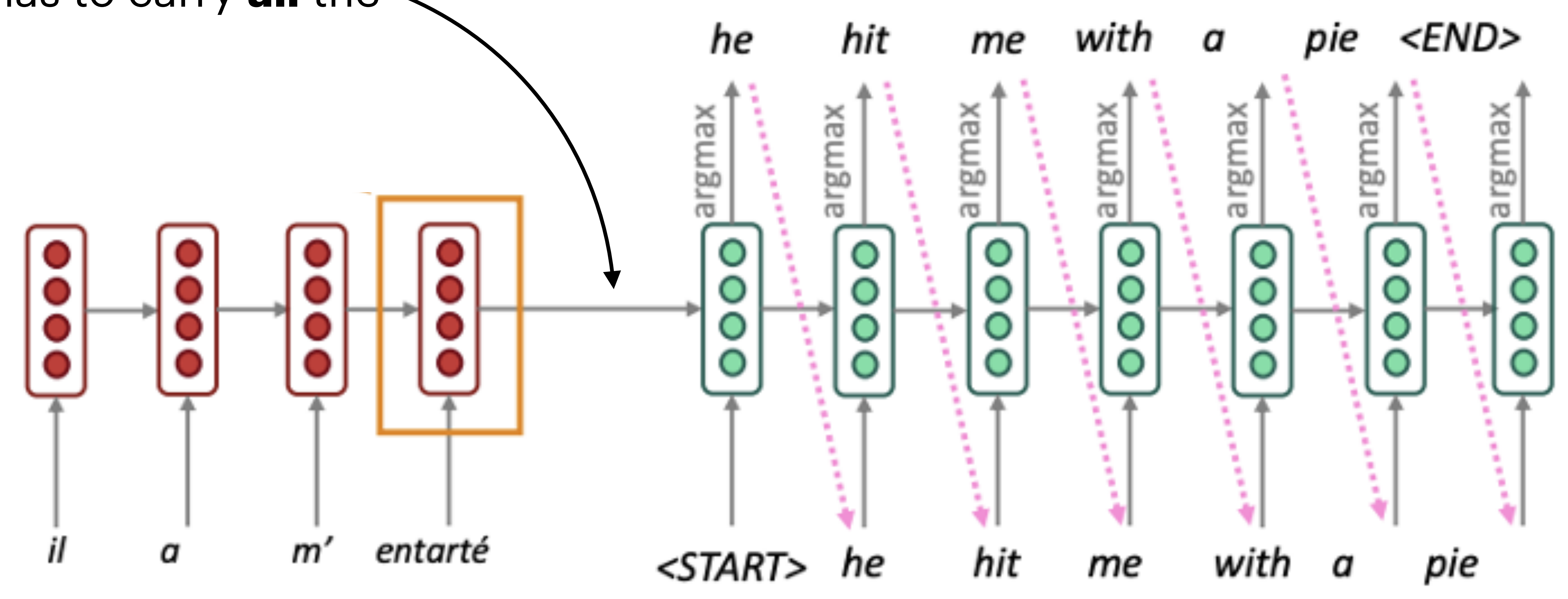


*at the time

Problems

Vanishing gradient problem:
If this word is wrong them the gradient has to flow through the the entire sequence

Bottleneck:
The hidden state passed from the encoder to the decoder has to carry **all** the information

