

# Lecture 05: Machine Translation

MADE, Moscow  
22.04.2020

**Radoslav Neychev**

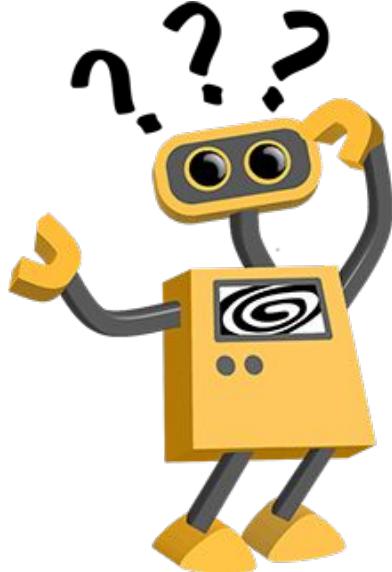
# Outline

1. Problem Statement
2. Before Deep Learning
3. Neural Machine Translation (NMT)
  - Seq2Seq
  - BeamSearch
4. Attention
5. Q & A

# Machine Translation

I'M GOING TO THE THEATER = ICH GEHE INS THEATER

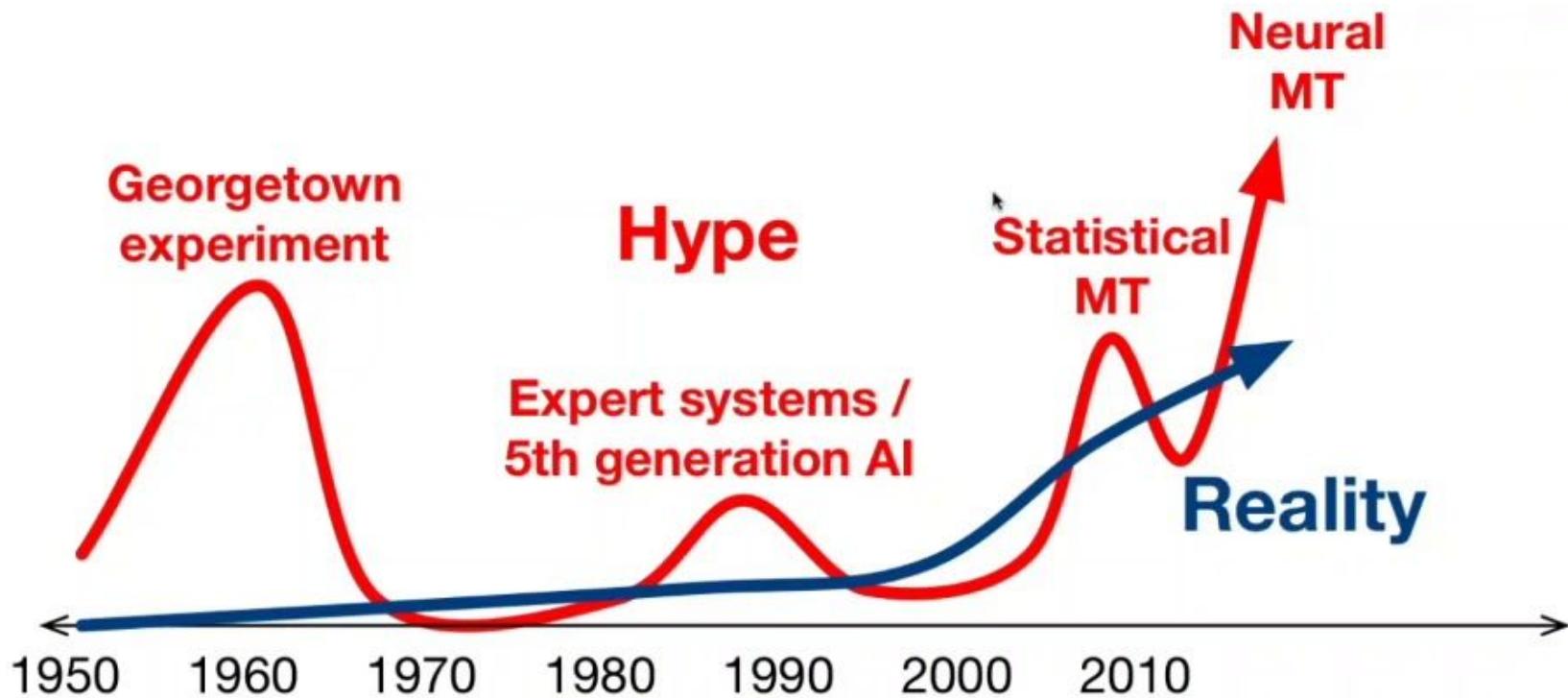
I'M GOING TO THE CINEMA = ICH GEHE INS KINO



???

KINO

# Historical overview



# Before Deep Learning

# 1950s: first Machine Translation

1. Georgetown experiment (7 Jan 1954)
  - a. Automatic Russian-English translation of 60 sentences
  - b. 250 vocabulary articles
  - c. 6 grammar rules
  - d. Calculated on Mainframe IBM 701
2. The same experiment in the USSR (1954 too)
  - a. Rule-based translation
  - b. Calculated on BESM

# 1990-2010: Statistical Machine Translation

We want to find best English sentence  $y$ , given French sentence  $x$

$$\operatorname{argmax}_y P(y|x)$$

Let's use Bayes Rule to break this down into two components:

$$= \operatorname{argmax}_y P(x|y)P(y)$$



# 1990-2010: Statistical Machine Translation

How to learn translation model from the parallel corpus?

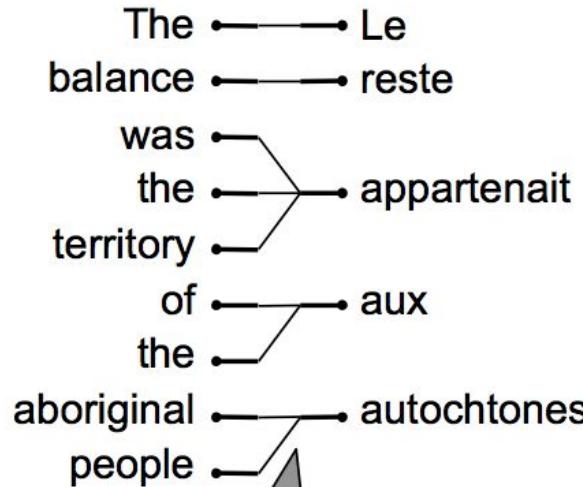
Let's calculate

$$P(x, a|y)$$

Where **a** is an **alignment** (word-level correspondence between French sentence x and English sentence y)

# 1990-2010: Statistical Machine Translation

Alignment can be: **many-to-one**

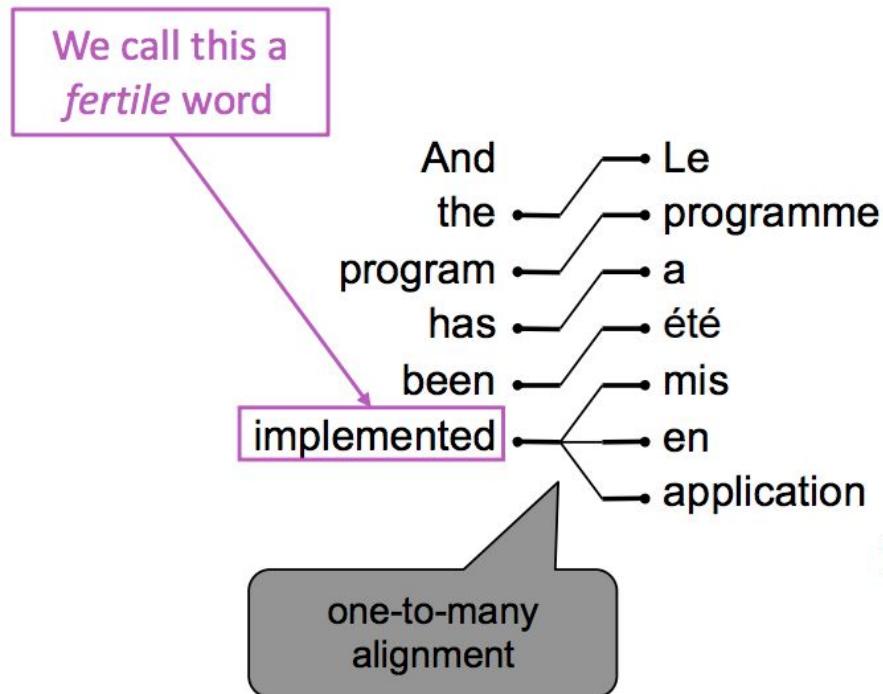


many-to-one  
alignments

	Le	reste	appartenait	aux	autochtones
The	■				
balance		■			
was			■		
the				■	
territory					■
of				■	
the					■
aboriginal					■
people					■

# 1990-2010: Statistical Machine Translation

Alignment can be: **one-to-many**

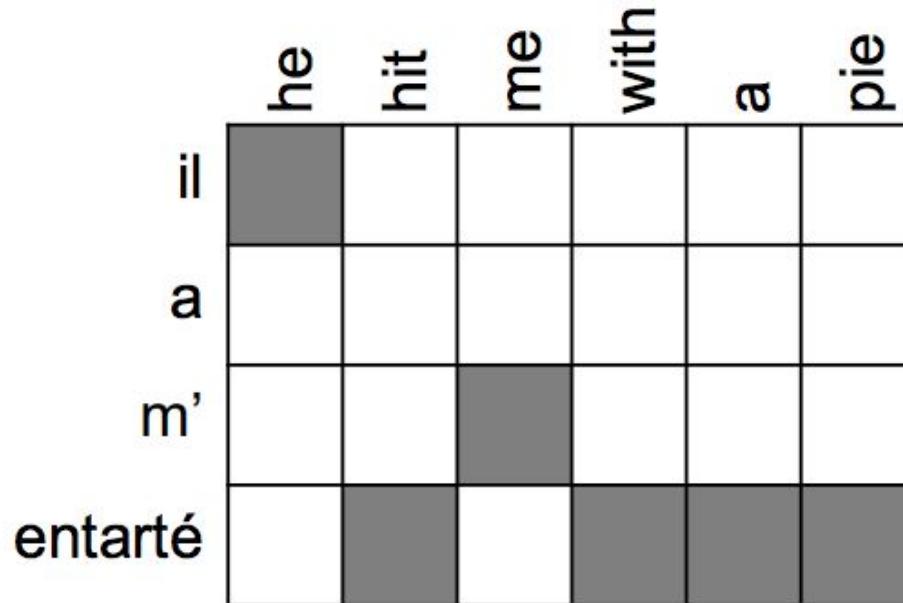
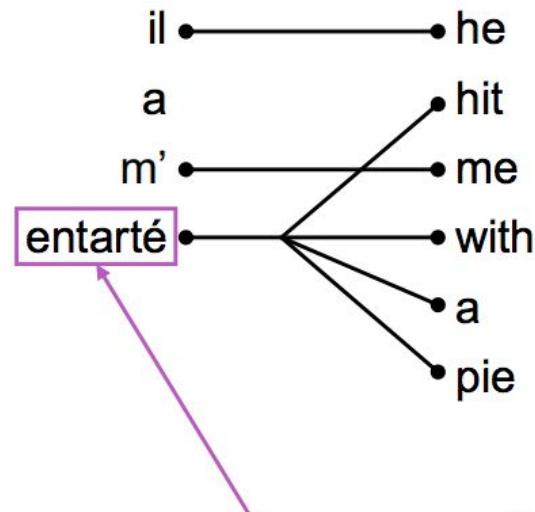


A grid matrix representing word alignment. The columns are English words and the rows are French words. Shaded cells indicate aligned pairs. The grid shows that "implemented" aligns with both "mis" and "en".

	Le	programme	a	été	mis	en	application
And							
the							
program							
has							
been							
implemented					mis	en	application

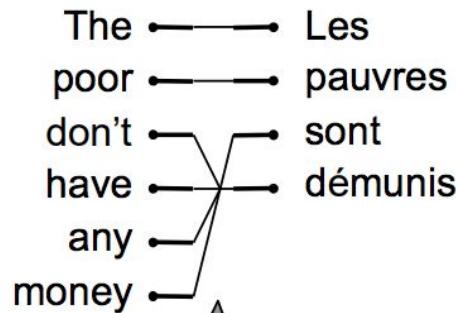
# 1990-2010: Statistical Machine Translation

Some words are very fertile!

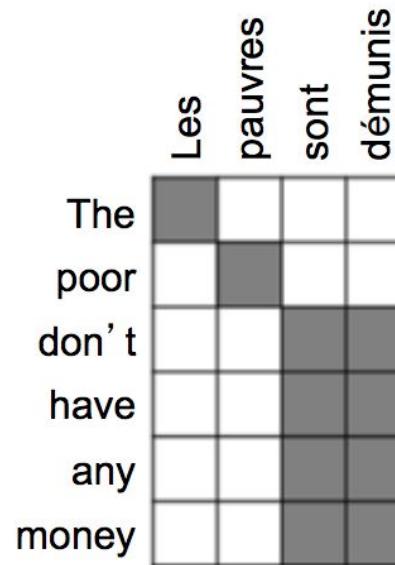


# 1990-2010: Statistical Machine Translation

Alignment can be: **many-to-many**

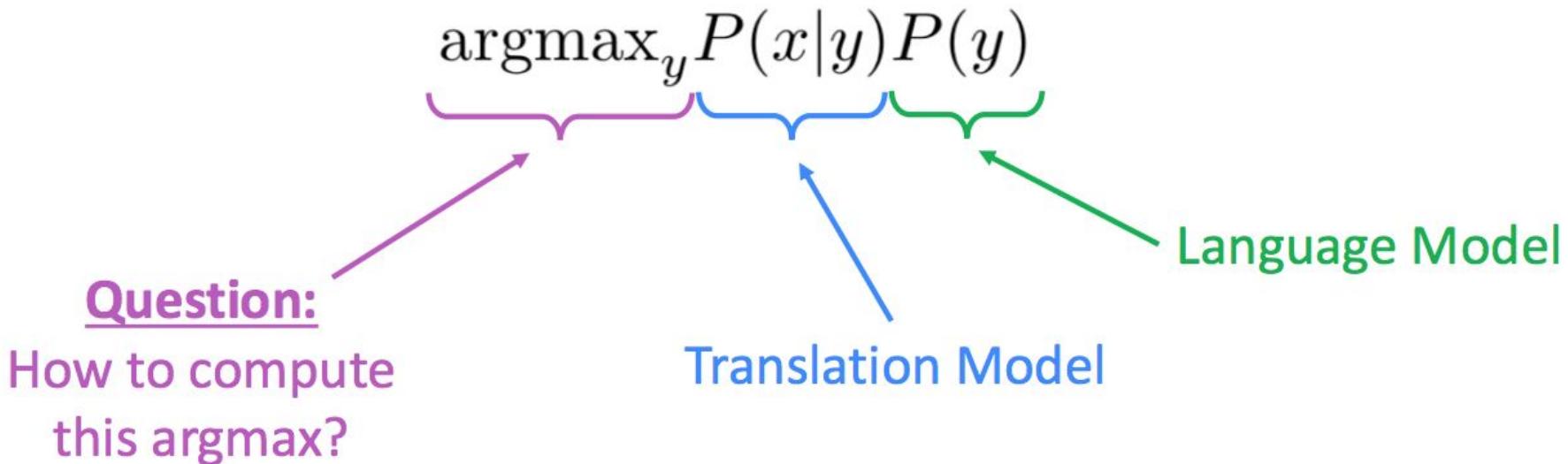


many-to-many  
alignment



phrase  
alignment

# 1990-2010: Statistical Machine Translation

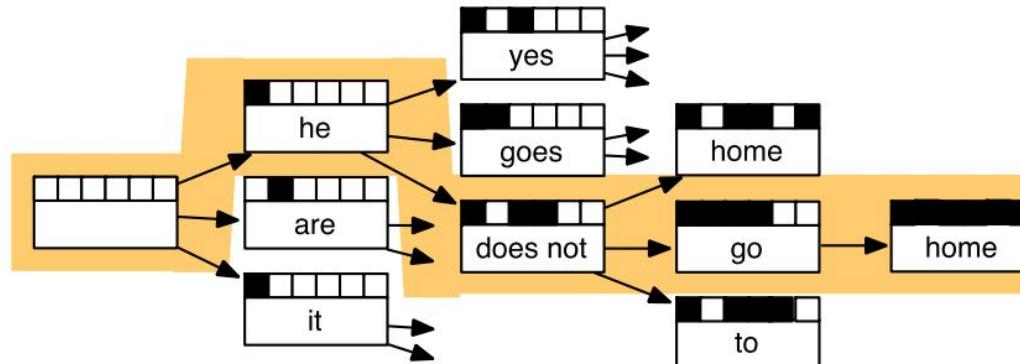


Enumerate every possible  $y$  and calculate the probability? No!

