



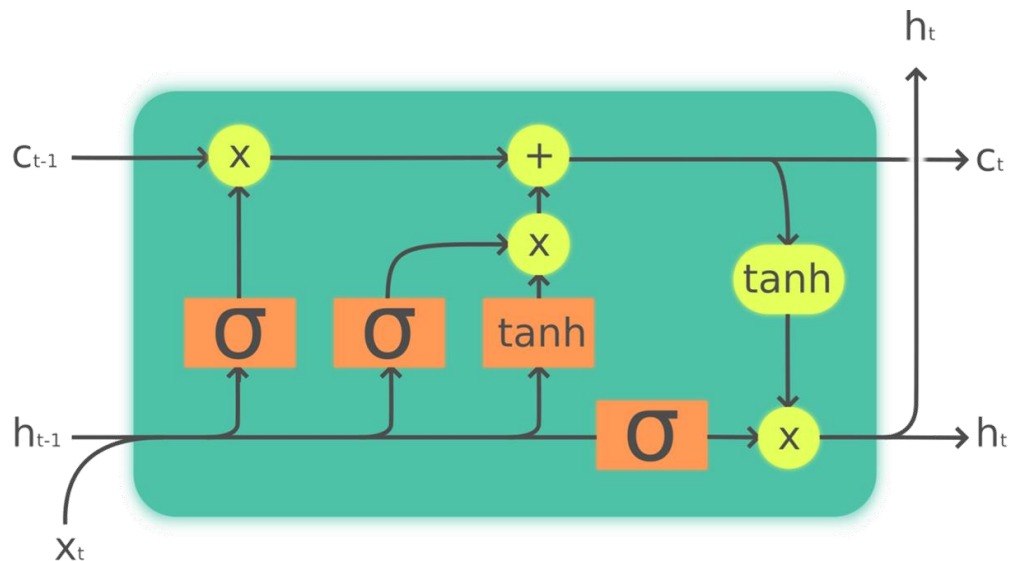
VietAI

# Sequence-to-Sequence model



VietAI teaching team

# 1 Ôn tập



Legend:

Layer



Pointwise op



Copy



# 1 Công thức LSTM

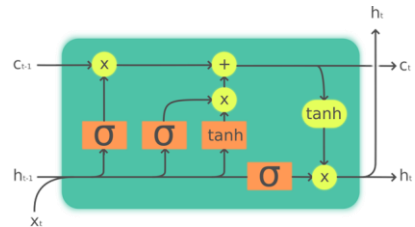
$$\begin{aligned}
 f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
 i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
 o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
 c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
 h_t &= o_t \circ \sigma_h(c_t)
 \end{aligned}$$

- $x_t \in \mathbb{R}^d$ : Vector đầu vào
- $f_t \in \mathbb{R}^h$ : Vector cổng forget
- $i_t \in \mathbb{R}^h$ : Vector cổng input
- $o_t \in \mathbb{R}^h$ : Vector cổng output
- $h_t \in \mathbb{R}^h$ : Vector đầu ra của mỗi cell
- $c_t \in \mathbb{R}^h$ : Vector trạng thái mỗi state

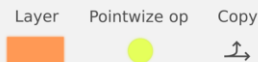
$t$ : Giá trị đang xét tại thời điểm  $t$

Khởi tạo:  $c_0 = h_0 = [0, \dots, 0]$

- $\sigma_g$ : sigmoid function.
- $\sigma_c$ : hyperbolic tangent



Legend:

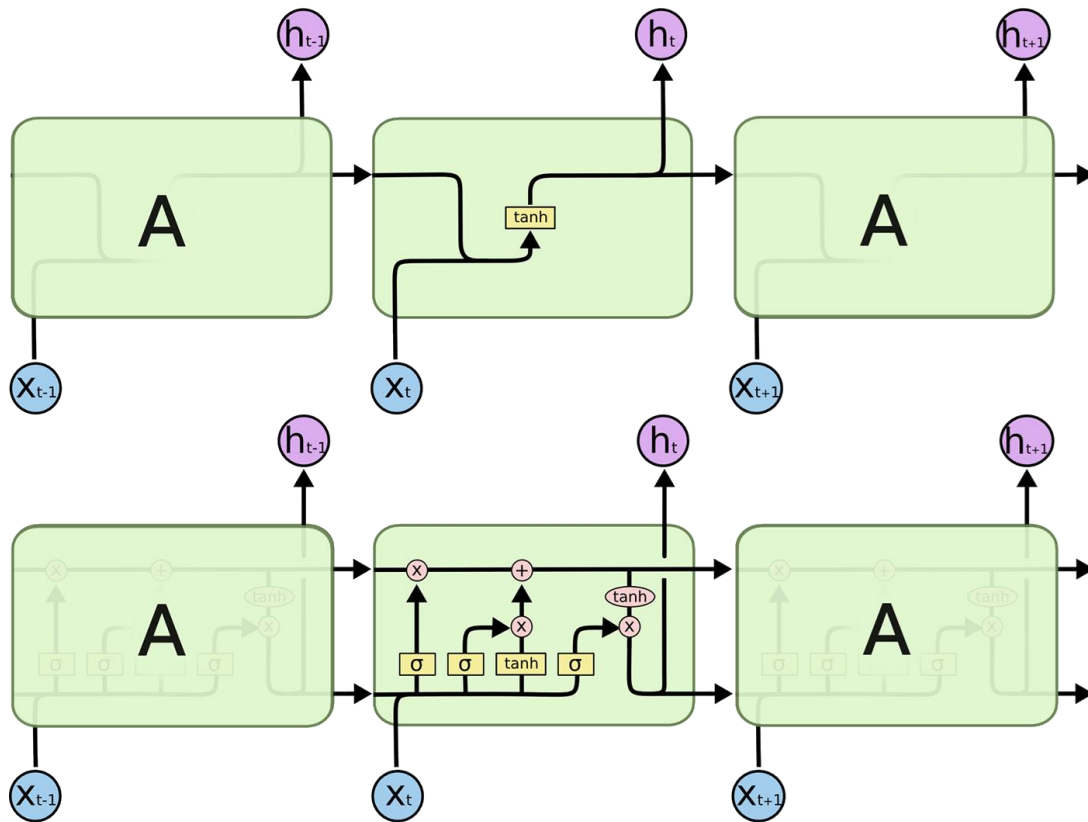


- $W \in \mathbb{R}^{h \times d}$ ,  $U \in \mathbb{R}^{h \times h}$  and  $b \in \mathbb{R}^h$ :

