

Lecture 11

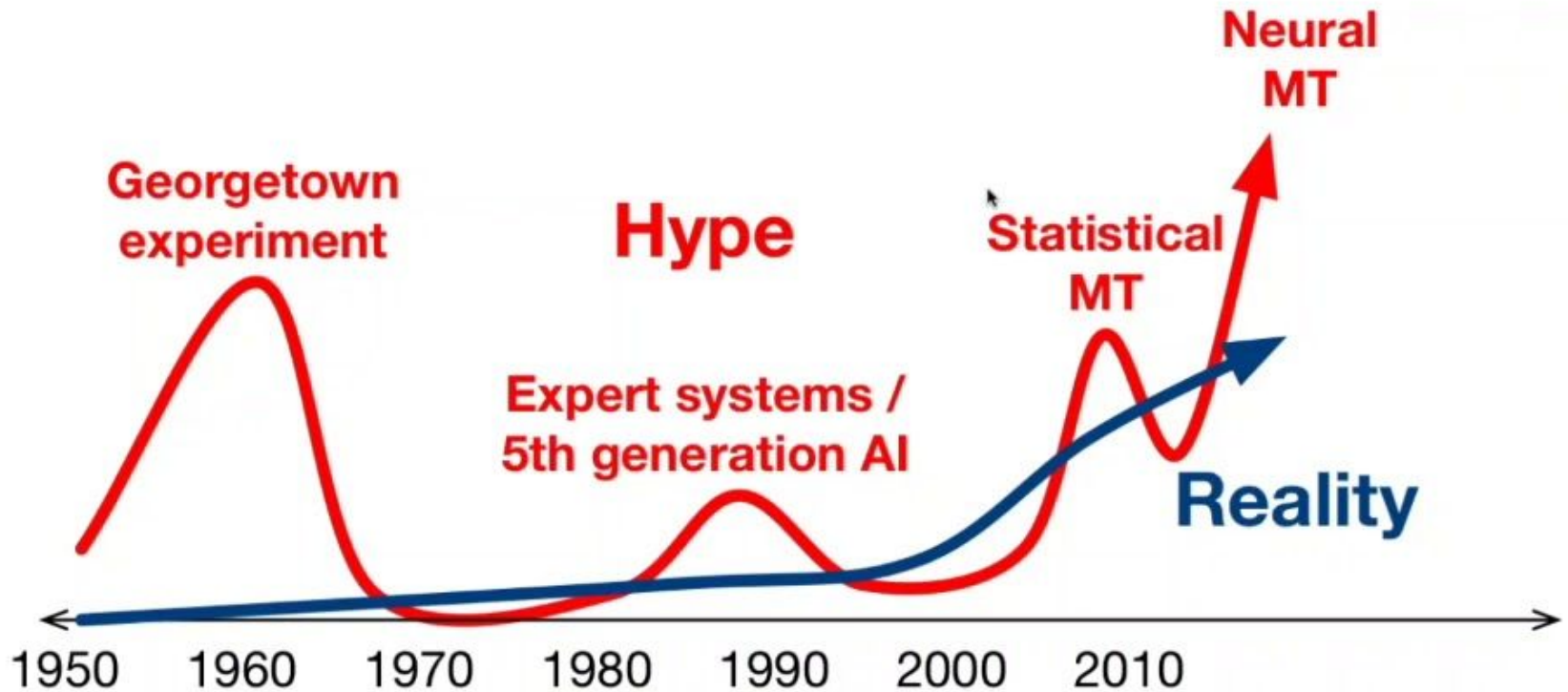
Machine Translation Attention Mechanism

Vladislav Goncharenko

- 2. Запишите формулу метода максимального правдоподобия для оценки параметра модели θ .
- 1. Назовите три различных метода валидации моделей
- 0. Какие функции активации используются в рекуррентных нейросетях? Почему именно они?
- 1. Какого размера будет выход свёрточного слоя над картинкой размера 8x5 (h x w) с четырьмя ядрами размера 3x3?
- 2. Как меняются паттерны, улавливаемые свёрточной нейросетью, в зависимости от глубины свёрточного слоя?
- 3. Что такое padding и stride в свёрточном слое? Зачем они используются?
- 4. Зачем нужен pooling слой в нейронных сетях? Каковы его преимущества перед свёрткой?
- 5. С какой основной проблемой сталкивались исследователи, увеличивавшие глубину нейросетей?

- Machine Translation historical overview
 - Statistical Machine Translation
 - Word alignments
- Neural Machine Translation (NMT)
 - Seq2Seq
 - Beam Search
- Attention mechanism

Historical overview



Before Deep Learning

1950s: first Machine Translation

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

MT Training Data

- Parallel corpora
 - Pairs <source, translation>
 - Typically sentence-level (although sometimes can be paragraph- or doc-level)
 - Can be manually curated by translators, crawled from web or synthetic

Source	角柱を過ぎる粘性流体の乱流をラージエディシミュレーションし、フィルタ幅と数値粘性の影響を調べた。
Reference	<u>The large eddy simulation</u> of a turbulent flow of a viscous fluid passing through a square column was conducted, and the effects of the filter width and numerical viscosity were examined.

1990-2010: Statistical Machine Translation

Want to find best English sentence y , given French sentence x

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

$$\begin{aligned} & \operatorname{argmax}_y P(y|x) \\ &= \operatorname{argmax}_y \underbrace{P(x|y)}_{\text{Translation Model}} \underbrace{P(y)}_{\text{Language Model}} \end{aligned}$$

Translation Model

Models how words and phrases
should be translated (*fidelity*).
Learnt from parallel data.

Language Model

Models how to write
good English (*fluency*).
Learnt from monolingual data.

1990-2010: Statistical Machine Translation

How to learn translation model from the parallel corpus?