Use a heuristic search algorithm to search for the best translation,  
discarding hypotheses that are too low-probability

# 1990-2010: Statistical Machine Translation

er	geht	ja	nicht	nach	hause
he	is	yes	not	after	house
it	are	is	do not	to	home
, it	goes	, of course	does not	according to	chamber
, he	go	,	is not	in	at home
it is		not		home	
he will be		is not		under house	
it goes		does not		return home	
he goes		do not		do not	
	is		to		
	are		following		
	is after all		not after		
	does		not to		
	not				
	is not				
	are not				
	is not a				



# 1990-2010: Statistical Machine Translation

- Systems had many separately-designed subcomponents
- Lots of feature engineering
- Need to design features to capture particular language phenomena
- Require compiling and maintaining extra resources (tables of equivalent phrases)

# 1990-2010: Statistical Machine Translation

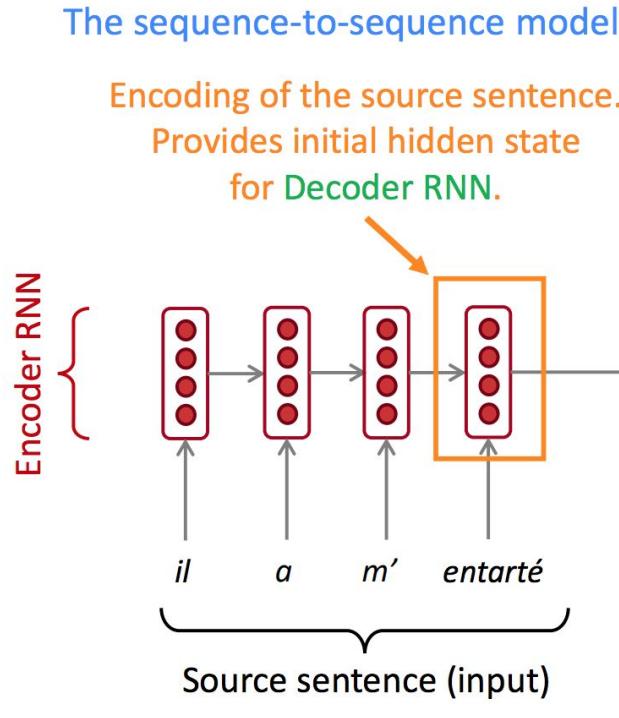
- Systems had many separately-designed subcomponents
- Lots of feature engineering
- Need to design features to capture particular language phenomena
- Require compiling and maintaining extra resources (tables of equivalent phrases)
- Lots of human effort to maintain
- Repeated effort for each language pair!

# Neural Machine Translation

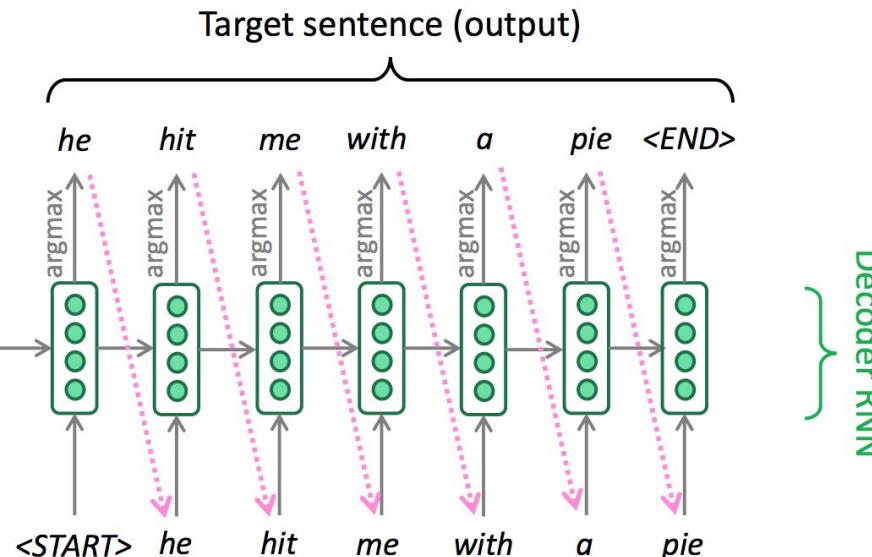
# What is Neural Machine Translation?

- Neural Machine Translation (NMT) is a way to do Machine Translation with a single neural network
- The neural network architecture is called sequence-to-sequence (aka **seq2seq**) and it involves two **RNNs**.

# Neural Machine Translation



Encoder RNN produces  
an encoding of the  
source sentence.



Decoder RNN is a Language Model that generates target sentence, *conditioned on encoding*.

Note: This diagram shows test time behavior:  
decoder output is fed in ..... as next step's input

# NMT: how does it work?

- NMT directly calculates  $P(y|x)$  ( $y$  - target sentence,  $x$  - source sentence)

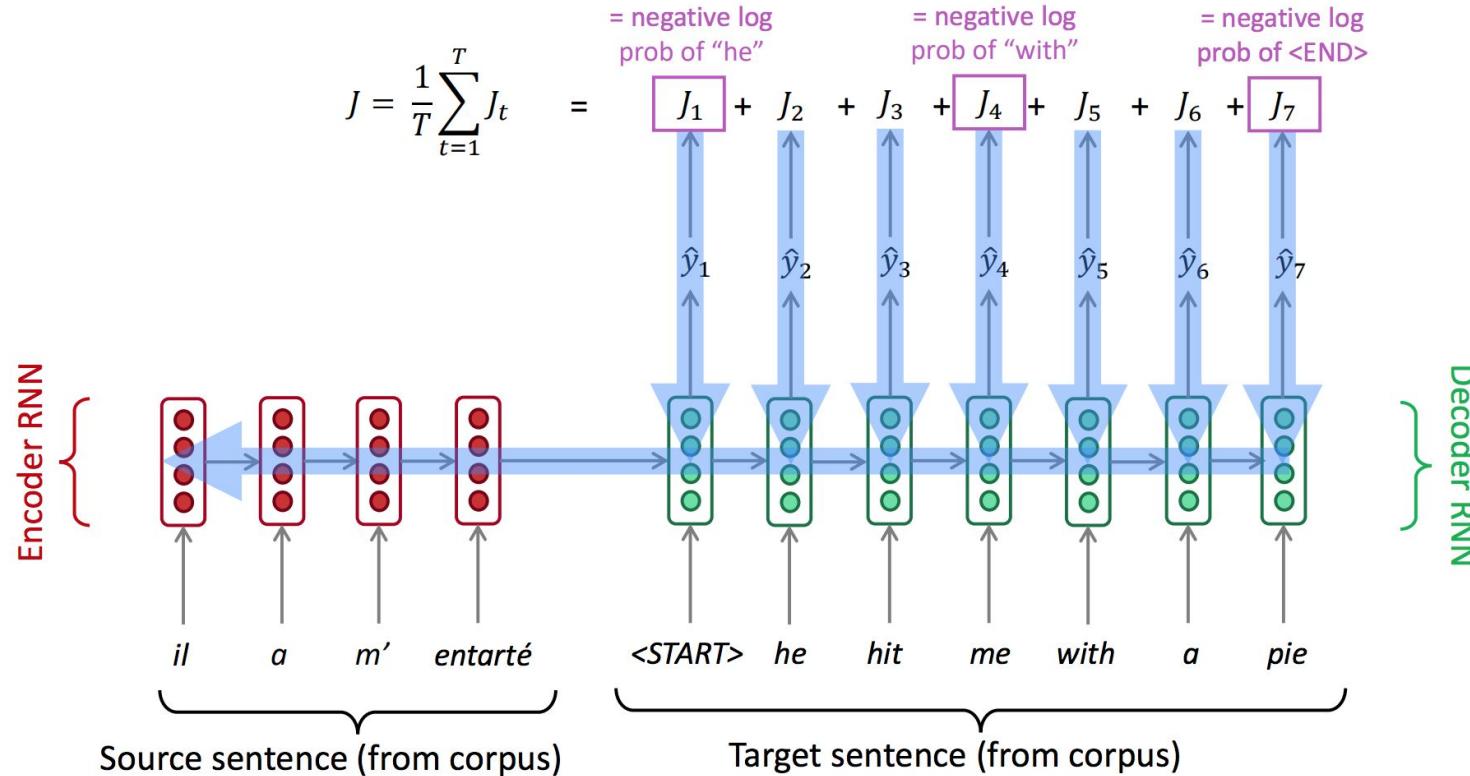
$$P(y|x) = P(y_1|x) P(y_2|y_1, x) P(y_3|y_1, y_2, x) \dots P(y_T|y_1, \dots, y_{T-1}, x)$$



Probability of next target word, given target words so far and source sentence  $x$

- To train it we need a huge parallel corpus.

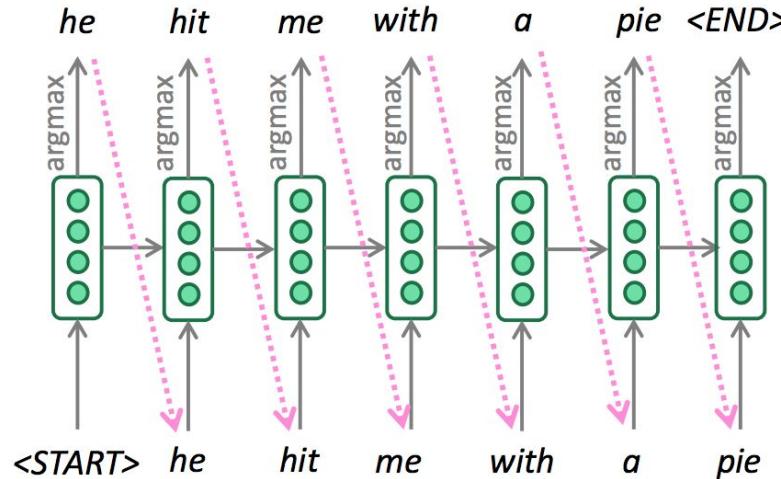
# Neural Machine Translation



Seq2seq is optimized as a single system.  
Backpropagation operates “end-to-end”.

# Greedy decoding

- We saw how to generate (or “decode”) the target sentence by taking argmax on each step of the decoder



- This is greedy decoding (take most probable word on each step)

# Greedy decoding: problems

- Greedy decoding has no way to undo decisions!

Input: *il a m'entarté*      (*he hit me with a pie*)

→ *he* \_\_\_\_

→ *he hit* \_\_\_\_

→ *he hit a* \_\_\_\_      (whoops! no going back now...)

# Exhaustive search

- Ideally we want to find a (length T) translation  $y$  that maximizes

$$P(y|x) = P(y_1|x) P(y_2|y_1, x) P(y_3|y_1, y_2, x) \dots, P(y_T|y_1, \dots, y_{T-1}, x)$$

$$= \prod_{t=1}^T P(y_t|y_1, \dots, y_{t-1}, x)$$

- We could try computing all possible sequences  $y$ . That's too slow!

- On each step of decoder, keep track of the k most probable partial translations (which we call hypotheses)
- k is the beam size (in practice around 5 to 10)
- A hypothesis has a score which is its log probability:

$$\text{score}(y_1, \dots, y_t) = \log P_{\text{LM}}(y_1, \dots, y_t | x) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

- We search for high-scoring hypotheses, tracking top k on each step
- Beam search is not guaranteed to find optimal solution

# Beam search decoding: example

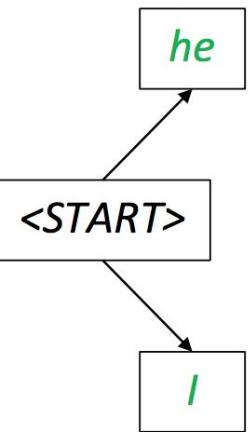
Beam size = k = 2. Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$

<START>

# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$

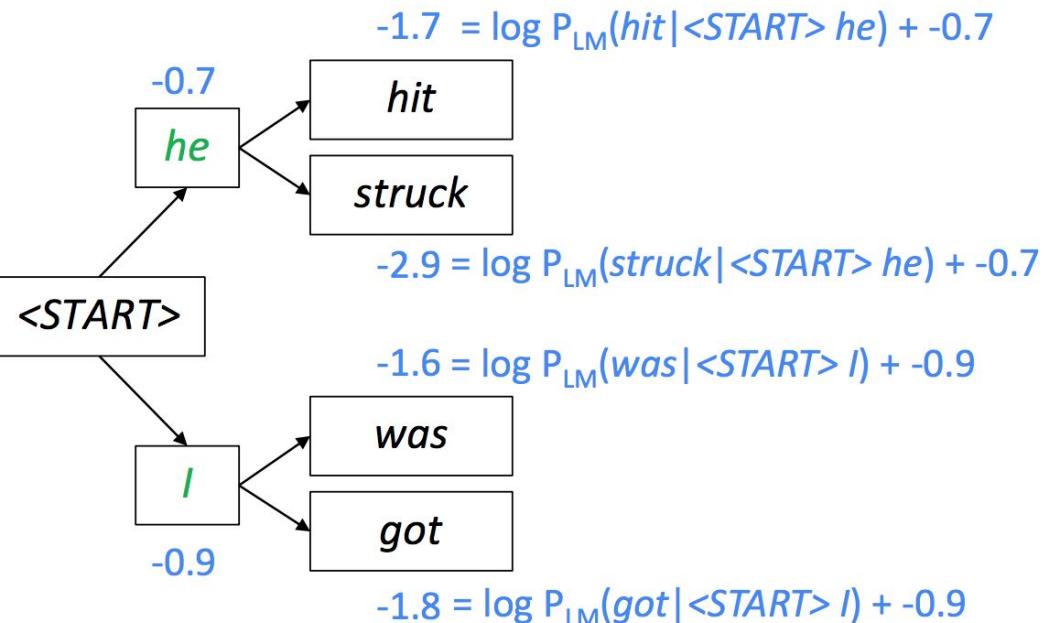
$$-0.7 = \log P_{\text{LM}}(he | <\text{START}>)$$



$$-0.9 = \log P_{\text{LM}}(I | <\text{START}>)$$

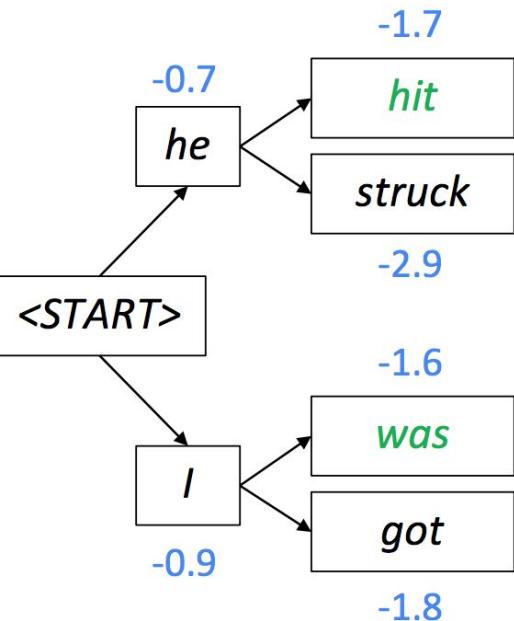
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