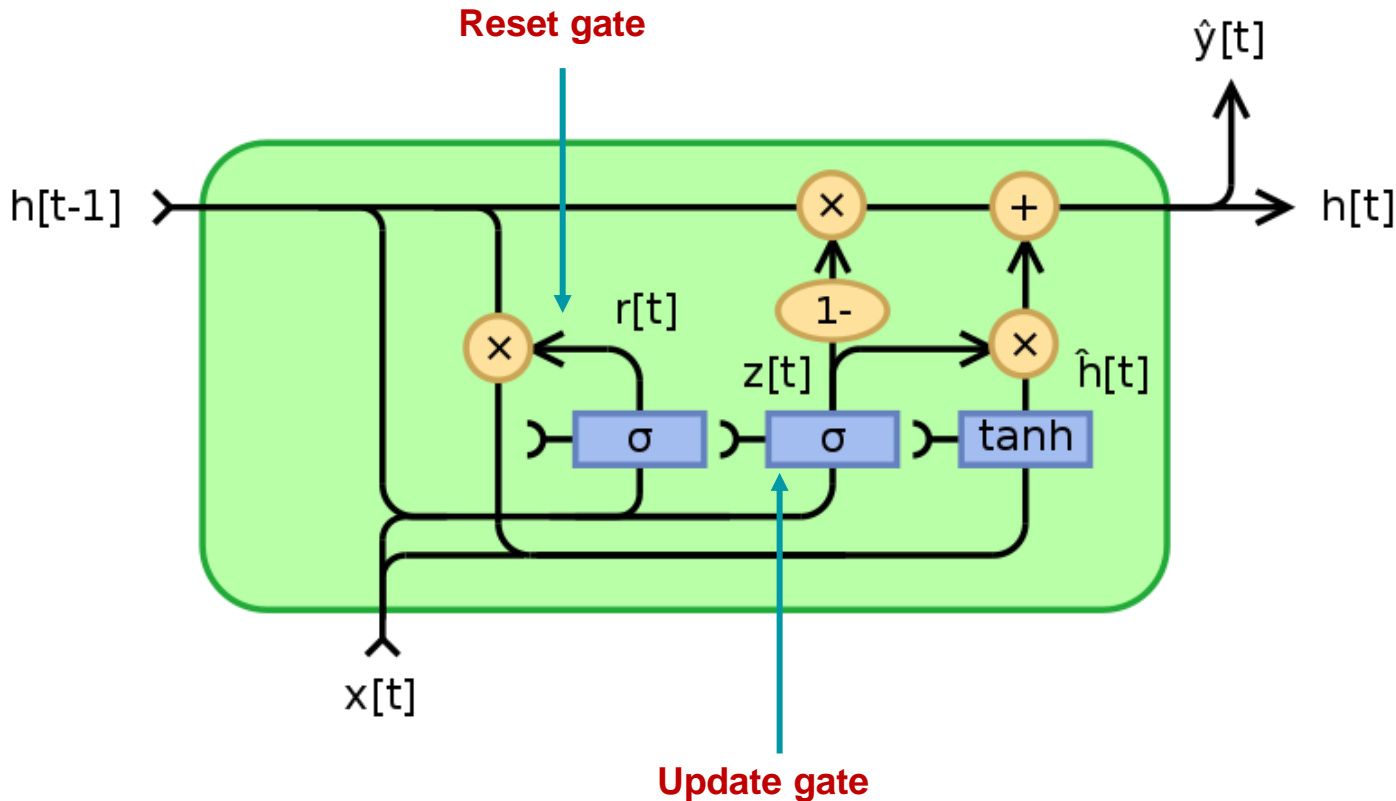
Ma trận weight và bias cần học

$d$  và  $h$  là số chiều vector đầu vào và đầu ra

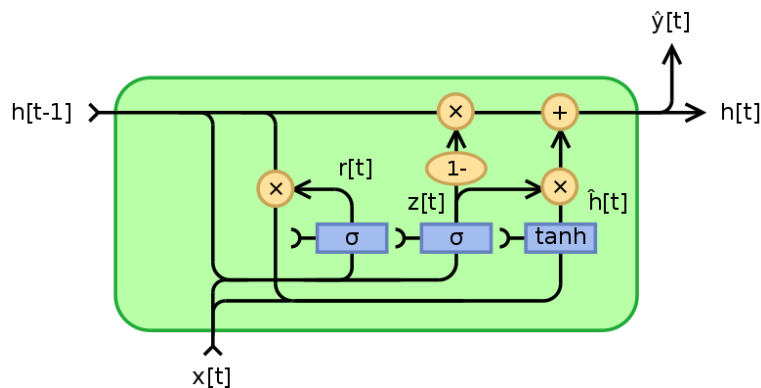
# 1 So sánh RNN và LSTM



## 2 Gated Recurrent Unit



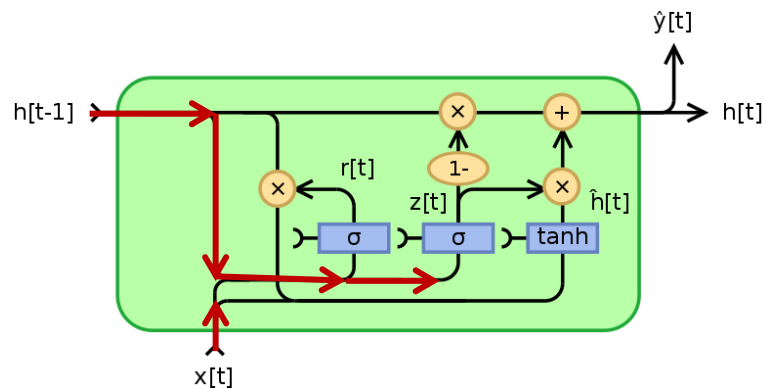
## 2 Công thức GRU



$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z)$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r)$$

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \sigma_h(W_h x_t + U_h (r_t \circ h_{t-1}) + b_h)$$