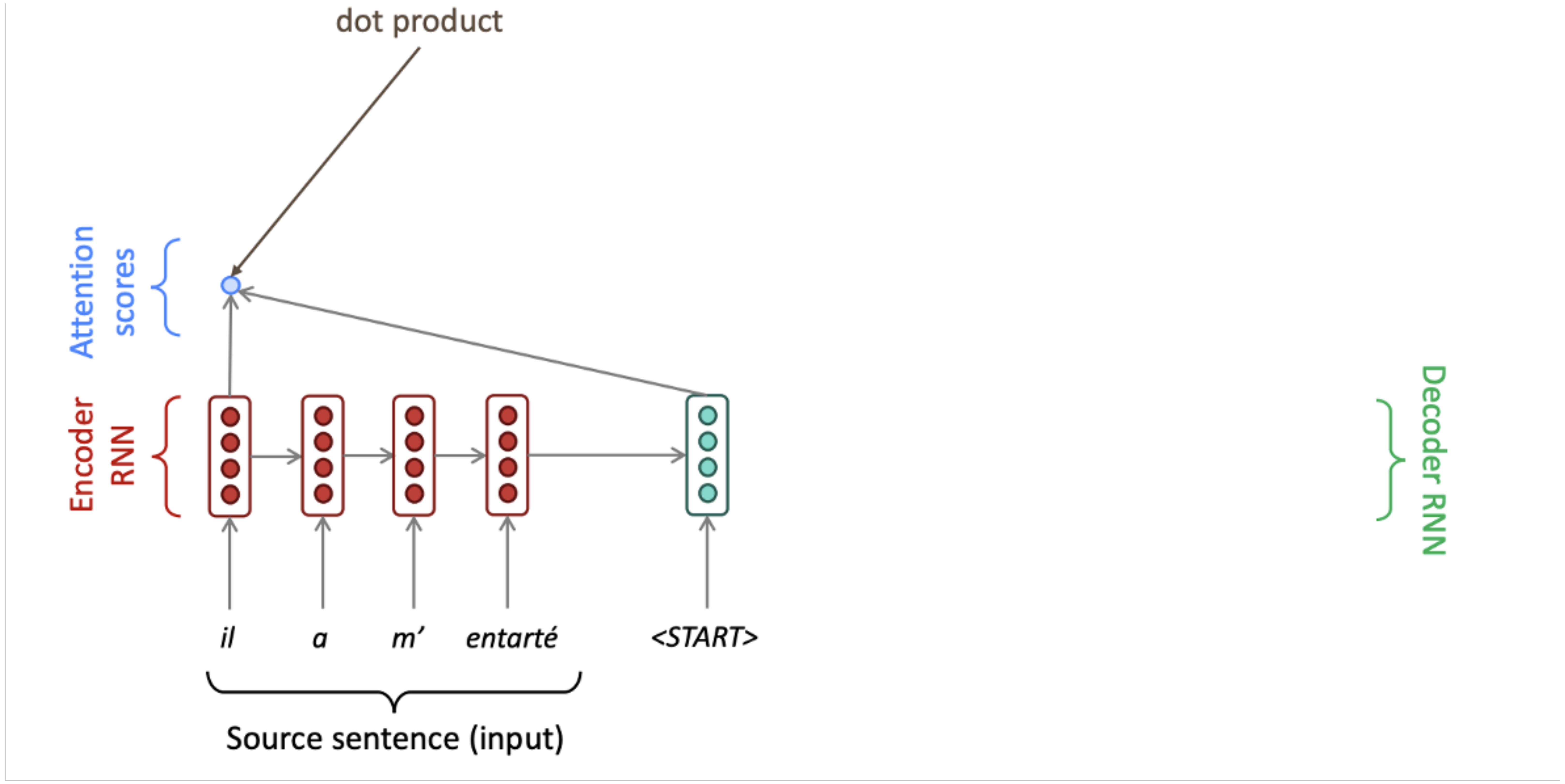


Sources & Notes

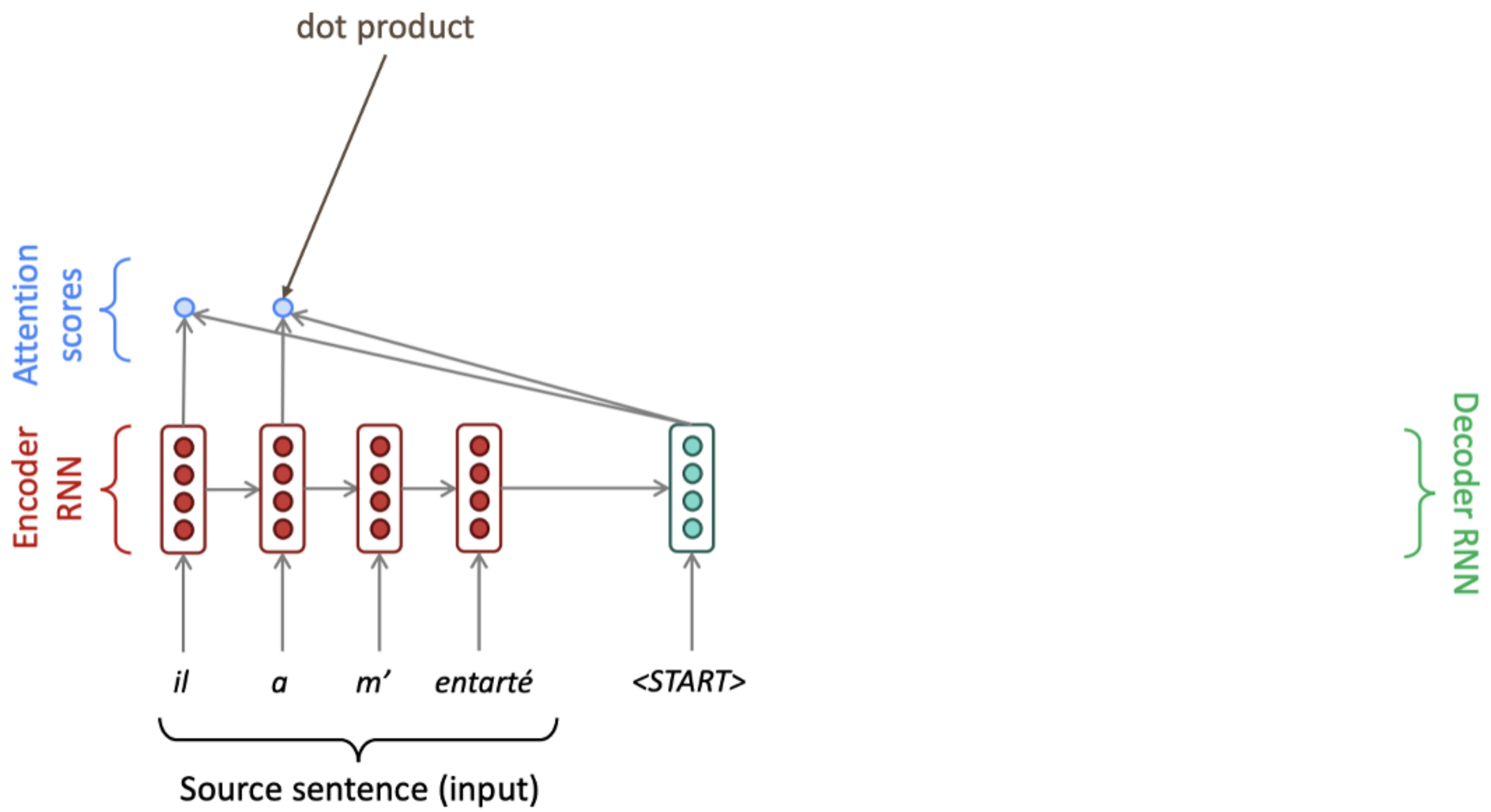
*at the time

- **Q:** how do we enable this model to use information about about specific, relevant words appearing earlier in the sequence to produce better predictions at a given time?