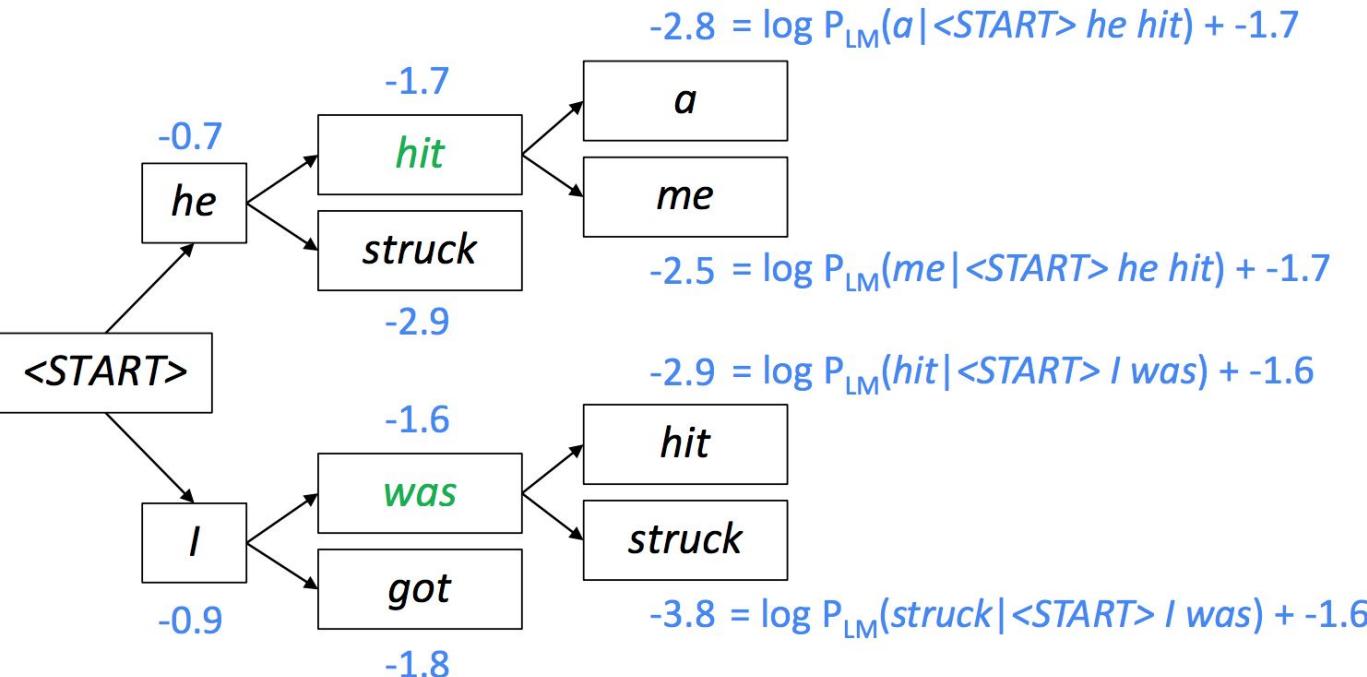
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



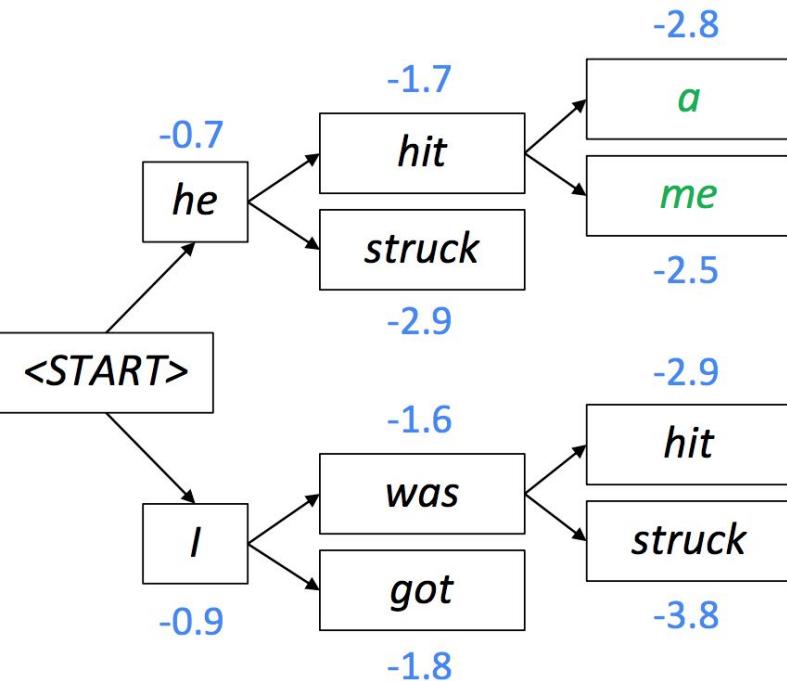
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



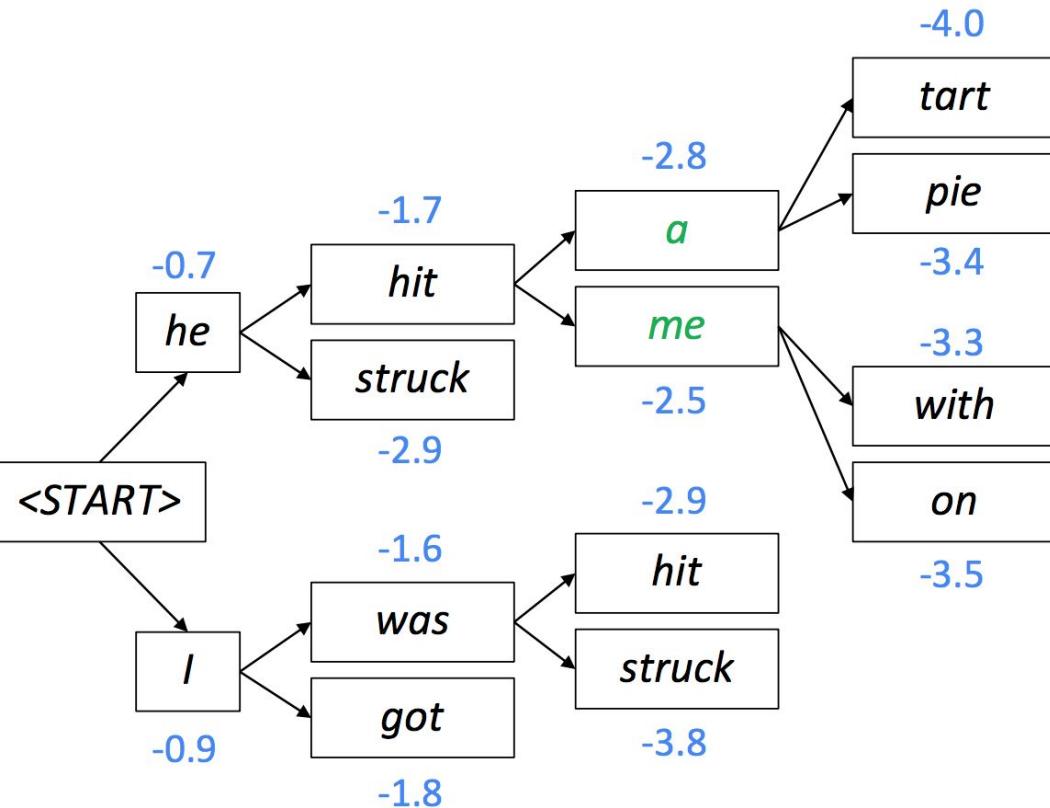
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



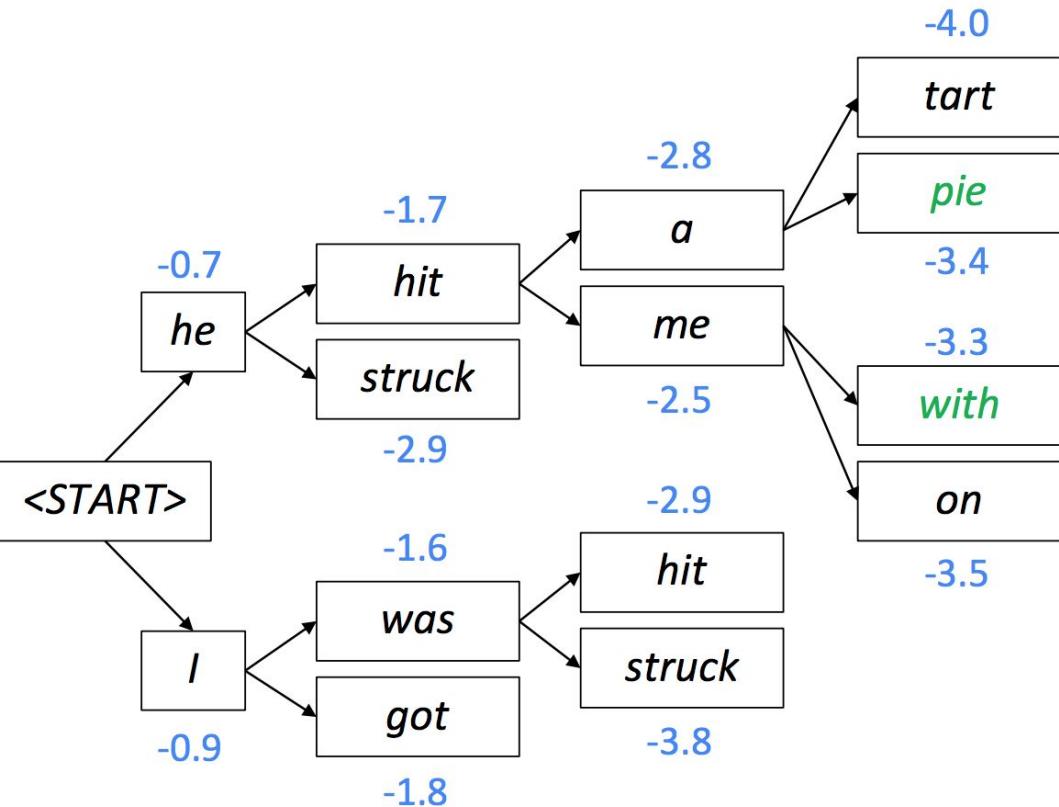
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



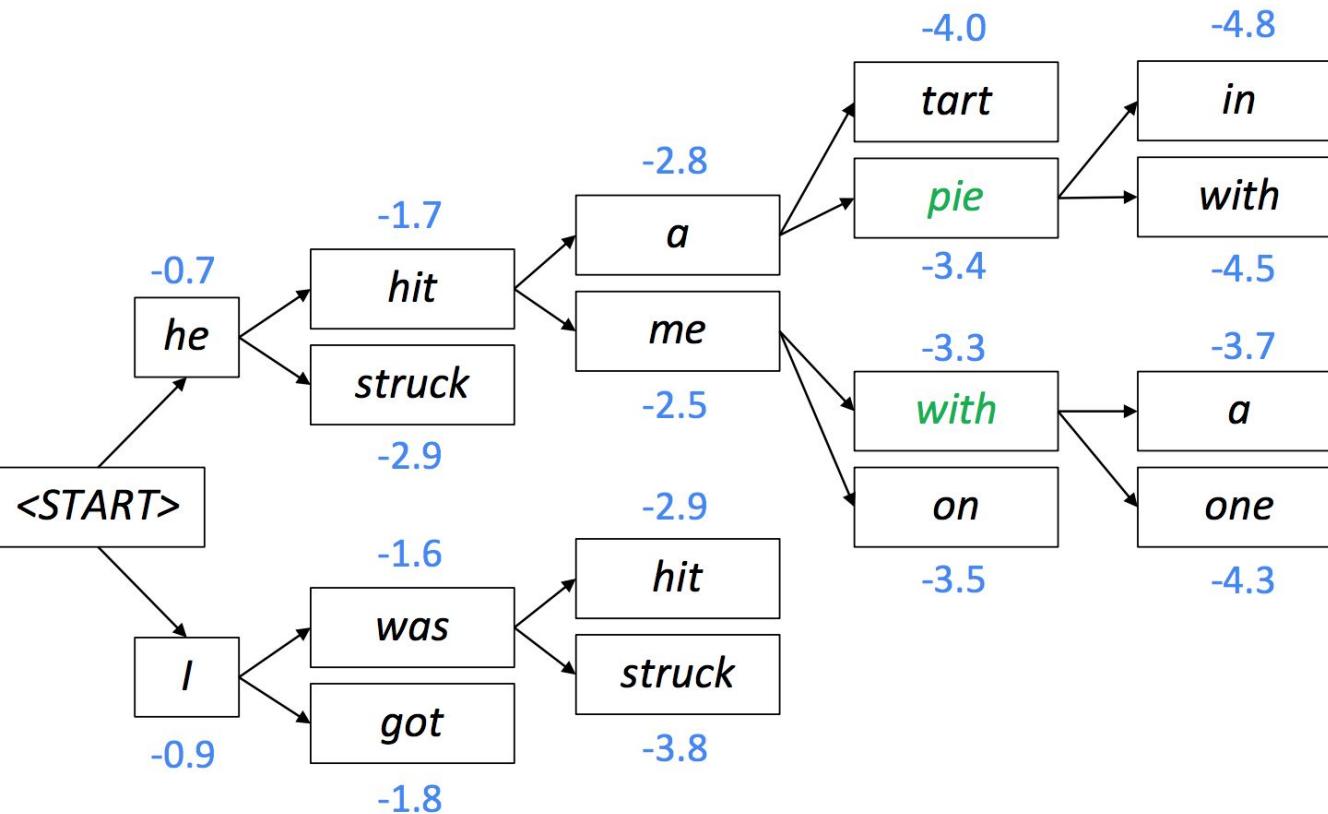
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



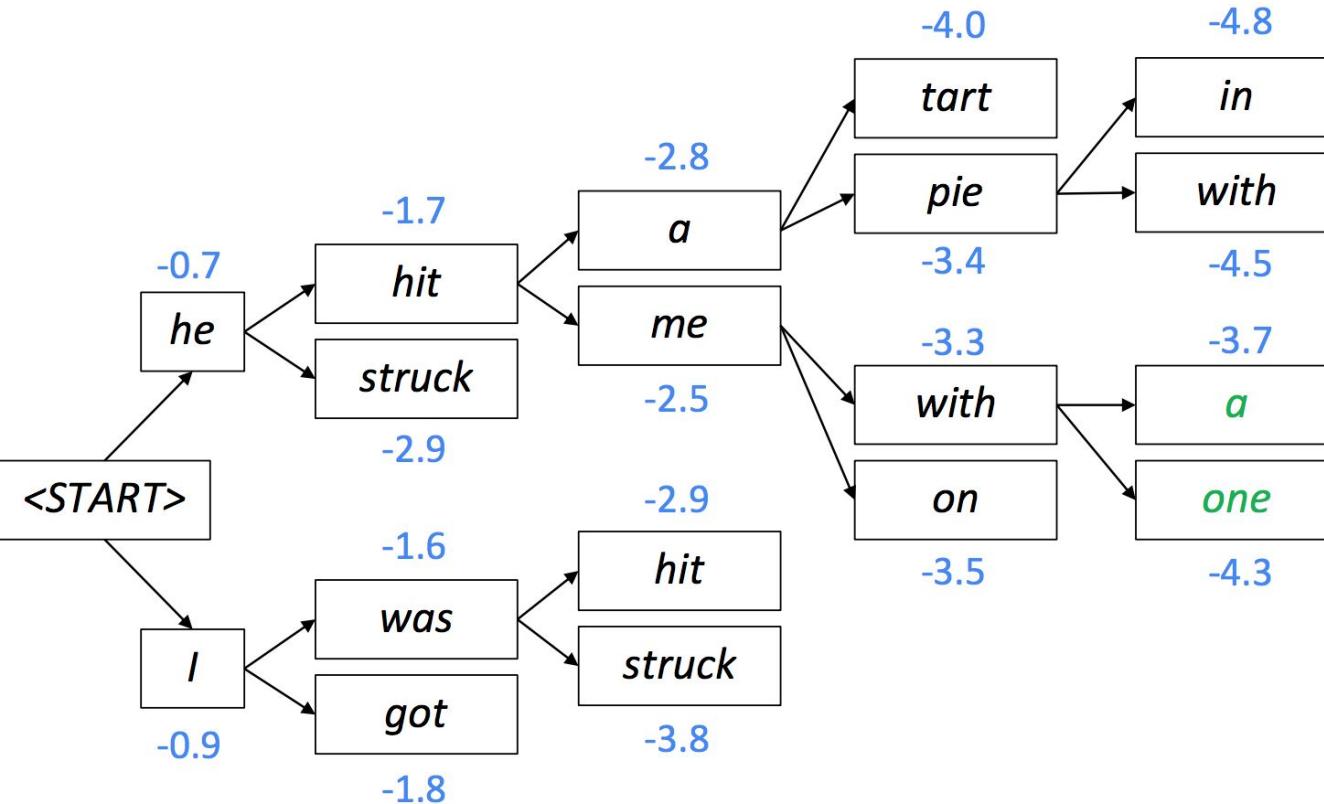
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



