Lesson 9
Neural machine translation

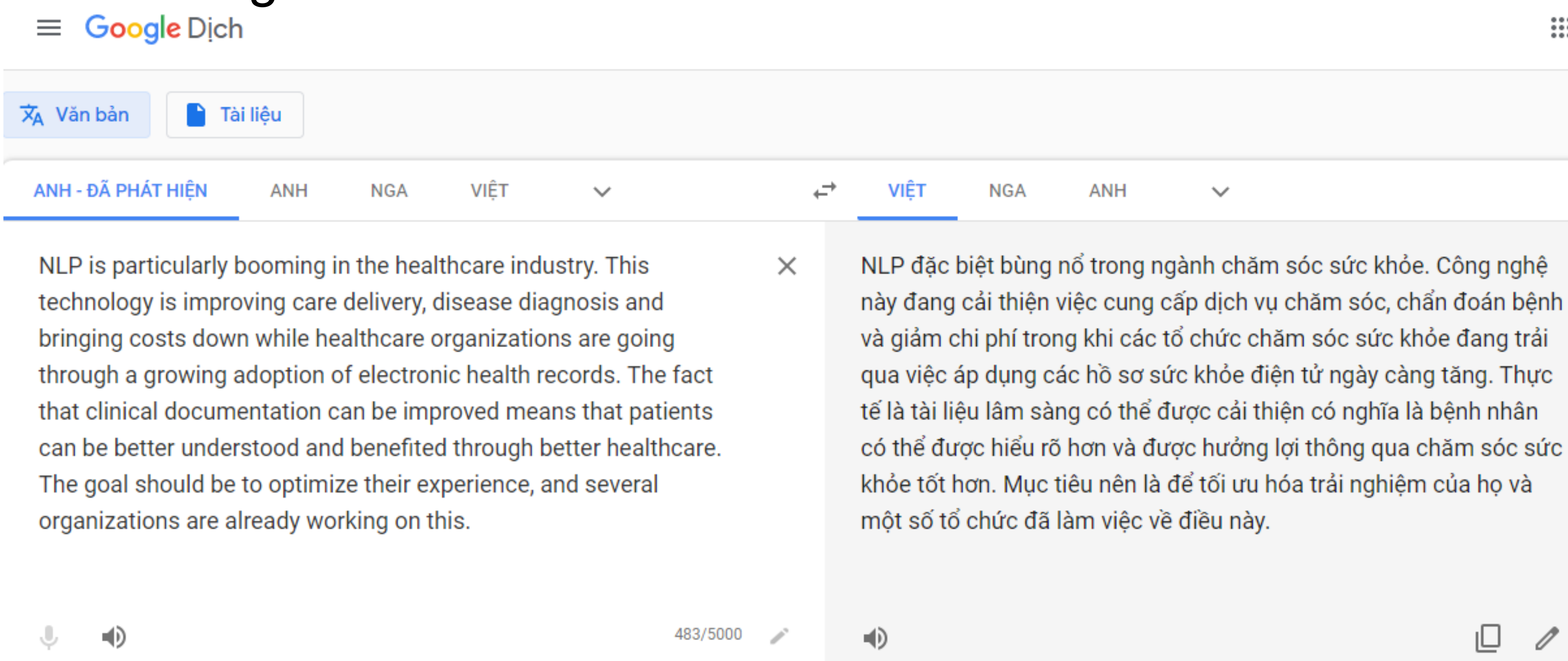
Outline

1. Introduction to machine translation
2. Neural machine translation
3. Attention mechanism

Introduction to machine translation

Machine translation

- Google translate



Machine Translation

- Machine translation (MT) is the operation of translating a sentence x from one language (called the source language) into a sentence y in another language (called the target language).

x: *L'homme est né libre, et partout il est dans les fers*

↓

y: *Man is born free, but everywhere he is in chains*

History of machine translation

- Beginning in the 1950s
- Translate from Russian to English (demand comes from the cold war)
- The translation system is mainly rule-based, using a dictionary to map Russian words to English

1 minute video showing 1954 MT:
https://youtu.be/K-HfpsHPmvw

Rule based translation

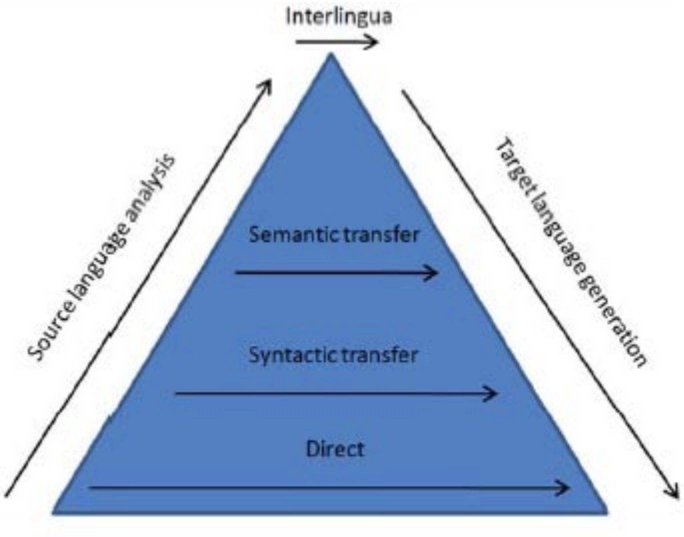
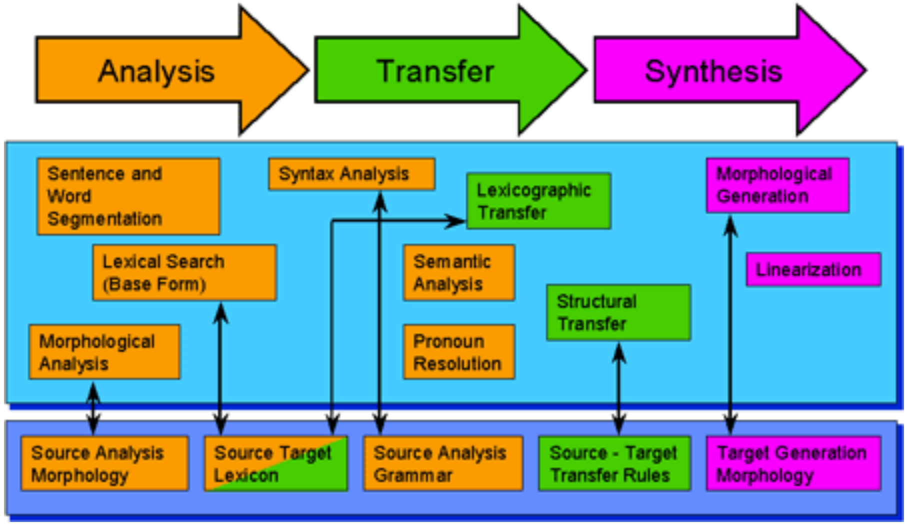


Figure 1: The Vauquois triangle



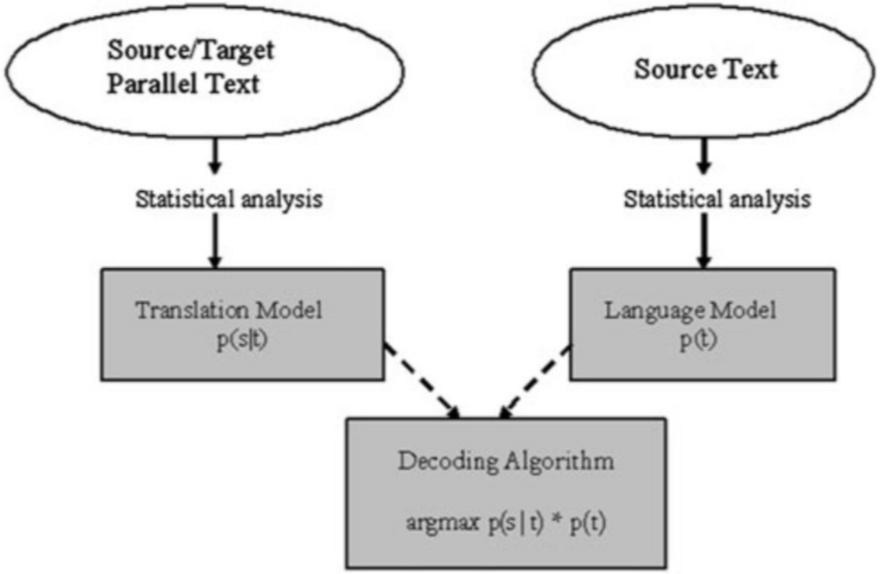
© https://www.coroflot.com/tuyenduong/machine-translation

Rule based translation

- Lots of manual handling and human effort
  - Mapping dictionary from Source – Destination
  - Transformation rules (lexical, structure)
  - Morphological rules
- Low quality

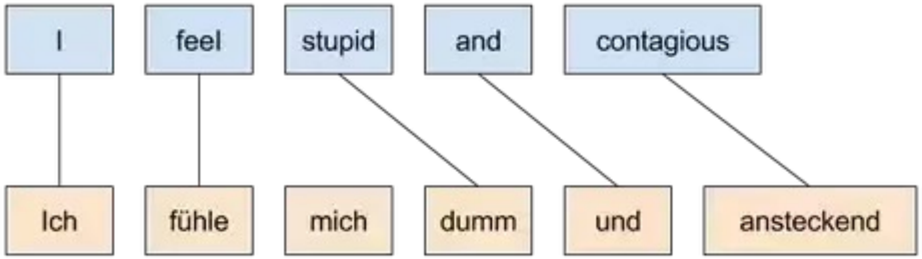
Statistical machine translation (1990-2010)

- Statistical machine translation learns a probabilistic model from data
- Objective: Find the best sentence in the target language for the input sentence in the source language



One way to model P(s|t)

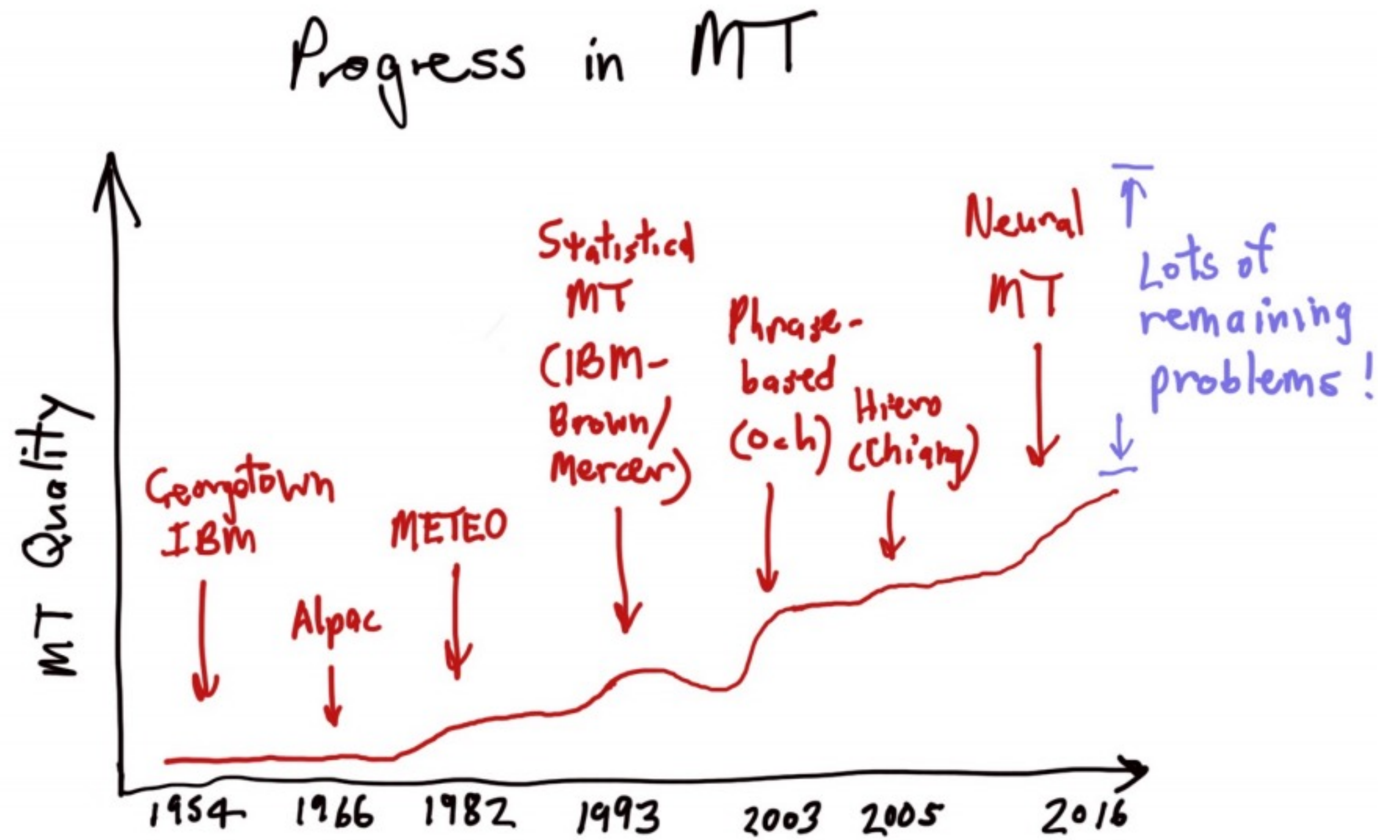
- Assumption: Match each word in the source sentence with the words in the target sentence
- Alignment vector a = [1,2,4,5,6]
- Objective: Find a way to maximize P(s,a|t)



Disadvantages of statistical machine translation

- The best systems following this approach are very complex, each containing many small, independently designed modules.
  - Still not as efficient as a human
- Requires a lot of manual handling and human effort
  - Feature engineering
  - External resources
- Maintenance costs are high
- Cannot reuse efforts when switching to another language pair,

Progress in machine translation

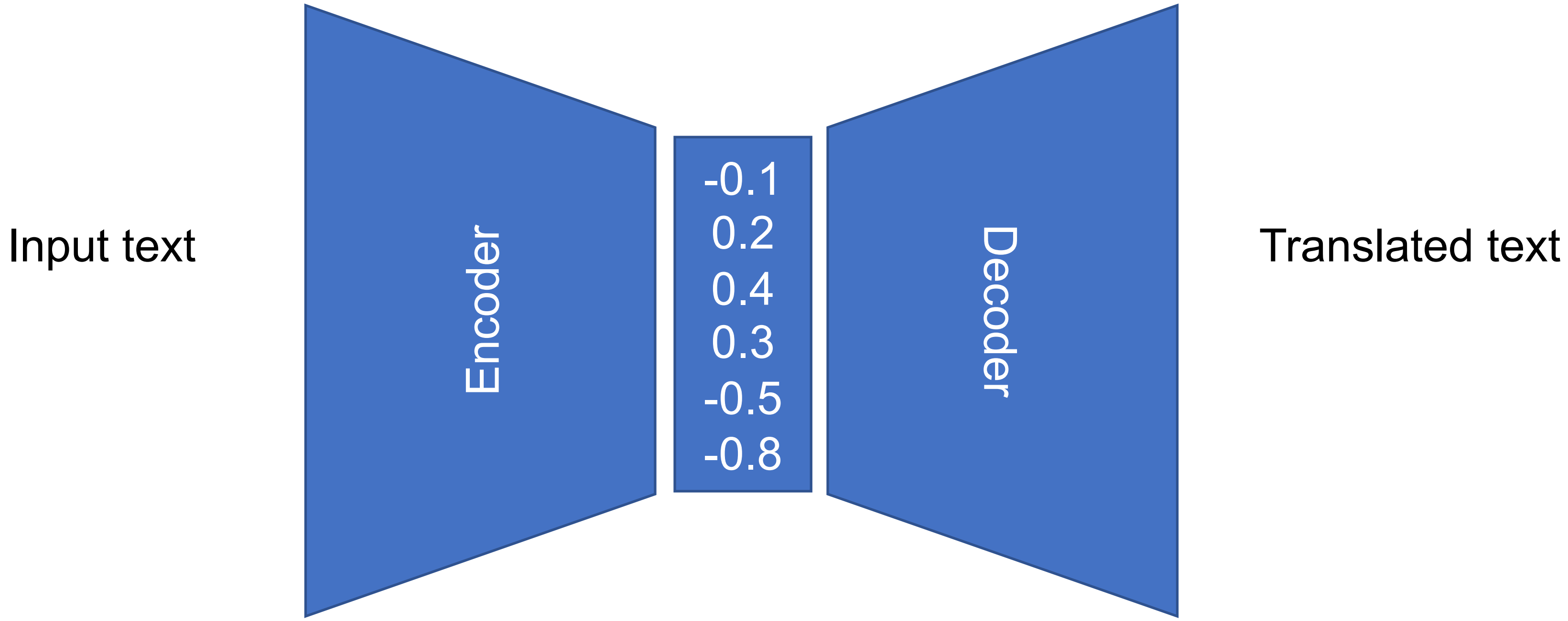


© https://sites.google.com/site/acl16nmt/home

Neural Machine Translation

Neural Machine Translation is the approach of modeling the entire MT process via one big artificial neural network (ACL 2016)