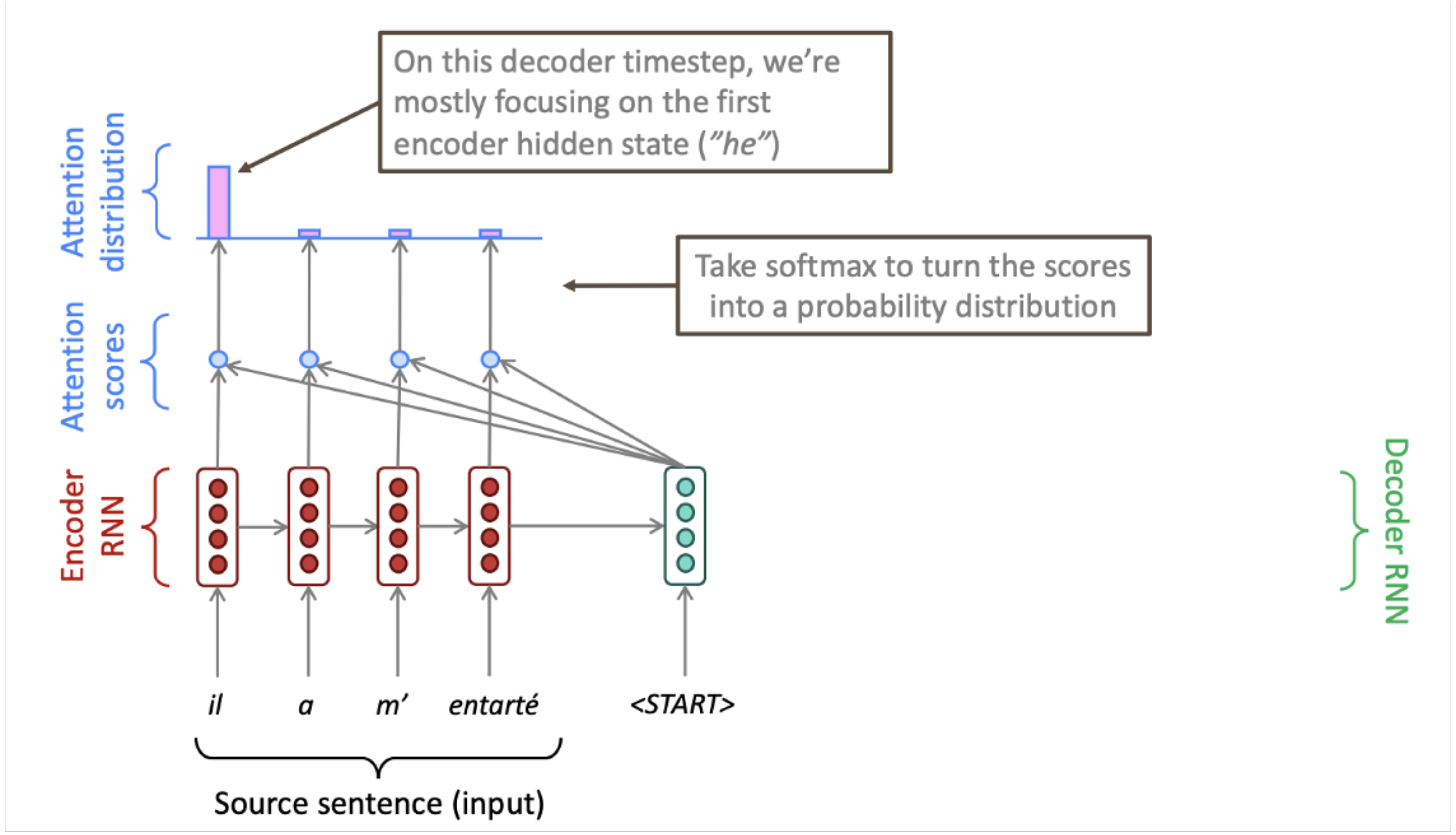
Solution: Attention



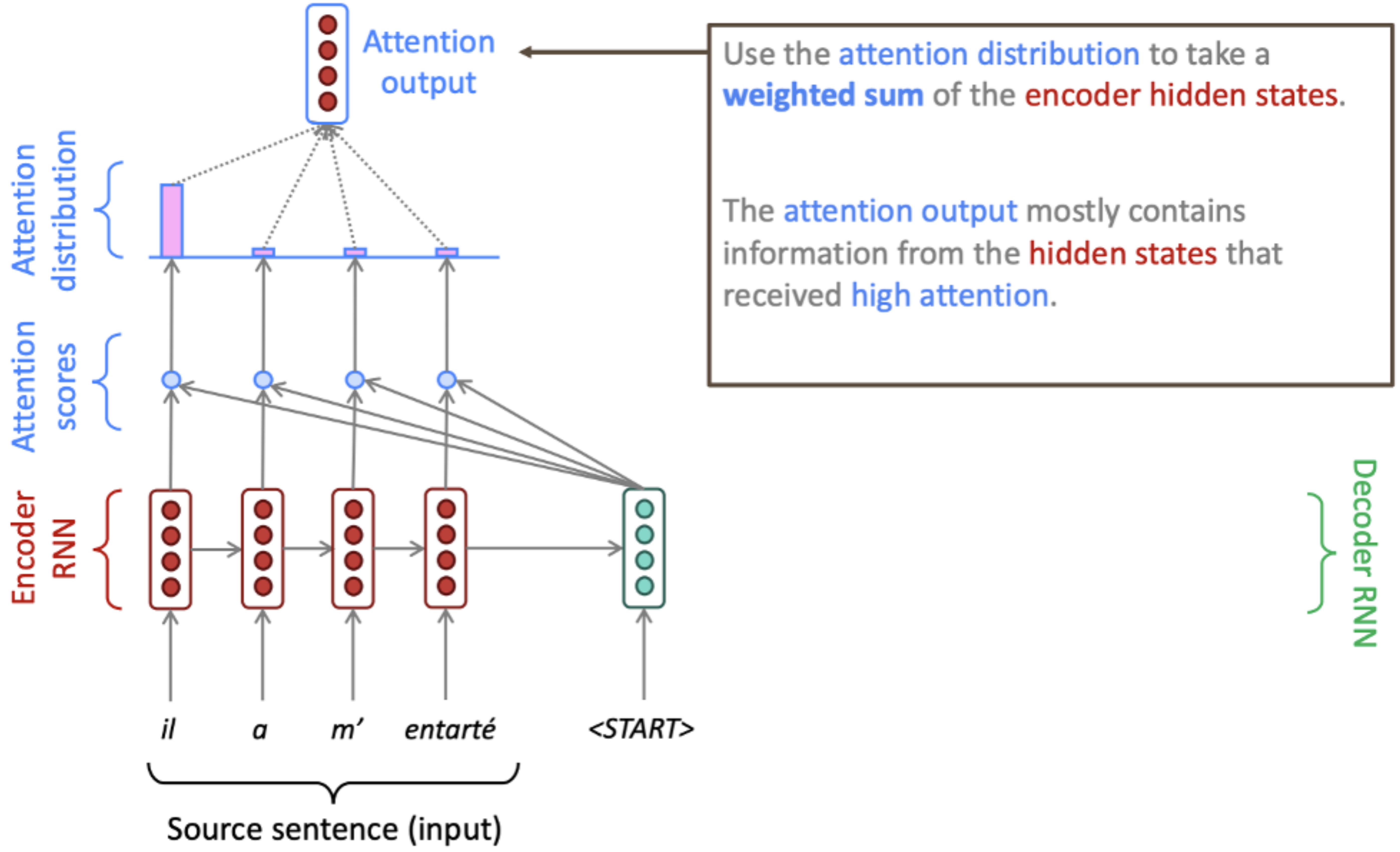
Attention for NMT



Attention for NMT



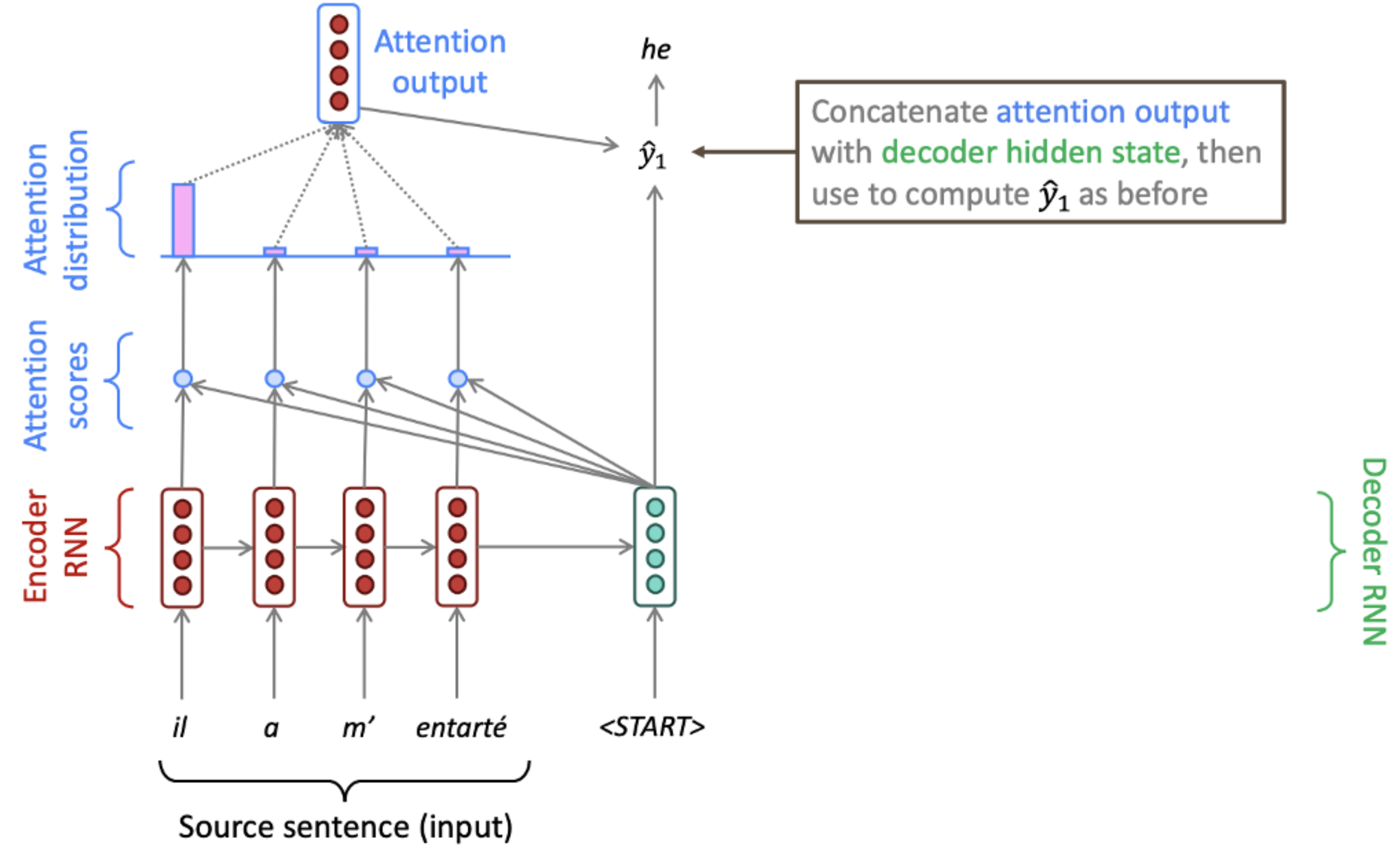
Attention for NMT



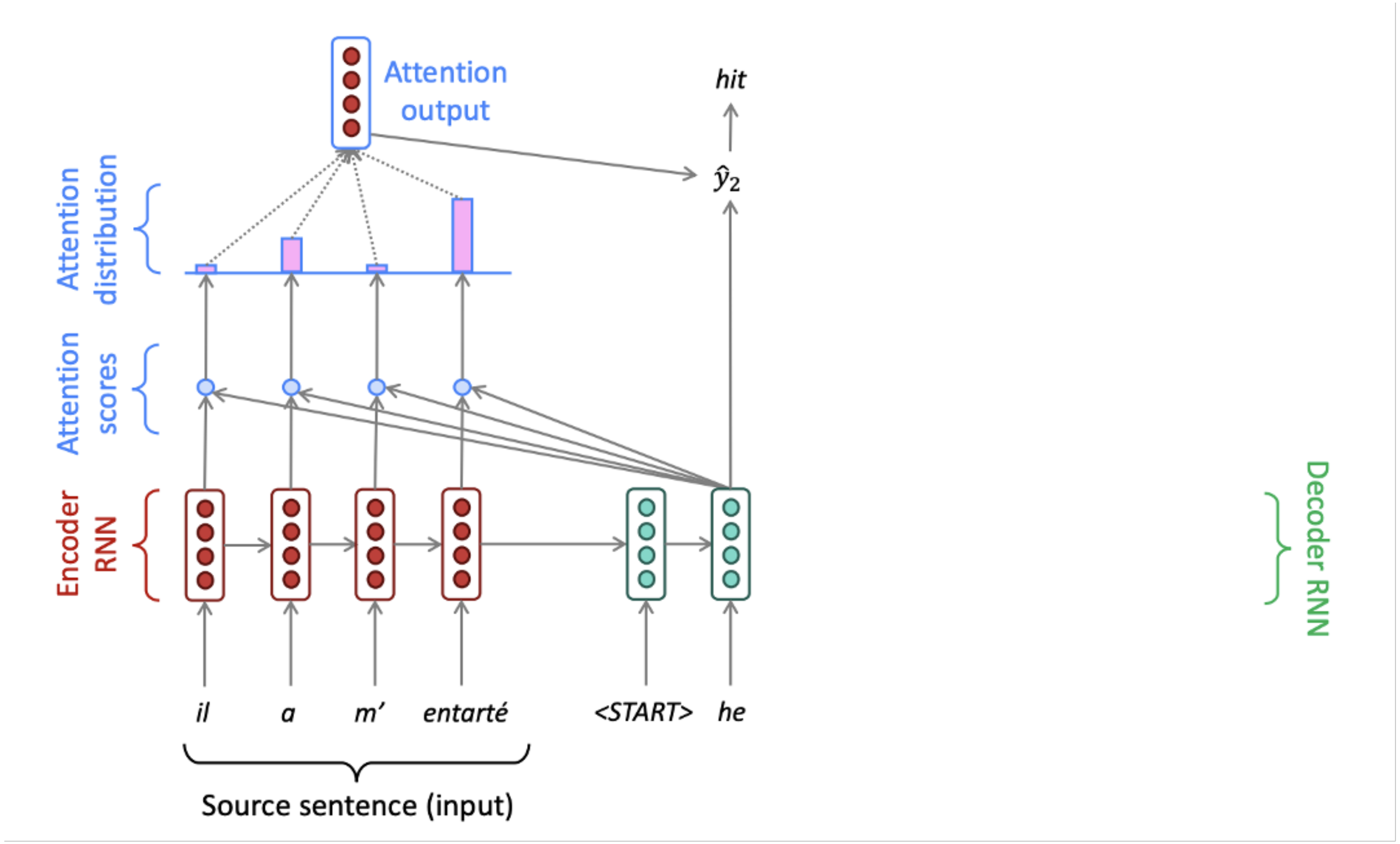
Use the attention distribution to take a **weighted sum** of the encoder hidden states.

The attention output mostly contains information from the hidden states that received high attention.