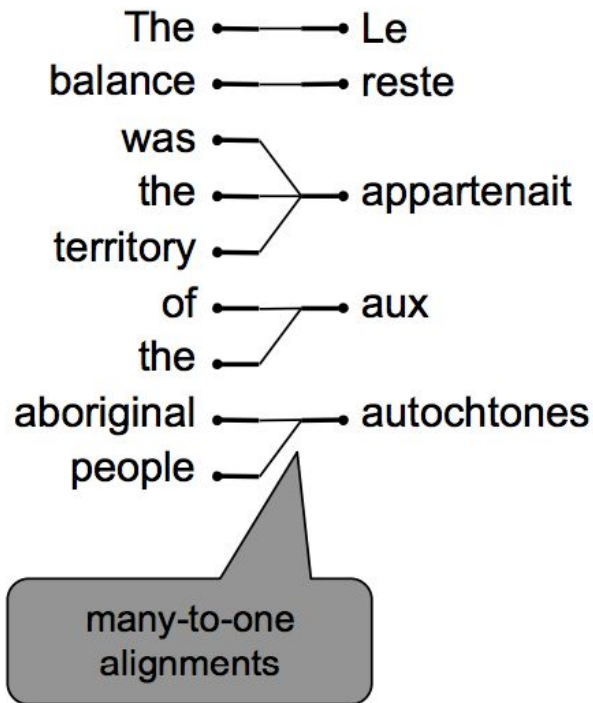
Let's calculate

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

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

1990-2010: Statistical Machine Translation

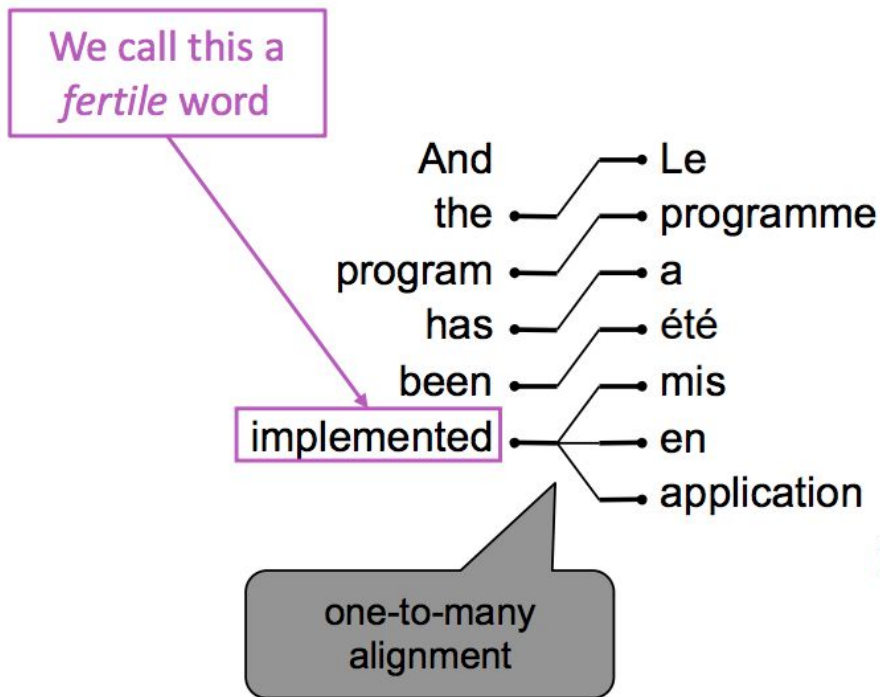
Alignment can be: **many-to-one**



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

1990-2010: Statistical Machine Translation

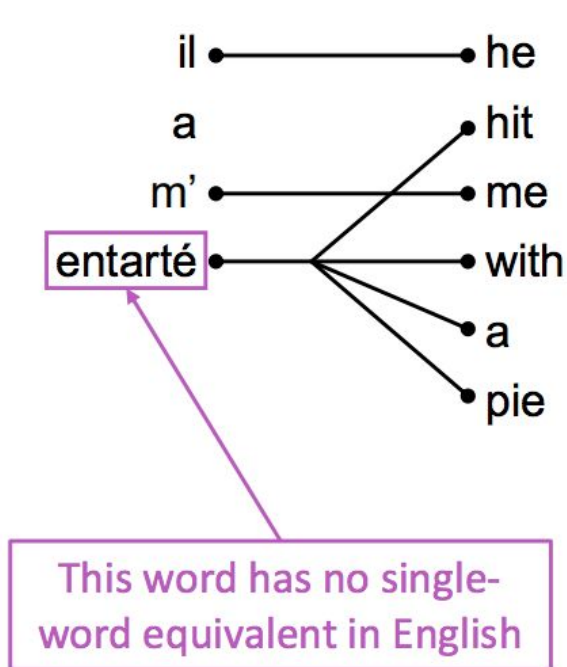
Alignment can be: **one-to-many**



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

1990-2010: Statistical Machine Translation

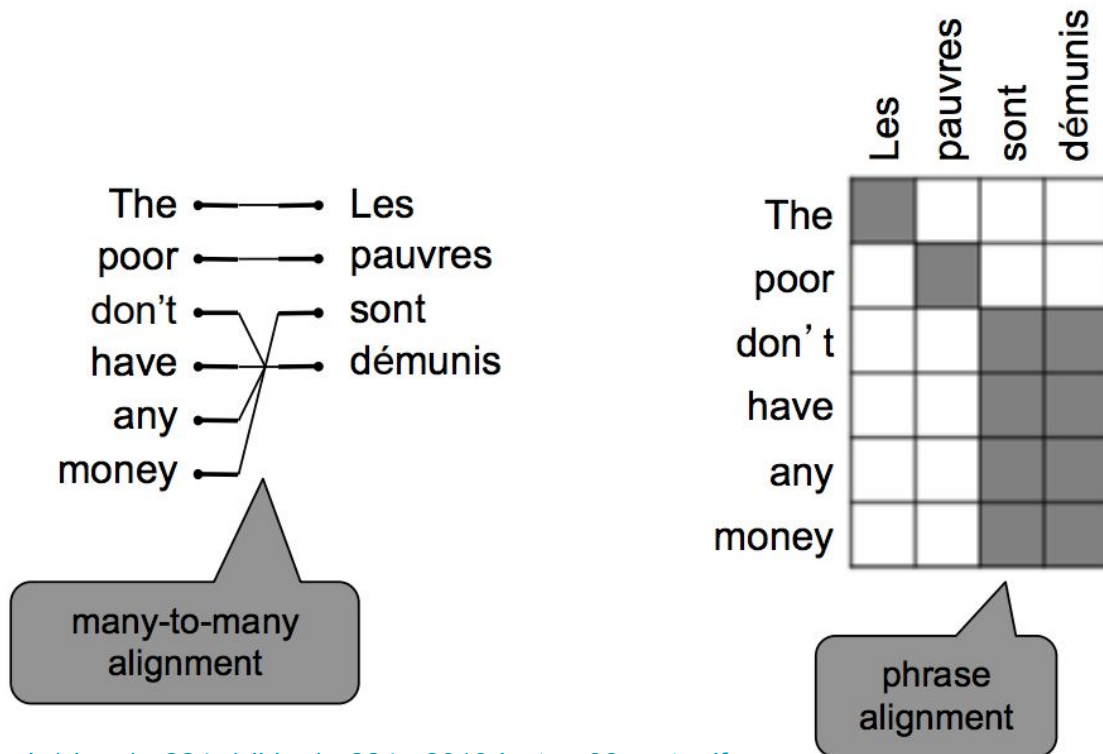
Some words are very fertile!




	he	hit	me	with	a	pie
il						
a						
m'						
entarté						

1990-2010: Statistical Machine Translation

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



1990-2010: Statistical Machine Translation

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


The equation is annotated with three colored curly braces underneath it. A purple brace is under the argmax_y term, a blue brace is under the $P(x|y)$ term, and a green brace is under the $P(y)$ term. Arrows point from these braces to their respective labels below.

Question:

How to compute
this argmax?

Translation Model

Language Model

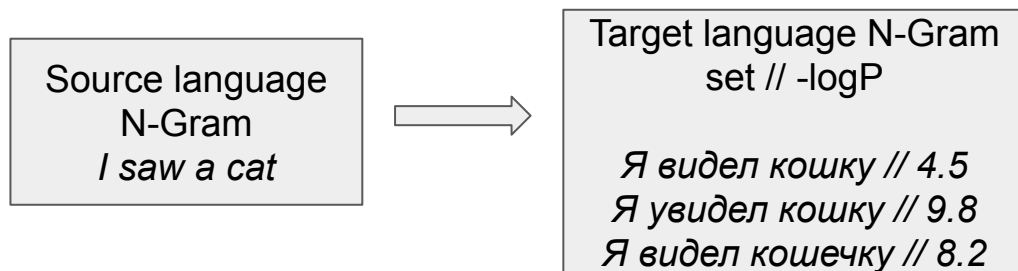
Enumerate every possible y and calculate the probability? No!

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

PBMT

- PBMT: phrase-based machine translation
 - N-Gram phrase table
 - N-Gram language model

Phase tables



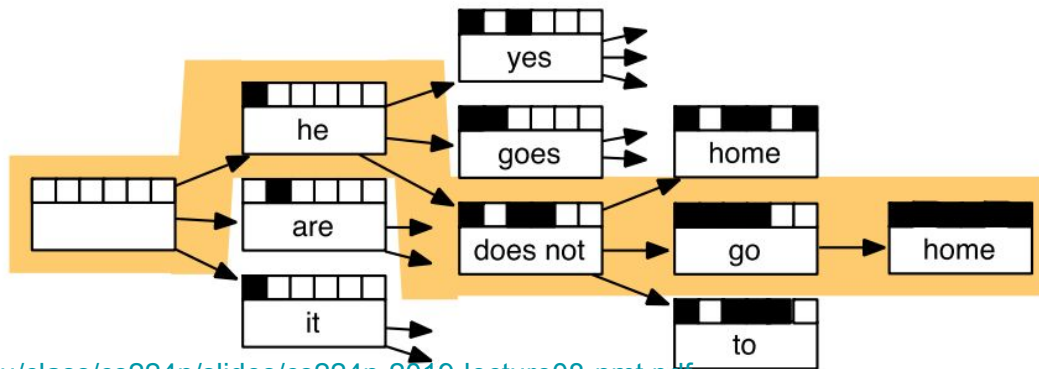
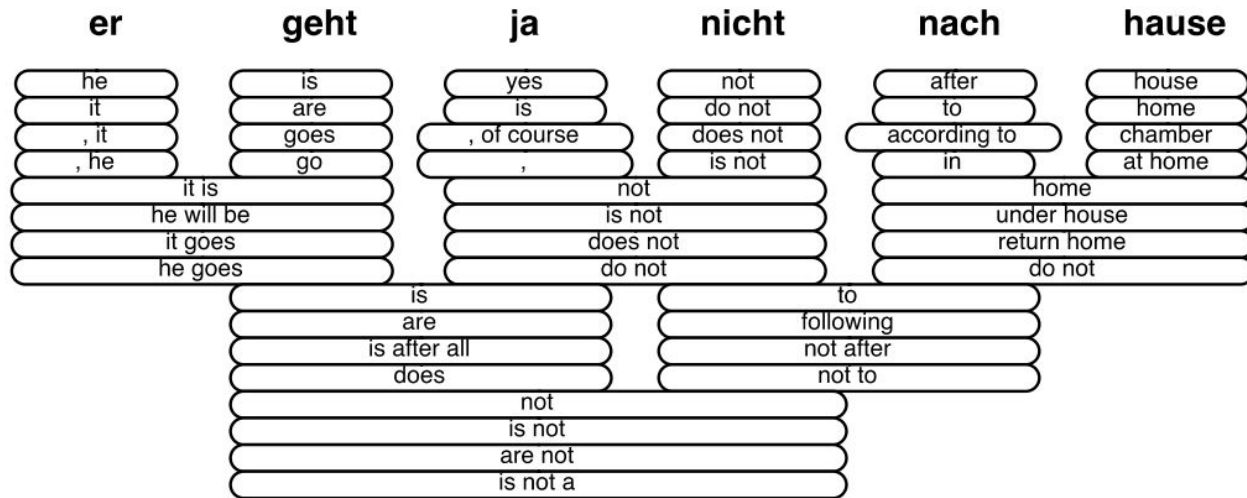
PBMT

- PBMT: phrase-based machine translation
 - N-Gram phrase table
 - N-Gram language model

NGram LM's

Source language
N-Gram // $-\log P$
(next | prefix)

I saw a cat
I saw a dog
I saw a bicycle



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!

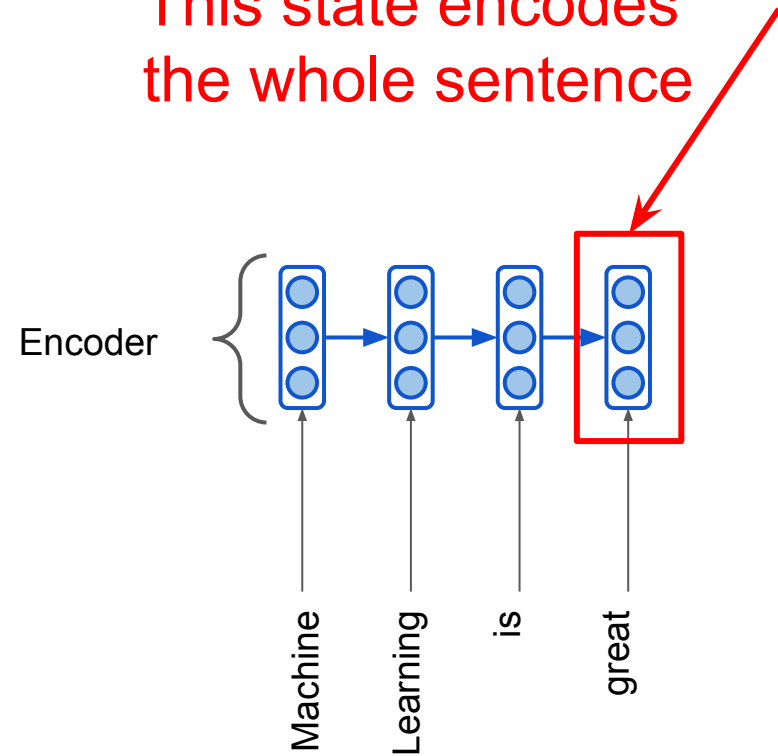
Neural Machine Translation

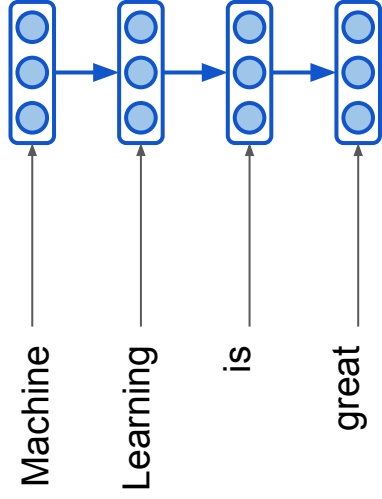
What is Neural Machine Translation?