Sequence-to-sequence model

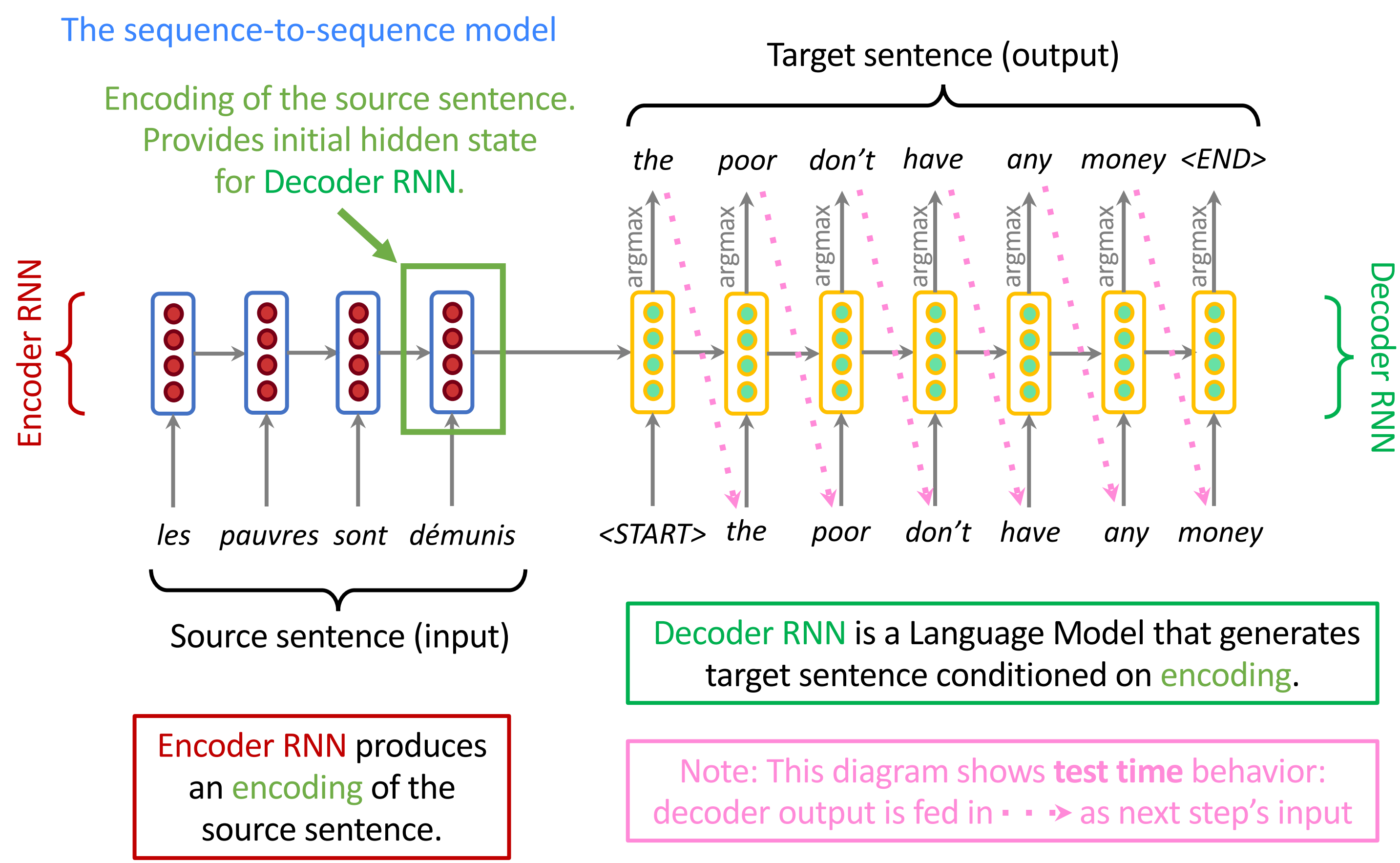


- The RNN encoder generates the “encoding information” of the source sentence
- The RNN decoder generates the target sentence based on the encoding information of the source sentence

Sequence-to-sequence model

- The seq2seq model can be used for many other problems such as:
  - Text summary (long text → short text)
  - Conversation (previous sentence → next sentence)
  - Code generation (natural language → python code)
  - …

Neural machine translation (NMT)



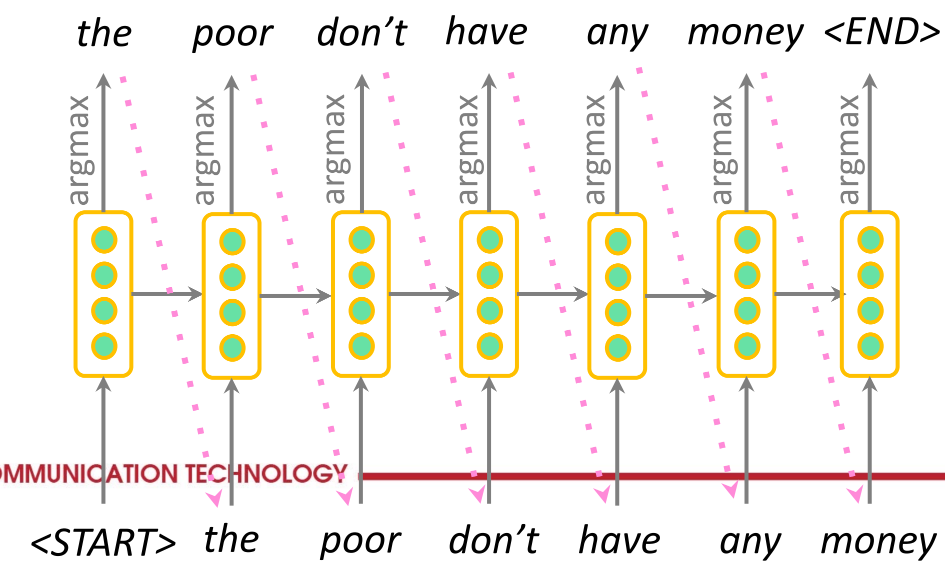
© Tensorflow for deep learning research. Standford

MT as conditional language model

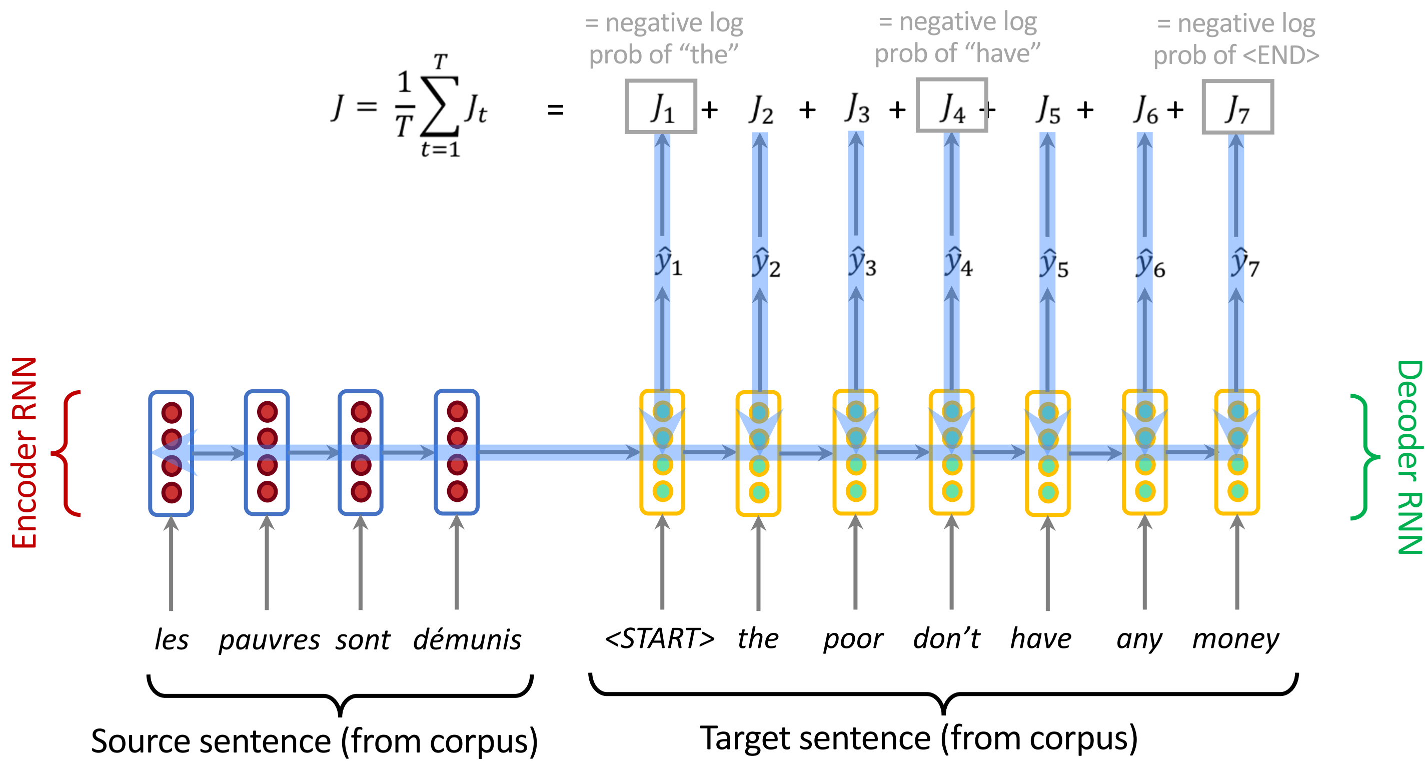
- NMT model: $P(y|x)$

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

- Conditional language model
  - Language modeling: predicting a word based on the context of surrounding words
  - Conditional: the prediction is based on additional conditions from the source sentence



Training seq2seq model

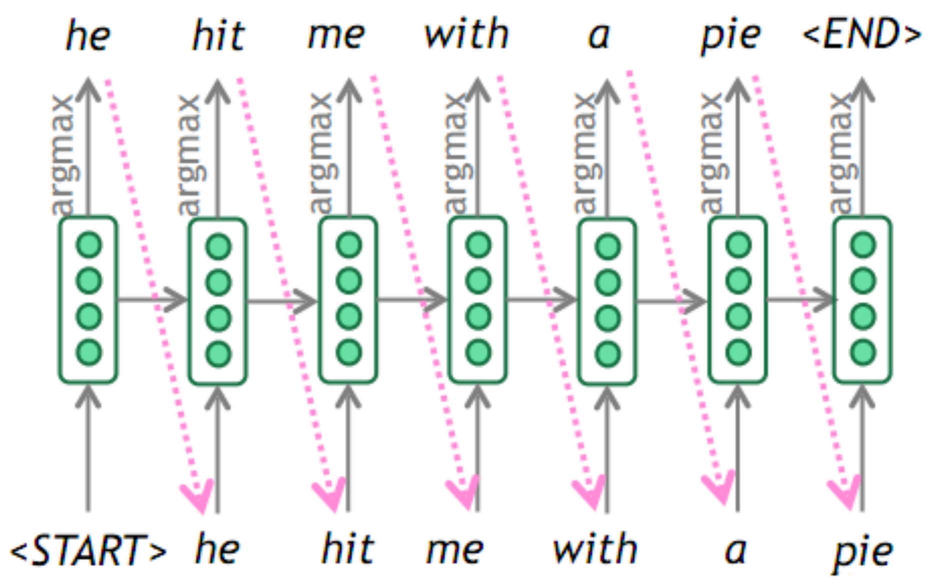


Seq2seq is optimized as a **single system.**
Backpropagation operates *“end to end”*.

© Tensorflow for deep learning research. Standford

Decoding in Seq2seq

- Decode the target sentence by taking argmax at each step
- This is greedy decoding
- If you make a mistake at a certain step, you will be wrong in the following steps, there is no way to go back to correct it.



SOICT SCHOOL OI

Beam search in decoding

Searching method

- We want to find the target sentence y (length T) that maximizes the posterior probability:

$$
\begin{aligned}
P(y|x) &= P(y_1|x)\, P(y_2|y_1, x)\, P(y_3|y_1, y_2, x) \ldots, P(y_T|y_1, \ldots, y_{T-1}, x) \\
&= \prod_{t=1}^{T} P(y_t|y_1, \ldots, y_{t-1}, x)
\end{aligned}
$$

- We can compute for all of y's alternatives.
- Complexity V^T where V is the size of the vocabulary set.