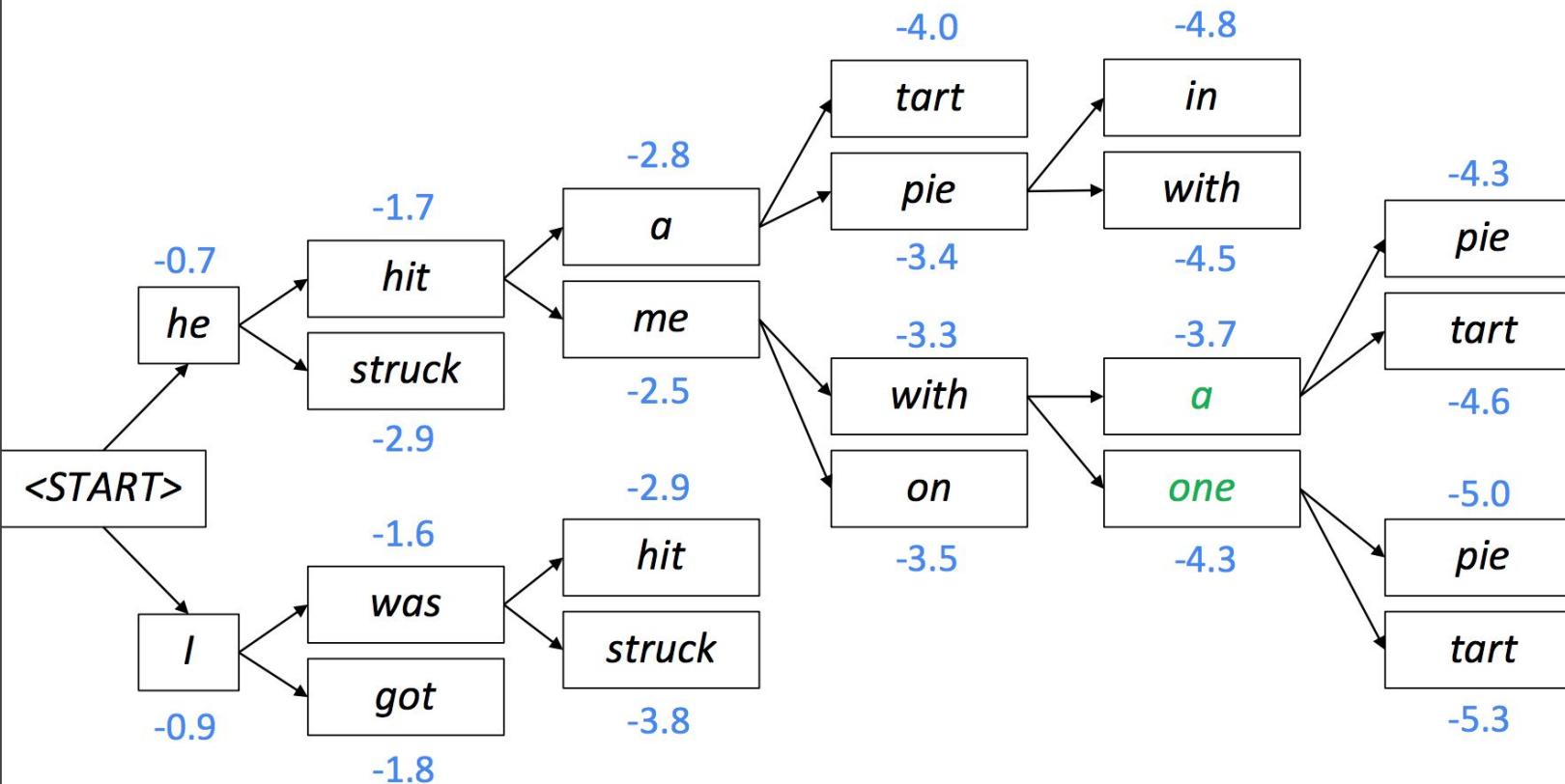
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



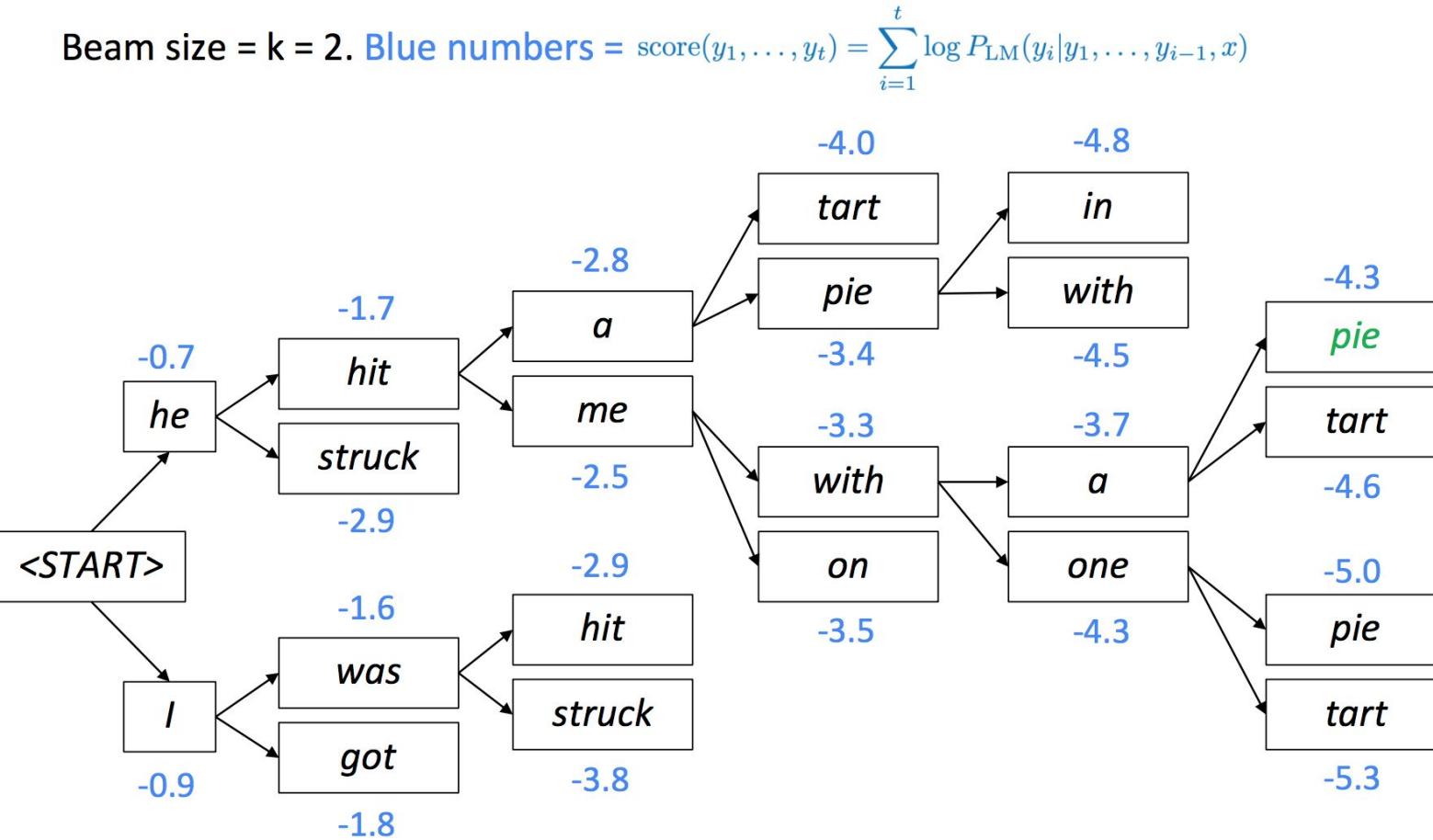
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



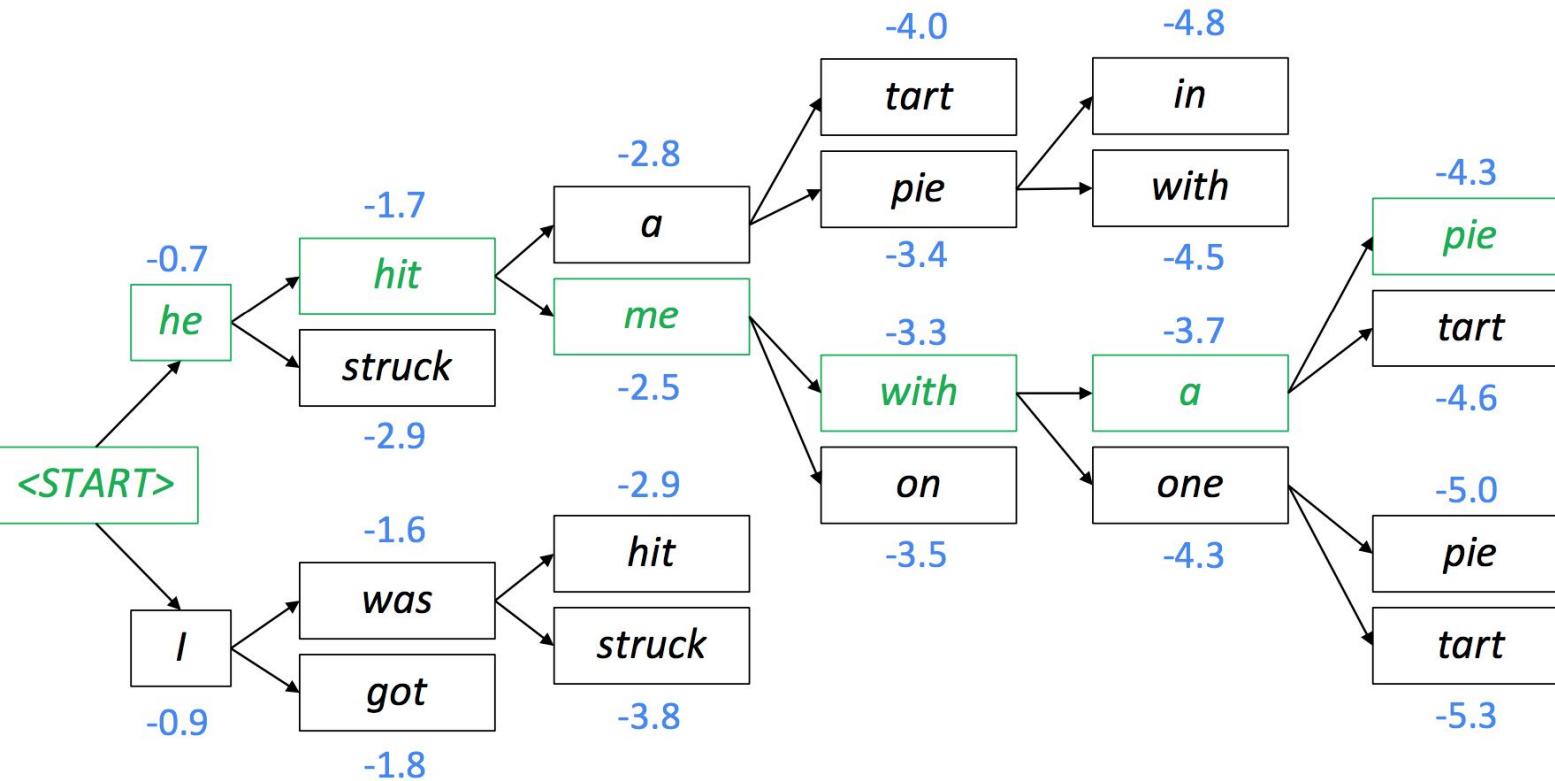
# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



# Beam search decoding: example

Beam size =  $k = 2$ . Blue numbers =  $\text{score}(y_1, \dots, y_t) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$



# Beam search decoding: stopping criterion

- In **greedy decoding**, usually we decode until the model produces <END> token
- In **beam search decoding**, different hypotheses may produce <END> tokens on different timesteps
  - When a hypothesis produces <END>, that hypothesis is complete.
  - Place it aside and continue exploring other hypotheses via beam search.
- Usually we continue beam search until:
  - We reach pre-defined timestep T
  - We have at least n completed hypotheses

# Beam search decoding: finishing up

- How to select top one with highest score?
- Each hypothesis on our list has a score:

$$\text{score}(y_1, \dots, y_t) = \log P_{\text{LM}}(y_1, \dots, y_t | x) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

- **Problems?**

# Beam search decoding: finishing up

- How to select top one with highest score?
- Each hypothesis on our list has a score:

$$\text{score}(y_1, \dots, y_t) = \log P_{\text{LM}}(y_1, \dots, y_t | x) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

- **Problem:** longer hypotheses have lower scores

# Beam search decoding: finishing up

- How to select top one with highest score?
- Each hypothesis on our list has a score:

$$\text{score}(y_1, \dots, y_t) = \log P_{\text{LM}}(y_1, \dots, y_t | x) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

- **Problem:** longer hypotheses have lower scores
- **Fix:** Normalize by length. Use this to select top one instead:

$$\frac{1}{t} \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

# NMT: Quality Evaluation

# Quality evaluation: Perplexity

$$PP(W) = P(w_1, w_2, \dots, w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1, w_2, \dots, w_N)}} = \sqrt[N]{\frac{1}{\prod_{i=1}^N P(w_i | w_1, \dots, w_{i-1})}}$$

# WER (Word Error Rate)

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}$$

- S is the number of substitutions,
- D is the number of deletions,
- I is the number of insertions,
- C is the number of correct words,
- N is the number of words in the reference ( $N = S + D + C$ )

BLEU (Bilingual Evaluation Understudy) compares the machine-written translation to human-written translation, and computes a similarity score based on:

- n-gram precision
- penalty for too-short system translations (brevity penalty)

$$BLEU = \text{brevity penalty} \cdot \left( \prod_{i=1}^n \text{precision}_i \right)^{1/n} \cdot 100\%$$

$$\text{brevity penalty} = \min \left( 1, \frac{\text{output length}}{\text{reference length}} \right)$$

BLEU (Bilingual Evaluation Understudy) compares the machine-written translation to human-written translation, and computes a similarity score based on:

- n-gram precision
- brevity penalty

SYSTEM A:	<b>Israeli officials</b> 2-GRAM MATCH	responsibility of <b>airport</b> safety 1-GRAM MATCH
REFERENCE: Israeli officials are responsible for airport security		
SYSTEM B:	<b>airport security</b> 2-GRAM MATCH	<b>Israeli officials are responsible</b> 4-GRAM MATCH

Metric	System A	System B
precision (1gram)	3/6	6/6
precision (2gram)	1/5	4/5
precision (3gram)	0/4	2/4
precision (4gram)	0/3	1/3
brevity penalty	6/7	6/7
BLEU	0%	52%

$$BLEU = \text{brevity penalty} \cdot \left( \prod_{i=1}^n \text{precision}_i \right)^{1/n} \cdot 100\%$$

BLEU is imperfect:

- There are many valid ways to translate a sentence
- So a good translation may get a poor BLEU score just because of low n-gram overlap with the human translation

# How to upgrade BLEU?

- **ROUGE** (Recall-Oriented Understudy for Gisting Evaluation)
- **METEOR** (Metric for Evaluation of Translation with Explicit ORdering)
  - Uses synonyms from WordNet
- **NIST** (of US National Institute of Standards and Technology)
  - More weight to rare n-grams, less punishment for short texts
- **TER**
  - Uses the number of changes that should be made to get to the reference translation

## ROUGE - Recall-Oriented Understudy for Gisting Evaluation

Recall in the context of ROUGE means how much of the reference summary is the system summary recovering or capturing