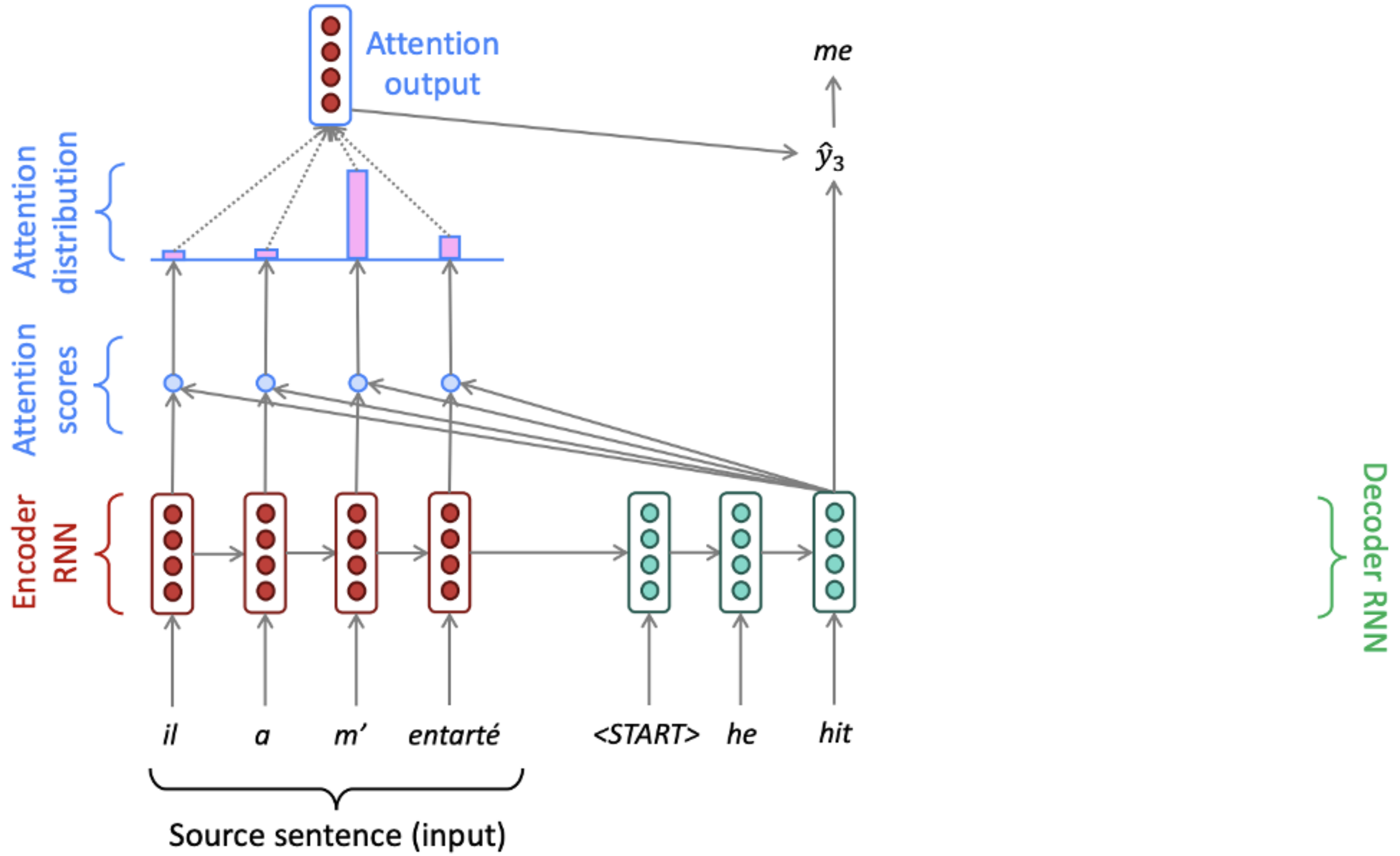
Attention for NMT



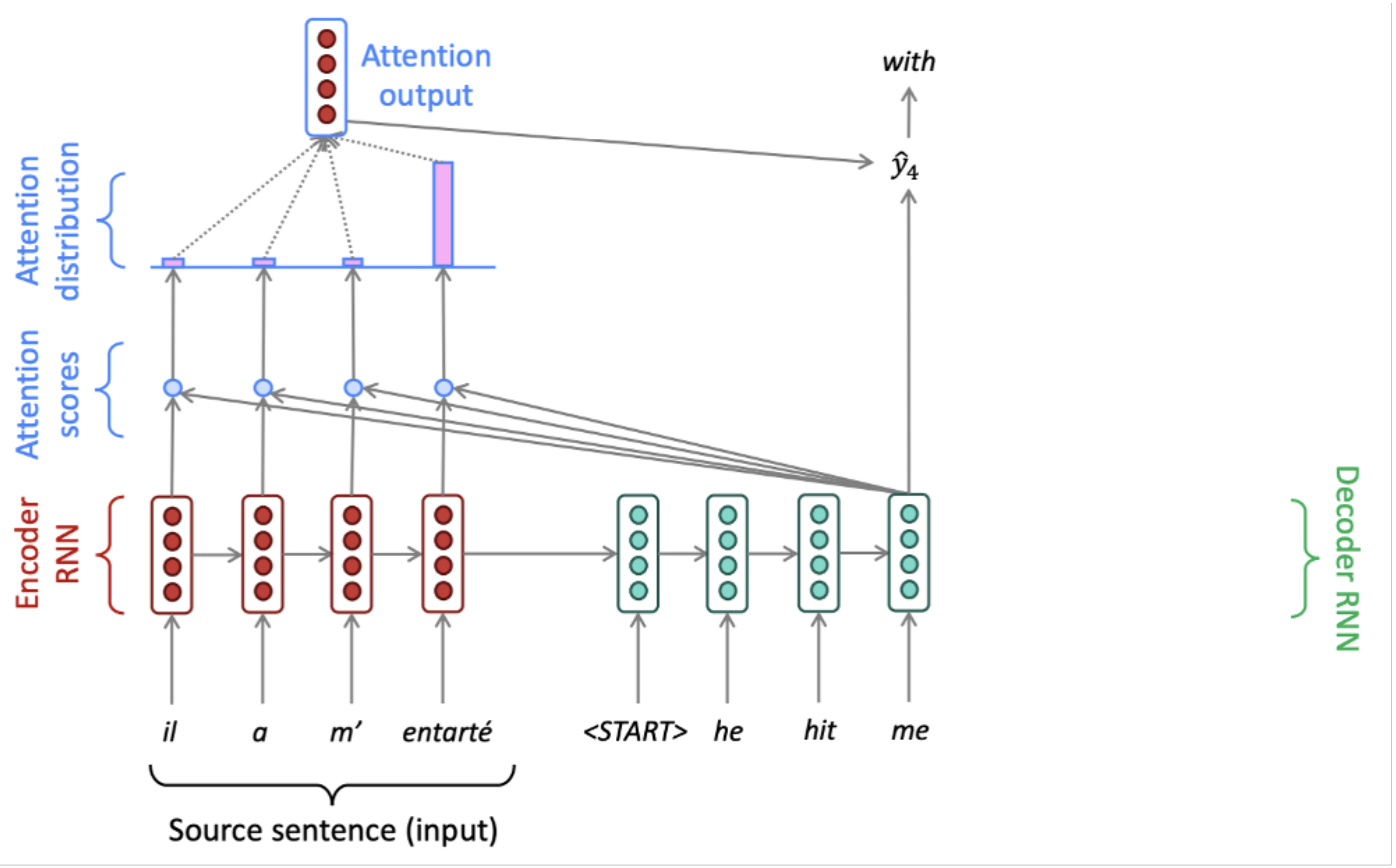
Attention for NMT



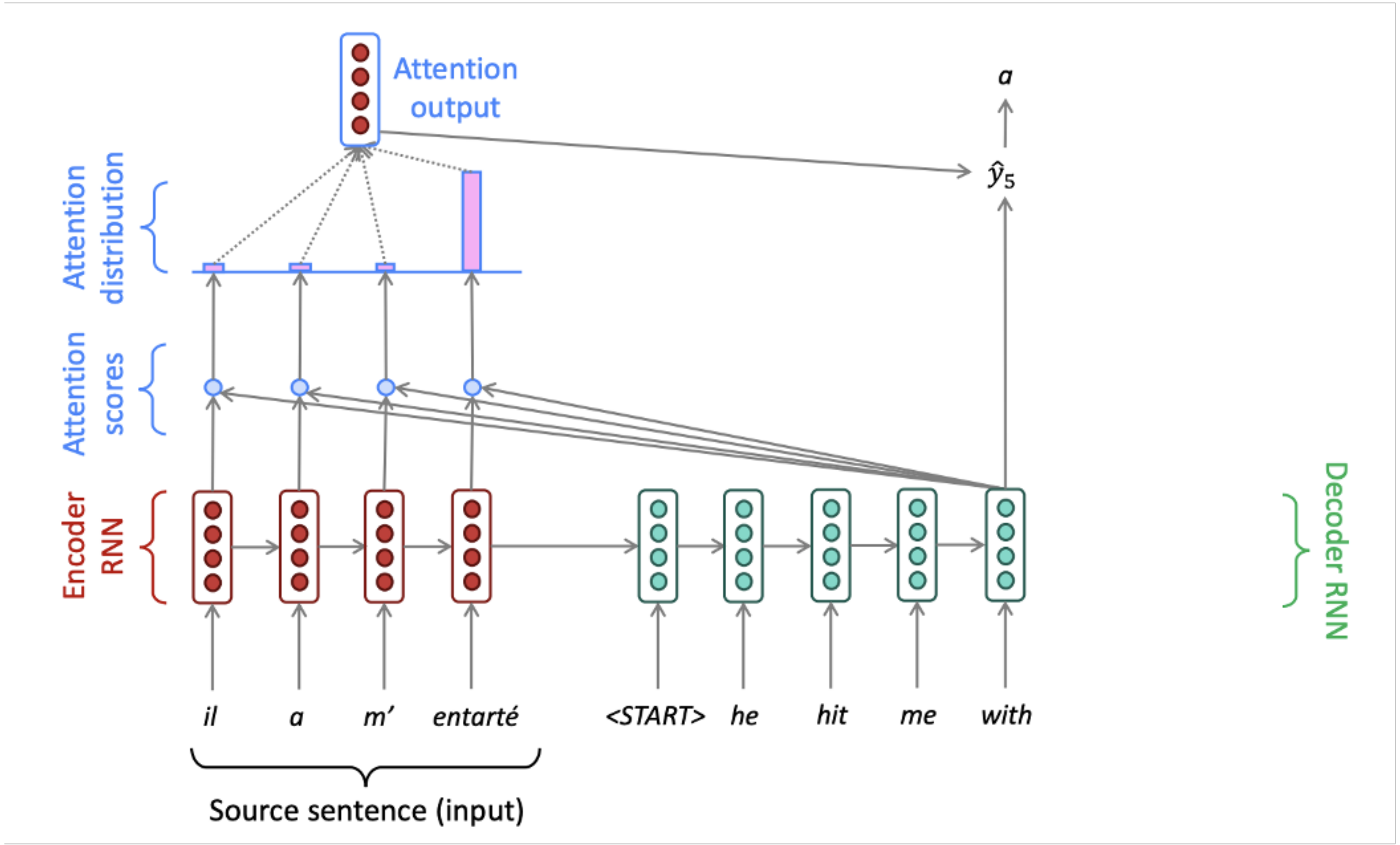
Attention for NMT



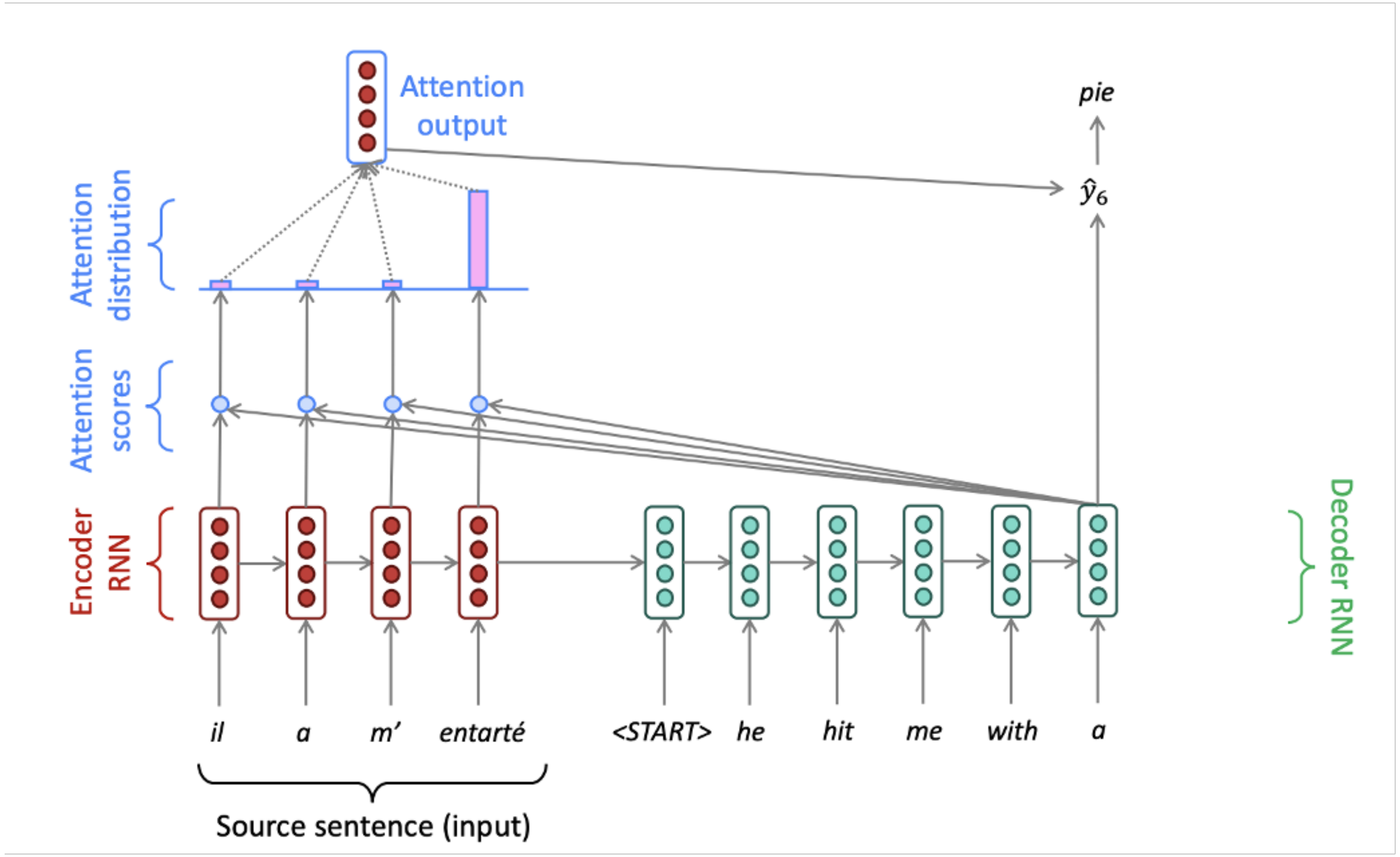
Attention for NMT



Attention for NMT



Attention for NMT



Attention in a Seq2Seq NMT Model

- We have **encoder hidden states** $h_1, \ldots, h_N$ (these are vectors)
- On timestep t, we have **decoder hidden state** s_t
- We get attention scores for this step using the **dot-product**

$$e^t = [s_t^T h_1, \ldots, s_t^T h_N]$$