Beam search method

- Idea: At each decoding step, we maintain k partial solutions with the highest probability (called hypotheses).
- k is the beam size
- A hypothesis $y_1, y_2, \ldots, y_t$ has a score equal to the log of its probability value:

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

- All scores are negative, the higher the better
- We will keep the k hypotheses with the highest score at each step
- Beam search does not guarantee optimal solution
- But much more efficient than the brute-force method

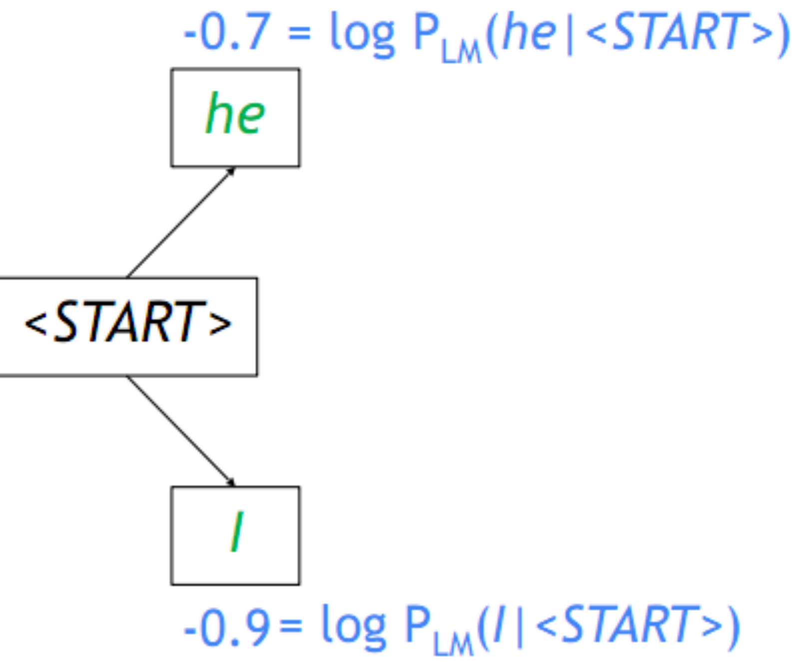
Example: beam search with k = 2

- Calculate the probability distribution of the next word

<START>

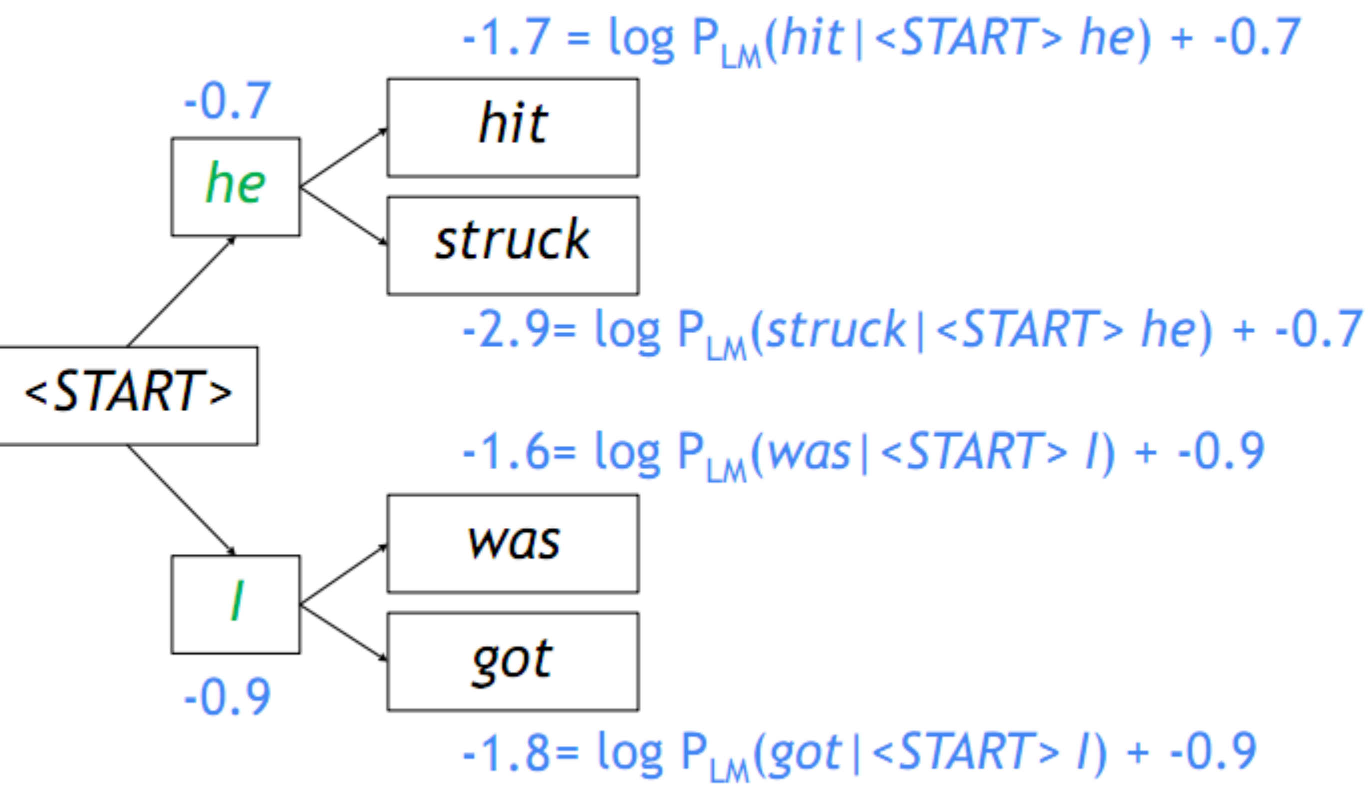
Example: beam search with k = 2

- Keep the two options with the highest score



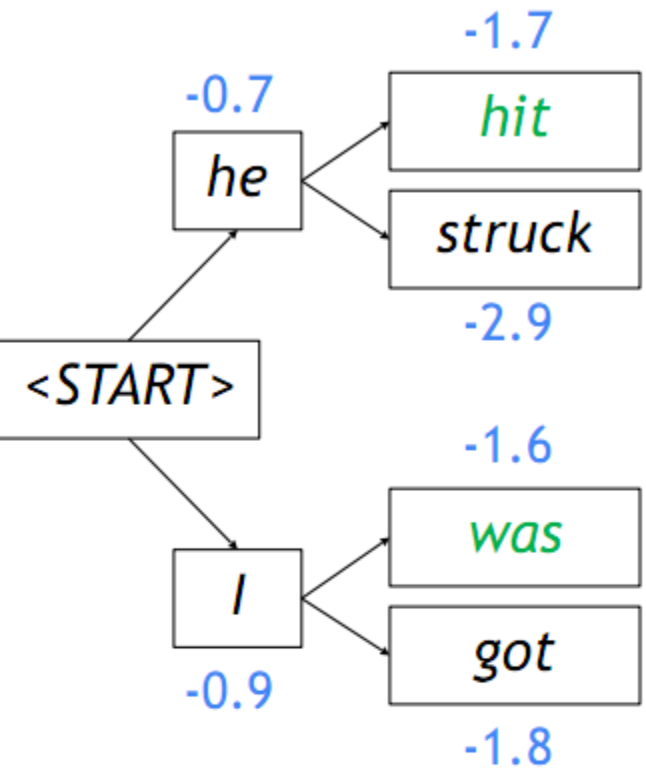
Example: beam search with k = 2

- For each hypothesis, find the next k with the highest scores



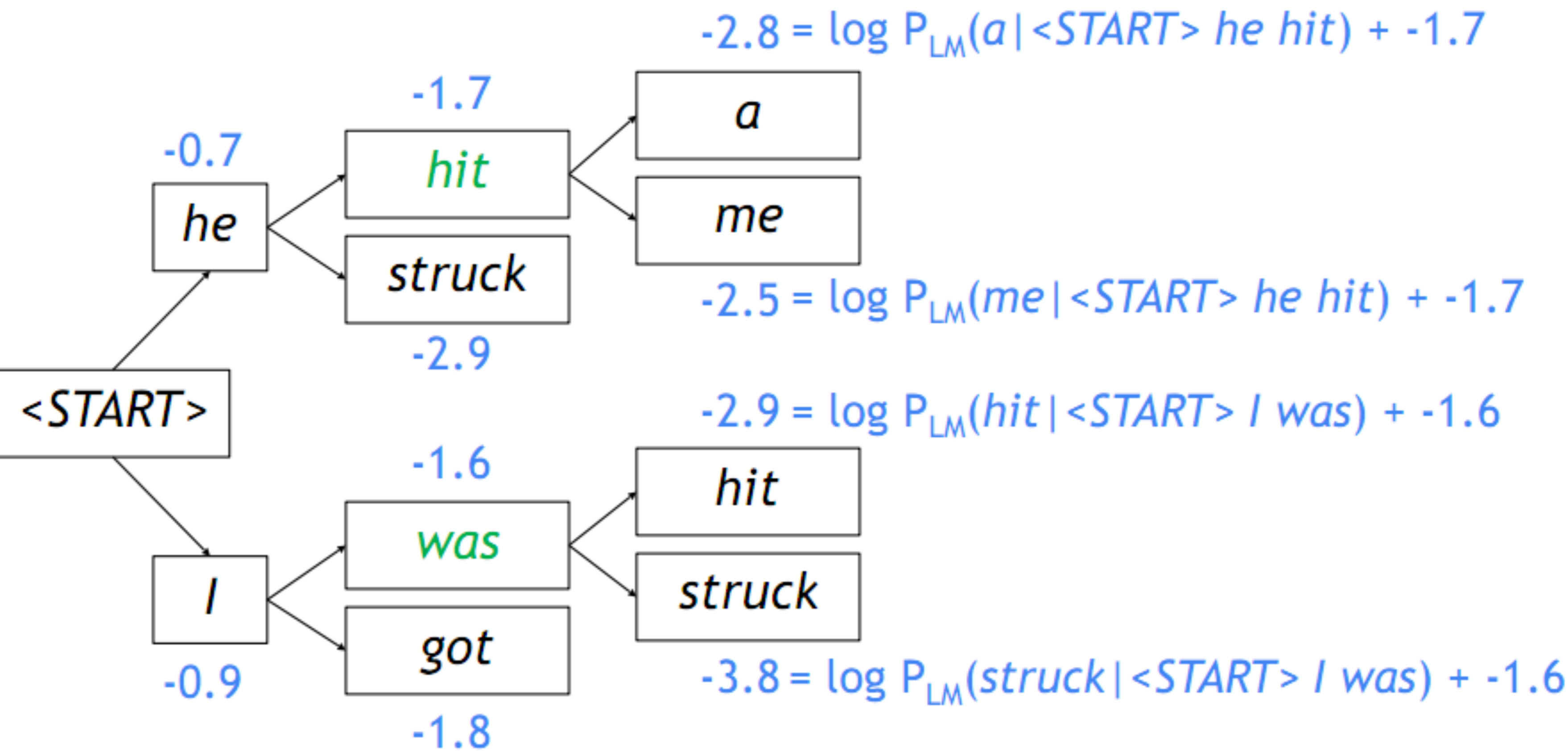
Example: beam search with k = 2

- In k^2 new hypothesis we keep only k highest score hypothesis



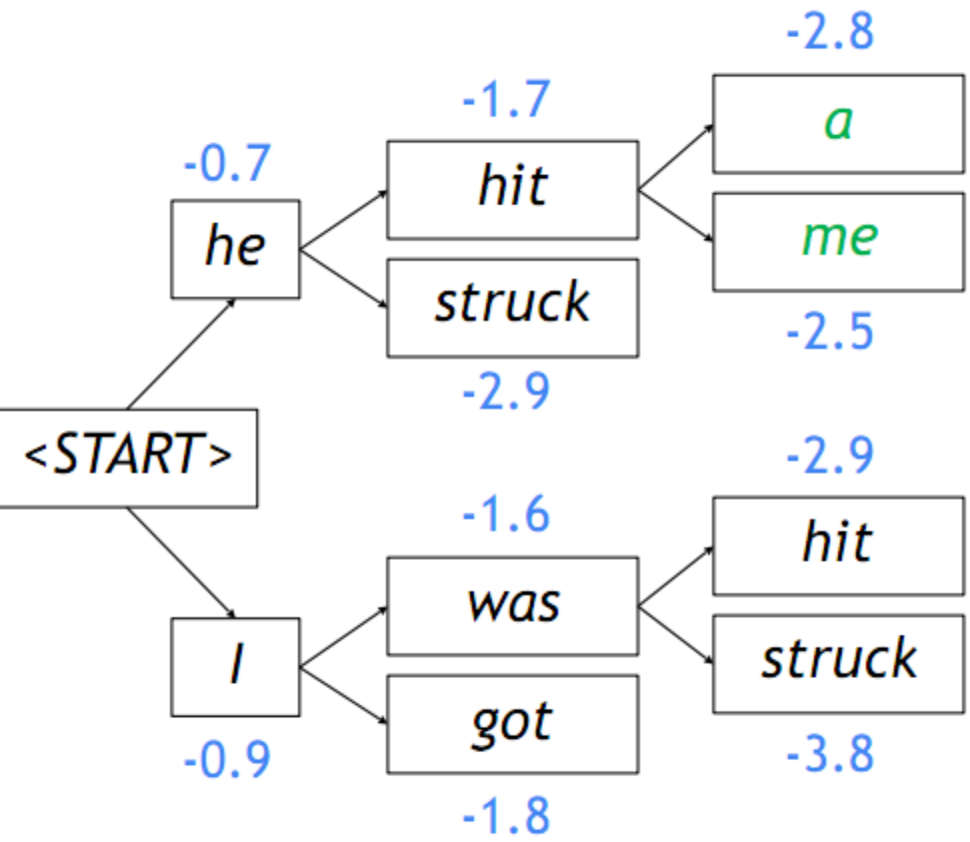
Example: beam search with k = 2

- For each hypothesis, find the next k with the highest scores



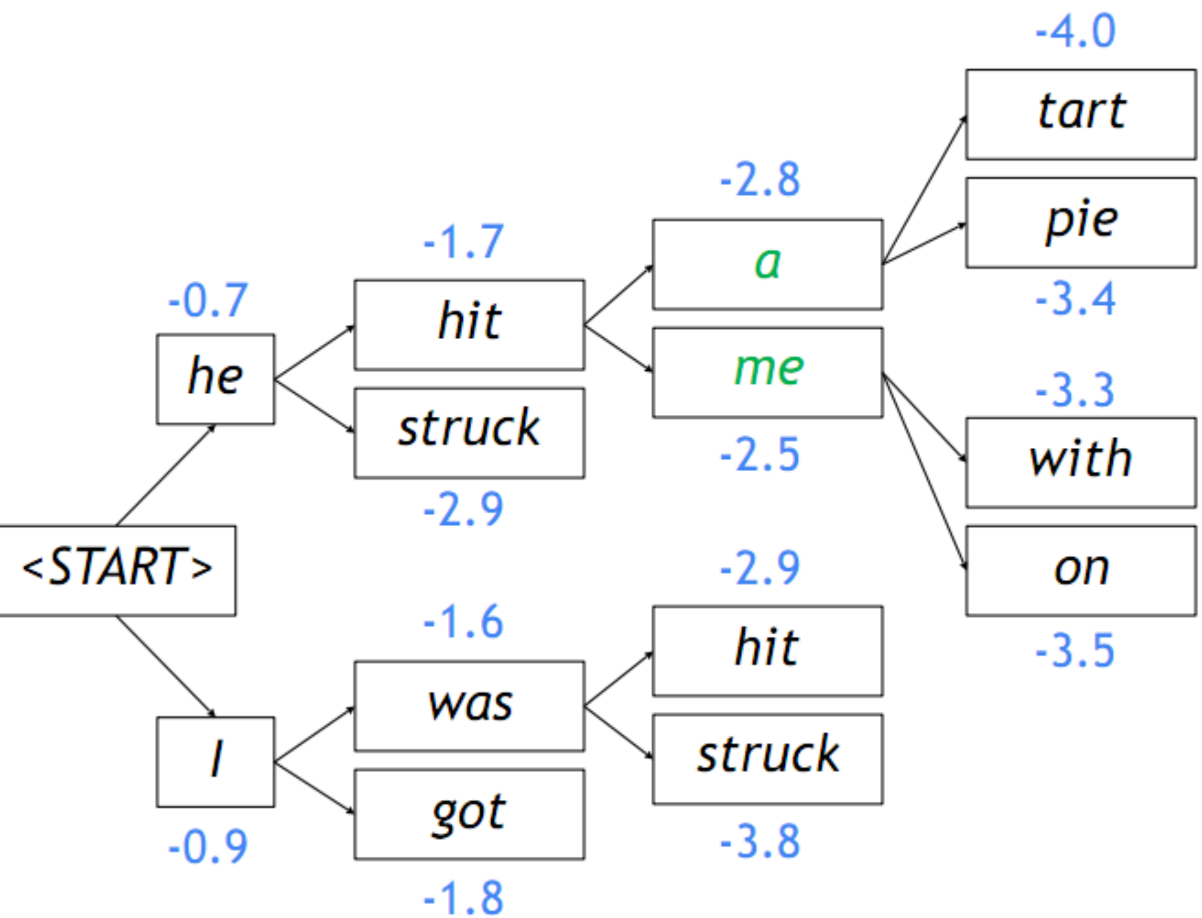
Example: beam search with k = 2

- In k^2 new hypothesis we keep only k highest score hypothesis



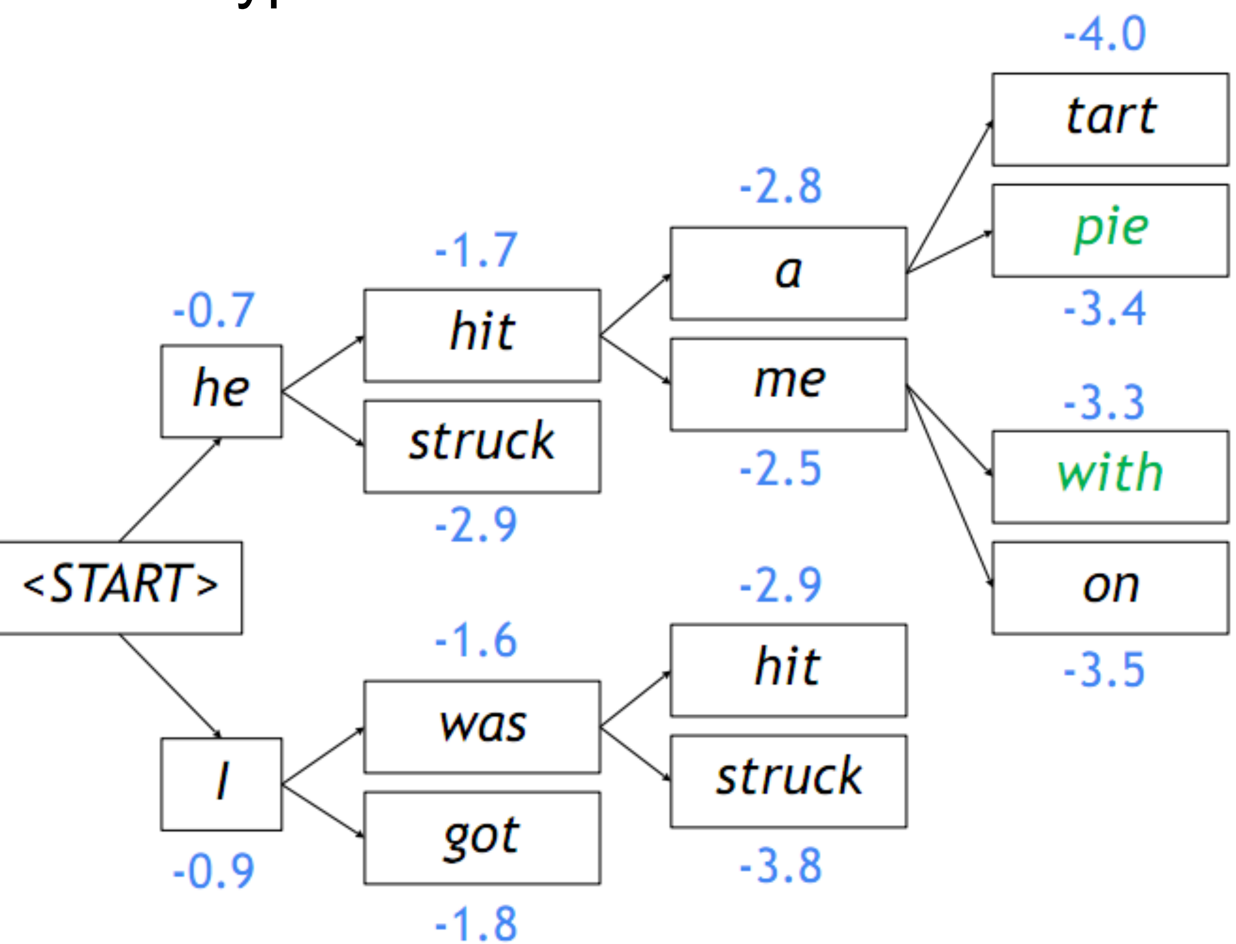
Example: beam search with k = 2

- For each hypothesis, find the next k with the highest scores



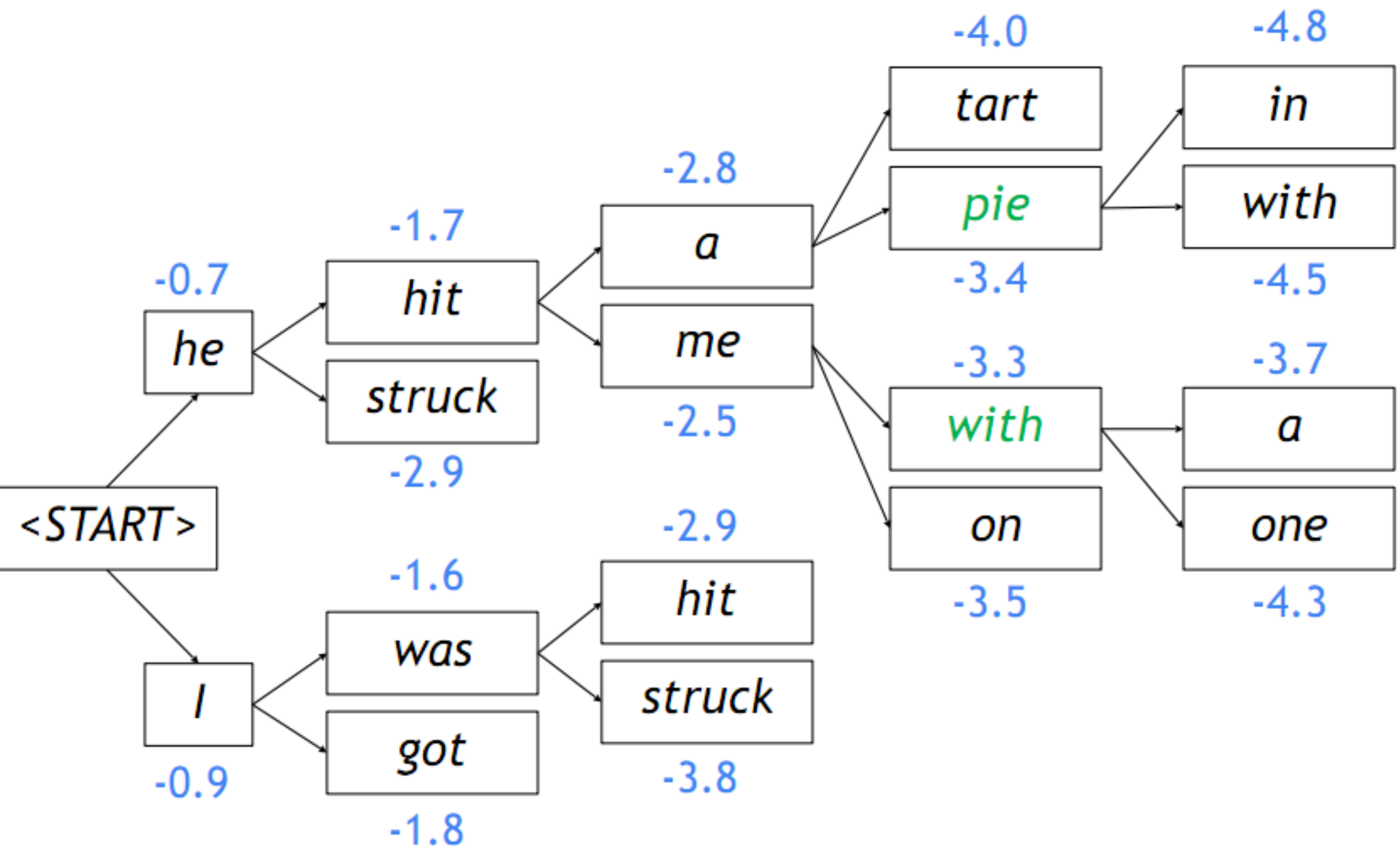
Example: beam search with k = 2

- In k^2 new hypothesis we keep only k highest score hypothesis