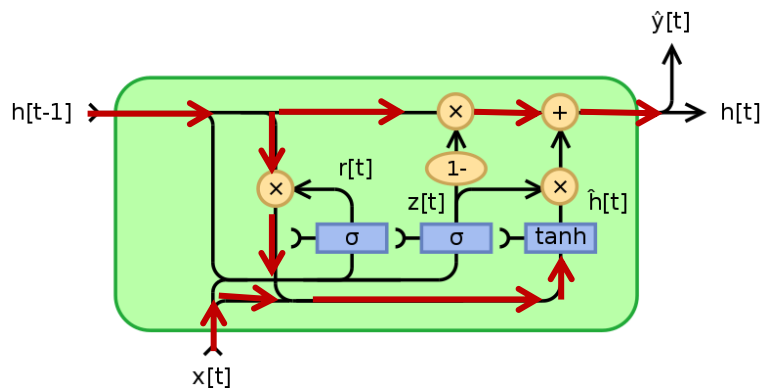


$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z)$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r)$$

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \sigma_h(W_h x_t + U_h (r_t \circ h_{t-1}) + b_h)$$

## 2 Công thức GRU



$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z)$$

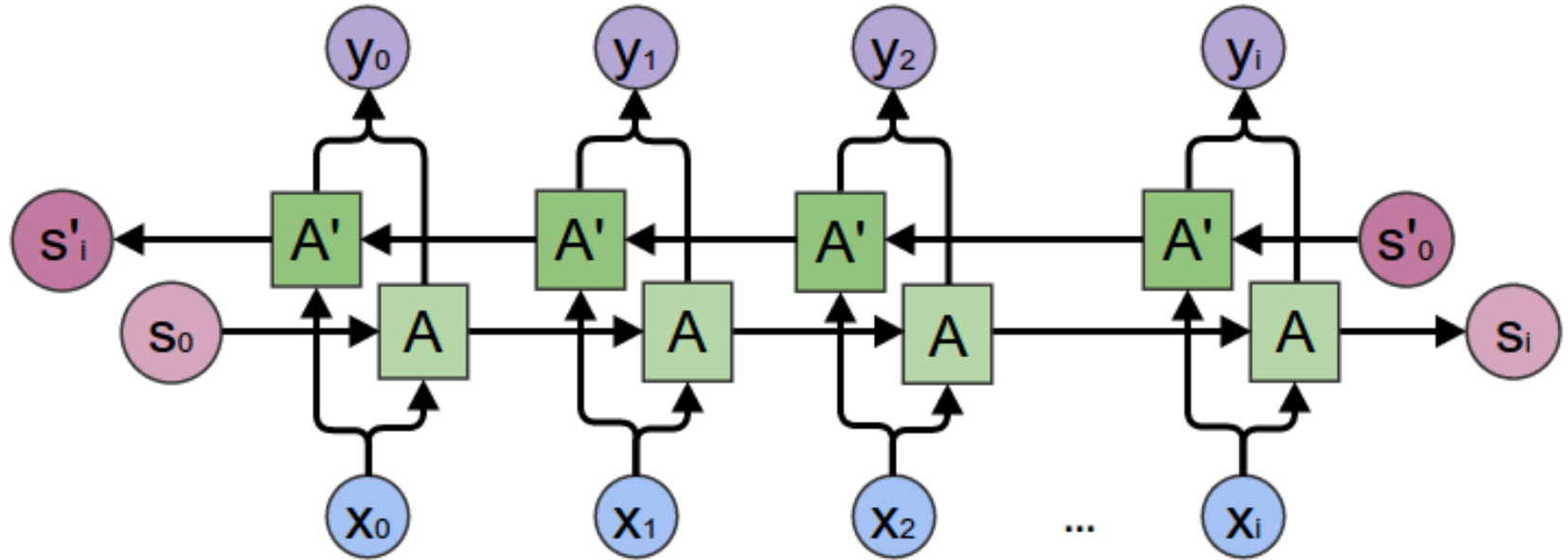
$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r)$$

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \sigma_h(W_h x_t + U_h(r_t \circ h_{t-1}) + b_h)$$