**BLEU** is focusing on **precision**:

$$\text{overlapping\_words} / \text{total\_words\_in\_system\_summary}$$

**ROUGE** is focusing on **recall**:

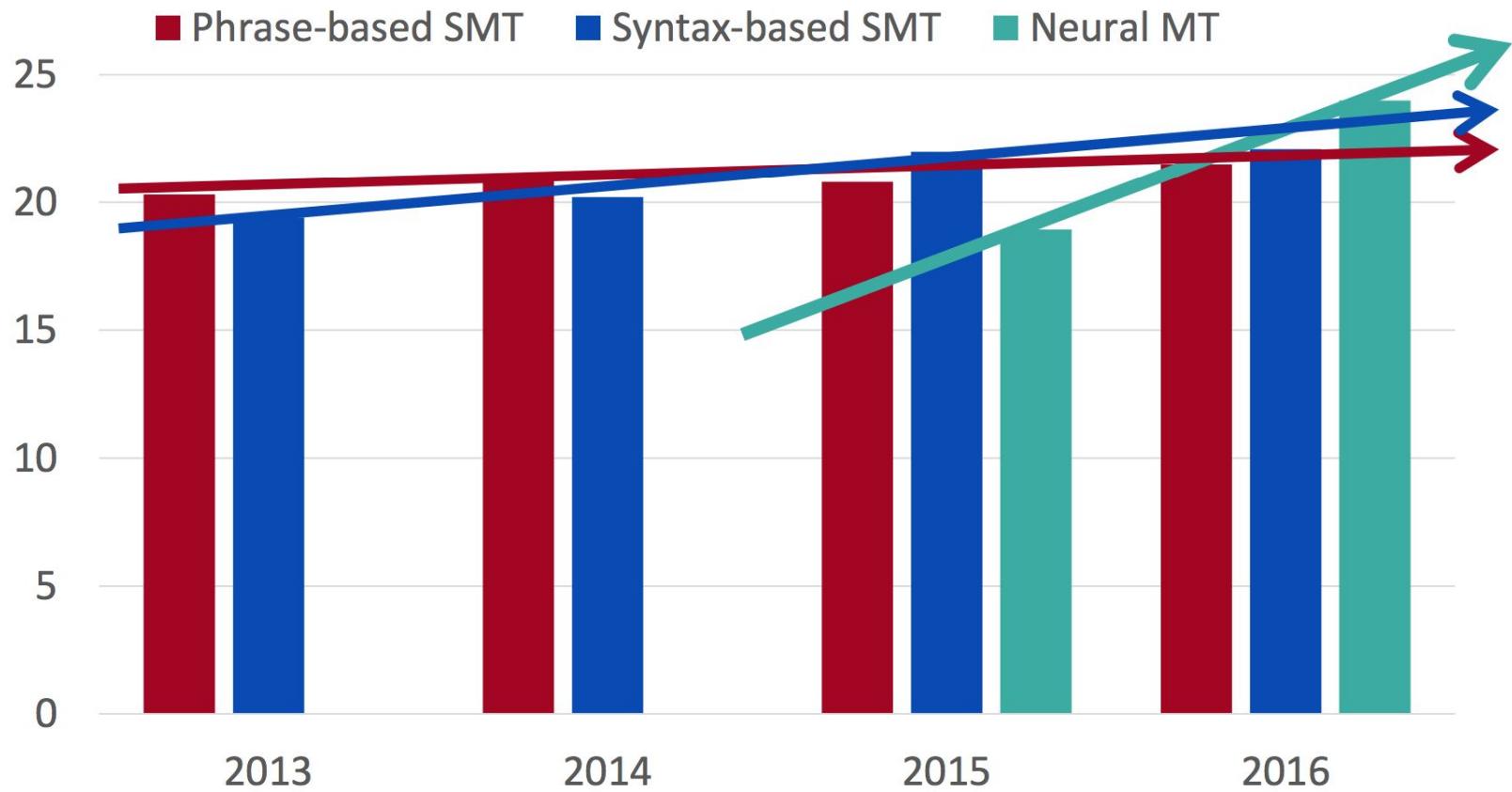
$$\text{overlapping\_words} / \text{total\_words\_in\_reference\_summary}$$

## ROUGE - Recall-Oriented Understudy for Gisting Evaluation

- **ROUGE-N:** Overlap of N-grams between the system and reference summaries.
- **ROUGE-L:** Longest Common Subsequence (LCS) based statistics.  
Longest common subsequence problem takes into account sentence level structure similarity naturally and identifies longest co-occurring in sequence n-grams automatically.
- **ROUGE-W:** Weighted LCS-based statistics
- etc.

- Better performance
  - More fluent
  - Better use of context
  - Better use of phrase similarities
- A single neural network to be optimized end-to-end
  - No subcomponents to be individually optimized
- Requires much less human engineering effort
  - No feature engineering
  - Same method for all language pairs

# NMT progress over time



# NMT: disadvantages

- NMT is less interpretable
  - Hard to debug
- NMT is difficult to control
  - For example, can't easily specify rules or guidelines for translation
  - Safety concerns!

# NMT: disadvantages

English ▾      Spanish ▾

paper jam Edit

Mermelada de papel

[Open in Google Translate](#)

*Feedback*



?

# NMT: disadvantages

Malay - detected▼



English▼



Dia bekerja sebagai jururawat.

Dia bekerja sebagai pengaturcara. Edit

She works as a nurse.

He works as a programmer.



Didn't specify gender

# NMT: disadvantages

Somali	↔	English
Translate from Irish		
ag ag ag ag ag ag ag ag ag ag ag ag ag ag	Edit	As the name of the LORD was written in the Hebrew language, it was written in the language of the Hebrew Nation

Open in Google Translate

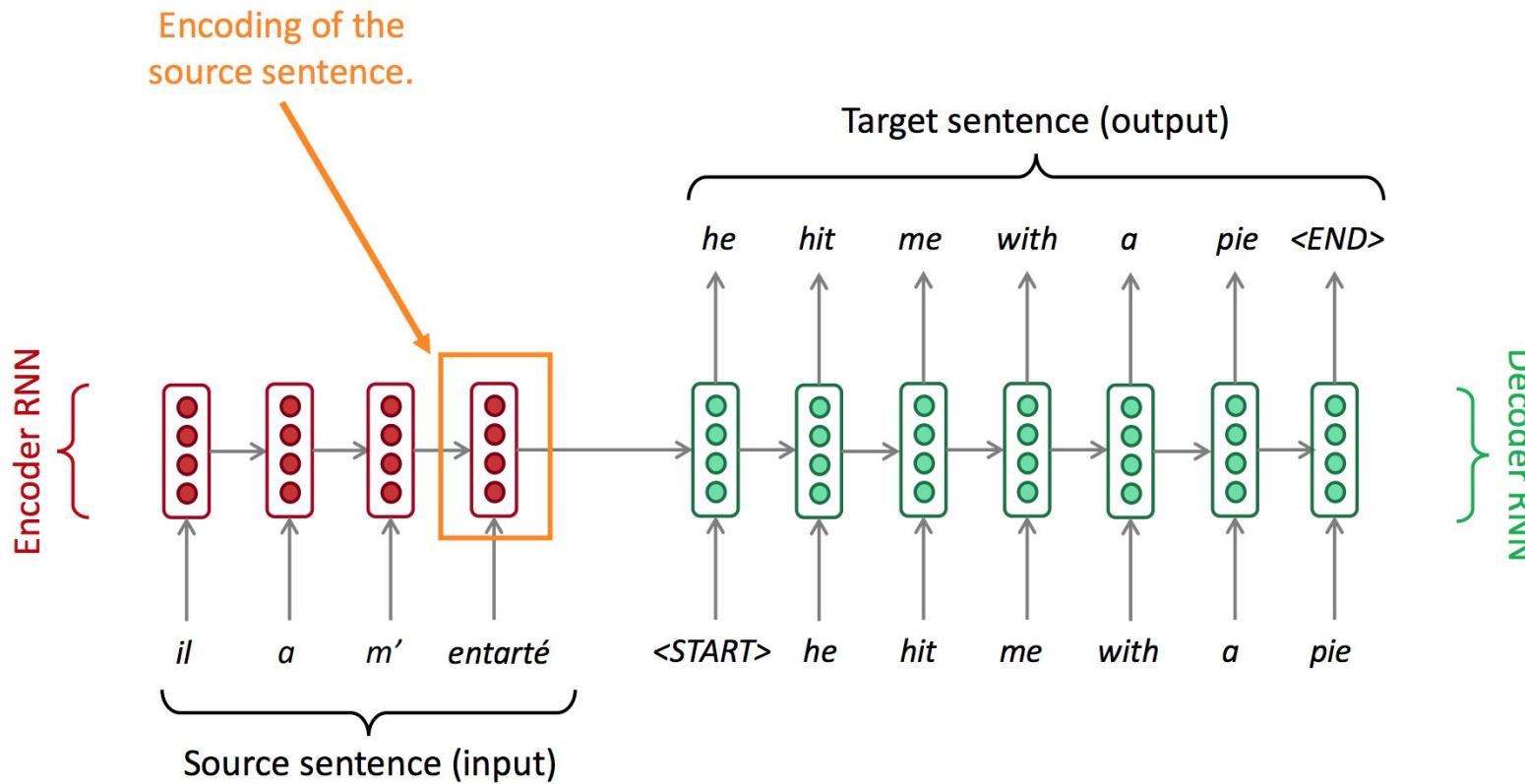
### *Feedback*

# Is Machine Translation solved?

- Many difficulties remain:
  - Out-of-vocabulary words
  - Domain mismatch between train and test data
  - Maintaining context over long texts
  - Low-resource language pairs (no big parallel corpora)

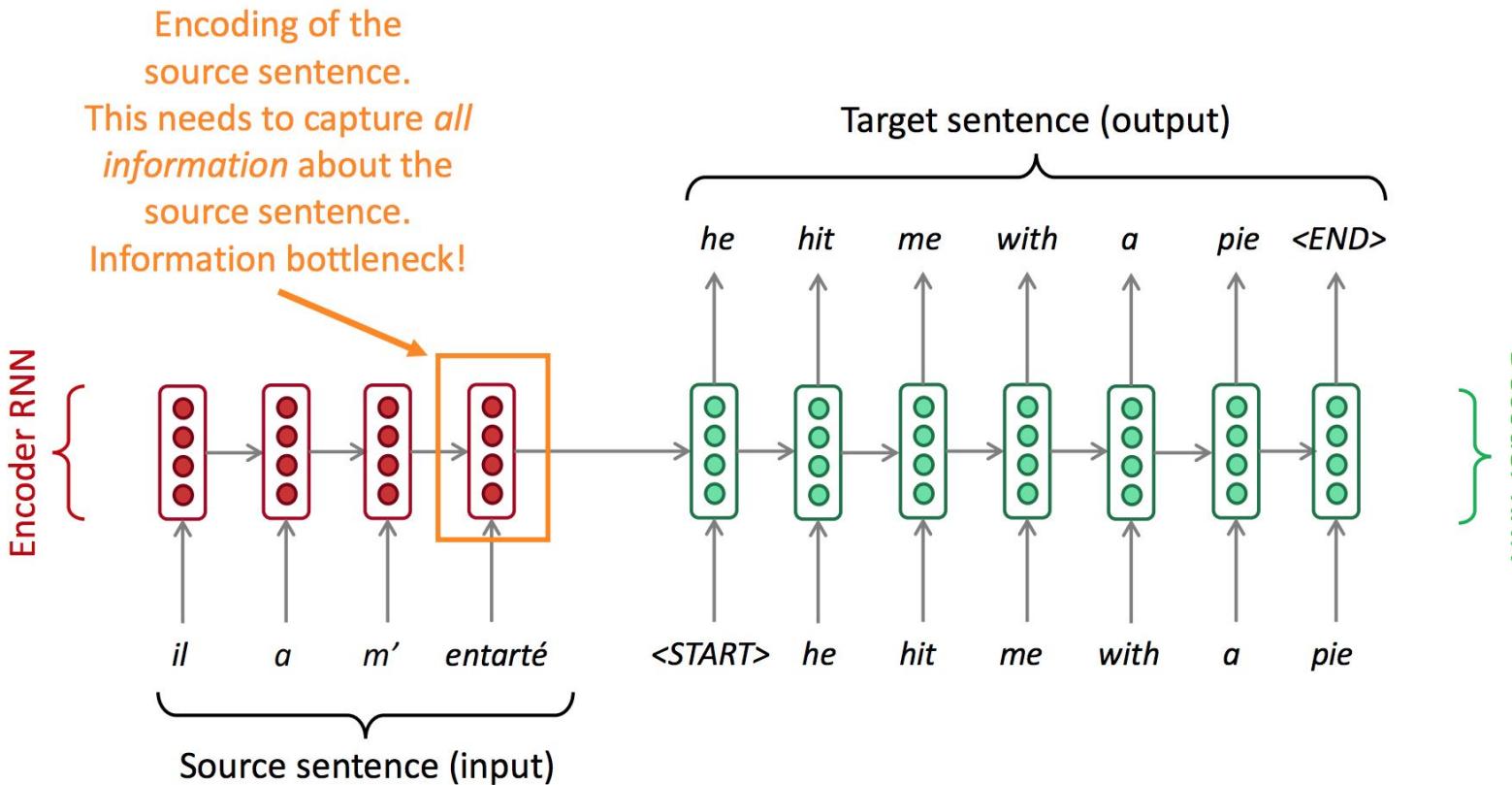
# Attention

# Seq2seq: the bottleneck problem



Problems with this architecture?