- 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**), it involves two **RNNs**

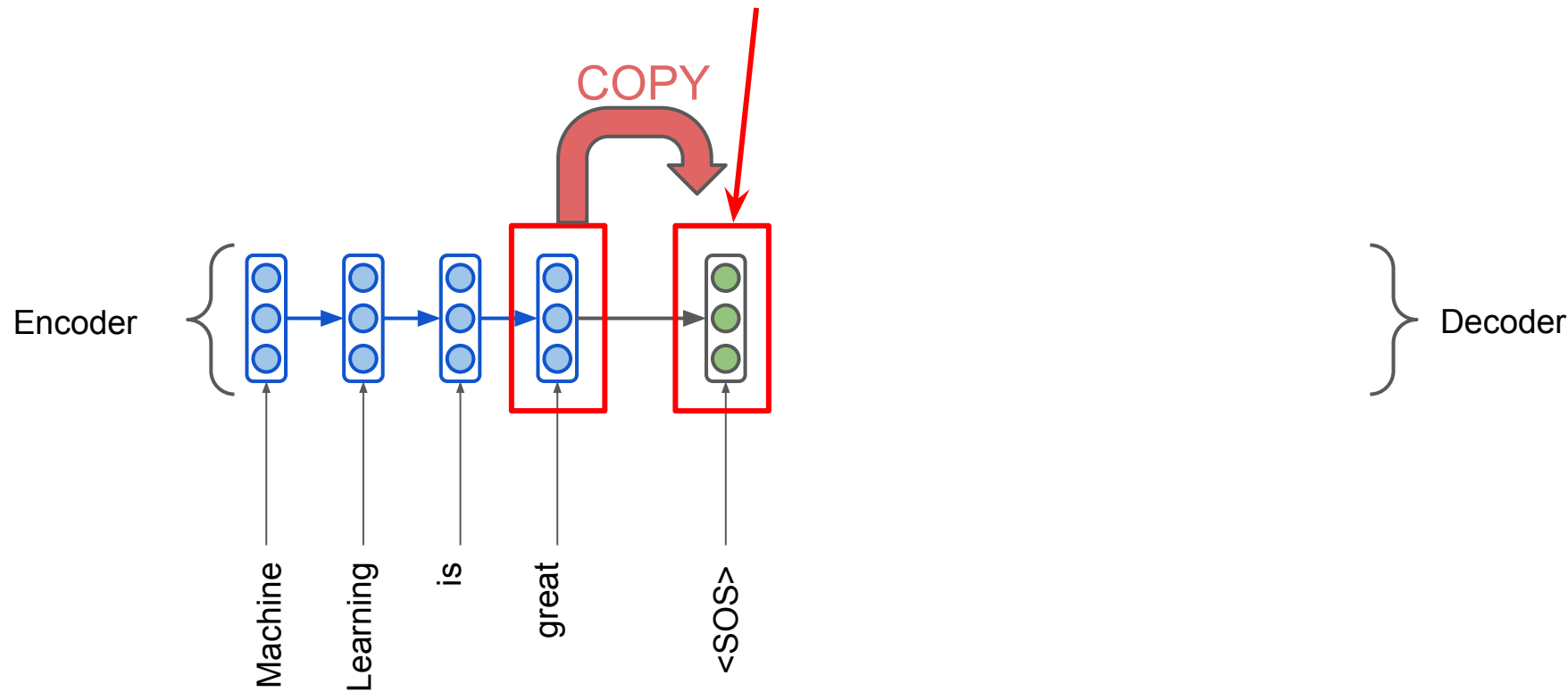
Seq2seq NMT

This state encodes
the whole sentence

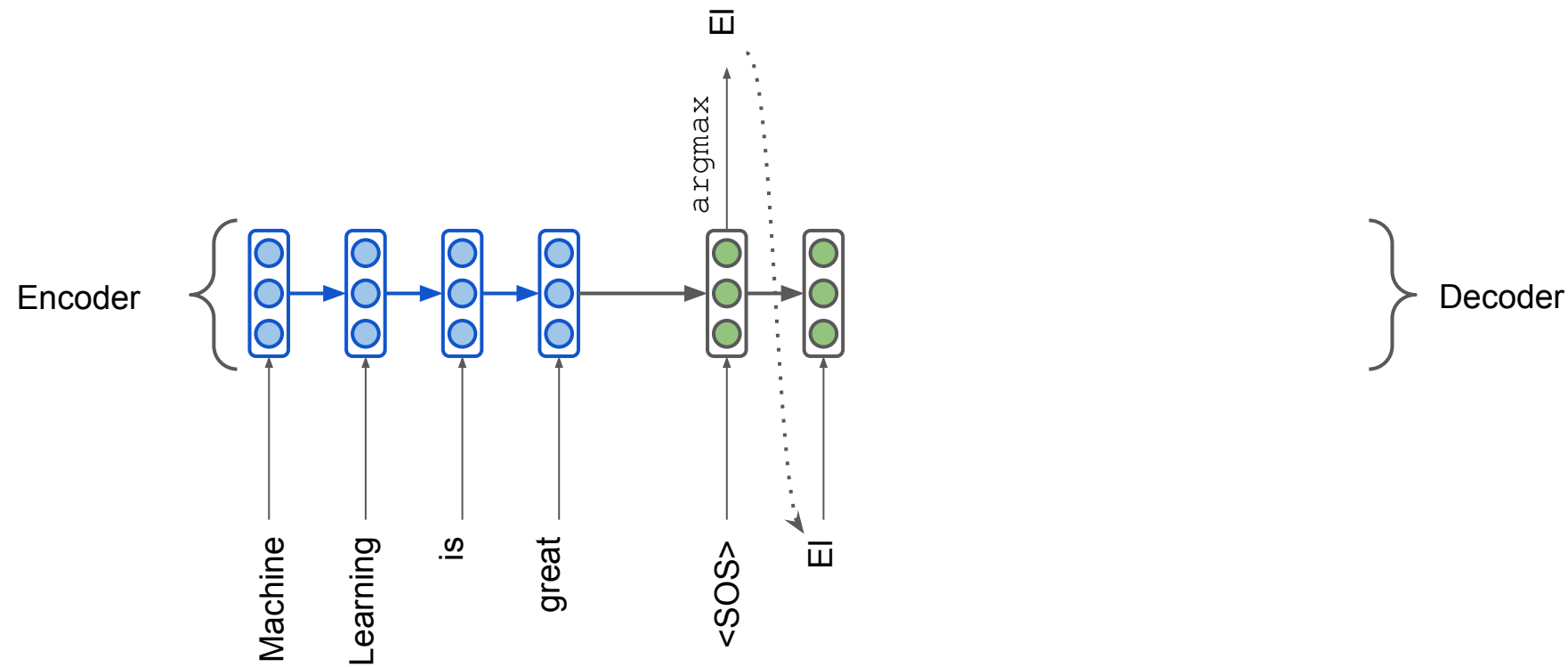




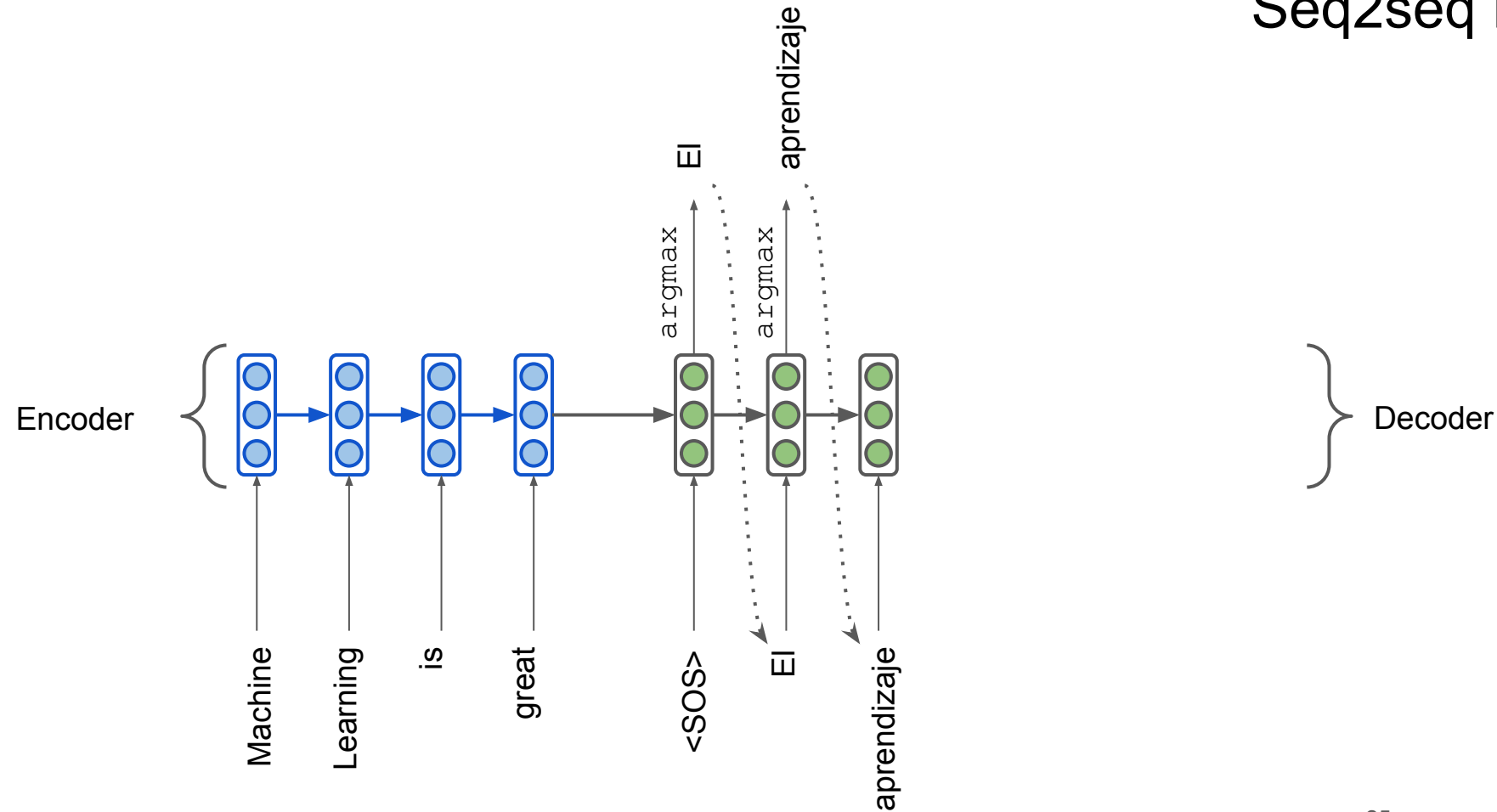
Forwarded as initial
hidden state to decoder