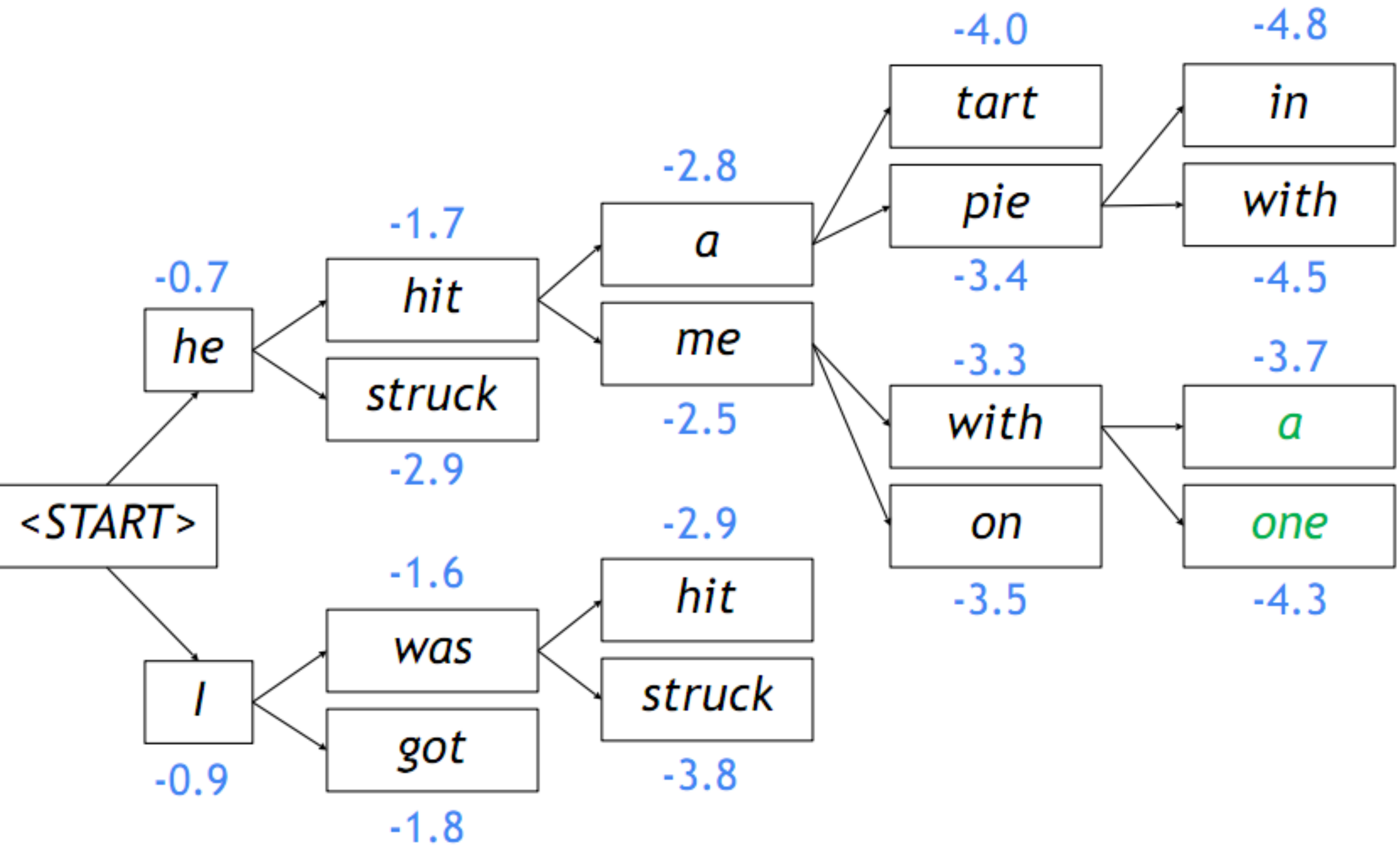
Example: beam search with k = 2

- For each hypothesis, find the next k with the highest scores



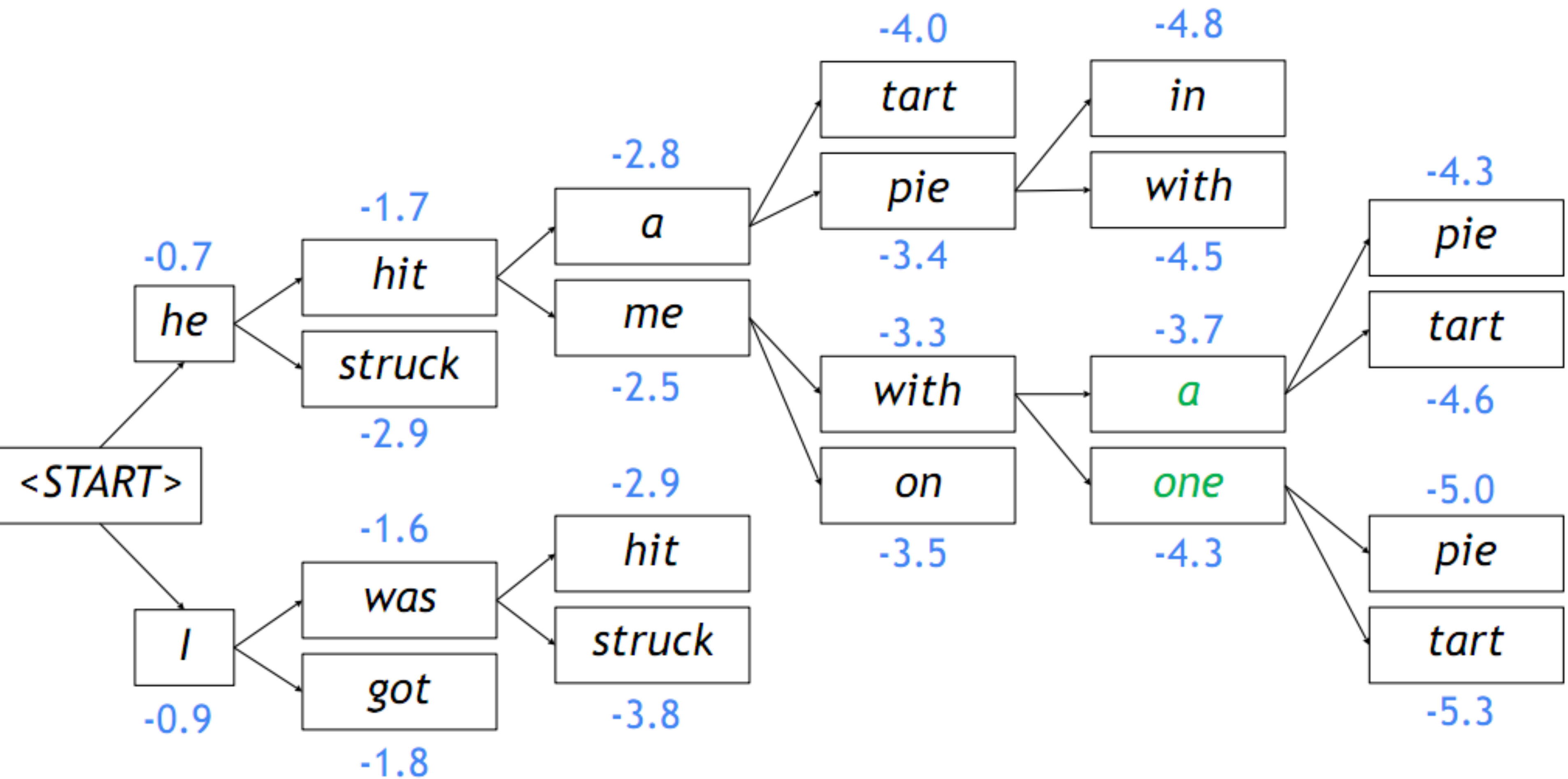
Example: beam search with k = 2

- In k^2 new hypothesis we keep only k highest score hypothesis



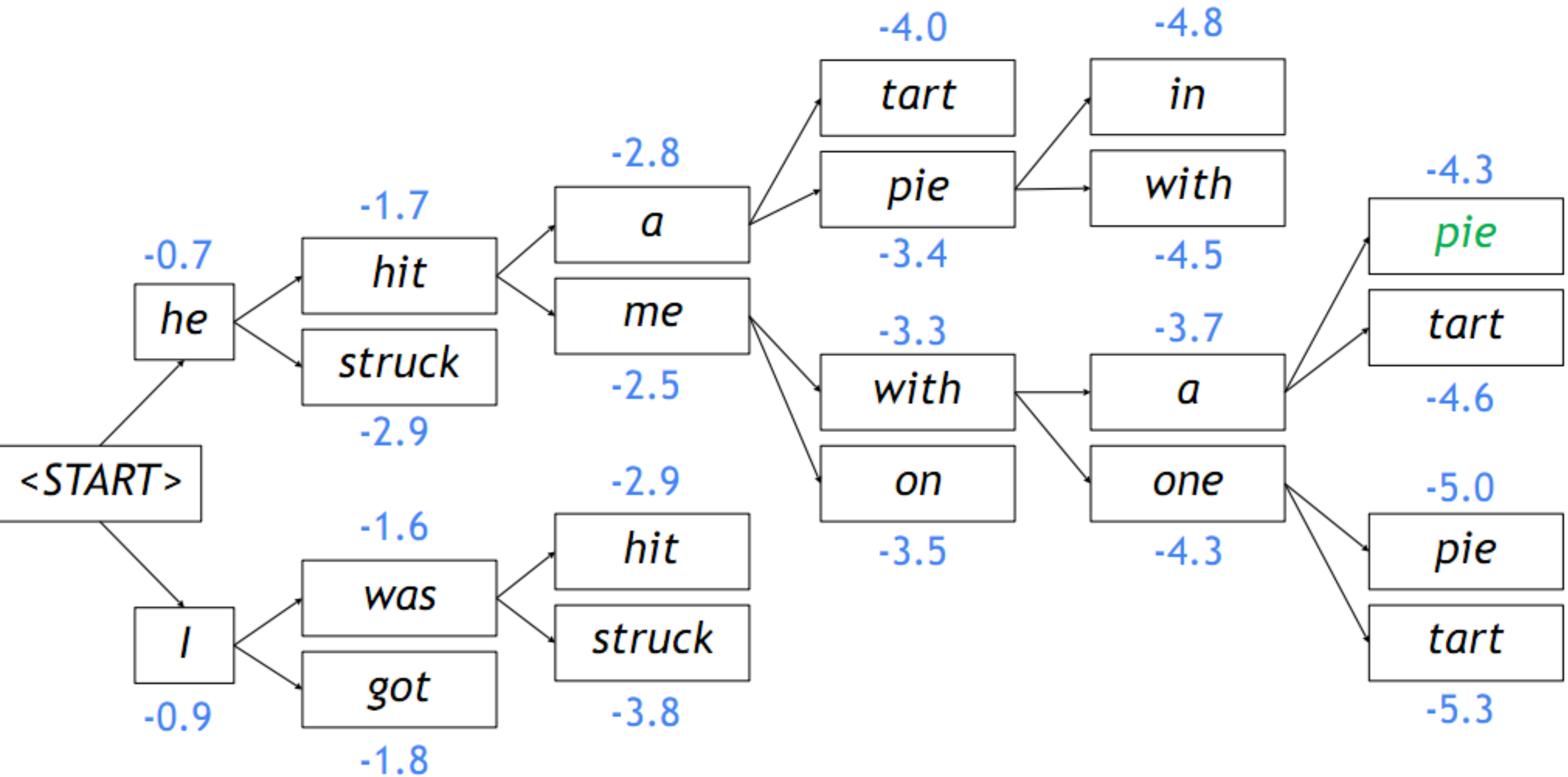
Example: beam search with k = 2

- For each hypothesis, find the next k with the highest scores



Example: beam search with k = 2

- The hypothesis with the highest score is the solution to be found!



Beam search stop condition

- In greedy decoding, usually stops when the model generates token <END>
- Example: <START> he hit me with a pie <END>
- For beam search, different hypotheses can generate <END> tokens at different times
- When a hypothesis generates <END> we call that hypothesis complete and set it aside to continue finding other hypotheses.
- Usually will stop beam search when:
  - Or reach a given step T
  - Or when at least n complete hypotheses have been found

Beam search ending

- After finding a complete set of hypotheses, which one to choose?
- Problem: the longer the hypothesis, the lower the score

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

- Solution: Normalize the score according to the hypothetical length

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

BLEU (Bilingual evaluation understudy) score

- BLEU calculates the similarity between the translated sentence generated by the model and the label sentence, created by the translator
- Measure accuracy of N-grams (N from 1 to 4)
- Penalties for translation sentences that are too short