- We compute **softmax** to get **attention distribution** (this is a probability distribution that sums to 1)

$$\alpha_t = \text{softmax}(e_t)$$

- We use that to create a weighted sum of encoder hidden states and get an attention **output**

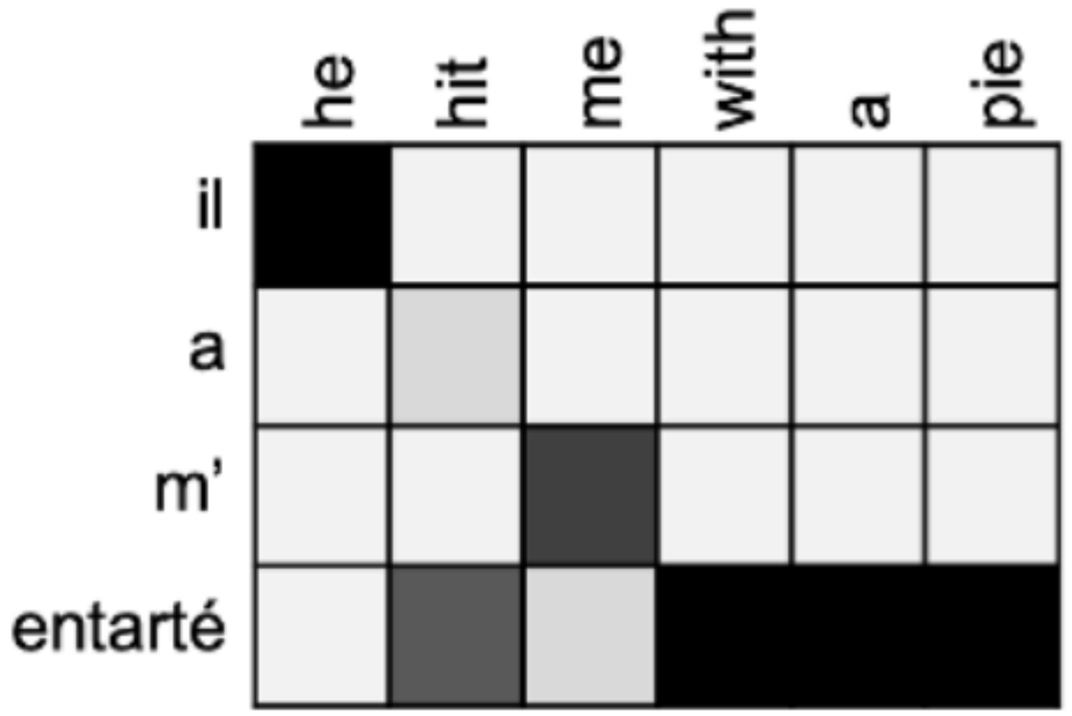
$$\alpha_t = \sum_{i=1}^{N} \alpha_i^t h_i$$

- We **concatenate the attention output with the decoder hidden state** and produce predictions for the next word

$$[\alpha_t; s_t]$$

Influence of Attention

- Solved the **bottleneck** problem
- Helps with the **vanishing gradients** — Q: Why?
- Provides a source for interpretability*