Seq2seq NMT

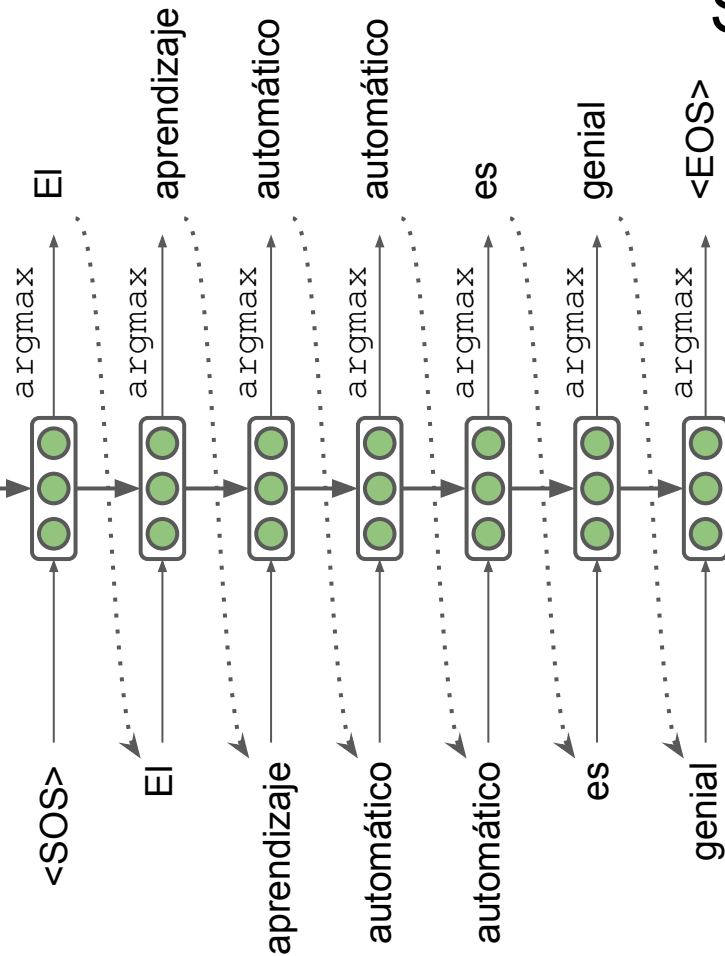
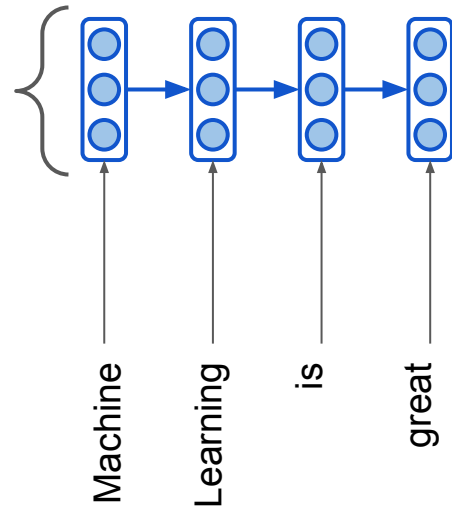


Seq2seq NMT



Seq2seq NMT

Encoder



Decoder

NMT: how does it work?

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

$$P(y|x) = P(y_2|y_1, x)P(y_3|y_1, y_2, x) \dots \underbrace{P(y_T|y_1, y_2, \dots, x)}$$

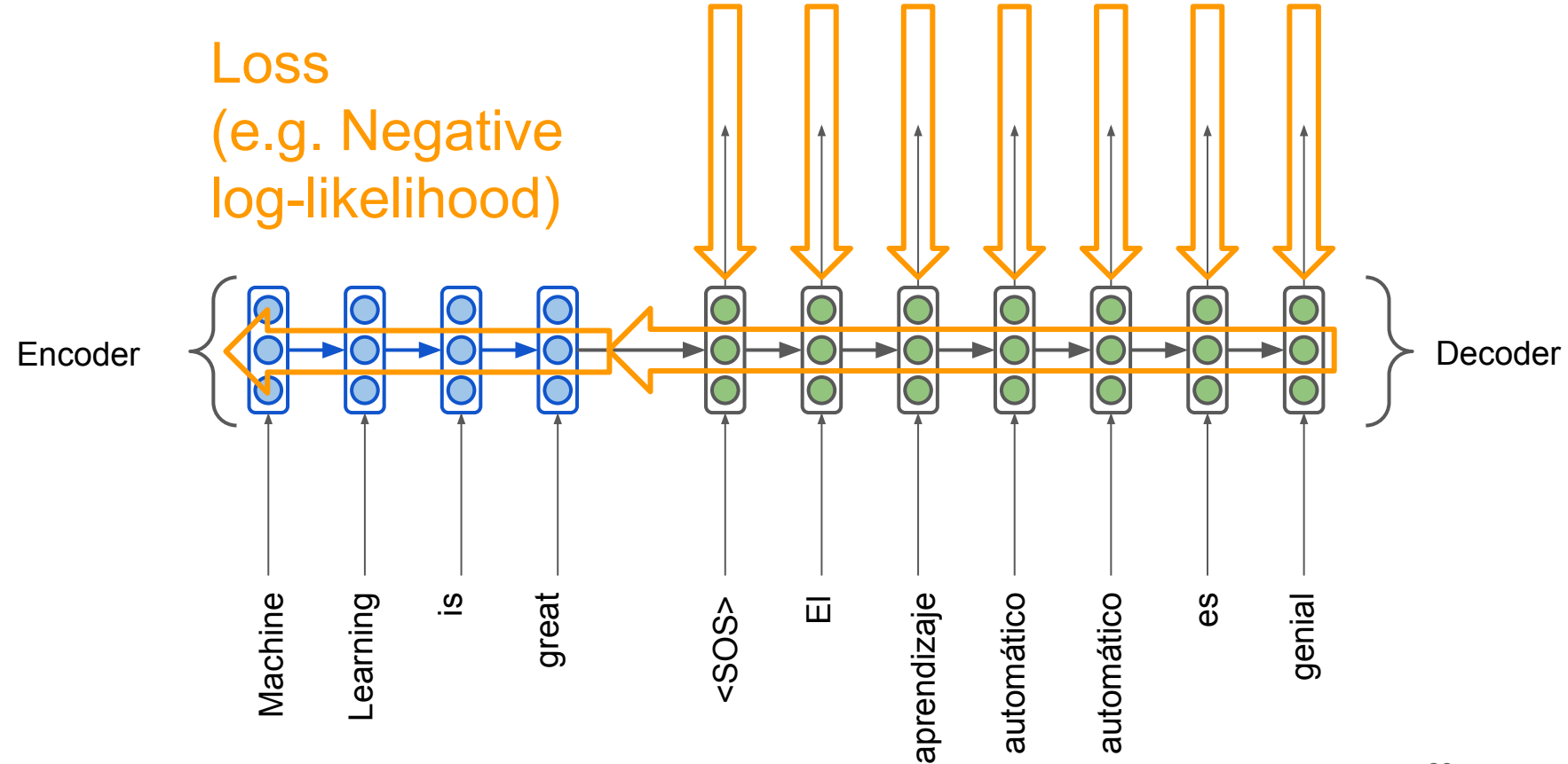
Probability of next word
in target language



- To train it we need a huge parallel corpus.