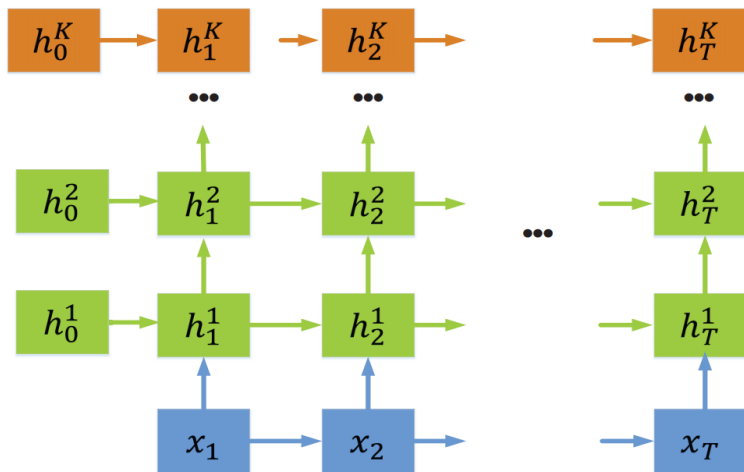
Thông tin update mới:  $\tilde{h}_t$



### 3 Bidirectional RNN



## 4 Deep-stacked RNN

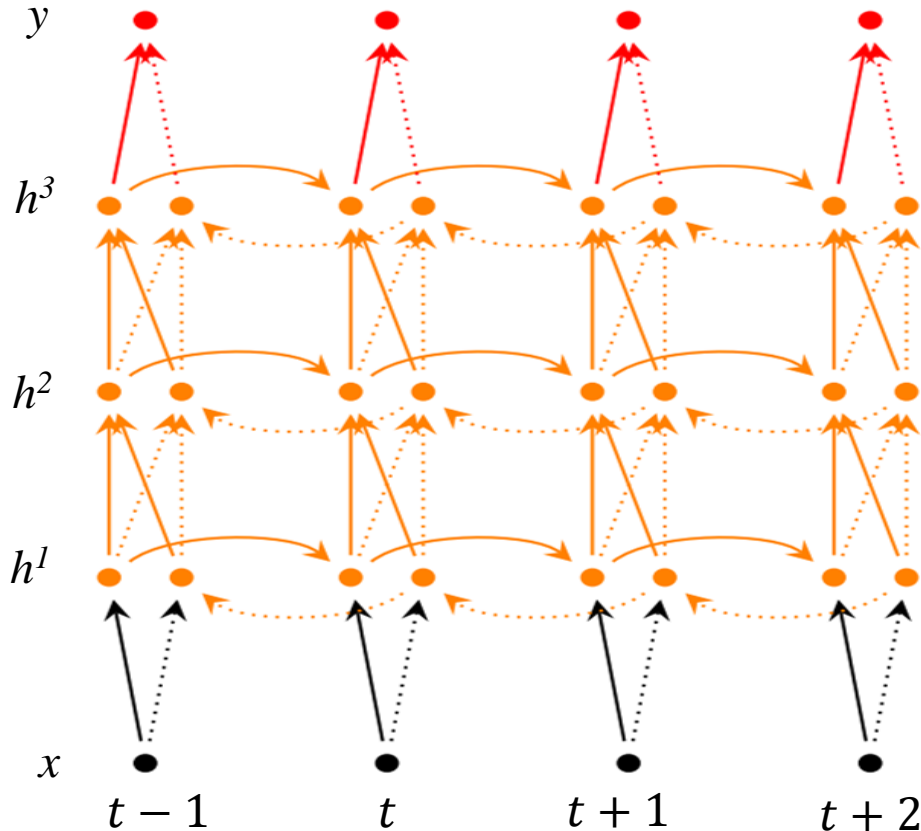


$$h_t^1 = \sigma (W^1 x_t + U^1 h_{t-1}^1)$$

$$h_t^2 = \sigma (W^2 h_t^1 + U^2 h_{t-1}^2)$$

$$h_t^3 = \sigma (W^3 h_t^2 + U^3 h_{t-1}^3)$$

## 4 Deep-stacked RNN



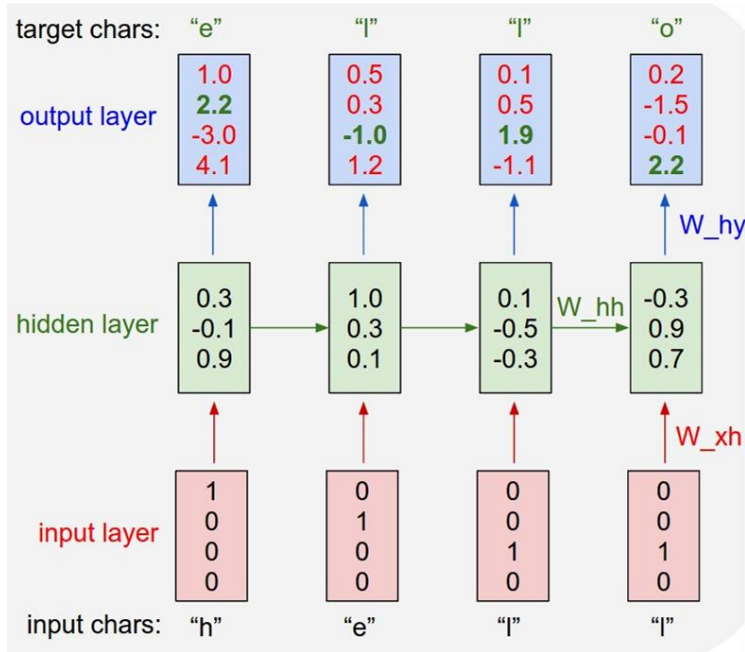
$$\vec{h}_t^i = f(\vec{W}^i x_t + \vec{U}^i \vec{h}_{t-1}^i)$$

$$\overleftarrow{h}_t^i = f(\overleftarrow{W}^i x_t + \overleftarrow{U}^i \overleftarrow{h}_{t+1}^i)$$

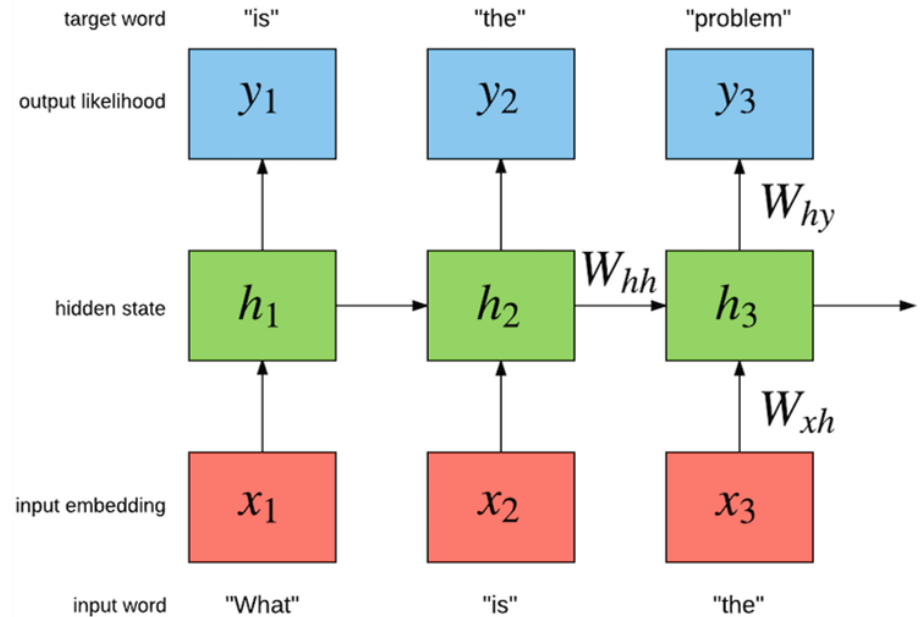
$$y_t = \text{softmax}\left(\left[\overleftarrow{h}_t^i, \vec{h}_t^i\right]\right)$$

# 4

## Deep-stacked RNN



Character-level RNN



Word-level RNN

# Nội dung

1. Pre - Neural Machine Translation
2. Neural Machine Translation
3. Sequence-to-Sequence model

# Nội dung

1. Pre - Neural Machine Translation
2. Neural Machine Translation
3. Sequence-to-Sequence model

# 1 Machine Translation - definition

- Machine translation:
  - Input x: 1 câu trong ngôn ngữ nguồn (source language)
  - Output y: 1 câu trong ngôn ngữ đích (destination language)
- Ví dụ:

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



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

- Rousseau

# 1 1950s: Early Machine Translation

- Nghiên cứu về machine translation bắt đầu từ năm 1950s
- Đa số được áp dụng trên cặp ngôn ngữ Russian → English
- Hệ thống dịch máy thường chỉ sử dụng luật (rule-based model)





# 1 1990s-2010s: Statistical Machine Translation

- Ý tưởng: Xây dựng một mô hình xác suất từ ngữ liệu
- Ví dụ: tiếng Anh  $\rightarrow$  tiếng Pháp
  - Cần tìm **một câu tiếng Anh  $y$  đúng nghĩa nhất**, với đầu vào là **một câu tiếng Pháp  $x$**

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

- Sử dụng mô hình Bayes để chia  $P(y|x)$  thành hai phần để học riêng lẻ

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

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

Translation Model

Language Model

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

# 1 1990s-2010s: Statistical Machine Translation

- **Bài toán trở thành:** Cách học được một translation model  $P(y|x)$ ?
  - Sử dụng một lượng **ngữ liệu song ngữ lớn** (parallel data) – bao gồm những cặp câu Anh/Pháp được người dịch sẵn

The Rosetta Stone



Ancient Egyptian

Demotic

Ancient Greek

All that glitters is not gold

↔ Tout ce qui brille n'est pas d'or

These violent delights have violent ends...

↔ Ces délices violents ont des fins violentes ...

Hell is empty and all the devils are here.

↔ L'enfer est vide et tous les démons sont ici.

...

↔ ...

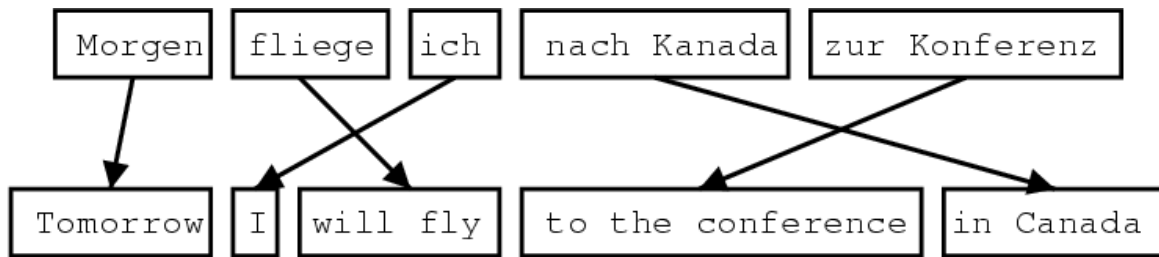
William Shakespeare

# 1 Learning alignment for SMT

- **Bài toán trở thành:** Cách học được một translation model  $P(y|x)$  từ parallel corpus?
  - Tiếp tục chia nhỏ công thức:

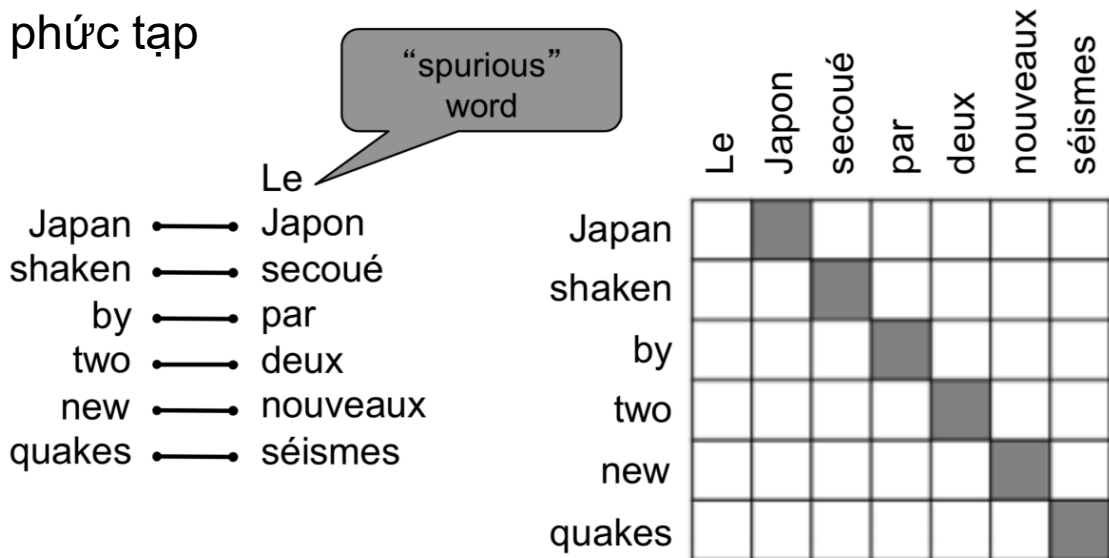
$$P(x, a|y)$$

Với  $a$  là alignment (cạnh nối giữa những từ trong ngôn ngữ nguồn và những từ trong ngôn ngữ đích)



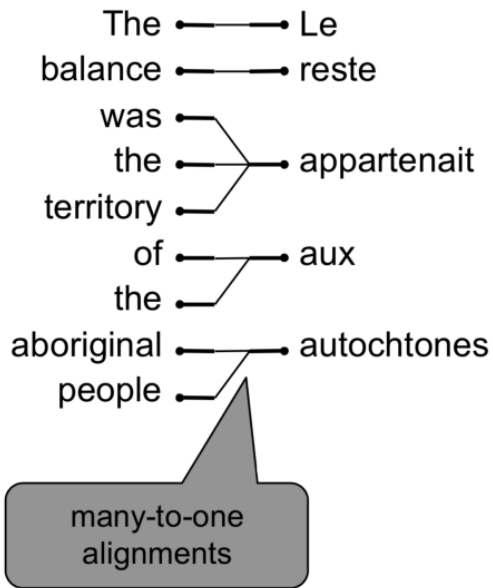
# 1 Learning alignment for SMT

- **Alignment** là sự tương ứng với nhau giữa những từ cụ thể trong một cặp câu của tập ngữ liệu song ngữ
- “Weak” point: sự khác nhau về typology của các ngôn ngữ khác nhau sẽ dẫn đến alignment phức tạp