$$\text{BLEU} = \min\left(1, \frac{\textit{output-length}}{\textit{reference-length}}\right) \left(\prod_{i=1}^{4} \textit{precision}_i\right)^{\frac{1}{4}}$$

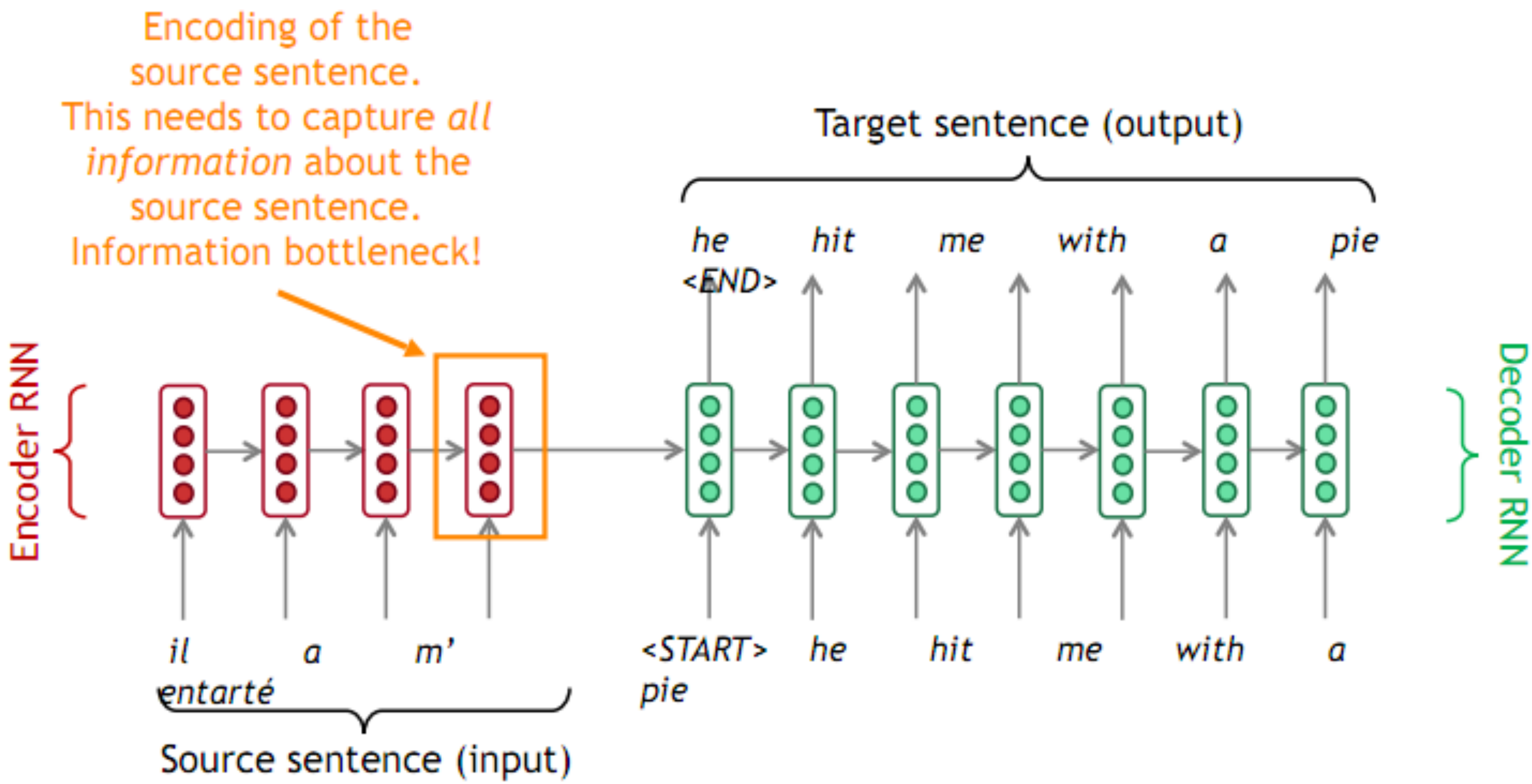
BLEU Score	Interpretation
30 - 40	Understandable to **good** translations
40 - 50	**High** quality translations
50 - 60	Very **high** quality, adequate, and fluent translations
> 60	Quality often **better** than human

NMT vs. SMT

- Advantages of NMT compared to SMT:
  - Better performance: more fluent translation, better context usage…
  - Using only a single network, it is possible to train end-to-end, no need to optimize other independent modules
  - Less human effort required: no manual feature extraction required; the same method is reusable for many different language pairs
- Disadvantages of NMT compared to SMT:
  - NMT is harder to explain, harder to debug
  - NMT is difficult to control. For example, it is not easy to give a rule or translation suggestion to NMT.

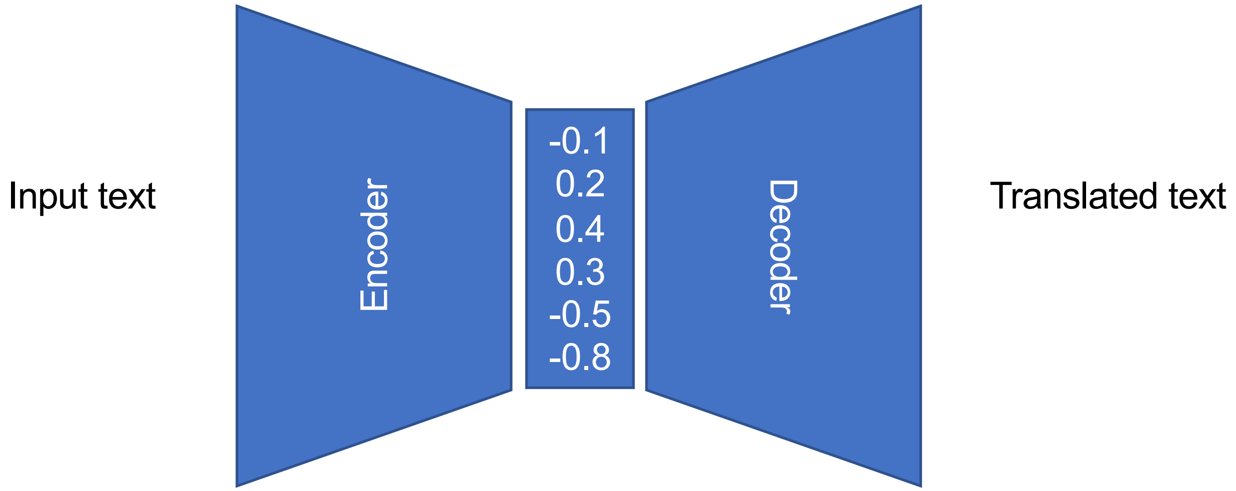
Attention mechanism

Bottleneck in seq2seq model

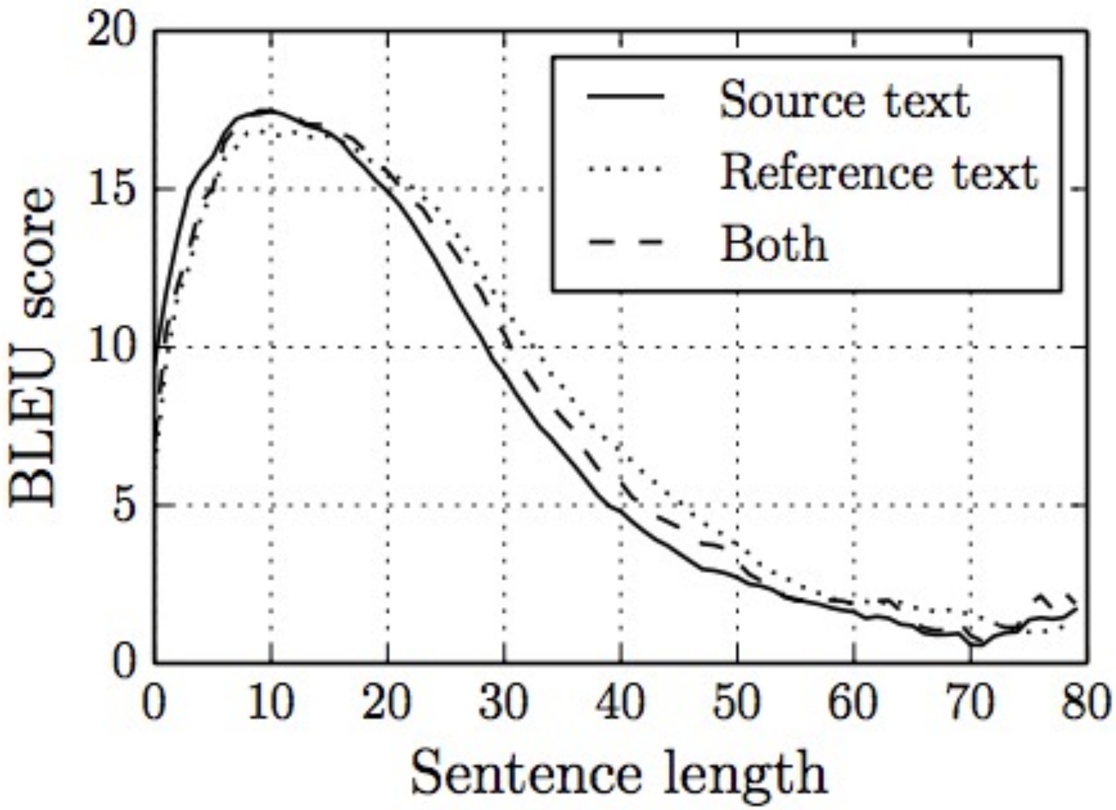


Long sentence translation

- Machine learning **has turned out to be** a very useful tool for translation, but it has a few weak spots. The tendency of translation models to do their work word by word is one of those, and can lead to serious errors.
- L'apprentissage automatique **s'est révélé être** un outil très utile pour la traduction, mais il comporte quelques points faibles. La tendance des modèles de traduction à faire leur travail mot à mot en fait partie et peut entraîner de graves erreurs.