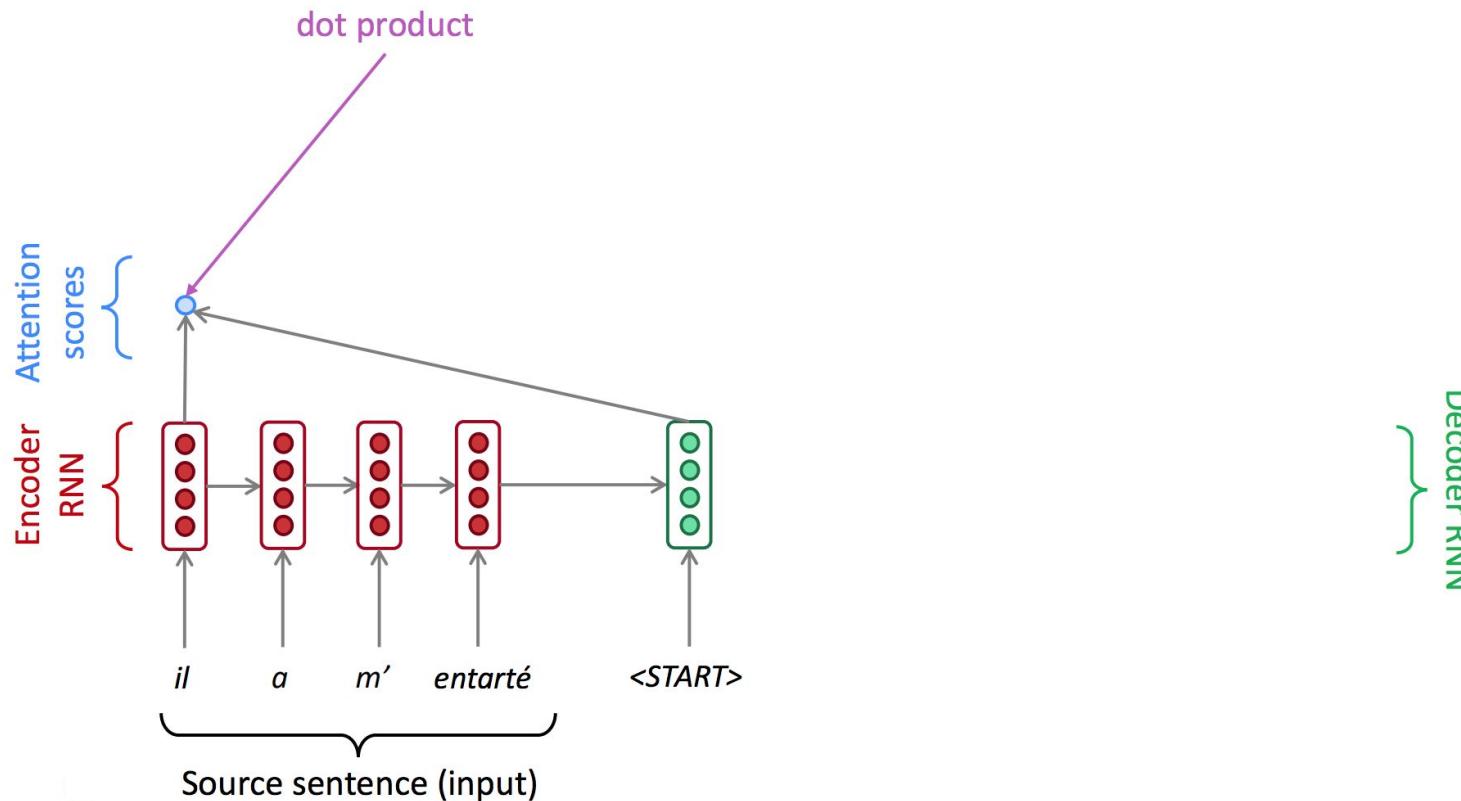
# Seq2seq: the bottleneck problem



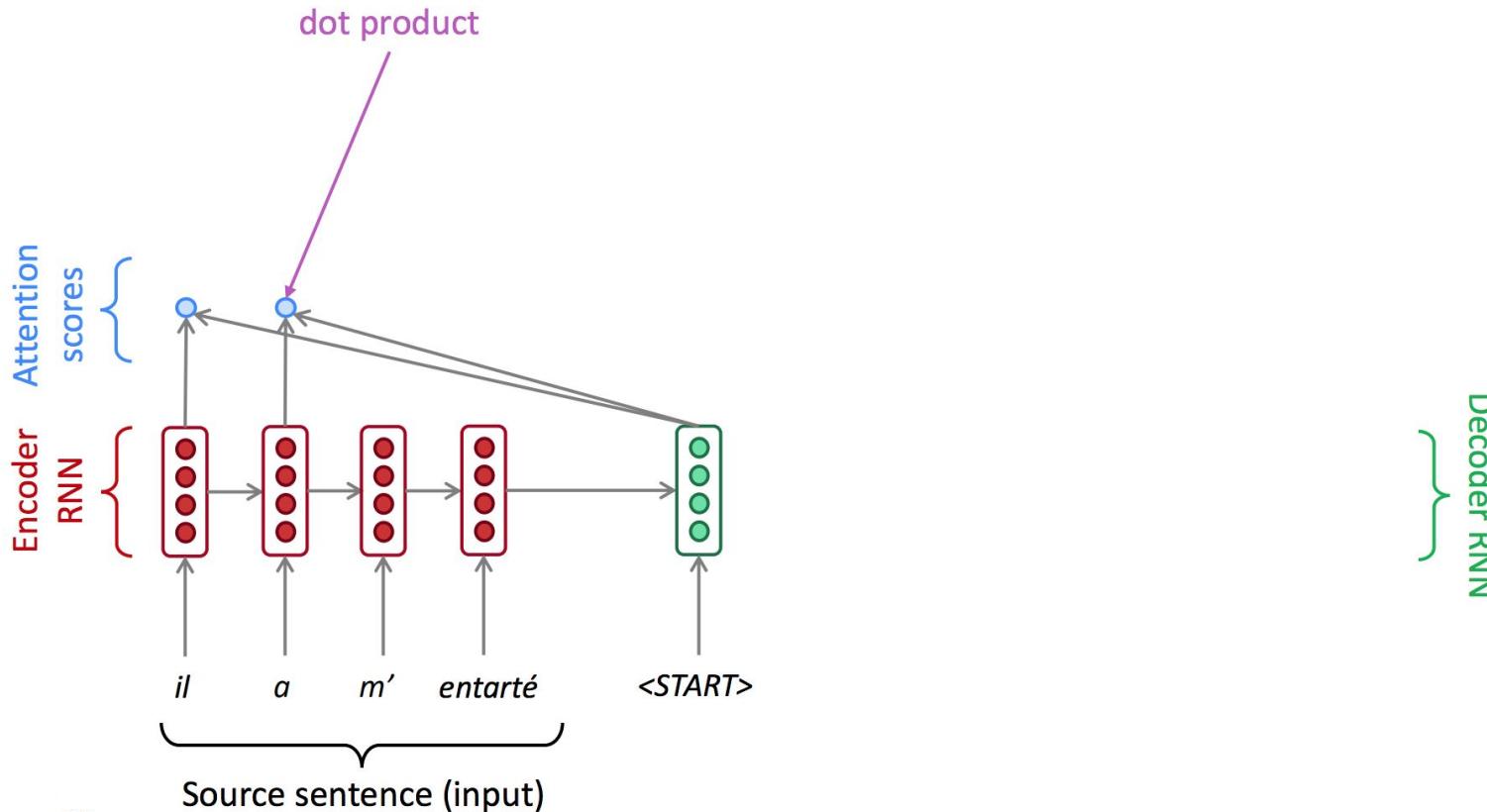
**Main idea:** on each step of the **decoder**, use **direct connection to the encoder** to focus on a particular part of the source sequence