# 1 Alignment is complex

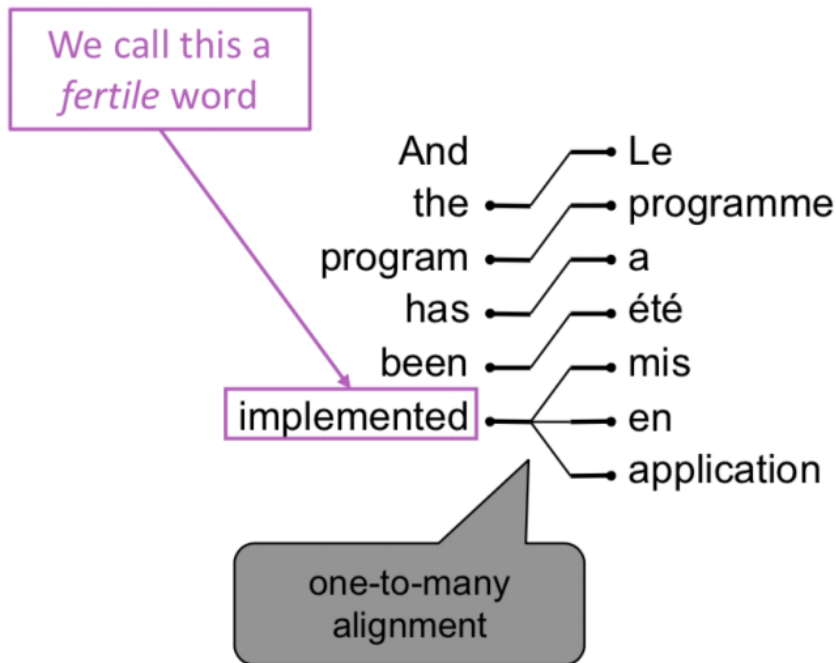
- **Alignment** có thể là quan hệ many-to-one



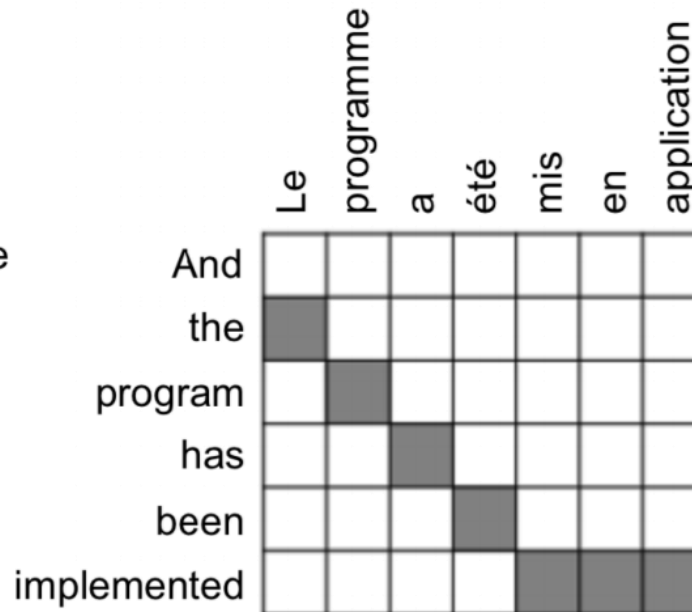
	Le	reste	appartenait	aux	autochtones
The					
balance					
was					
the					
territory					
of					
the					
aboriginal					
people					

# 1 Alignment is complex

- **Alignment** có thể là quan hệ one-to-many

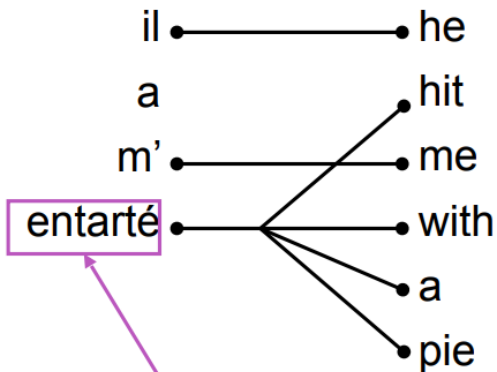


	Le	programme	a	été	mis	en	application
And							
the							
program							
has							
been							
implemented							