Seq2seq is trained end-to-end

Loss
(e.g. Negative
log-likelihood)

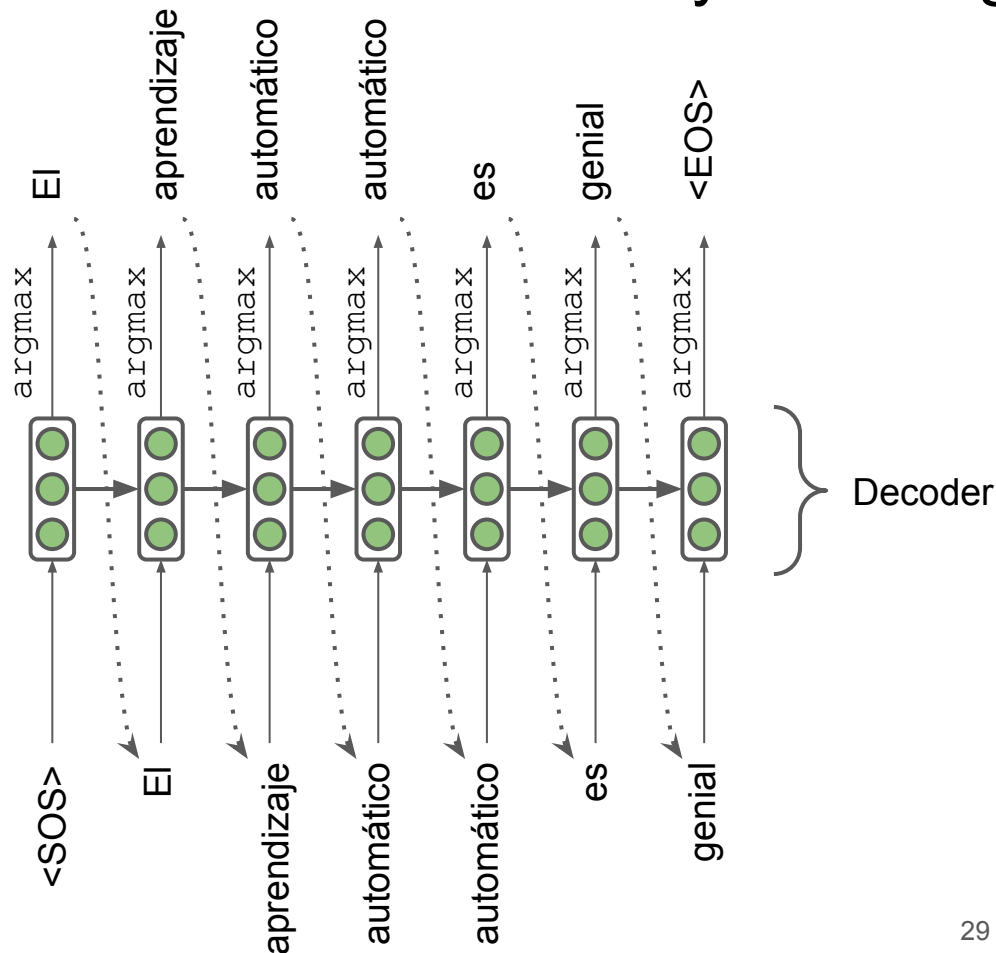


Greedy decoding

- Decoder predicts the most probable token (argmax) on each step
- The approach is **greedy**

Any problems with it?

Any mistake is treated as input on the next step!



- We want the translation that maximizes the likelihood:

$$P(y|x) = P(y_1|x) \prod_{t=2}^T P(y_t|y_1, \dots, y_{t-1}, x)$$

- We cannot compute all the possible sequences (exponential complexity)

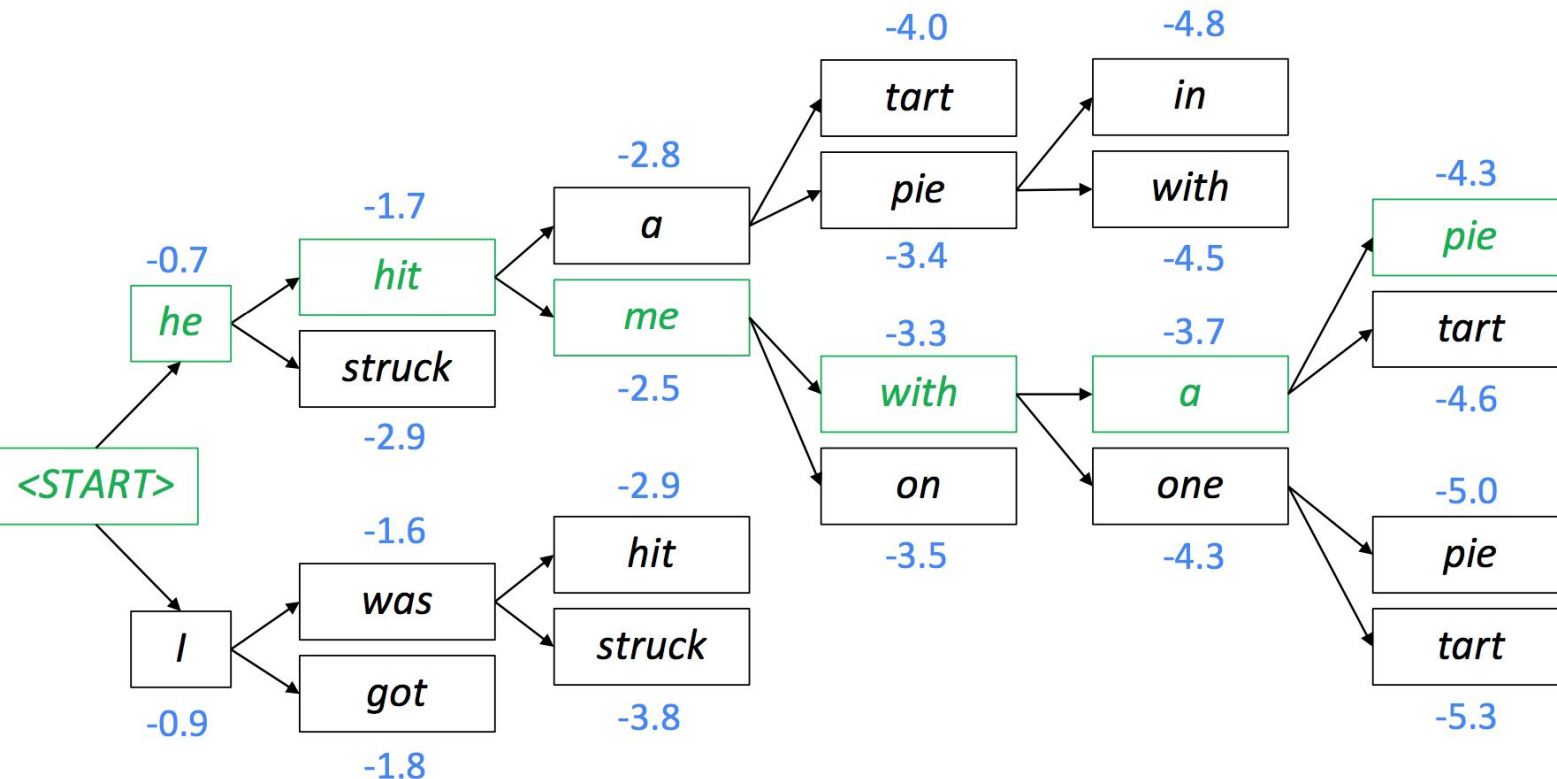
- On each step of decoder, keep track of the k most probable partial translations (which we call hypotheses)
- k is the beam size (in practice around 5 to 10)
- A hypothesis has a score which is its log probability:

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

- We search for high-scoring hypotheses, tracking top k on each step
- Beam search does not guarantee finding optimal solution

Beam search decoding: example

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



Beam search decoding: stopping criterion

- In **greedy decoding**, usually we decode until the model produces <EOS> token
- In **beam search decoding**, different hypotheses may produce <EOS> tokens on different timesteps
 - When a hypothesis produces <EOS>, 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?**

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

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: Israeli officials responsibility of airport safety
 2-GRAM MATCH 1-GRAM MATCH

REFERENCE: Israeli officials are responsible for airport security

SYSTEM B: airport security Israeli officials are responsible
 2-GRAM MATCH 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

- **ROUGE** (Recall-Oriented Understudy for Gisting Evaluation)
- **METEOR** (Metric for Evaluation of Translation with Explicit ORdering)
 - Uses synonyms from WordNet

Quality metrics for MT

- **Human evaluation**

Side-by-side (SbS HE)

Q: Pick the best translation

Src: Beam search decoding

System1: декодирование с поиском луча декодирование поиска луча

System2: декодирование поиска луча

Direct Assessment (DA)

Q: Rate this translation of 5-point scale

Src: Beam search decoding

Translation: декодирование поиска луча

Quality metrics for MT

Human eval is expensive

Solution: ML models trained to approximate HE

1. BLEURT (Google) - BERT finetuned on DA ratings
2. XCOMET (Unbabel) - XLM finetuned on DA + MQM ratings
3. LLM-based estimators - prompt LLM to estimate MT quality

- 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 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 ▾

paper jam Edit

[Open in Google Translate](#)

Spanish ▾

Mermelada de papel

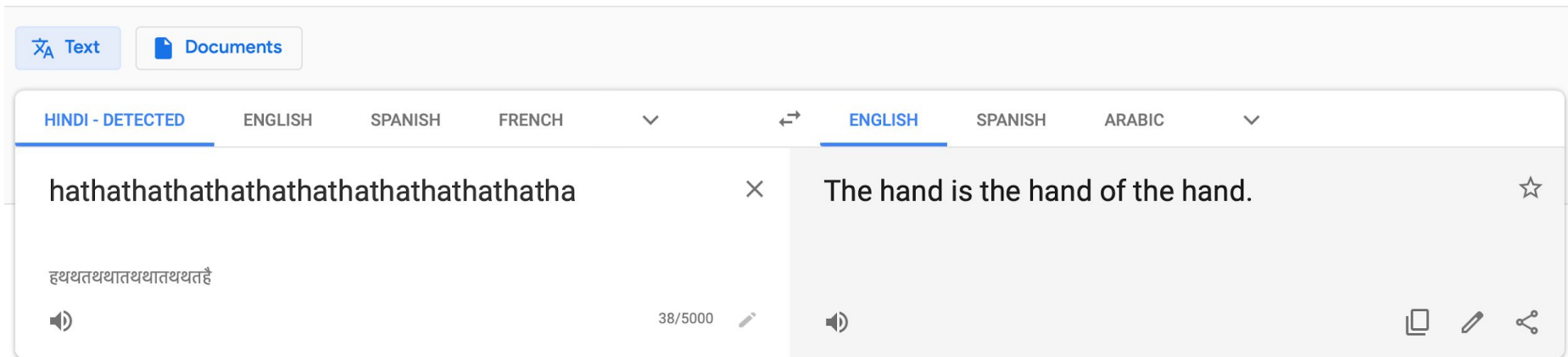
[Feedback](#)



NMT: disadvantages

A screenshot of the Google Translate web application. The interface is split into two main sections. On the left, under the 'Somali' language header, there is a text input area containing the phrase 'ag ag ag ag ag ag ag ag ag ag ag ag ag ag ag ag ag'. Below this text, there is an 'Edit' link. Above the text, there is a button labeled 'Translate from Irish'. On the right, under the 'English' language header, the translated text is displayed: 'As the name of the LORD was written in the Hebrew language, it was written in the language of the Hebrew Nation'. At the bottom right corner, there is a 'Feedback' link.