Model performance w.r.t. sentence length

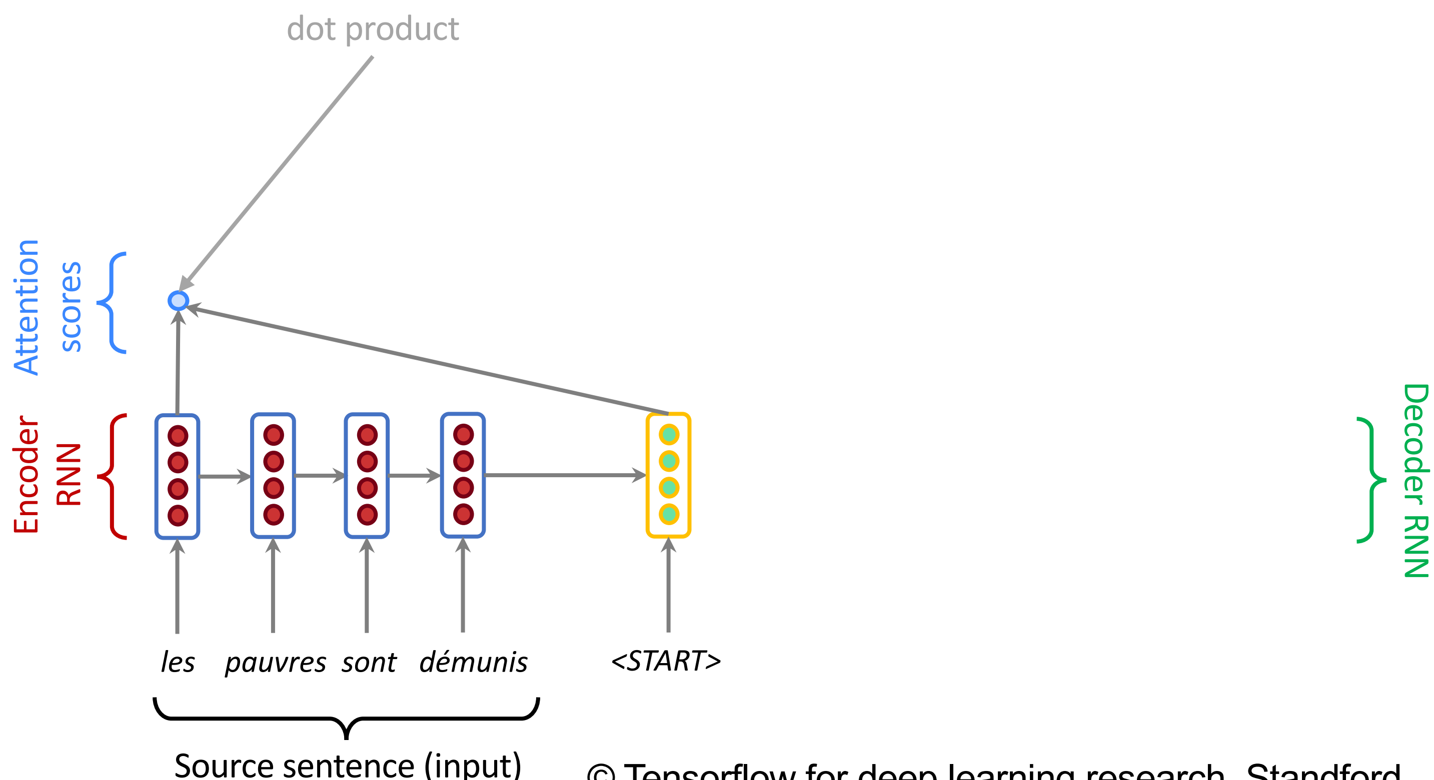


© On the Properties of Neural Machine Translation: Encoder-Decoder Approaches

Attention

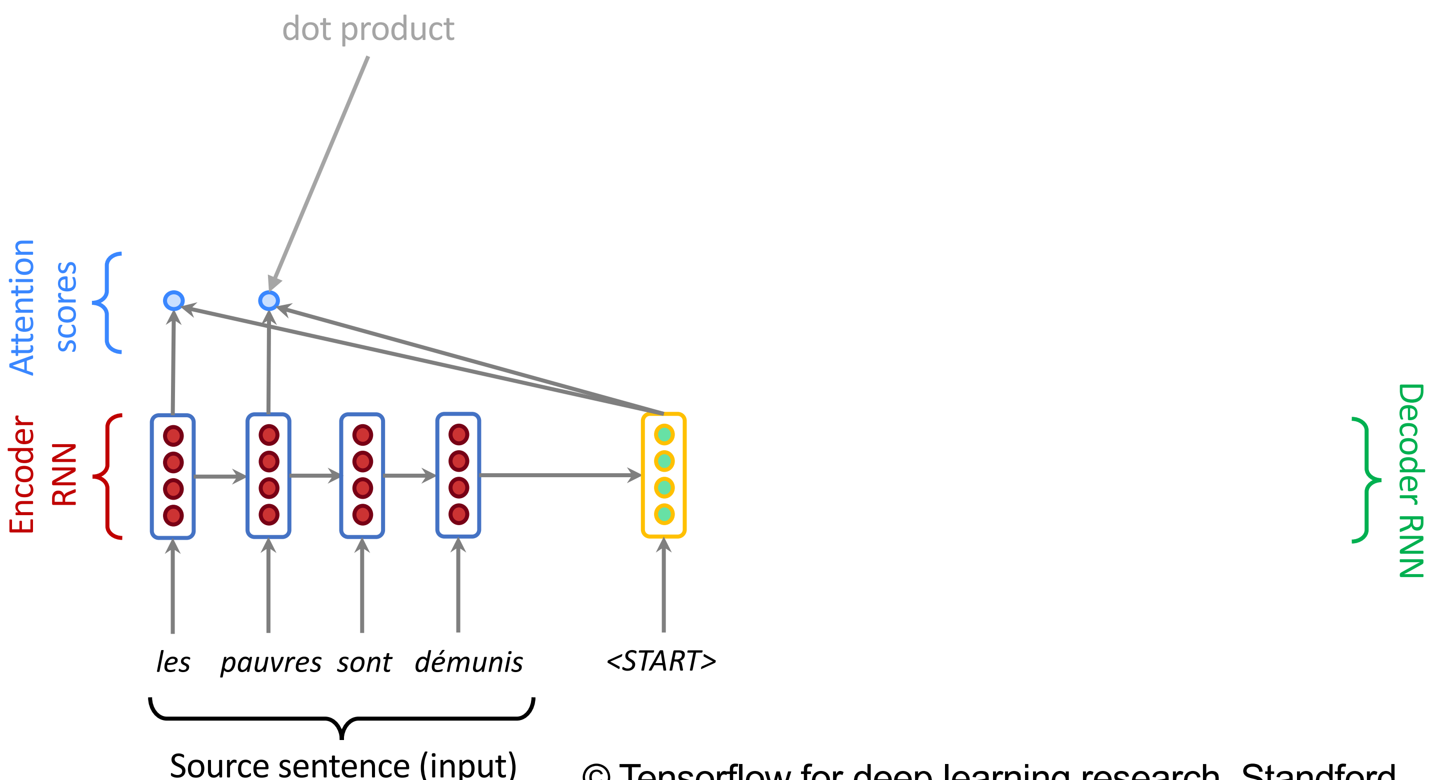
- Attention solves the bottleneck problem of seq2seq
- The idea
  - At each decoding step, **use a direct connection to the encoder network part** for computation and from there focus (pay attention) only on a specific part of the source sentence, ignoring irrelevant parts.
- One of the most influential ideas in deep learning for NLP
  - Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

Sequence-to-sequence with attention



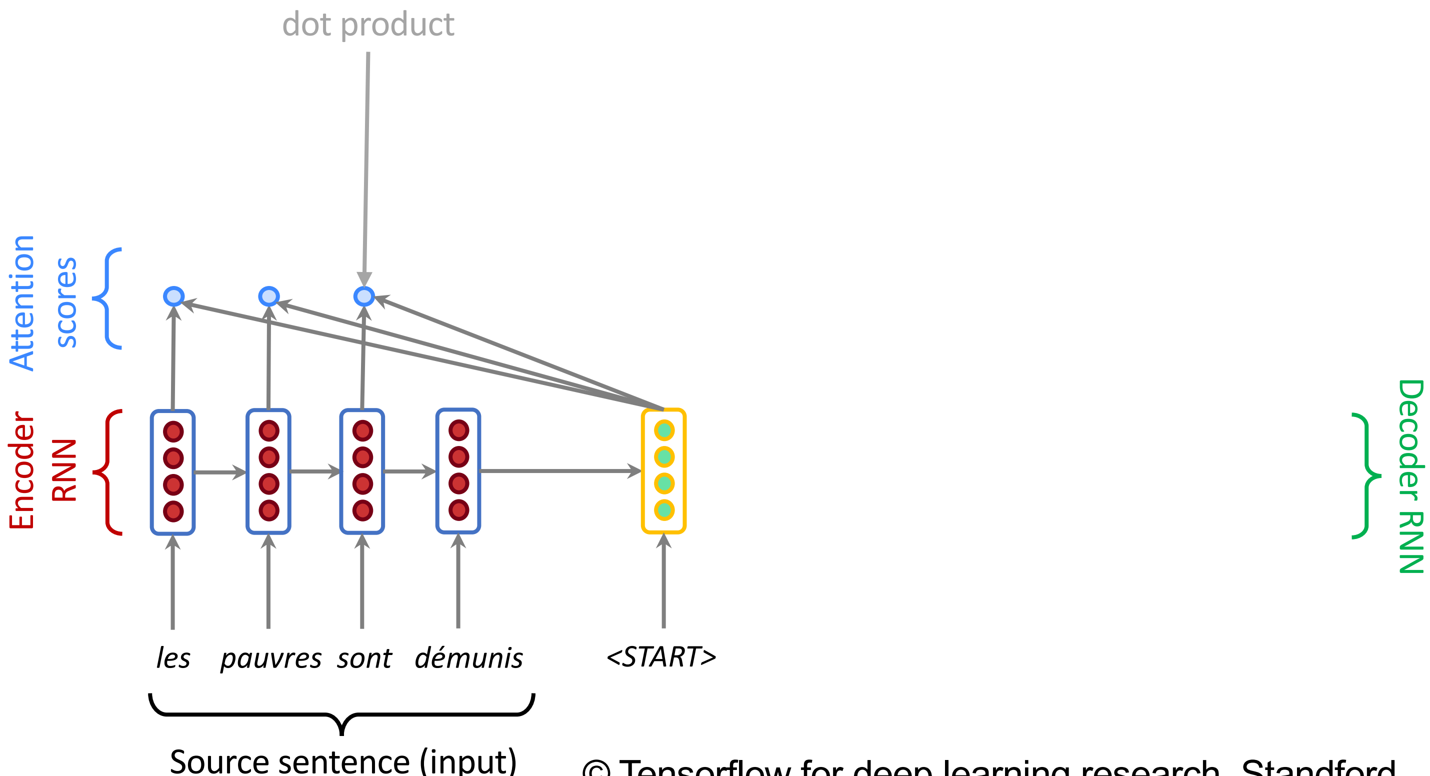
© Tensorflow for deep learning research. Standford

Sequence-to-sequence with attention



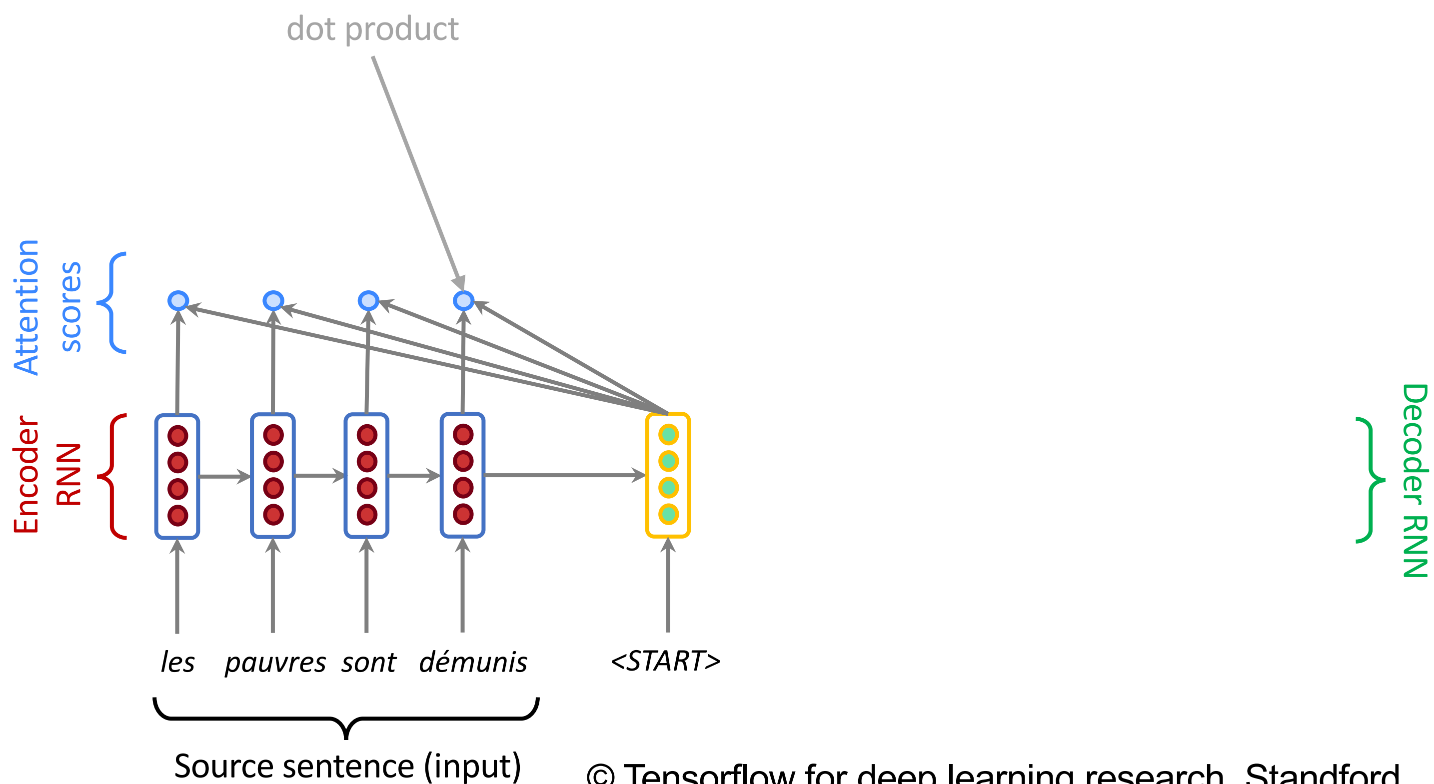
© Tensorflow for deep learning research. Standford