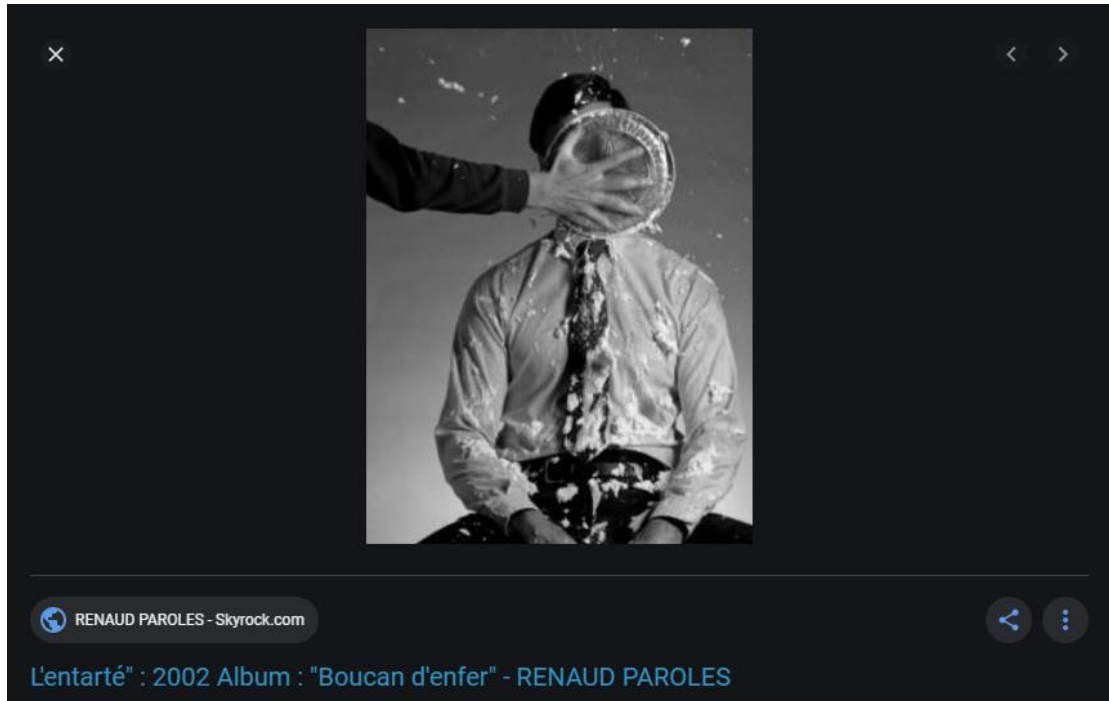


# 1 Alignment is complex

- Trường hợp đặc biệt

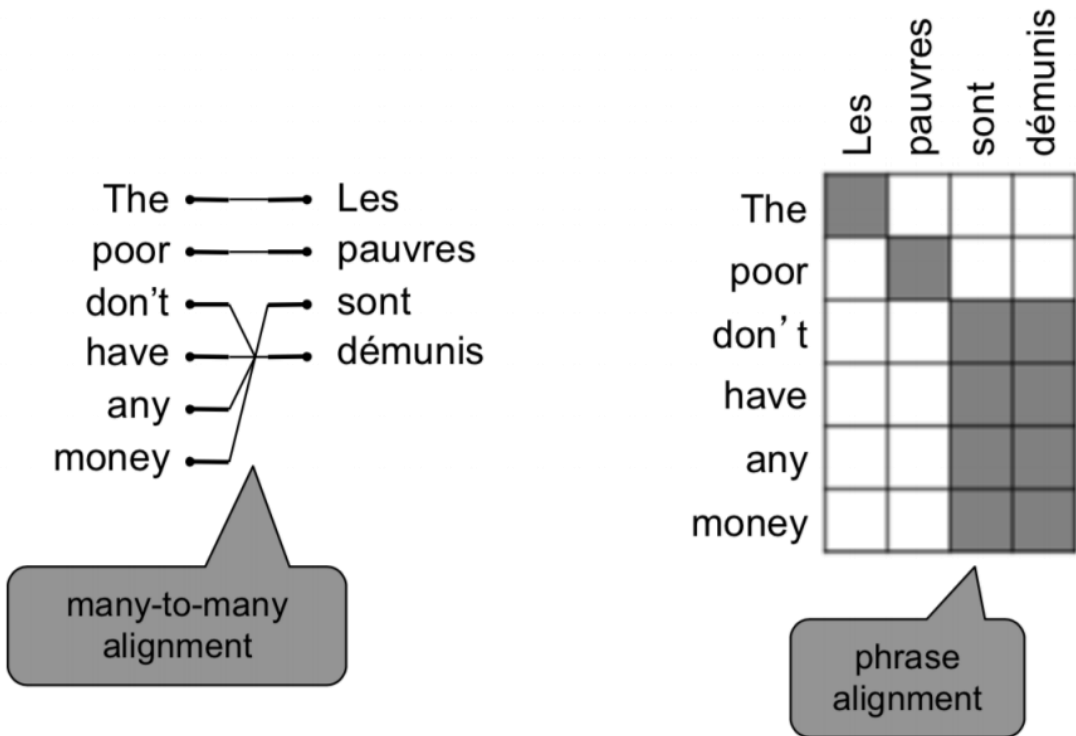


This word has no single-word equivalent in English



# 1 Alignment is complex

- **Alignment** có thể là quan hệ many-to-many (phrase level)





# 1 Learning alignment for SMT

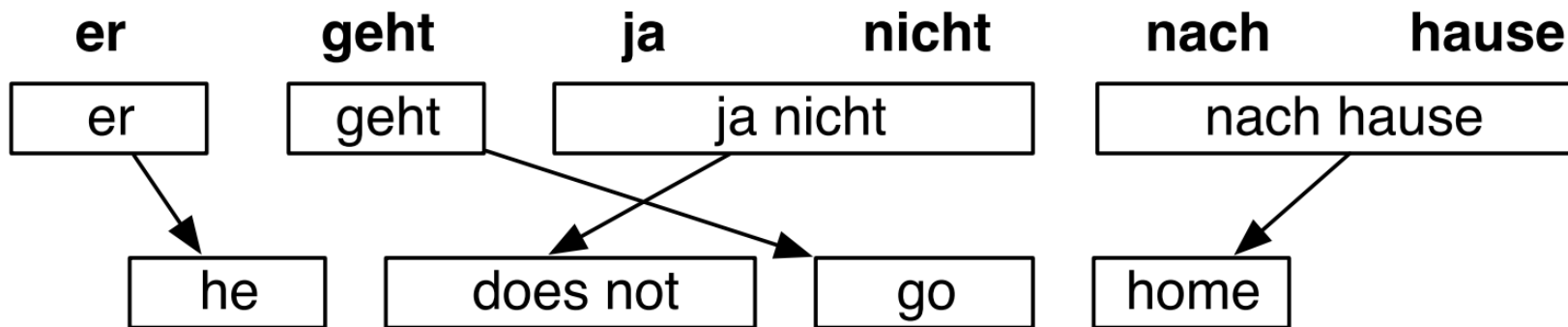
- Để xây dựng được mô hình xác suất  $P(x, a|y)$ :
  - Xác suất của một từ trong ngôn ngữ nguồn được align với **một** từ khác trong ngôn ngữ đích (và có phụ thuộc vào vị trí)
  - Xác suất của một từ trong ngôn ngữ nguồn được align với **nhiều** từ khác trong ngôn ngữ đích (và có phụ thuộc vào số lượng từ)
  - ...
- Alignment không hiển thị trong ngữ liệu – latent variable

# 1 Decoding for SMT

$$\text{argmax}_y \underbrace{P(x|y)}_{\substack{\text{Cách tìm được} \\ \text{xác suất lớn nhất?}}} \underbrace{P(y)}_{\substack{\text{Translation Model}}} \underbrace{P(y)}_{\substack{\text{Language Model}}}$$