NMT: disadvantages



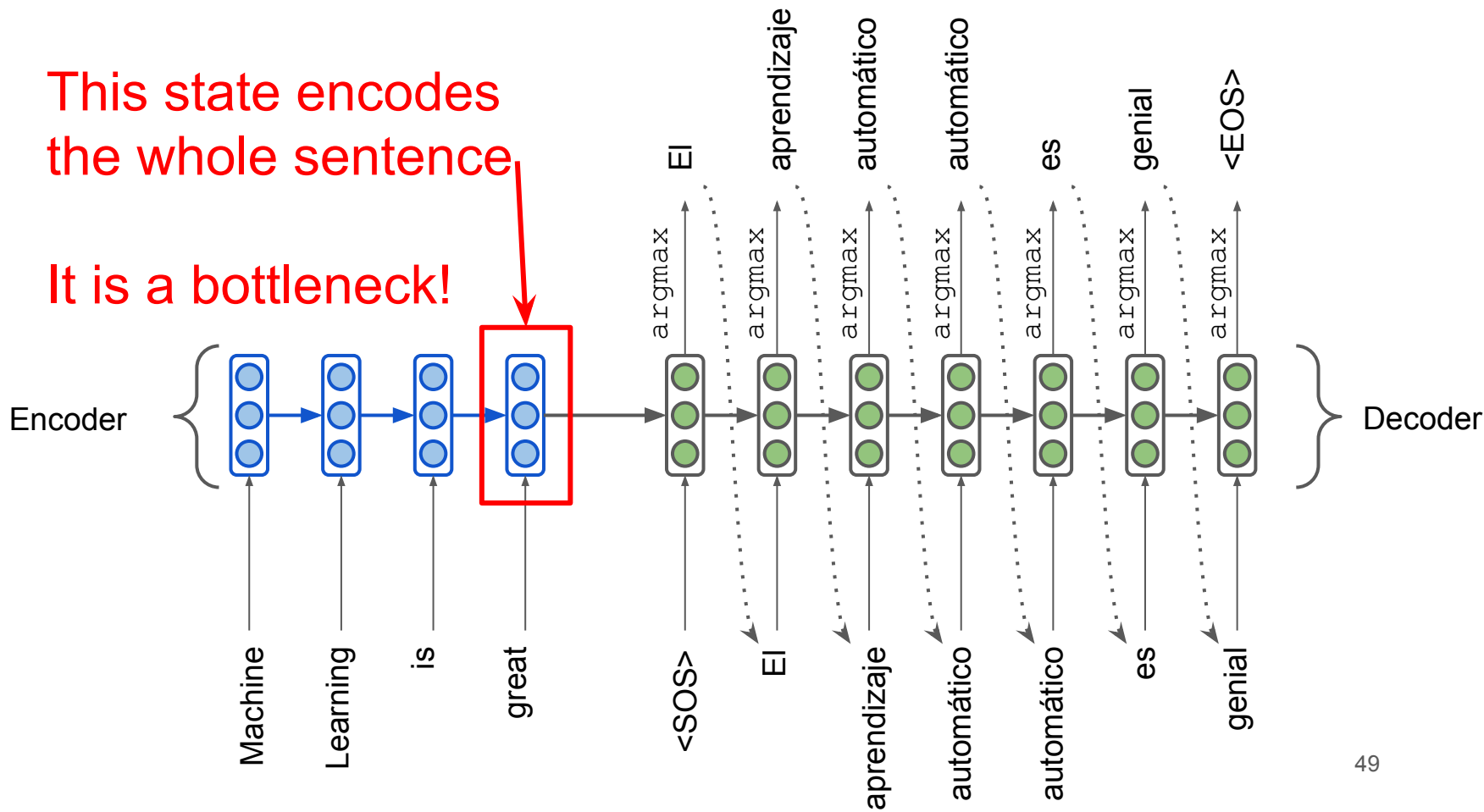
Is Machine Translation solved?

- Challenges yet to be resolved
 - Use of large context in translation
 - Low-resource language pairs (no big parallel corpora)
 - Slang and narrow domains
 - Robustness to errors and typos

Attention

This state encodes
the whole sentence

It is a bottleneck!

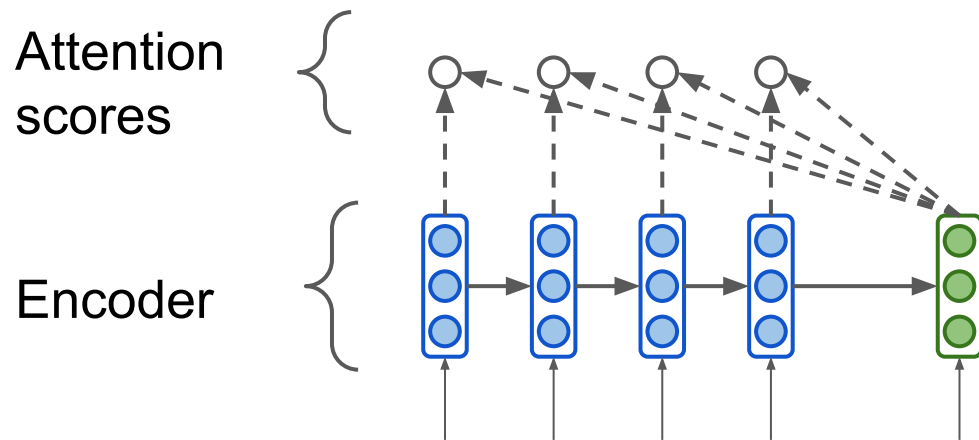


Main idea:

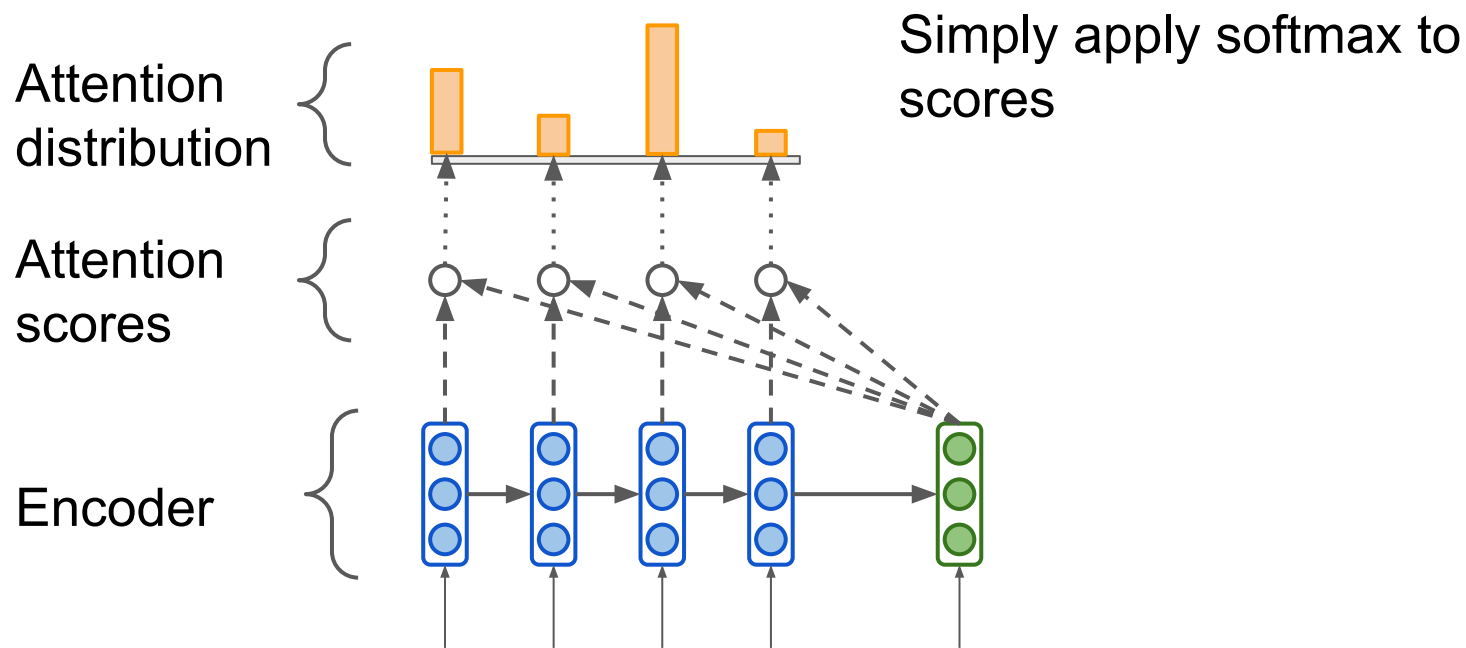
on each step of the **decoder**, use **direct connection to the encoder** to focus on a particular part of the source sequence