Sequence-to-sequence with attention

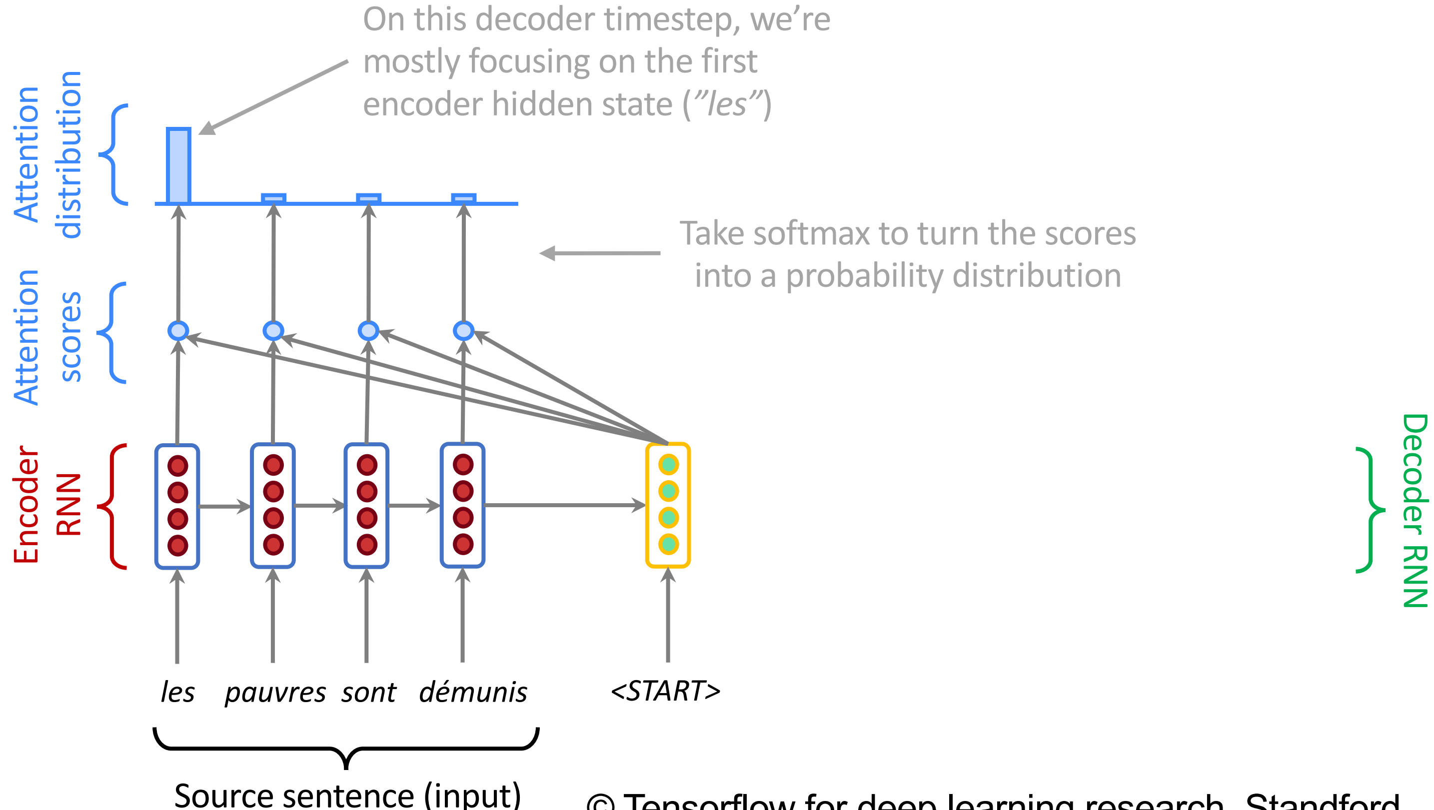


© Tensorflow for deep learning research. Standford

Sequence-to-sequence with attention

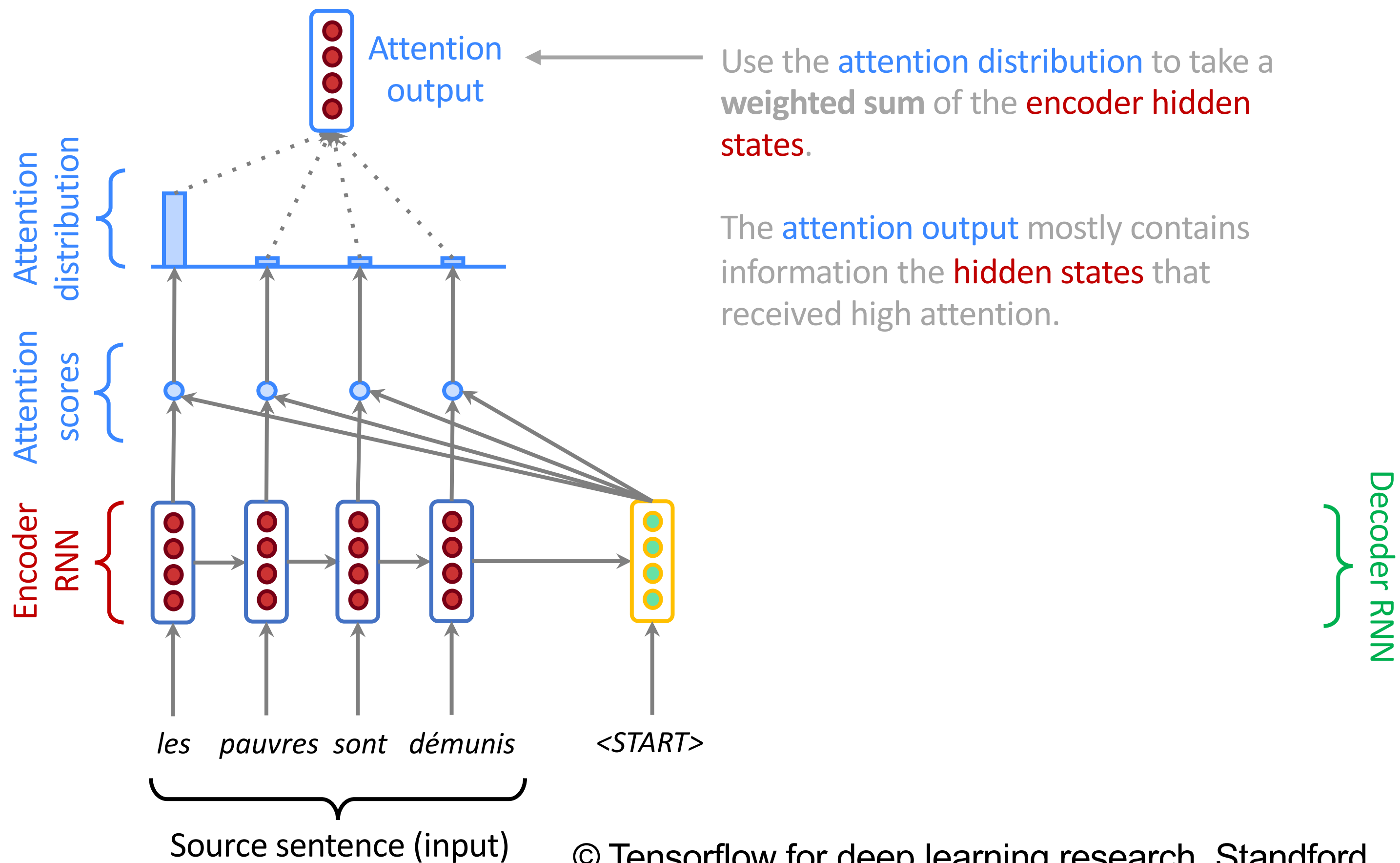


Sequence-to-sequence with attention



© Tensorflow for deep learning research. Standford

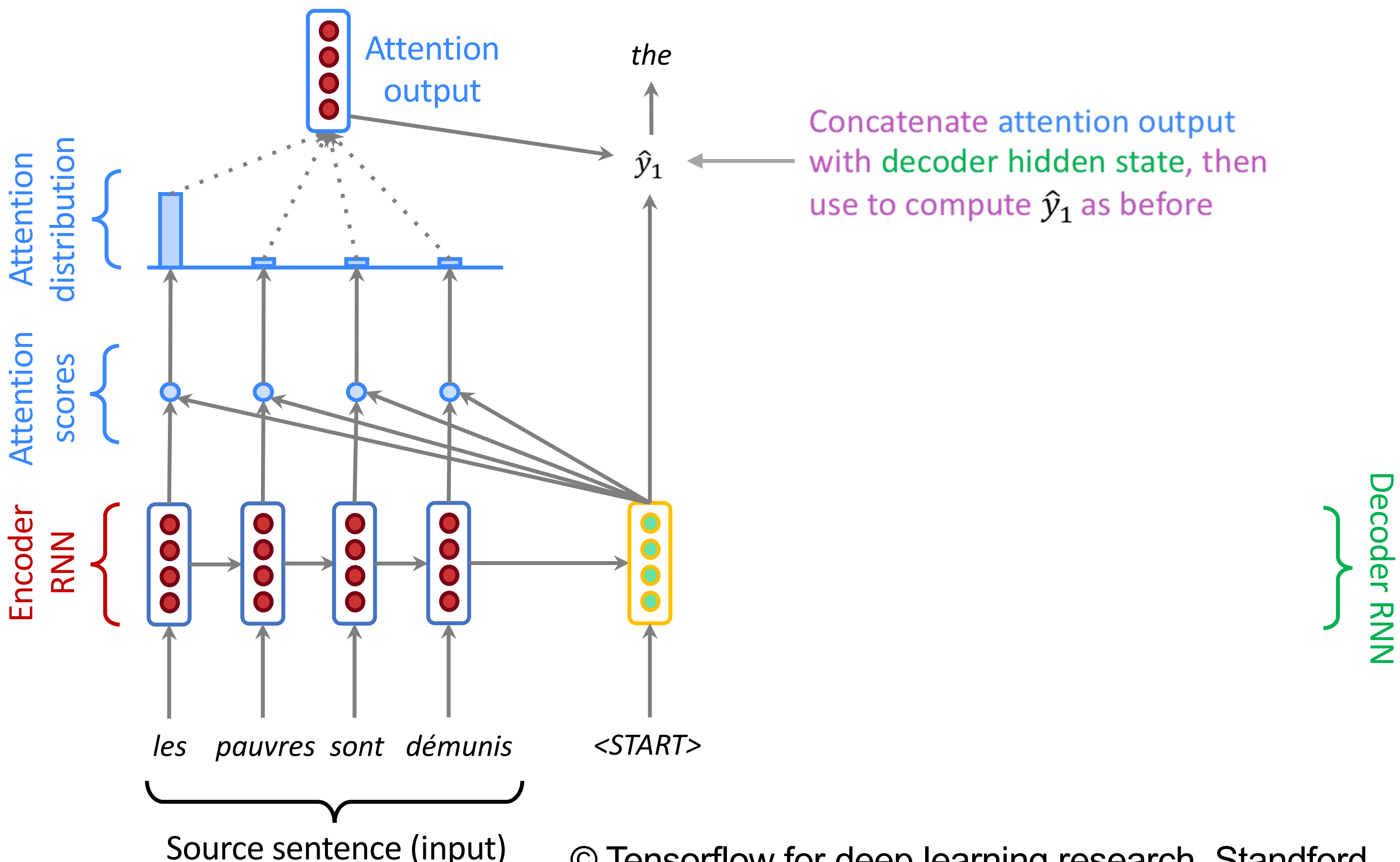
Sequence-to-sequence with attention



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

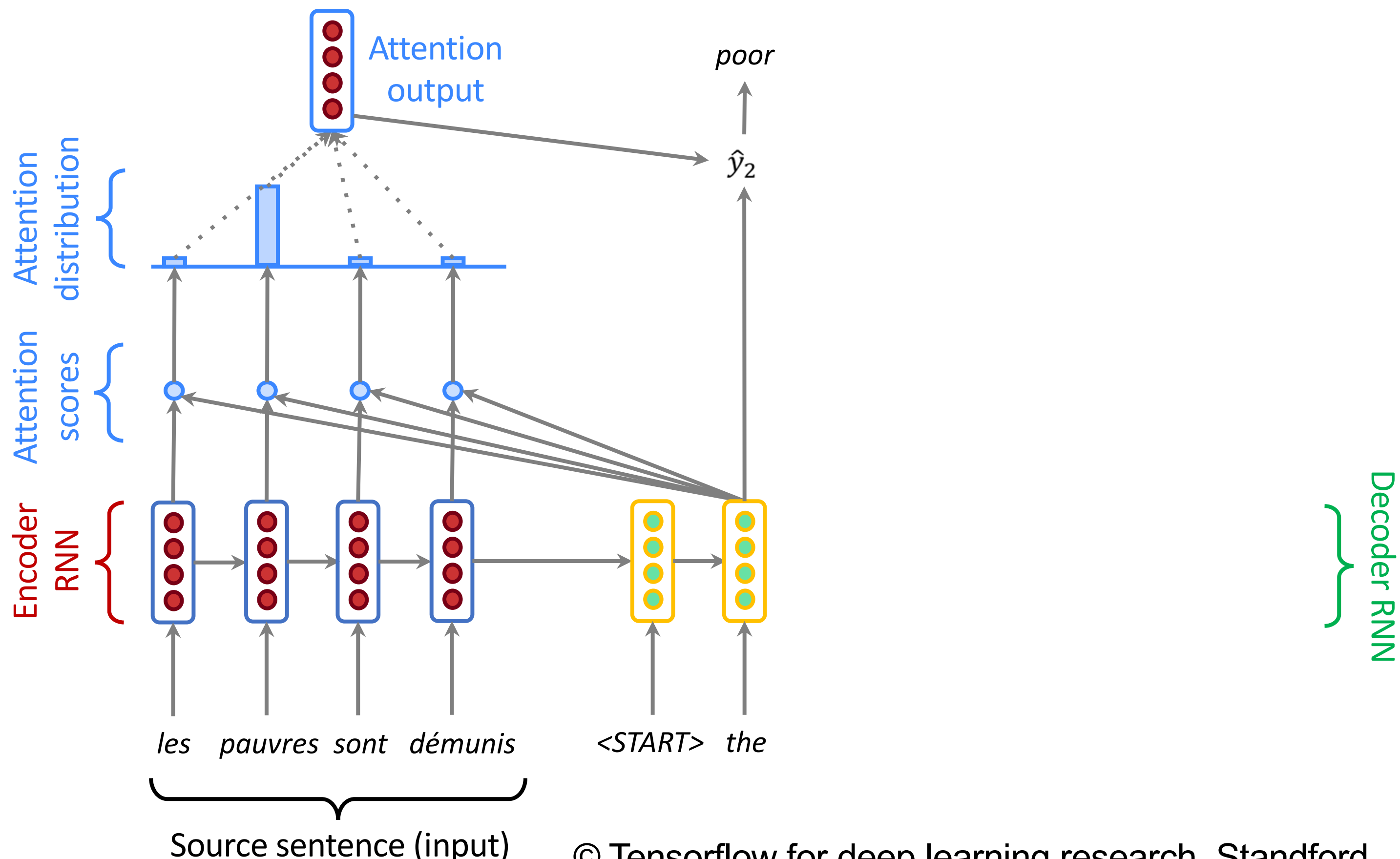
The attention output mostly contains information the hidden states that received high attention.