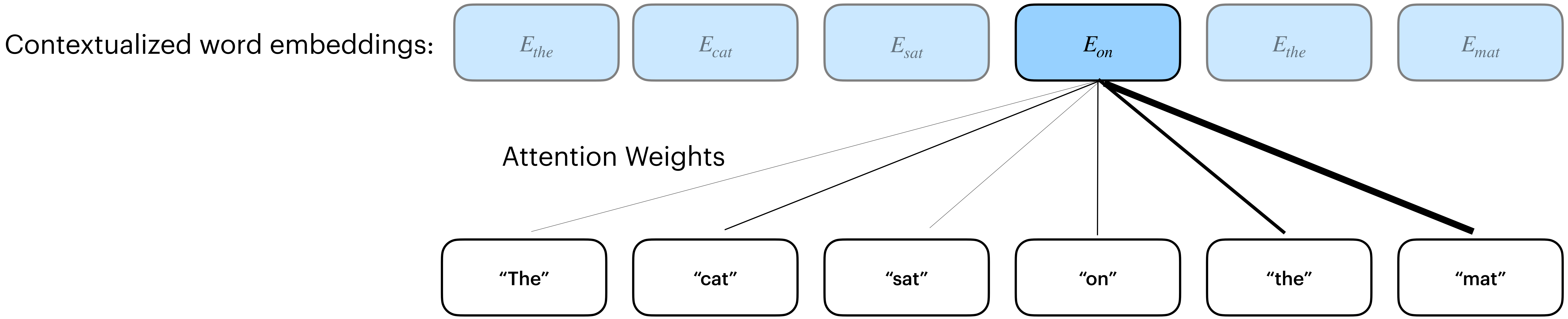
Sources & Notes

*But not always: If you want to learn about this debate check out:
Jain, S., & Wallace, B.C. (2019). Attention is not Explanation. North American Chapter of the Association for Computational Linguistics.
Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not Explanation. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 11–20, Hong Kong, China. Association for Computational Linguistics.

CENTER FOR HUMANITIES COMPUTING

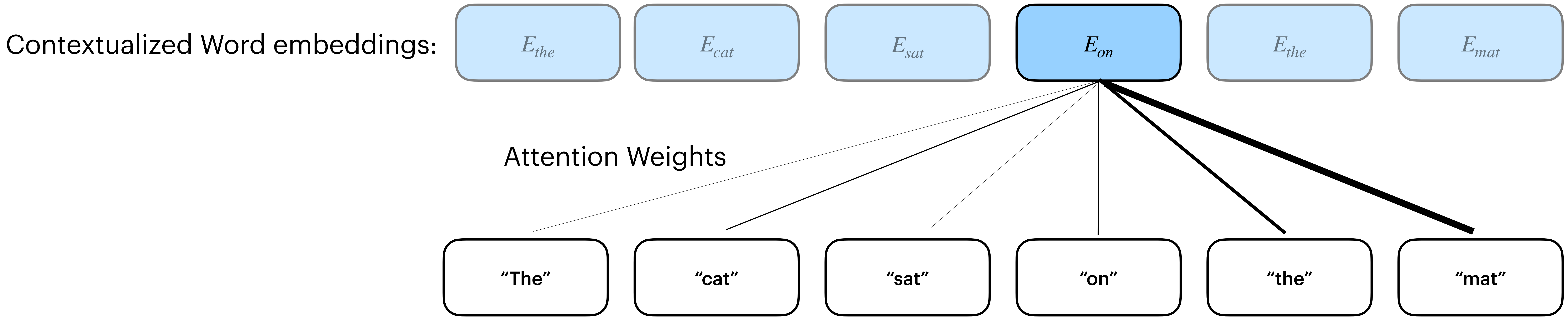
Intuition of Attention

- We previously discussed attention as a **weighted mean**
- We want to produce **contextualized** word embedding
- We want to update it based on words in its context with some **attention** score α



Intuition of Attention

- How could this look?
- It could be any function which produces an attention value $\alpha \in [0,1]$
  $f(E_{on}, E_{the}) \rightarrow \alpha$
- Dot-product attention is very popular for NMT, but **what is the problem?**



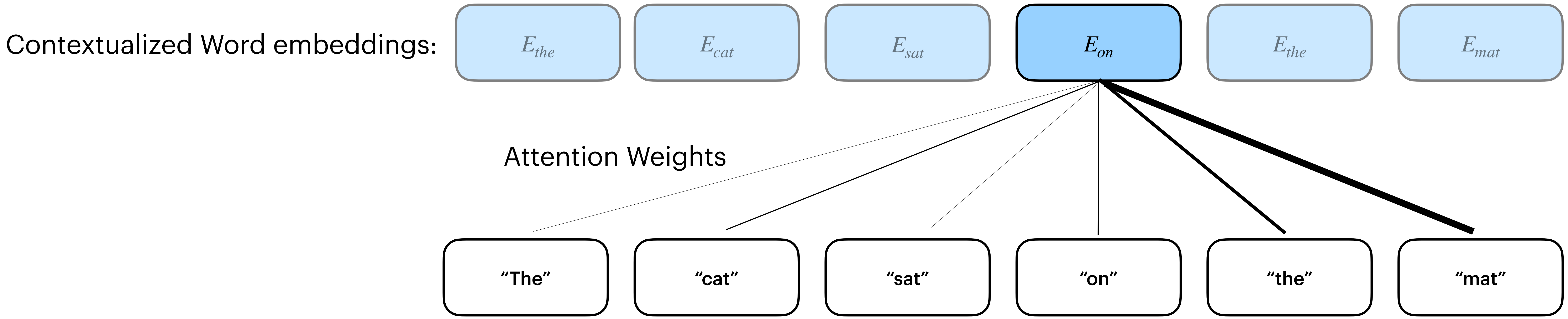
CENTER FOR
HUMANITIES
COMPUTING

Attention for sequence modelling

- The dot product between E_{on} and E_{the} is high if they are similar
  - We do not only want to attend to similar words!
    $\sigma(E_{on}^T E_{the})$
- Solution we create an transformation of each and call it **query** and **key**
  $\sigma(Q_{on}^T K_{the})$
  - Where $Q_{on}^T = E_{on}^T X$ (a linear transformation, aka. Neural network)
- We can intuitively see it as:
  query: "what am I looking for?"
  key: "what do I have to give?"

Dot-product Attention for Sequence Models

- Current formula:
  $\sigma(Q^T K)E$
- Adding normalization:
  $$\sigma\left(\frac{Q^T K}{\sqrt{d}}\right)E$$



Sources & Notes

The normalization ensures that the variance does not explore. To see this similar Q and K from a normal with mean 0 and variance 1 and see that $Q^T K$ has a variance $\sqrt{d}^{-1}$ where d is the dimension of the embeddings.

CENTER FOR HUMANITIES COMPUTING

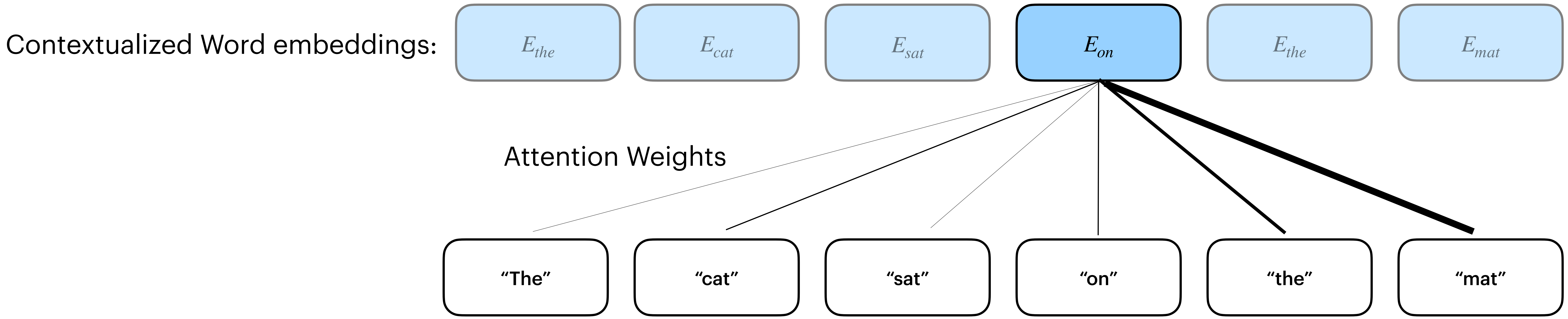
Sidenote on normalization

[4]To illustrate why the dot products get large, assume that the components of q and k are independent random variables with mean 0 and variance 1. Then their dot product, $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$, has mean 0 and variance d_k.

```
import numpy as np

n_samples = 10000
d = 200 # embedding dimensions

attention_weights = np.zeros(n_samples)

for i in range(n_samples):

    q = np.random.normal(0, 1, size=[d])
    k = np.random.normal(0, 1, size=[d])

    attention_weights[i] = q.T @ k


attention_weights.shape # (1000,)

attention_weights.mean() # 0.067  # close to zero
attention_weights.var()  # 195.34 # ~ 200 = d
attention_weights.std()  # 13.976 # ~sqrt(d) = 14.14
```

Problem with a weighted mean?