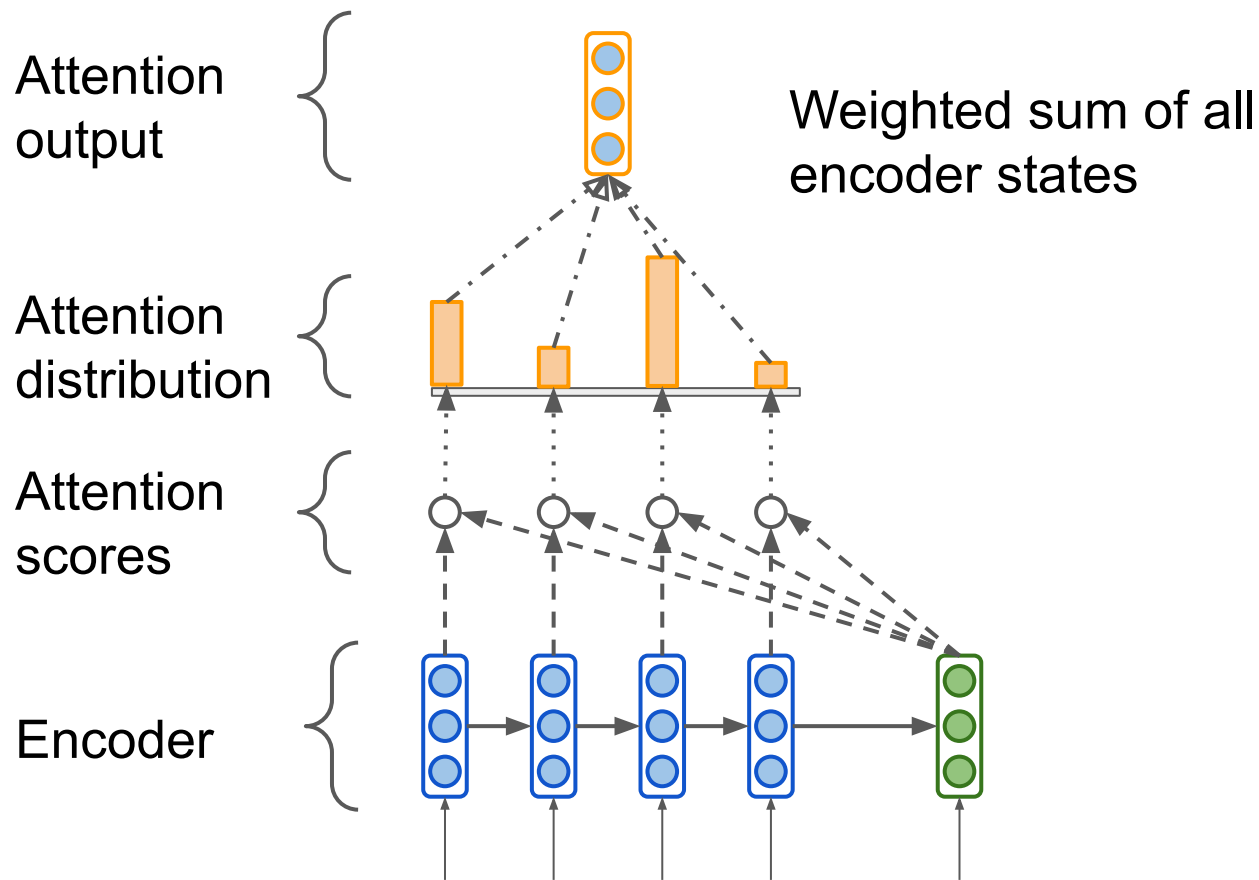
Seq2seq with attention



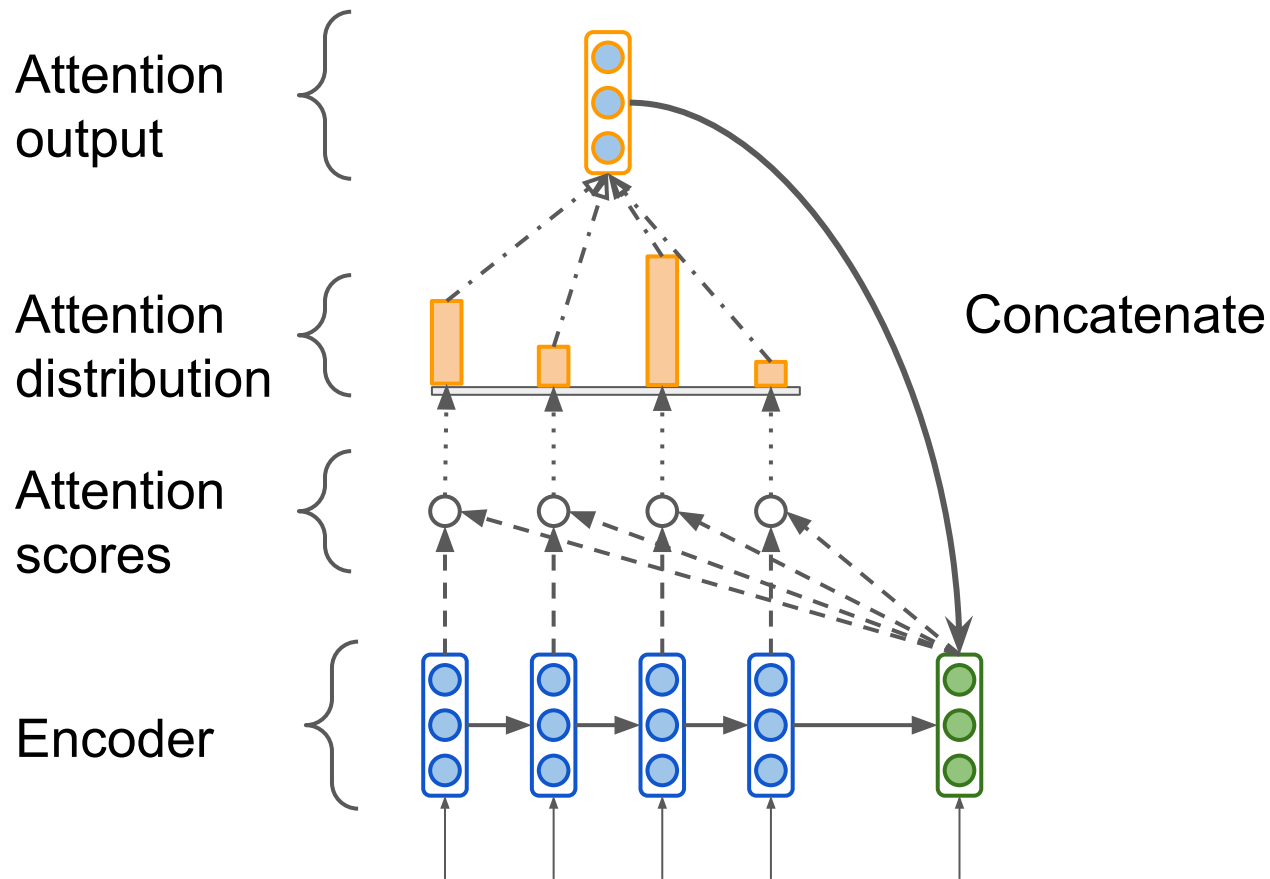
Seq2seq with attention



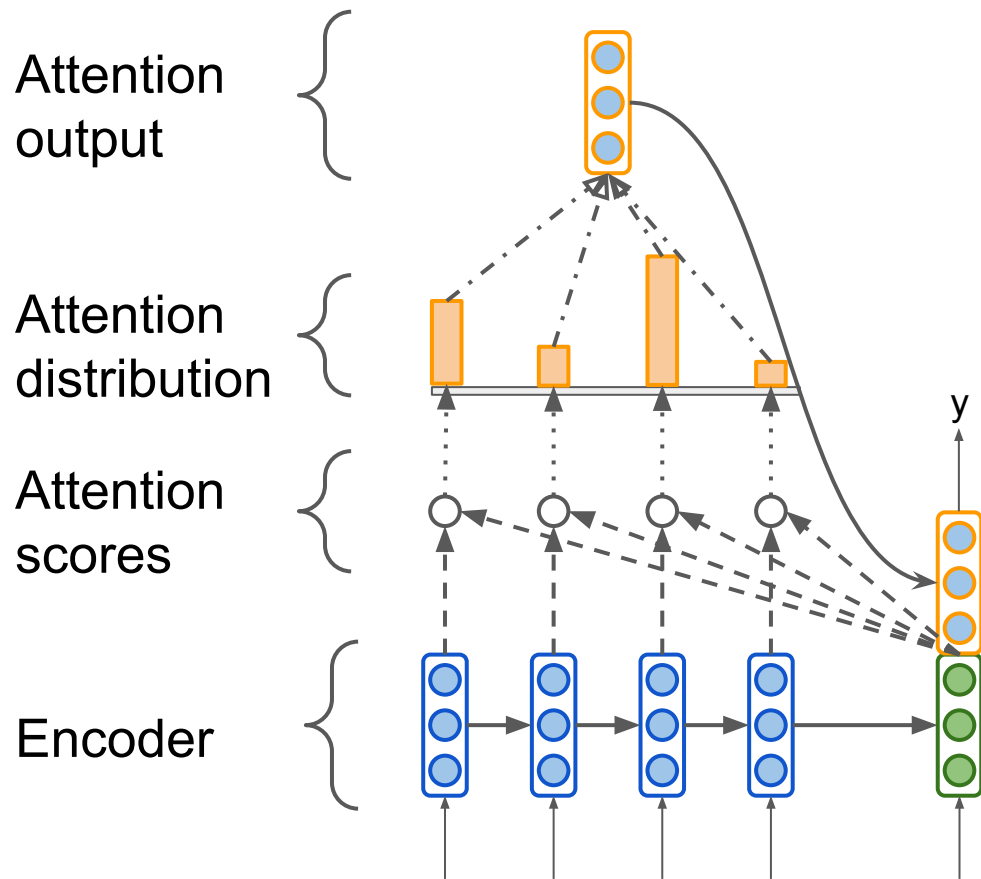
Seq2seq with attention



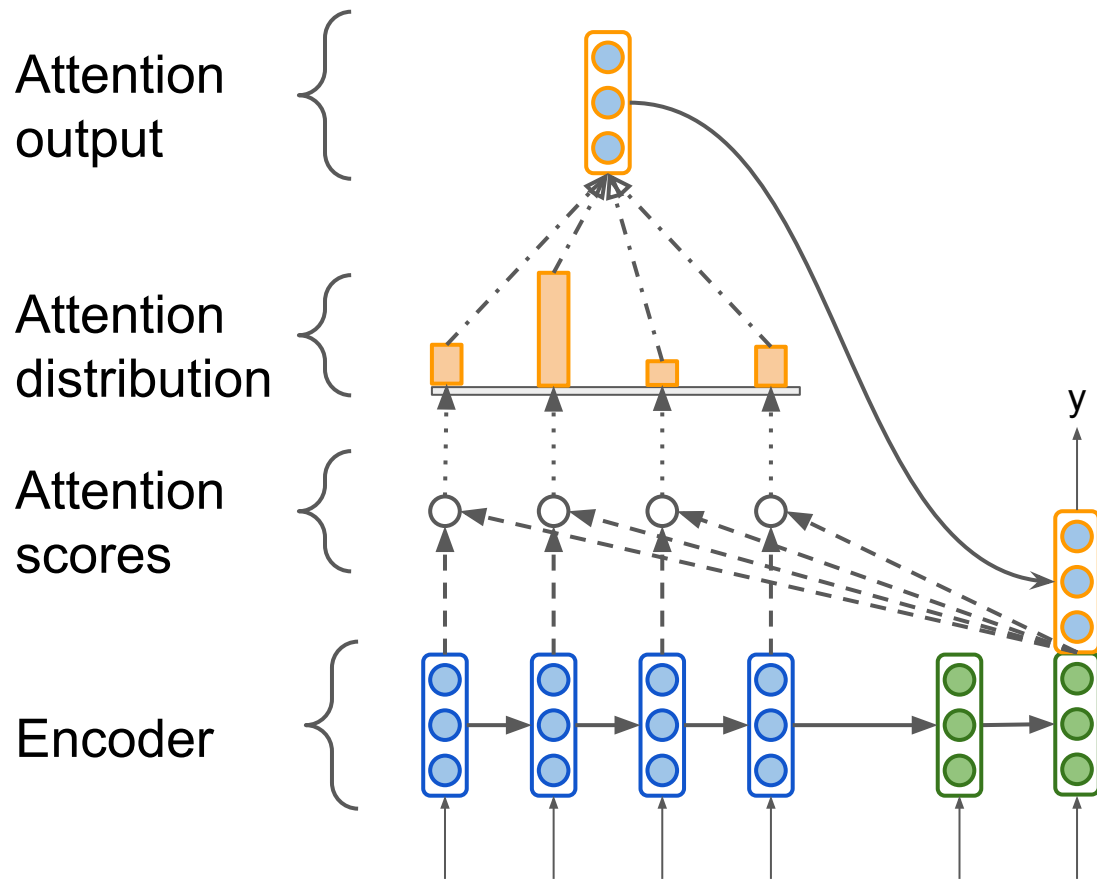
Seq2seq with attention



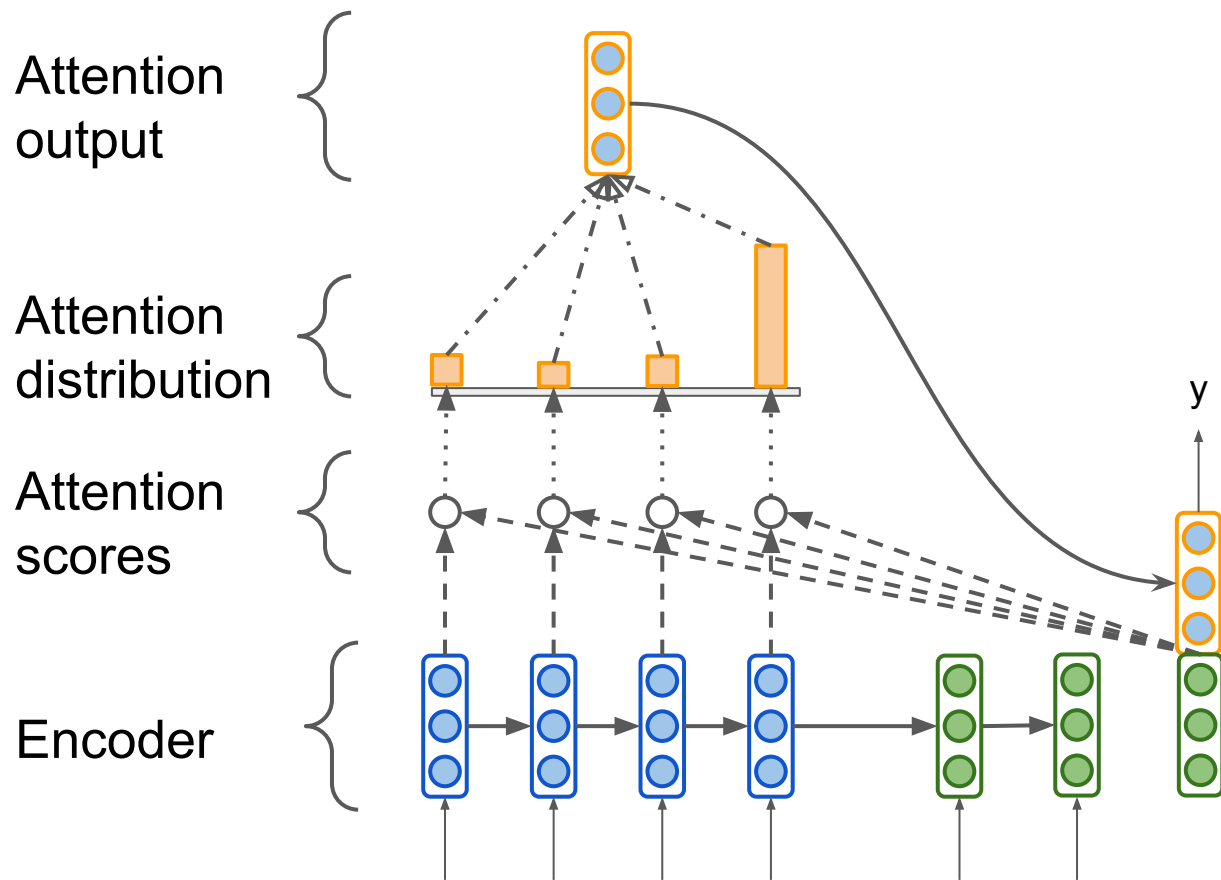
Seq2seq with attention



Seq2seq with attention



Seq2seq with attention



Denote encoder hidden states $\mathbf{h}_1, \dots, \mathbf{h}_N \in \mathbb{R}^k$
and decoder hidden state at time step t $\mathbf{s}_t \in \mathbb{R}^k$

The attention scores \mathbf{e}^t can be computed as dot product

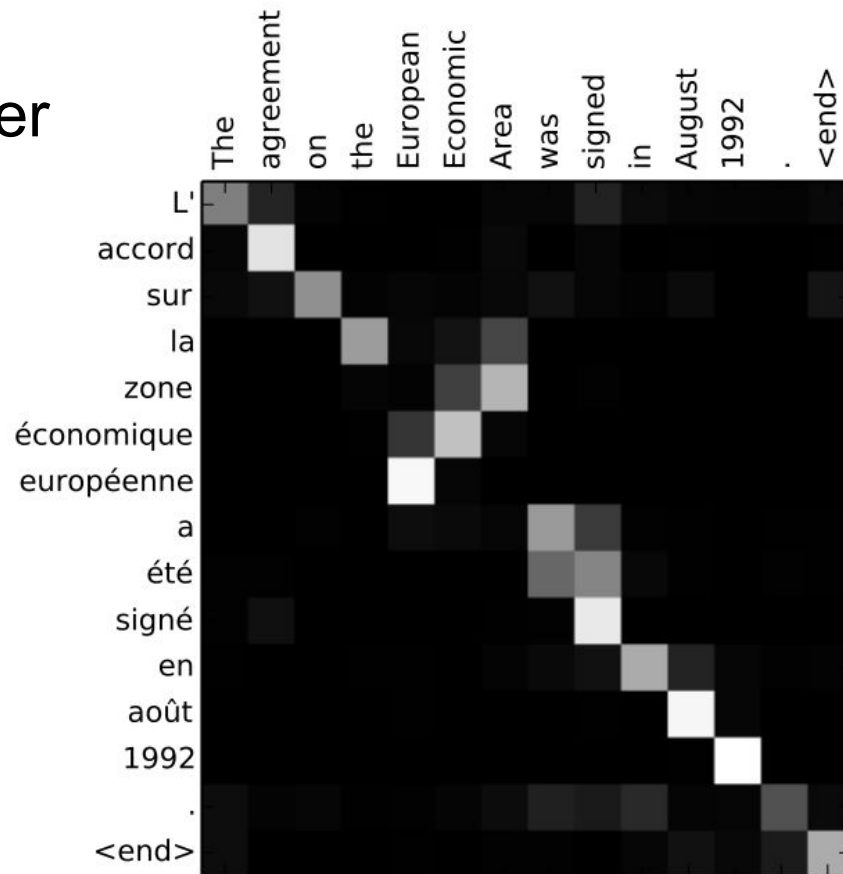
$$\mathbf{e}^t = [\mathbf{s}^T \mathbf{h}_1, \dots, \mathbf{s}^T \mathbf{h}_N]$$

Then the attention vector is a linear combination of encoder states

$$\mathbf{a}_t = \sum_{i=1}^N \alpha_i^t \mathbf{h}_i \in \mathbb{R}^k, \text{ where } \boldsymbol{\alpha}_t = \text{softmax}(\mathbf{e}_t)$$

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): $e_i = s^T h_i \in \mathbb{R}$
- Multiplicative attention: $e_i = s^T W h_i \in \mathbb{R}$
 - $W \in \mathbb{R}^{d_2 \times d_1}$ - weight matrix
- Additive attention: $e_i = v^T \tanh(W_1 h_i + W_2 s) \in \mathbb{R}$
 - $W_1 \in \mathbb{R}^{d_3 \times d_1}, W_2 \in \mathbb{R}^{d_3 \times d_2}$ - weight matrices
 - $v \in \mathbb{R}^{d_3}$ - weight vector