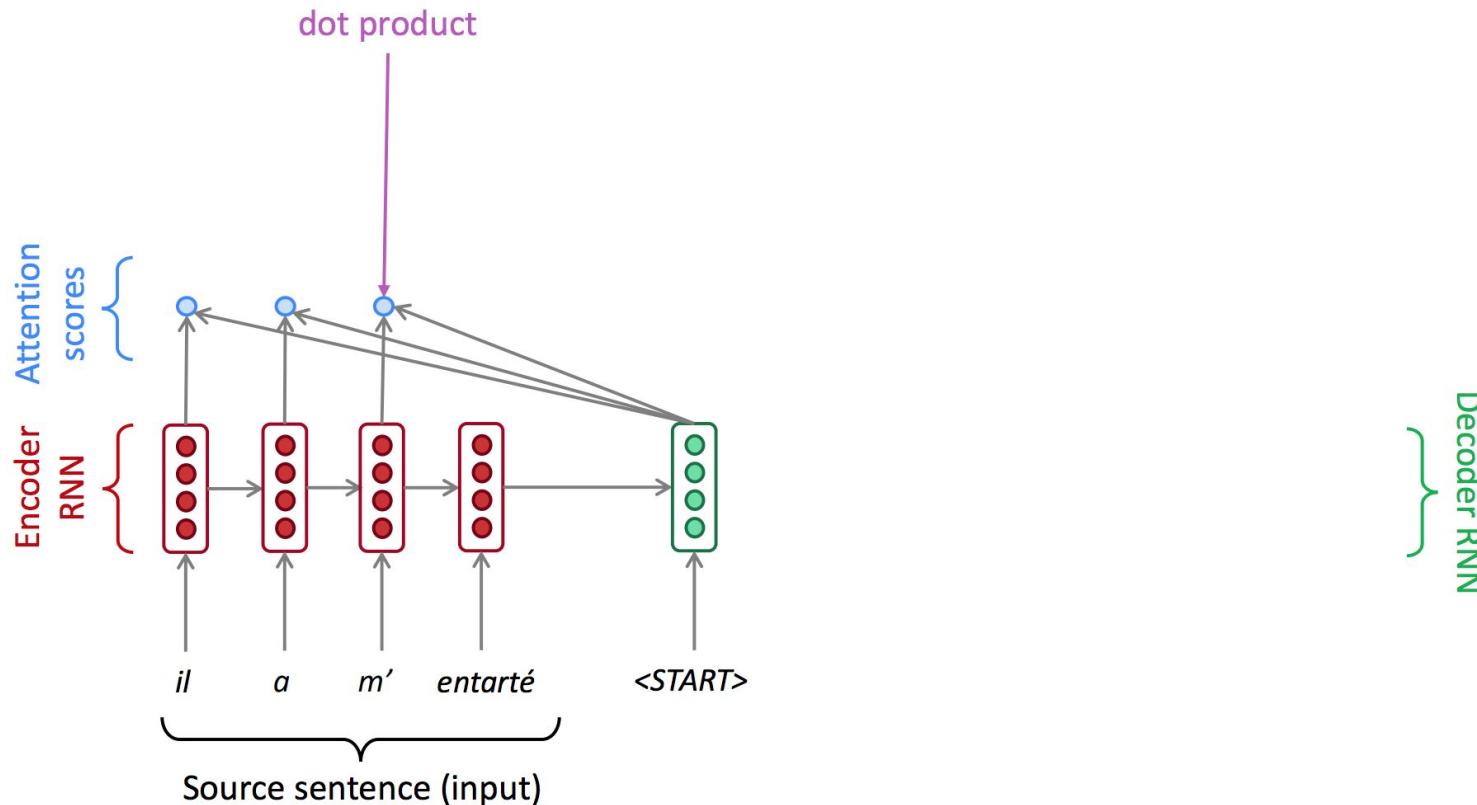
# Seq2seq with attention



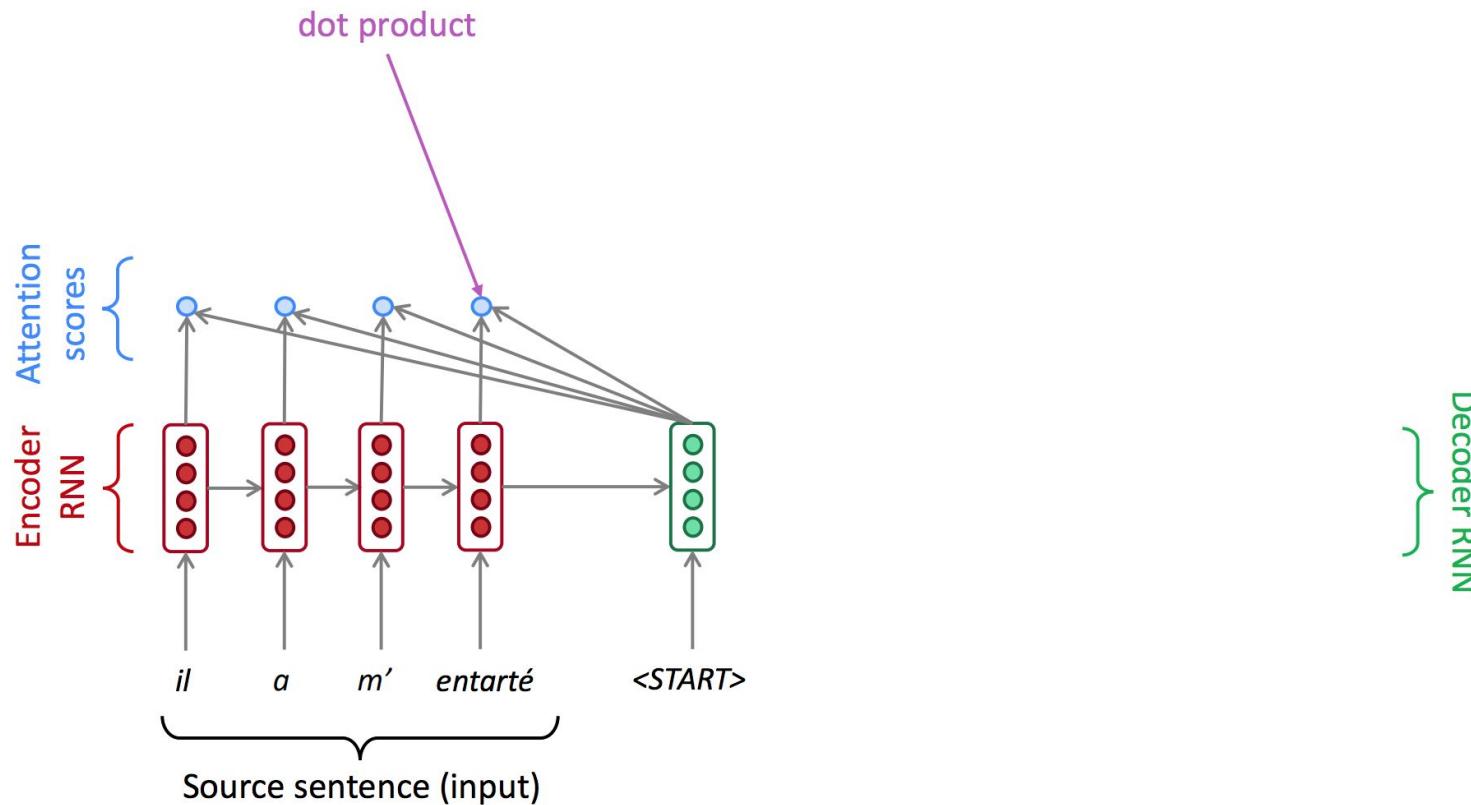
# Seq2seq with attention



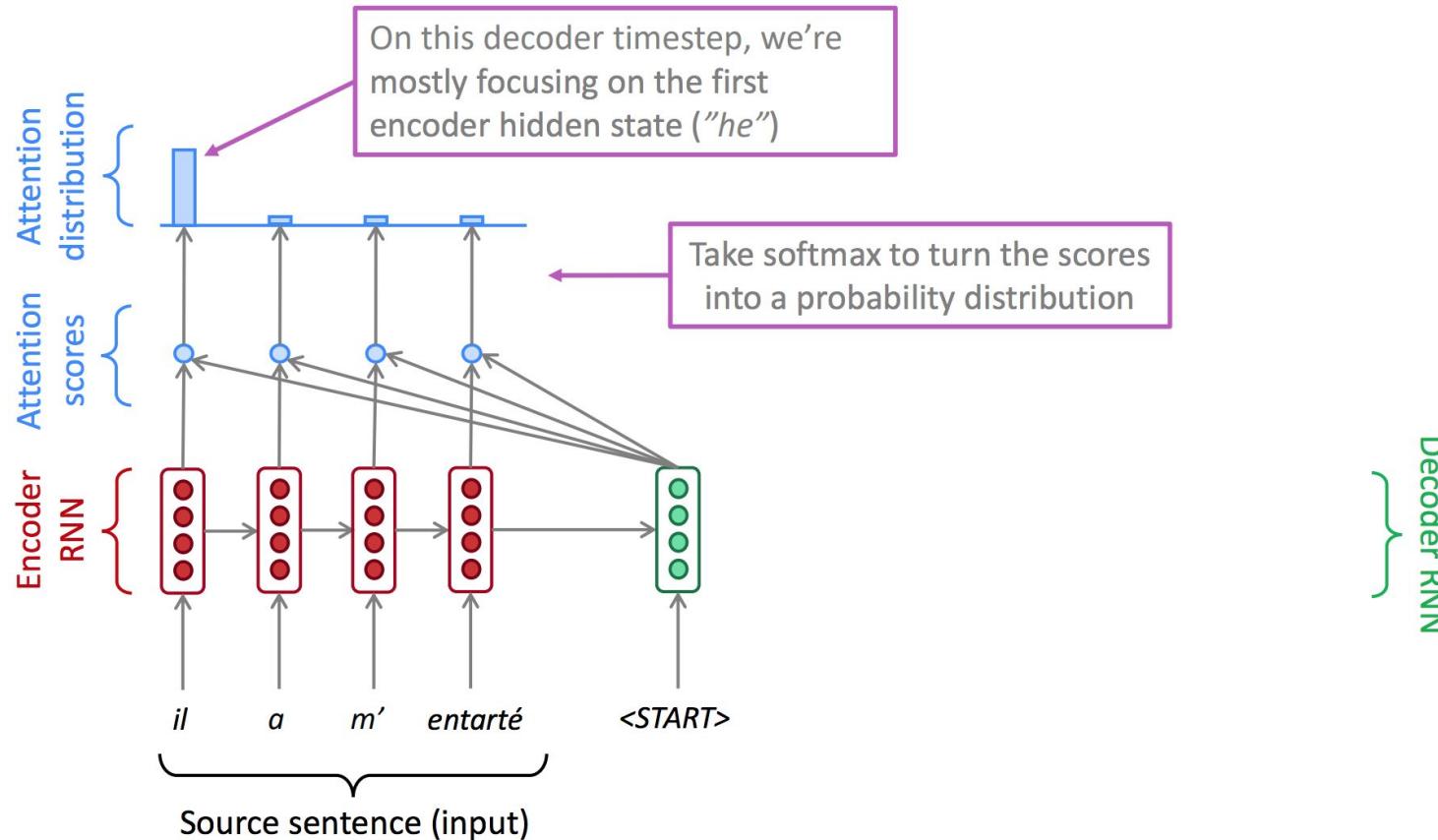
# Seq2seq with attention



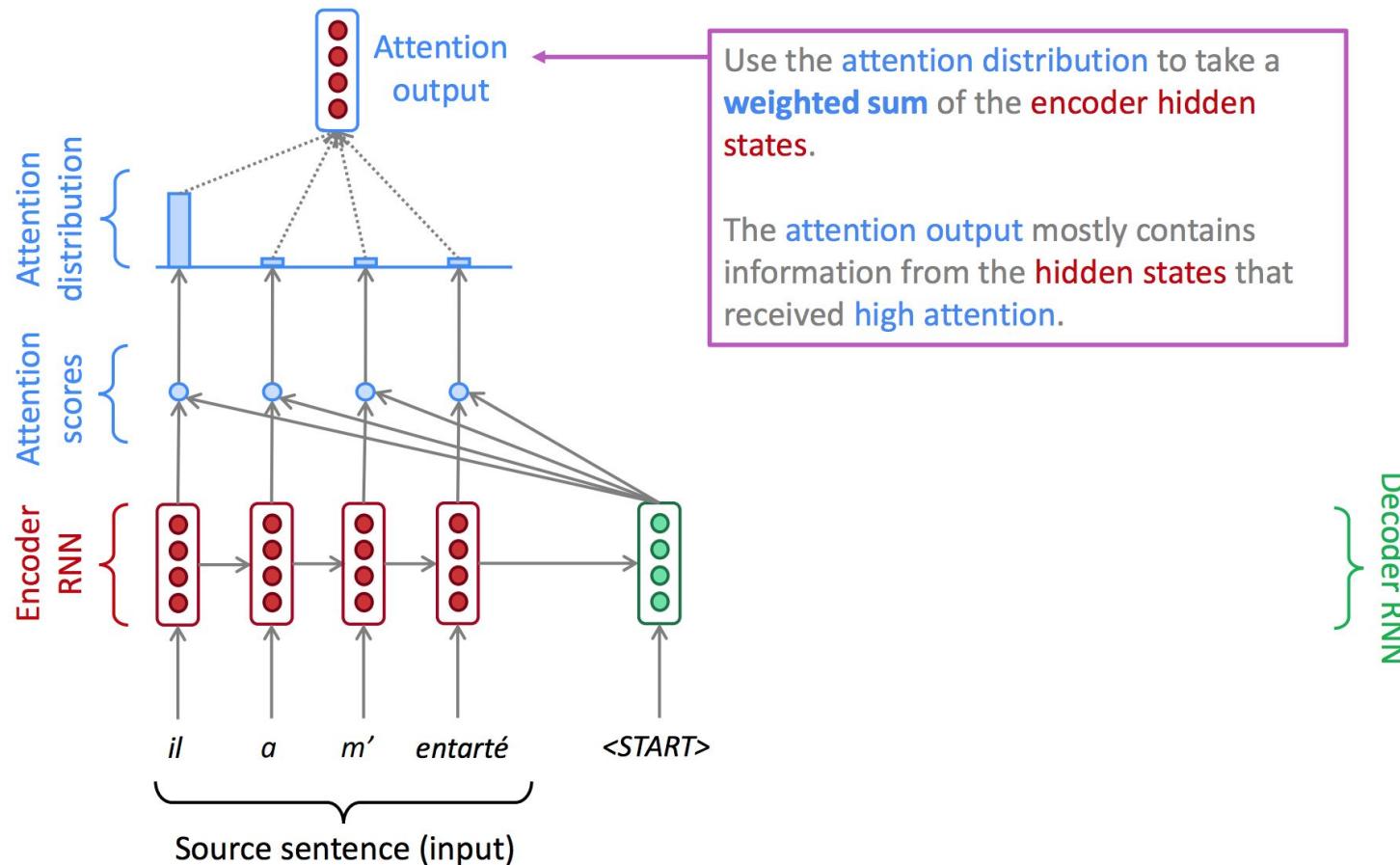
# Seq2seq with attention



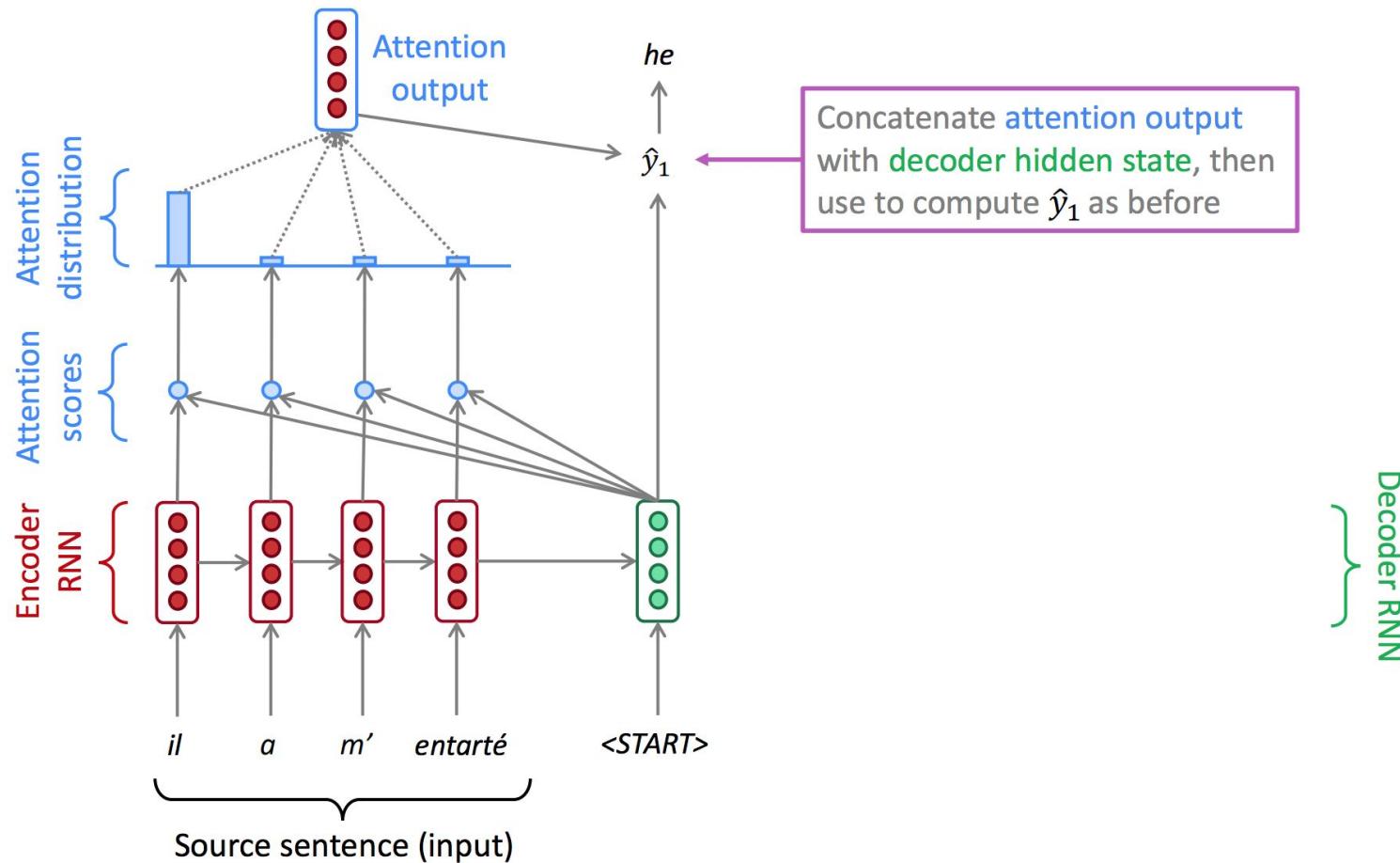
# Seq2seq with attention



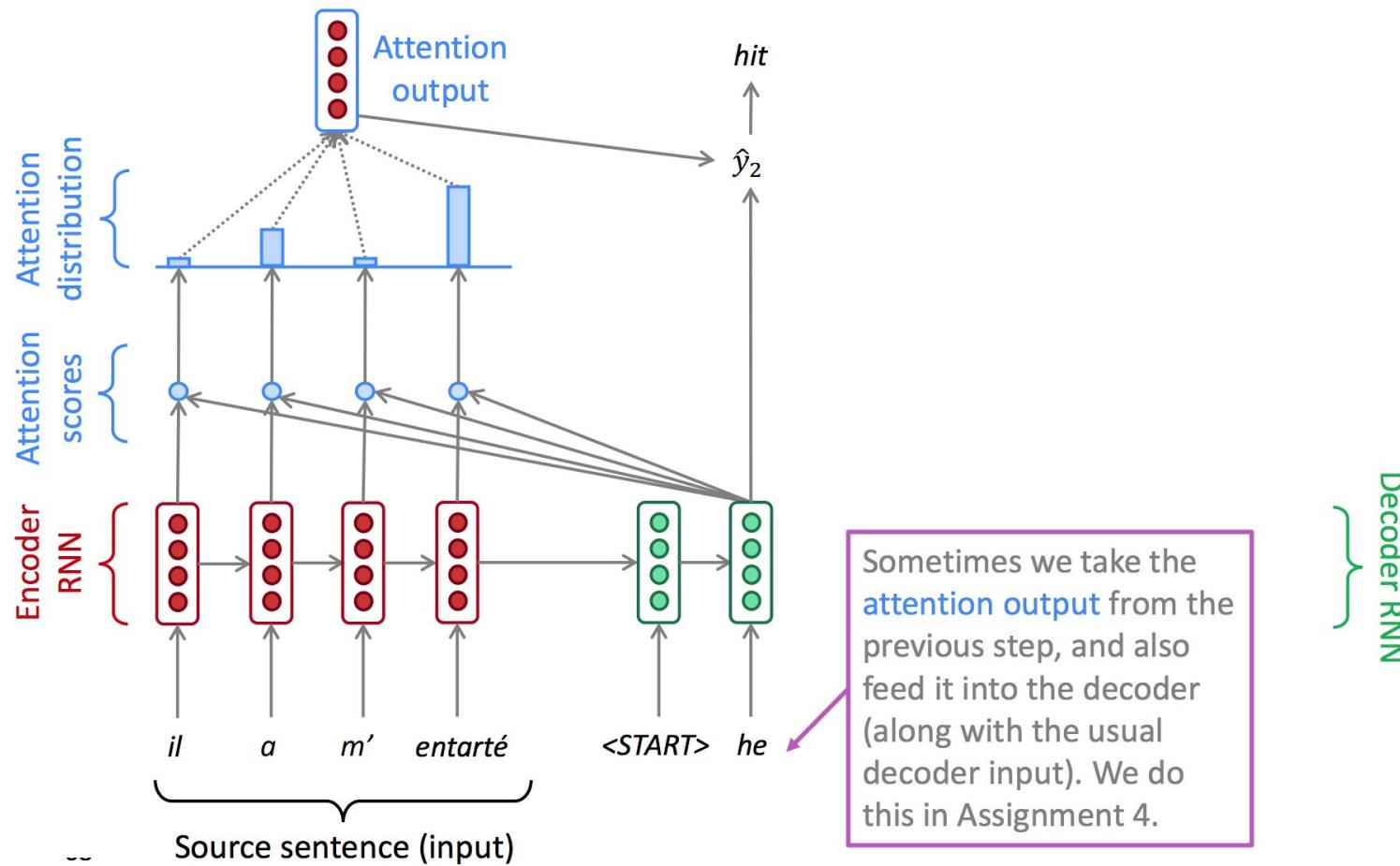
# Seq2seq with attention



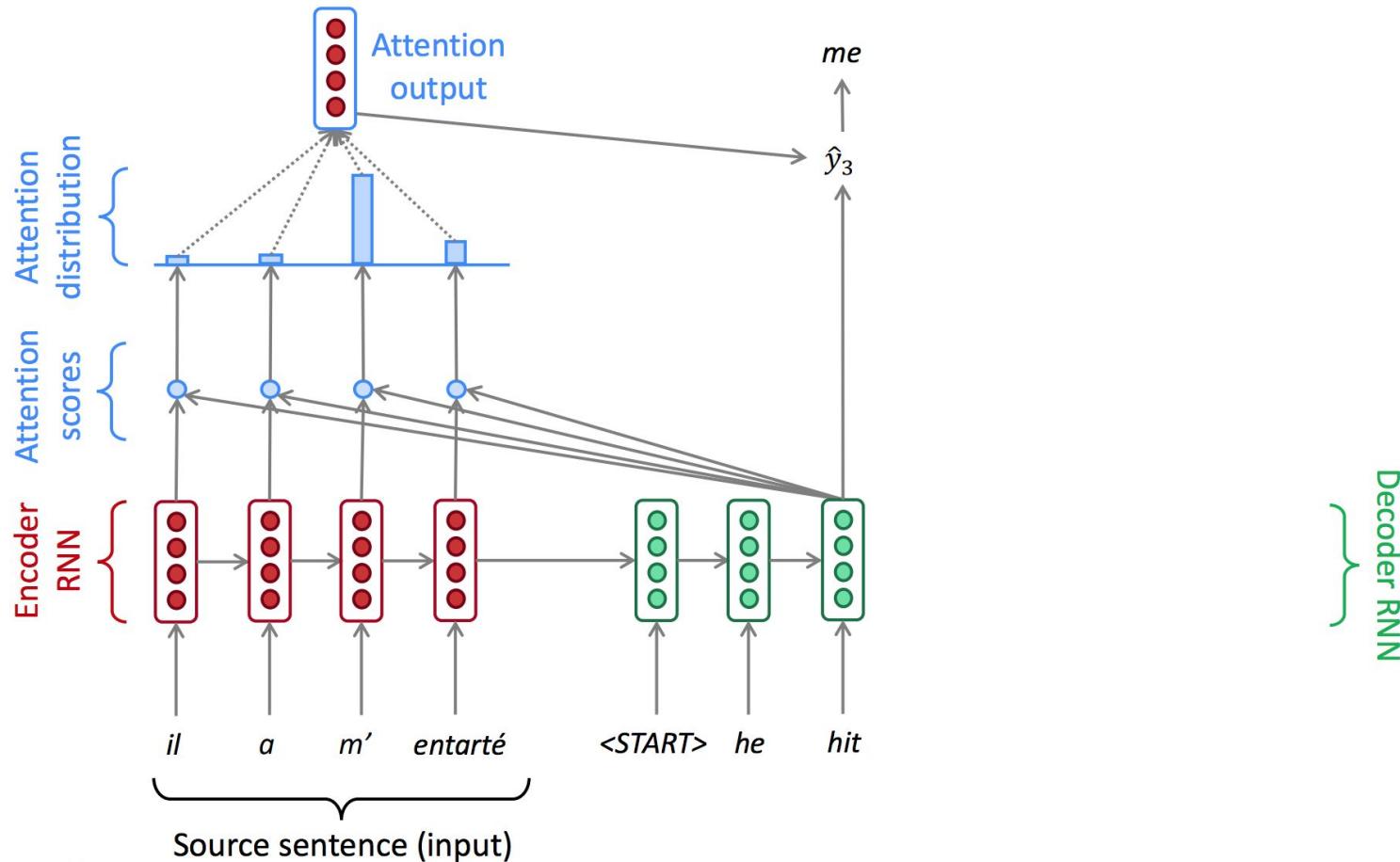
# Seq2seq with attention



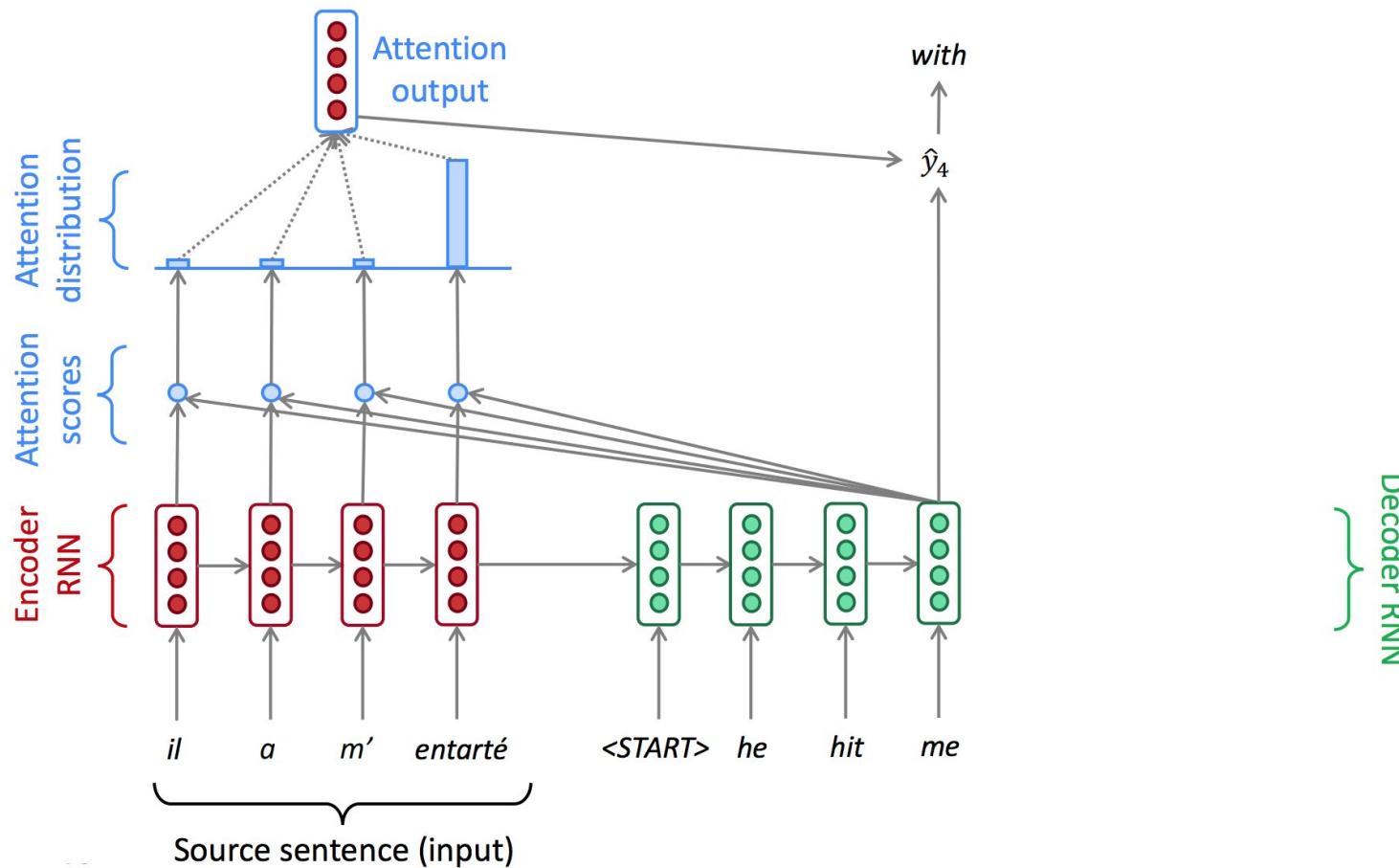
# Seq2seq with attention



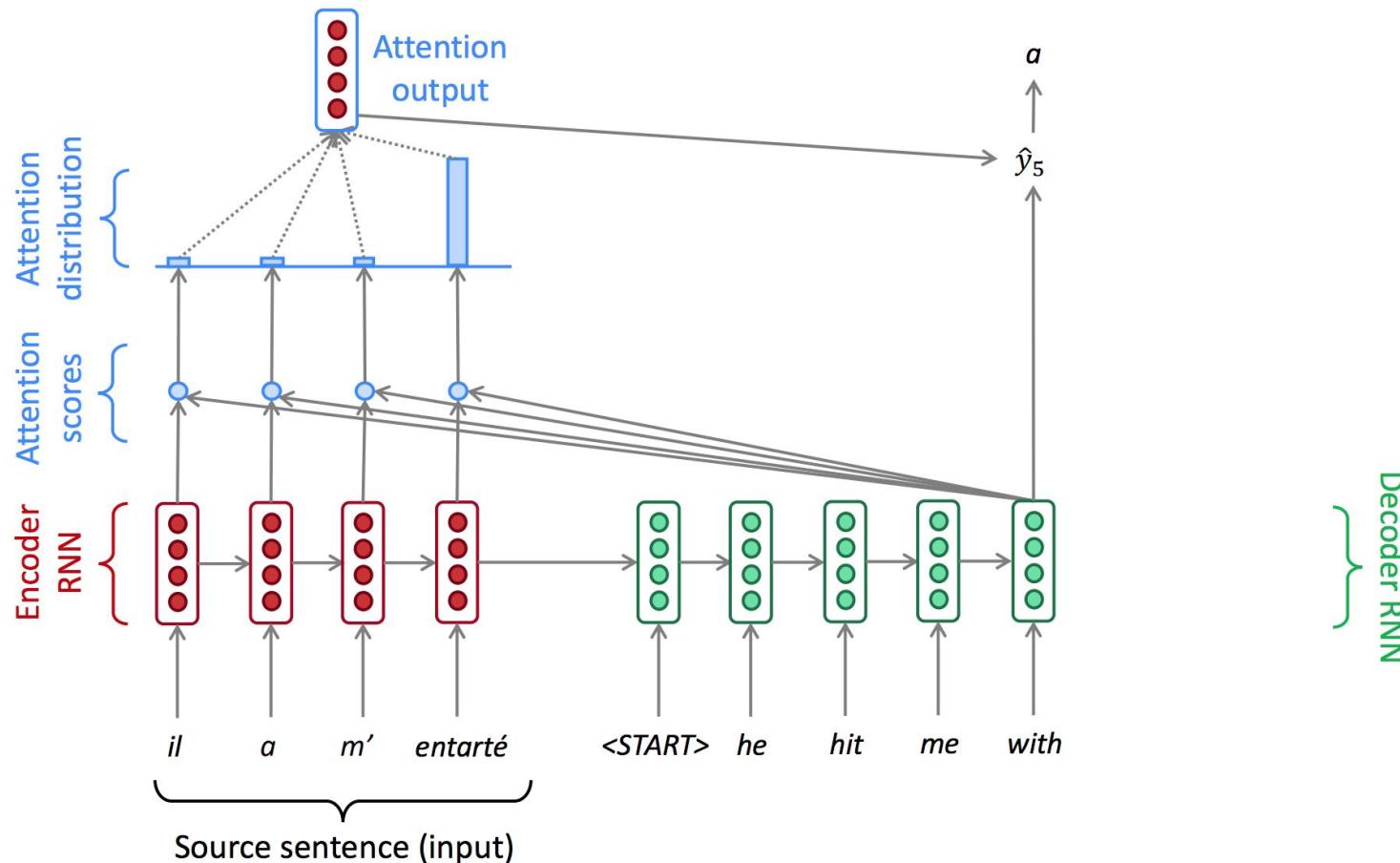
# Seq2seq with attention



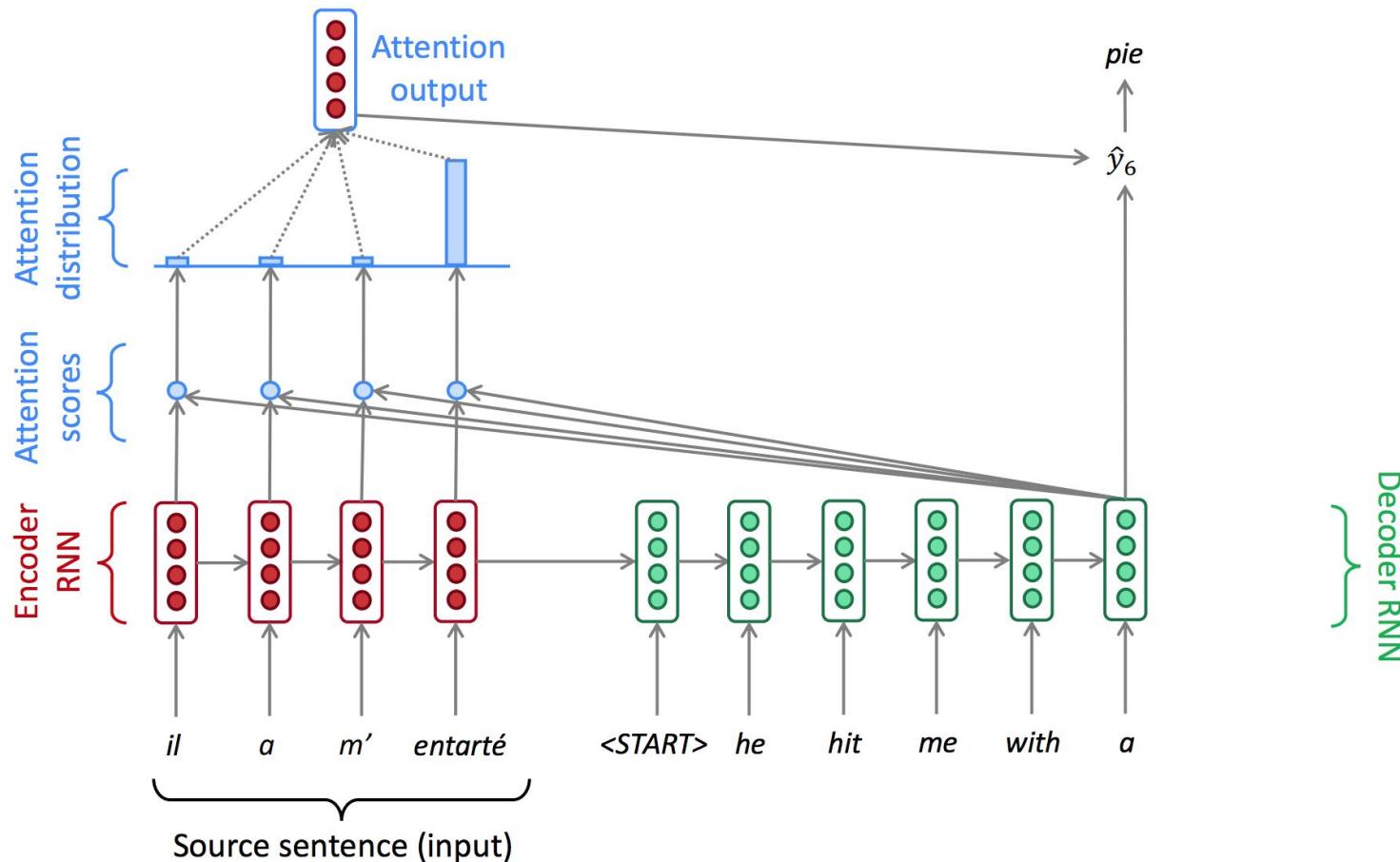
# Seq2seq with attention



# Seq2seq with attention



# Seq2seq with attention



# Attention in equations

We have encoder hidden states  $h_1, \dots, h_N \in \mathbb{R}^h$

On timestep  $t$ , we have decoder hidden state  $s_t \in \mathbb{R}^h$

We get the attention scores  $e^t$  for this step:

$$e^t = [s_t^T h_1, \dots, s_t^T h_N] \in \mathbb{R}^N$$

We take softmax to get the attention distribution  $\alpha^t$  for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(e^t) \in \mathbb{R}^N$$