- Current formula:
  $\sigma(Q^T K)E$
- Adding normalization:
  $\sigma\left(\frac{Q^T K}{\sqrt{d}}\right)E$



Sources & Notes

The normalization ensures that the variance does not explore. To see this similar Q and K from a normal with mean 0 and variance 1 and see that $Q^T K$ has a variance $\sqrt{d}^{-1}$ where d is the dimension of the embeddings.

CENTER FOR HUMANITIES COMPUTING

Lack of positional information

Classification:
Is this a question?

are
you
today
happy

"you are happy today(!)"

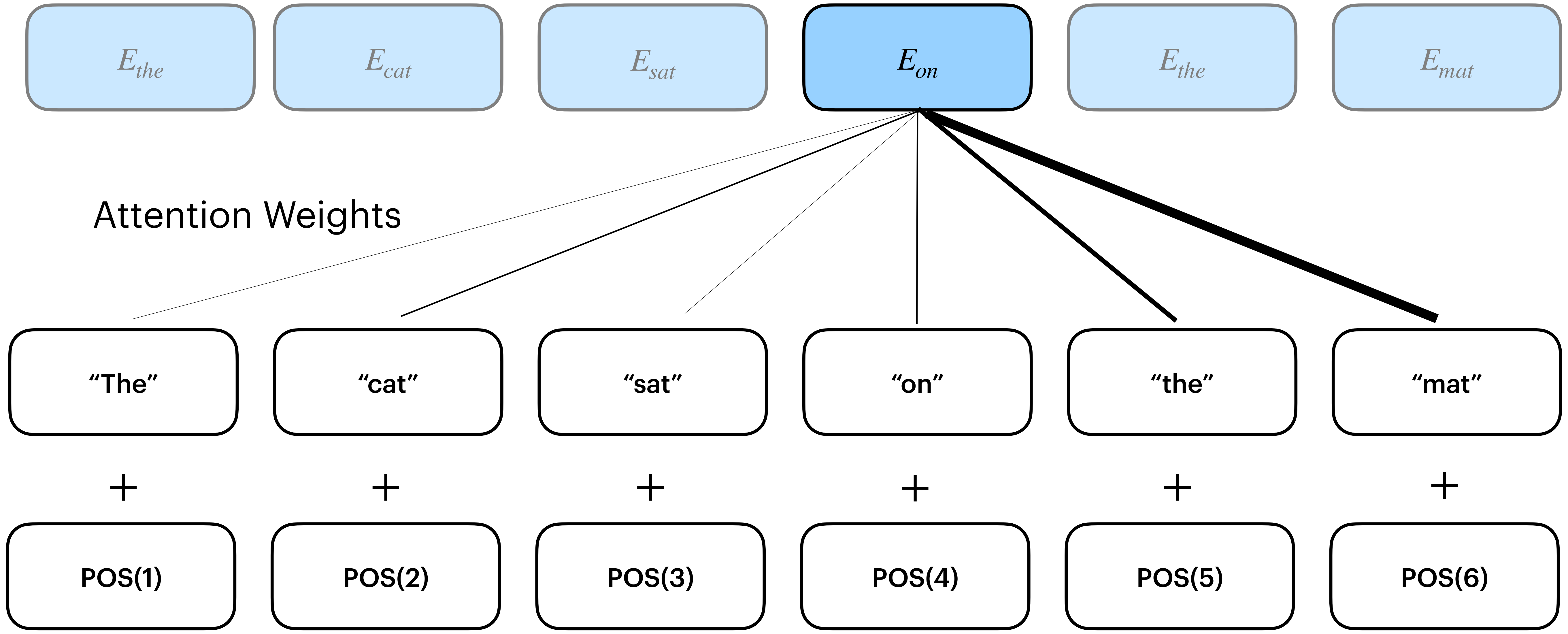
"Are you happy today(?)"

Text generation:
What is the next word

the
to
teacher
told
open
the
students

"The teacher told the students to open their ___"

Positional Encodings

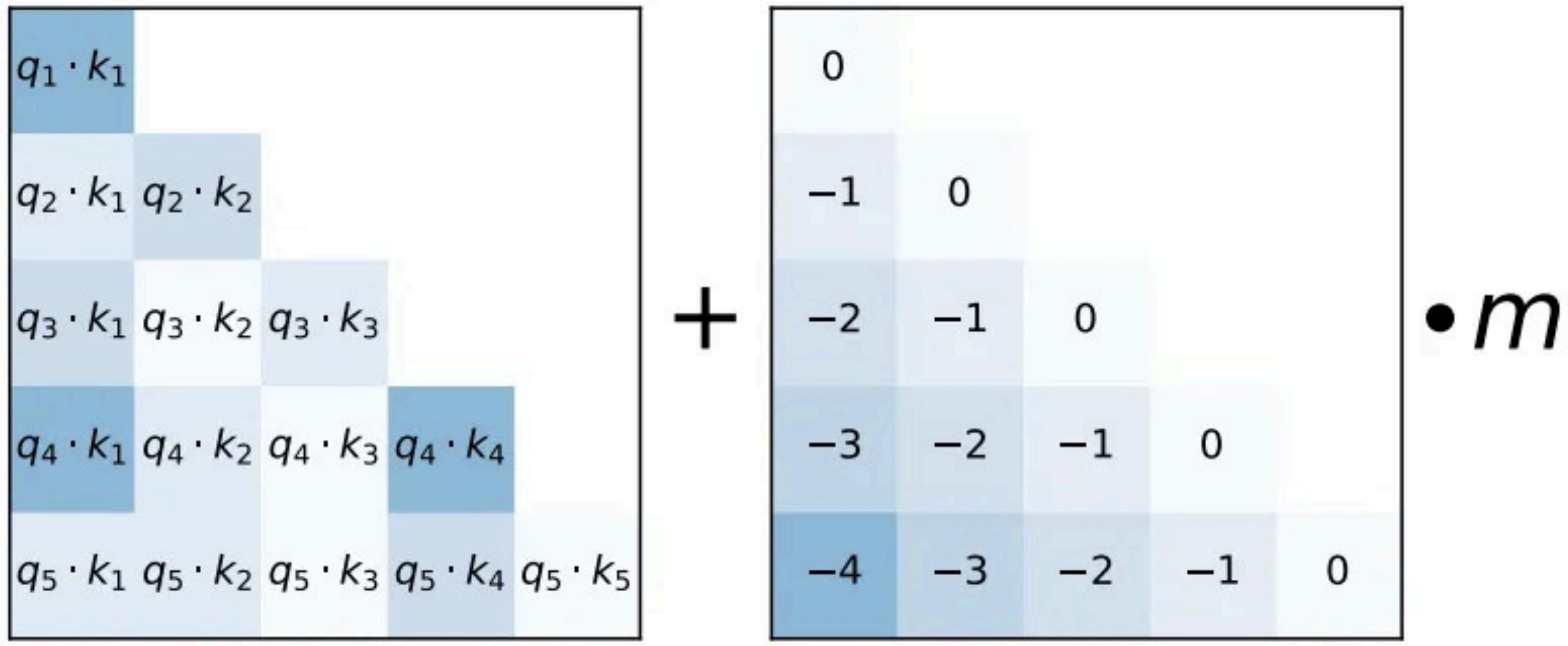


- How do we construct the positional encoding?
- Brute force approach: Create it as a word vector (we can just learn it using gradient descent)
- Idea: typically words which appear closer are more relevant
  - Construct positional embeddings to appear “closer” when they are close*

This idea is also expressed in Alibi positional embeddings

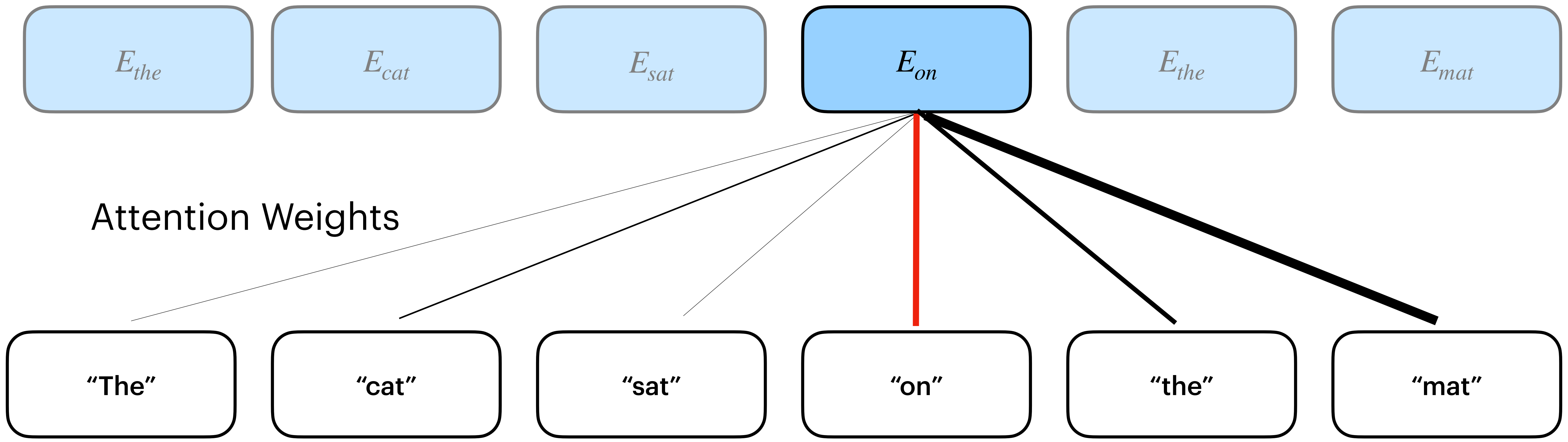
Alibi positional Embeddings

- Some positional encodings work directly with the attention weights:
- Same Idea: typically words which appear closer are more relevant