Sequence-to-sequence with attention



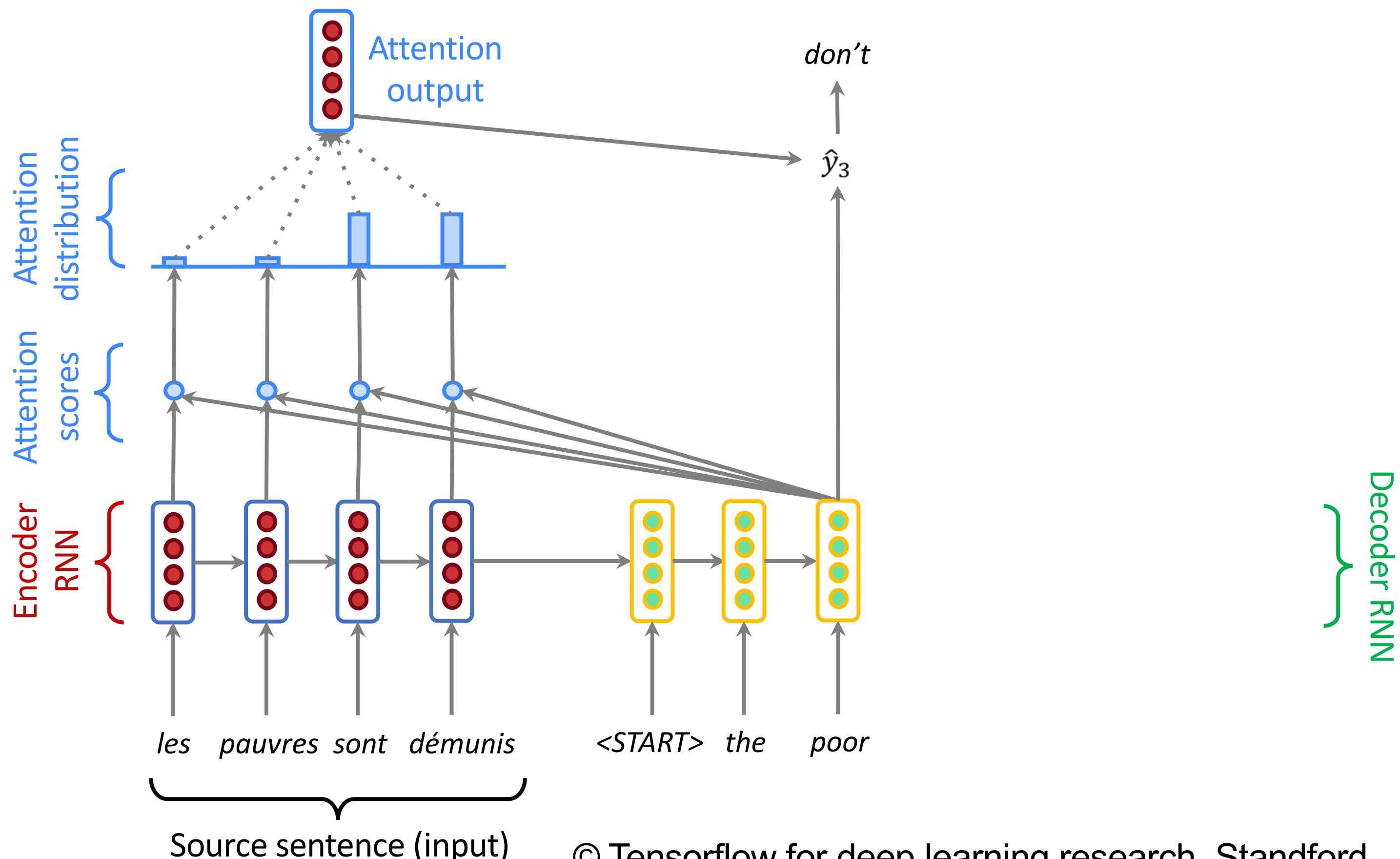
© Tensorflow for deep learning research. Standford

Sequence-to-sequence with attention



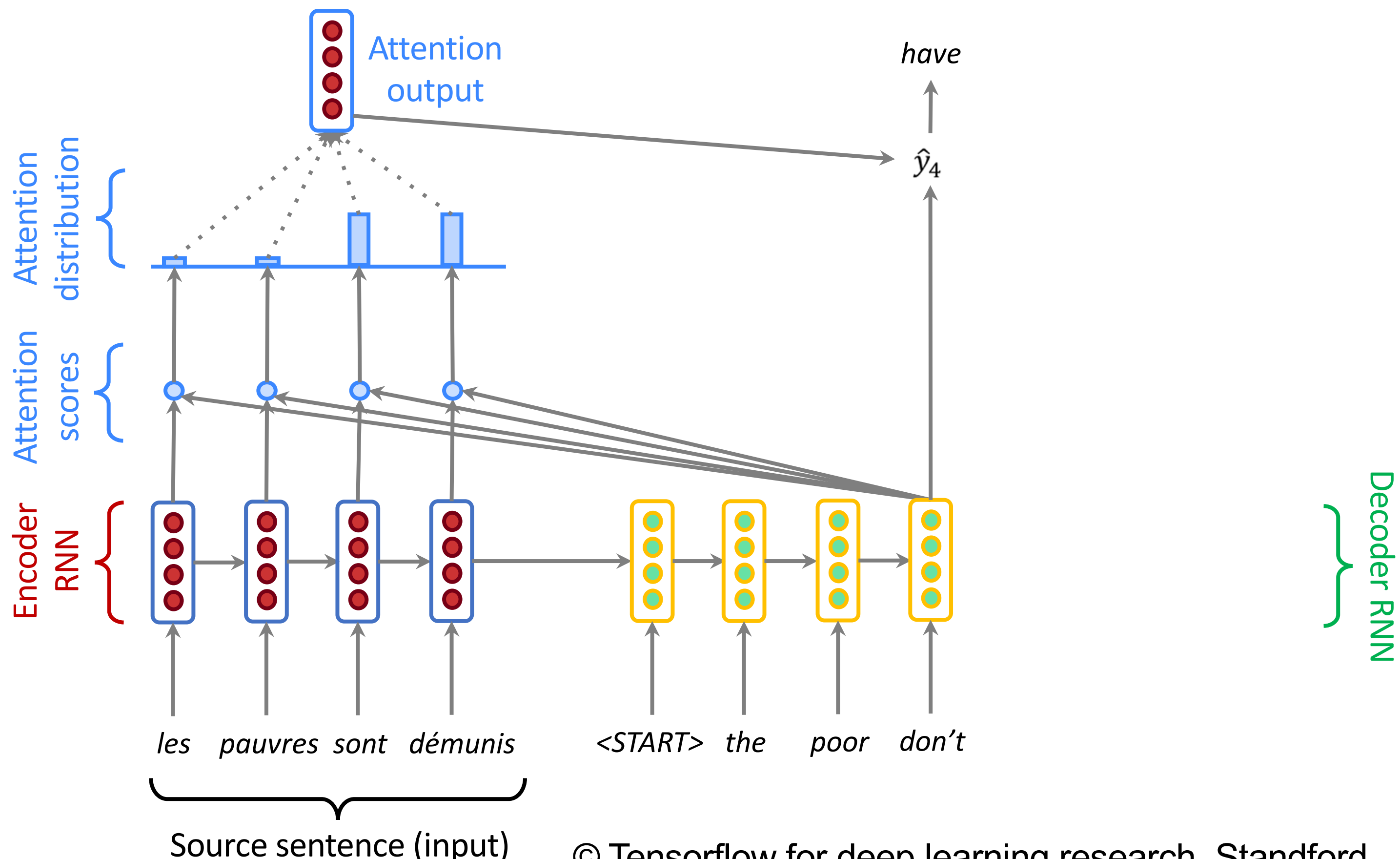
© Tensorflow for deep learning research. Standford

Sequence-to-sequence with attention

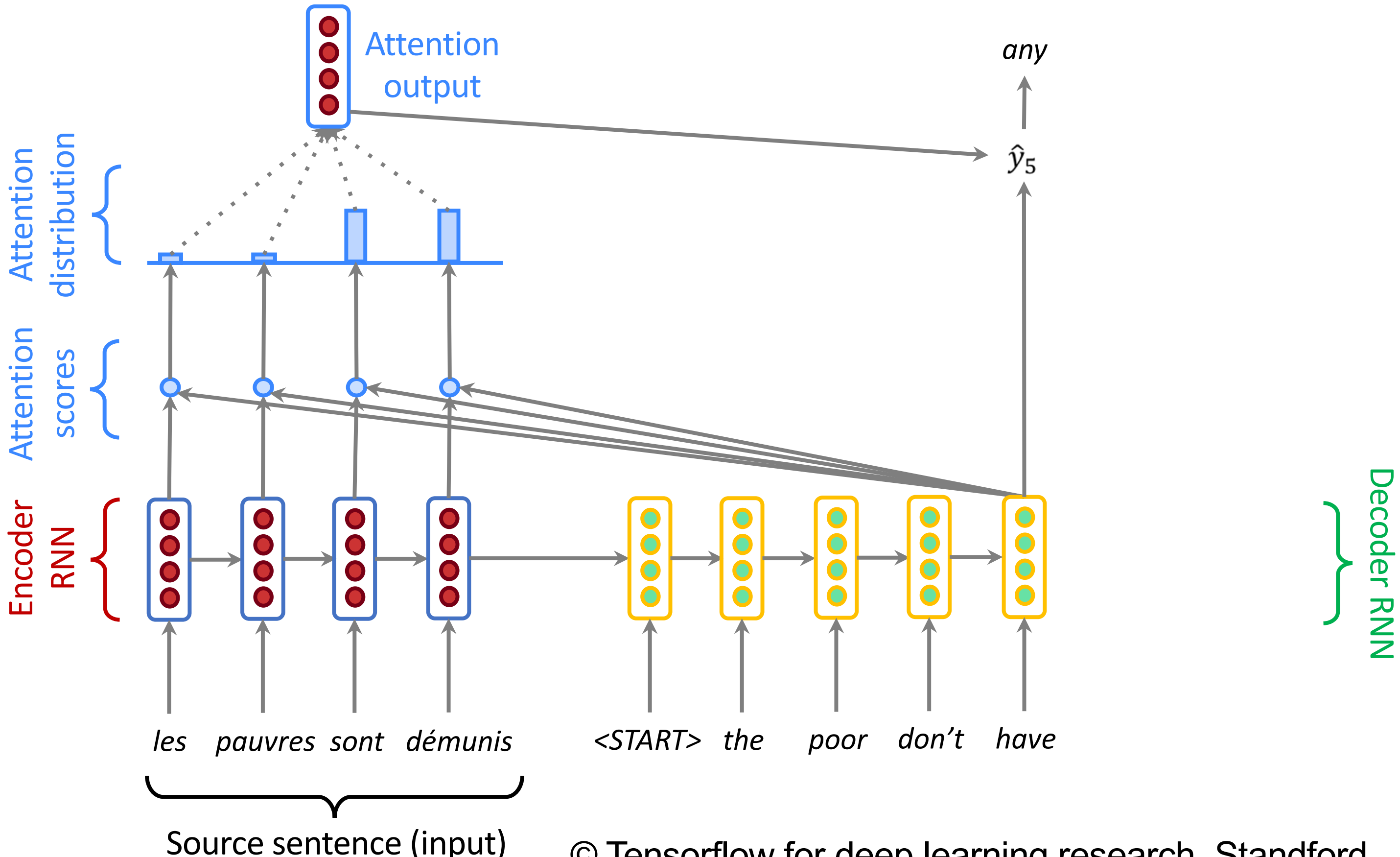


© Tensorflow for deep learning research. Standford

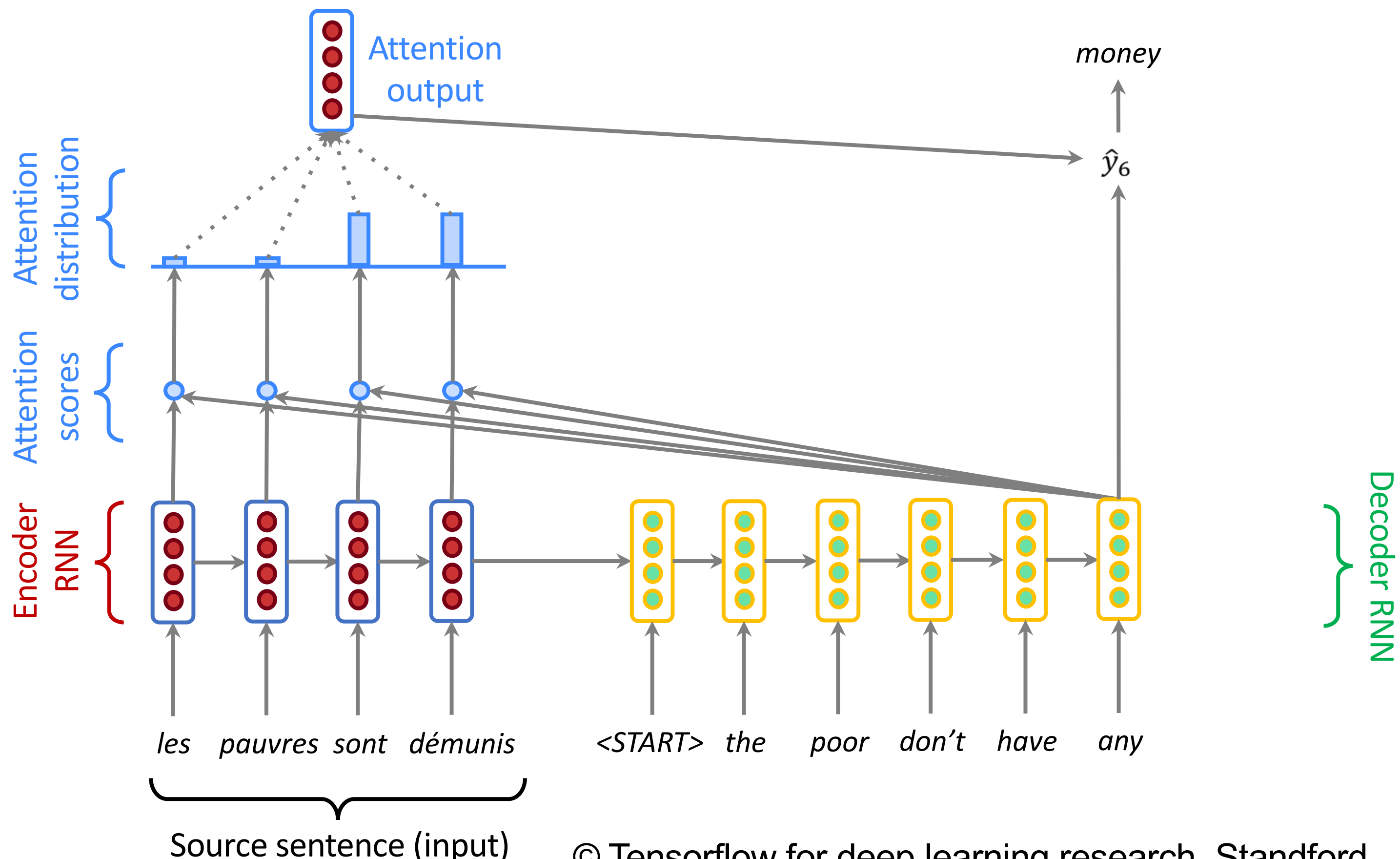
Sequence-to-sequence with attention



Sequence-to-sequence with attention



Sequence-to-sequence with attention



© Tensorflow for deep learning research. Standford

Formulations

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores $\boldsymbol{e}^t$ for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

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

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

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output

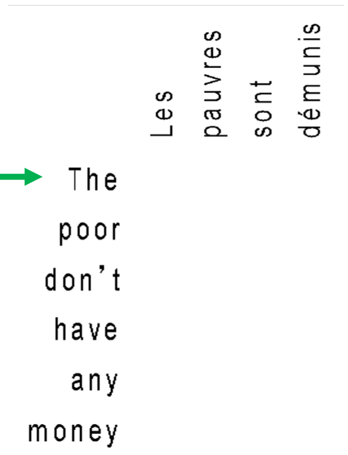
$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

Advantages of attention mechanism

- Attention significantly improves NMT performance
    - It’s very useful to allow decoder to focus on certain parts of the source
- Attention solves the bottleneck problem
    - Attention allows decoder to look directly at source; bypass bottleneck
- Attention helps with vanishing gradient problem
    - Provides shortcut to faraway states
- Attention provides some interpretability
    - By inspecting attention distribution, we can see what the decoder was focusing on
    - We get alignment for free!
    - This is cool because we never explicitly trained an alignment system
    - The network just learned alignment by itself



© Tensorflow for deep learning research. Standford

References

1. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/
2. http://cs231n.stanford.edu/slides/2020/lecture_10.pdf