We use  $\alpha^t$  to take a weighted sum of the encoder hidden states to get the attention output  $a_t$

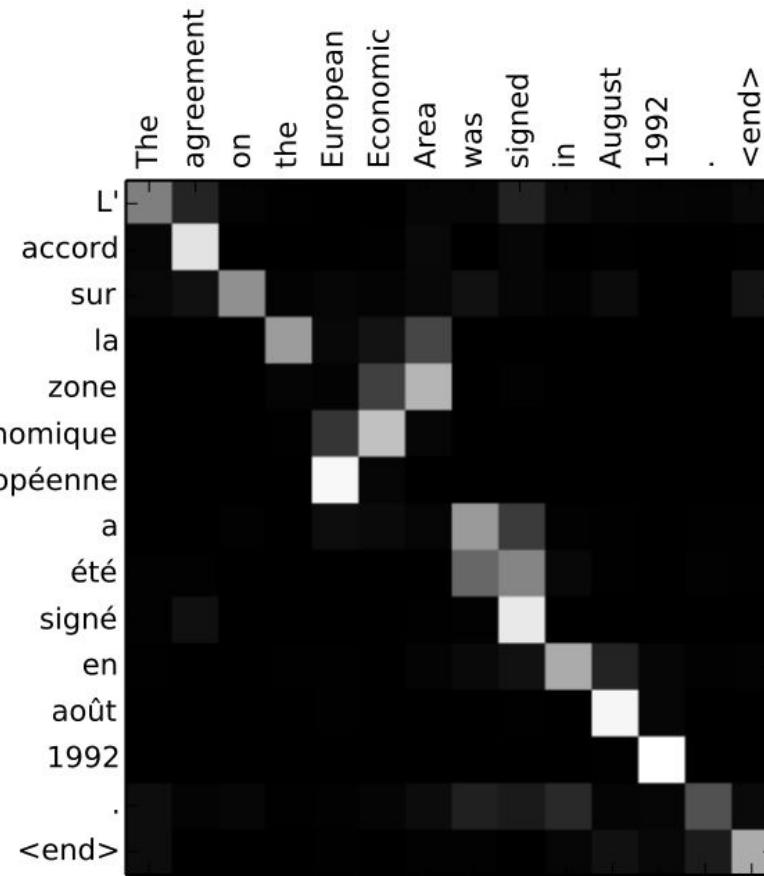
$$a_t = \sum_{i=1}^N \alpha_i^t h_i \in \mathbb{R}^h$$

Finally we concatenate the attention output  $a_t$  with the decoder hidden state  $s_t$  and proceed as in the non-attention seq2seq model

$$[a_t; s_t] \in \mathbb{R}^{2h}$$

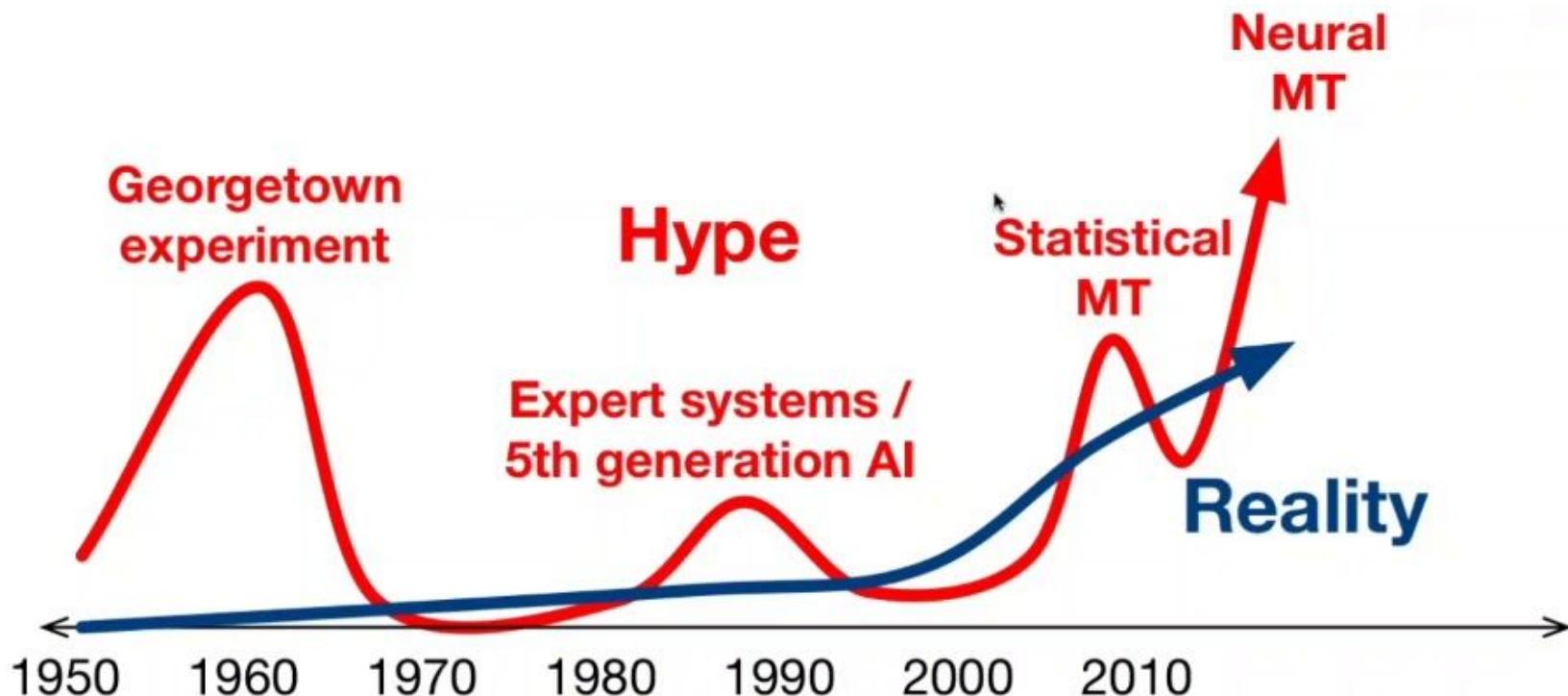
# Attention provides interpretability

- We may see what the decoder was focusing on
- We get word alignment for free!



- Basic dot-product (the one discussed before):  $e_i = \mathbf{s}^T \mathbf{h}_i \in \mathbb{R}$
- Multiplicative attention:  $e_i = \mathbf{s}^T \mathbf{W} \mathbf{h}_i \in \mathbb{R}$ 
  - $\mathbf{W} \in \mathbb{R}^{d_2 \times d_1}$  - weight matrix
- Additive attention:  $e_i = \mathbf{v}^T \tanh(\mathbf{W}_1 \mathbf{h}_i + \mathbf{W}_2 \mathbf{s}) \in \mathbb{R}$ 
  - $\mathbf{W}_1 \in \mathbb{R}^{d_3 \times d_1}, \mathbf{W}_2 \in \mathbb{R}^{d_3 \times d_2}$  - weight matrices
  - $\mathbf{v} \in \mathbb{R}^{d_3}$  - weight vector

# Summary



- Seq2seq is an architecture for NMT (2 RNNs)
- Attention is a way to focus on particular parts of the input