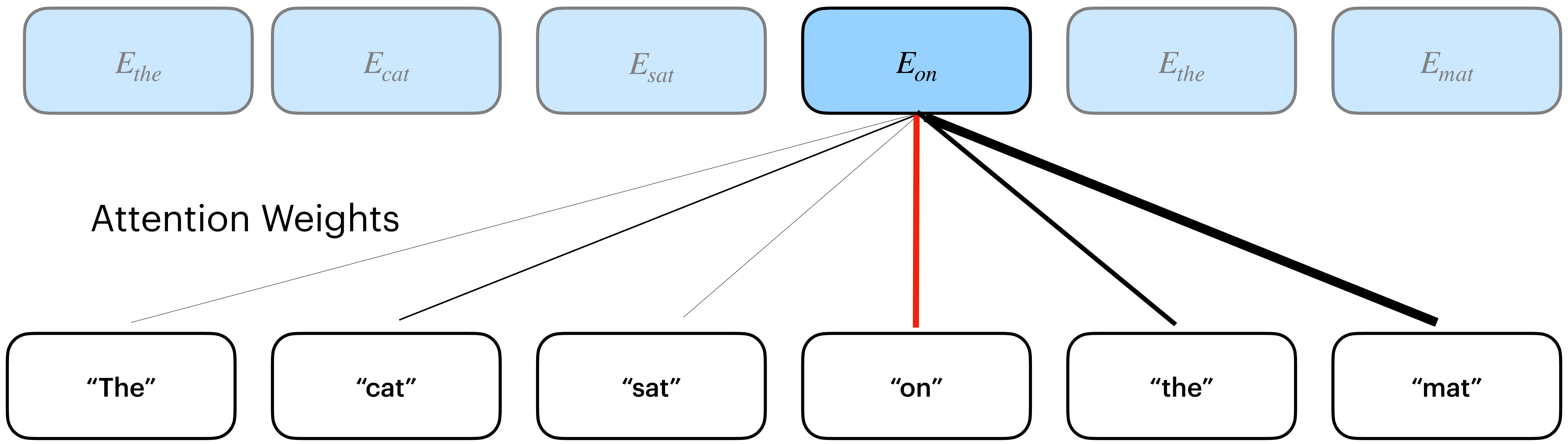
Multi-head **Self-**Attention

- Why is it called “self-“?
- Hard Question: Why might you want to pay attention to the token itself?



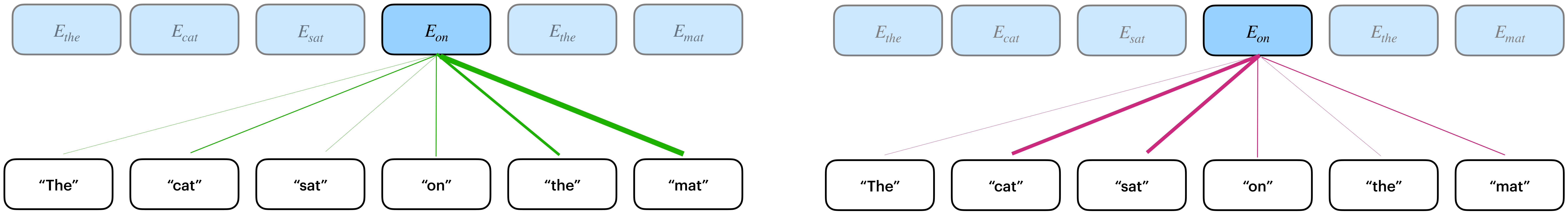
Multi-head **Self**-Attention

- Why is it called “self-“?
- Hard Question: Why might you want to pay attention to the token itself?
  - **Allows downweighting information** by highly attending to itself
  - It makes it a simple matrix multiplication



Multi-head Self-Attention

- You might need to attend to different things
- Easy to run in parallel
- Example



- Perspective: Very similar to multiple filters in CNN

Interpreting Attention

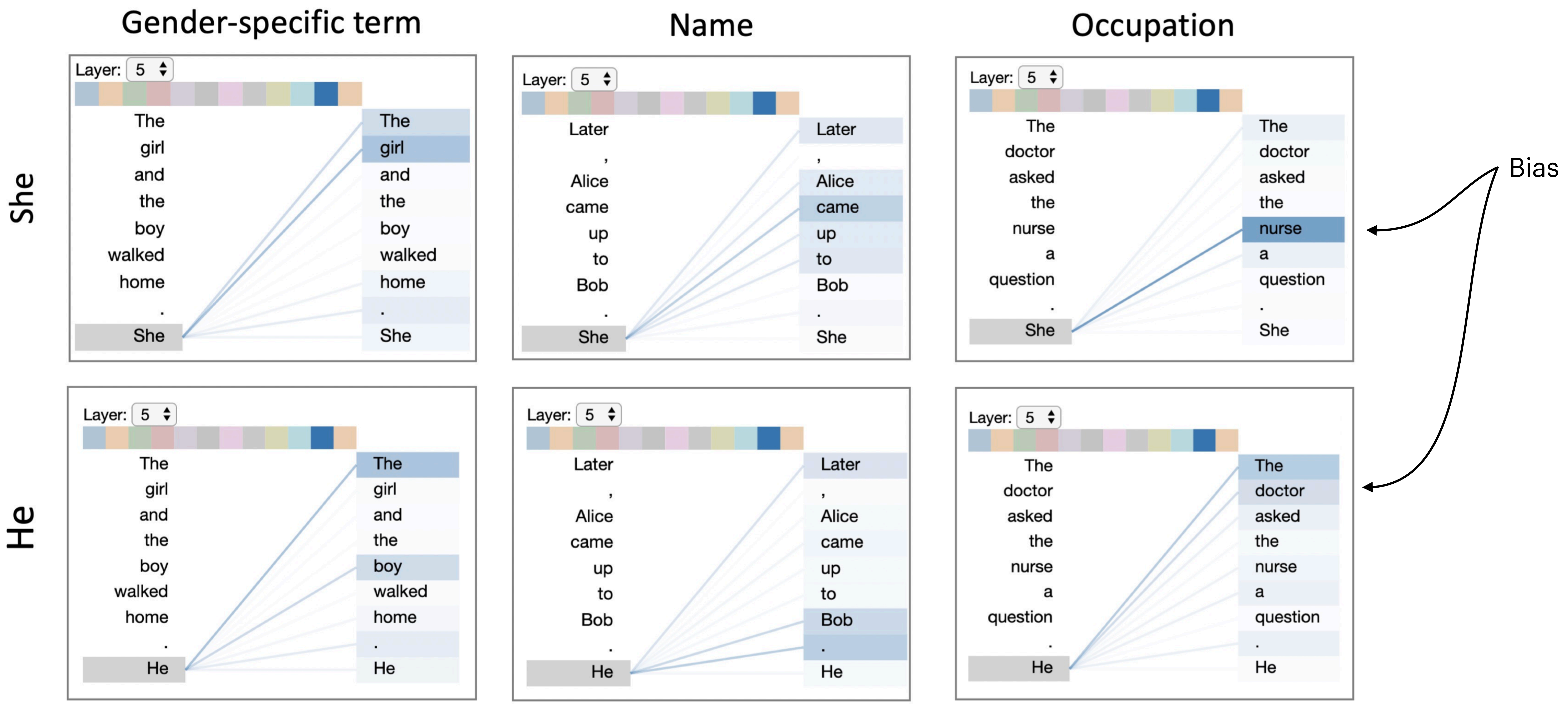


Figure 4: Attention pattern in GPT-2 related to coreference resolution suggests the model may encode gender bias.

Source: https://aclanthology.org/P19-3007.pdf

Attention: How is does it look when you put it all together

- Vaswani, et al. (2017) Attention
- https://colab.research.google.com/drive/1rPk3ohrmVclqhH7uQ7qys4oznDdAhpzF#scrollTo=qegb9M0KbnRK

Next up:

- The remainder of the Transformers archtitecture
- Back to Language generation

Additional Visual Blogs

- The Illustrated Transformer – Jay Alammar – Visualizing machine learning one concept at a time.
- The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)
- The Illustrated GPT-2 (Visualizing Transformer Language Models) – Jay Alammar – Visualizing machine learning one concept at a time.
- Illustrated Guide to Transformers- Step by Step Explanation | by Michael Phi