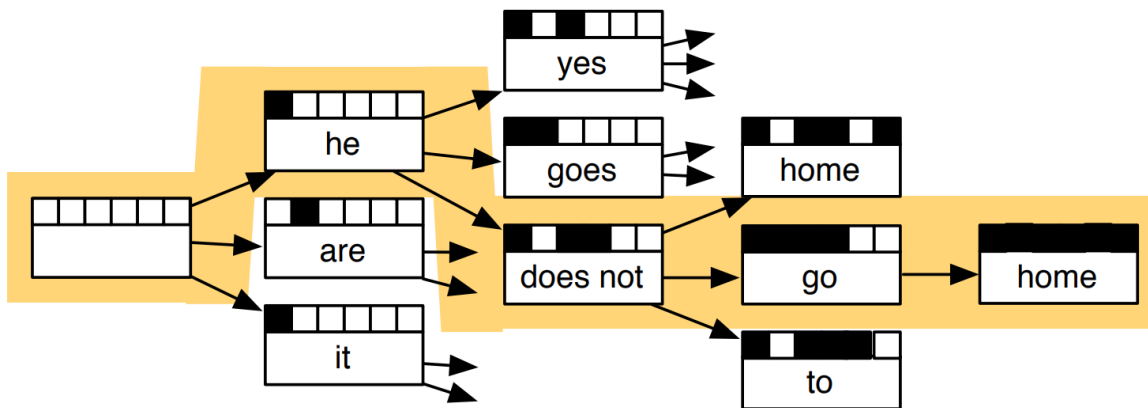
- Vét cạn  $\rightarrow$  expensive computational cost
- Solution: sử dụng thuật toán heuristic search để tìm ra translation phù hợp nhất, loại bỏ đi những hypotheses có xác suất thấp
  - Quá trình này được gọi là decoding

# 1 Heuristic search for SMT



# 1 Heuristic search for SMT

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				



# 1 Statistical Machine Translation

- Hệ thống SMT tốt sẽ cực kỳ phức tạp
  - Cần nhiều feature engineerings
  - Sẽ có nhiều ad-hoc features để có thể xử lý được những trường hợp đặc biệt của mỗi ngôn ngữ
  - Cần nhiều chi phí để lưu trữ resource (vd bảng của những phrase tương ứng nhau)
  - Cần nhiều nhân lực để bảo trì hệ thống, làm những việc tương tự nhau cho từng cặp ngôn ngữ

→ Neural Machine Translation

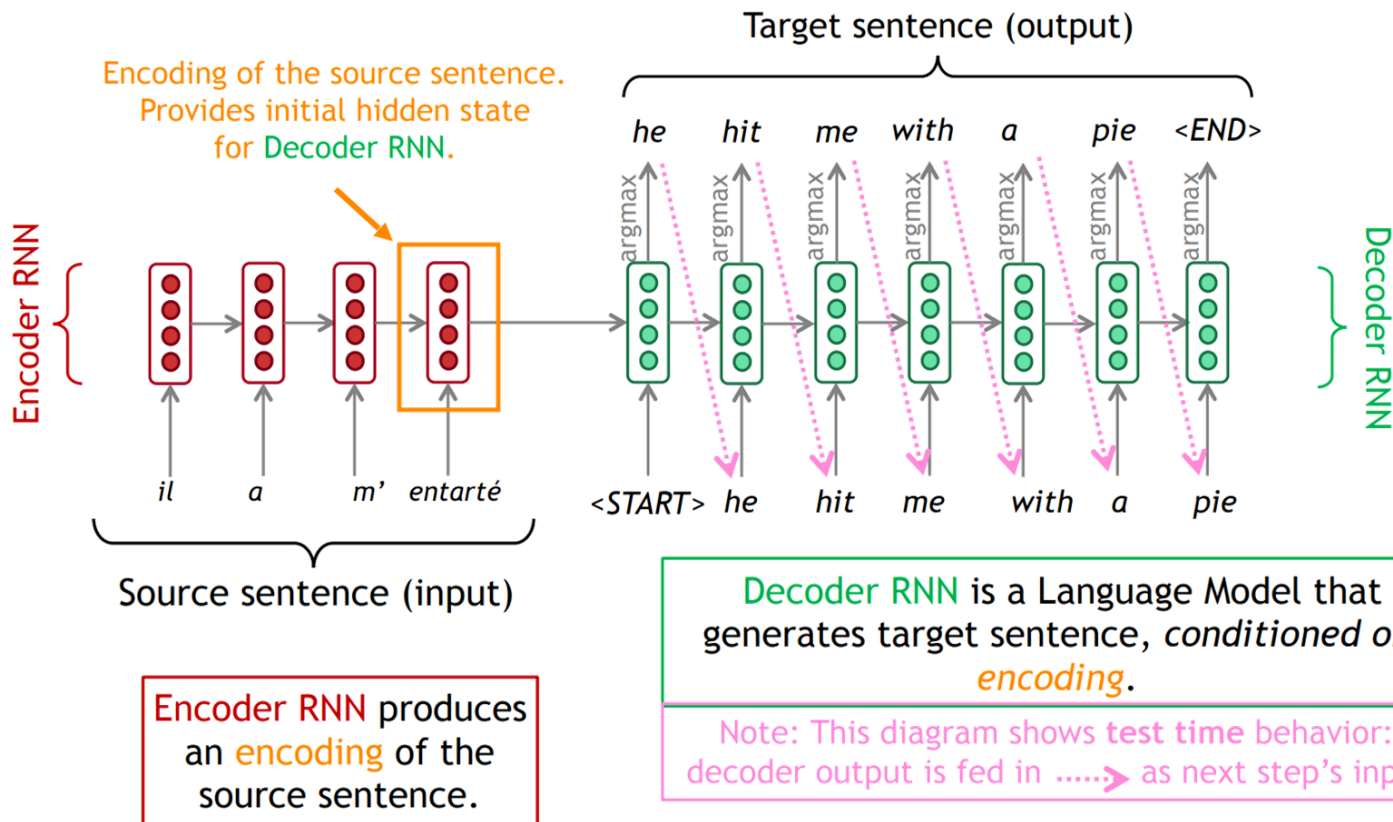
# Nội dung

1. Pre - Neural Machine Translation
2. **Neural Machine Translation**
3. Sequence-to-Sequence model

## 2 Neural Machine Translation

- Neural Machine Translation là hệ thống dịch máy sử dụng một mạng neurons
- Cấu trúc mạng neurons sử dụng cho bài toán neural machine translation được gọi là sequence-to-sequence model (seq2seq) và bao gồm 2 RNNs

## 2 Neural Machine Translation





## 2 Seq2seq model

- Seq2seq model còn có thể được dùng cho các bài toán khác
- Ví dụ:
  - Text summarization (long text → short text)
  - Dialogue (previous utterances → next utterances)
  - Parsing (input text → output parse as sequence)
  - Code generation (natural language → Python code)

## 2 Conditional language model

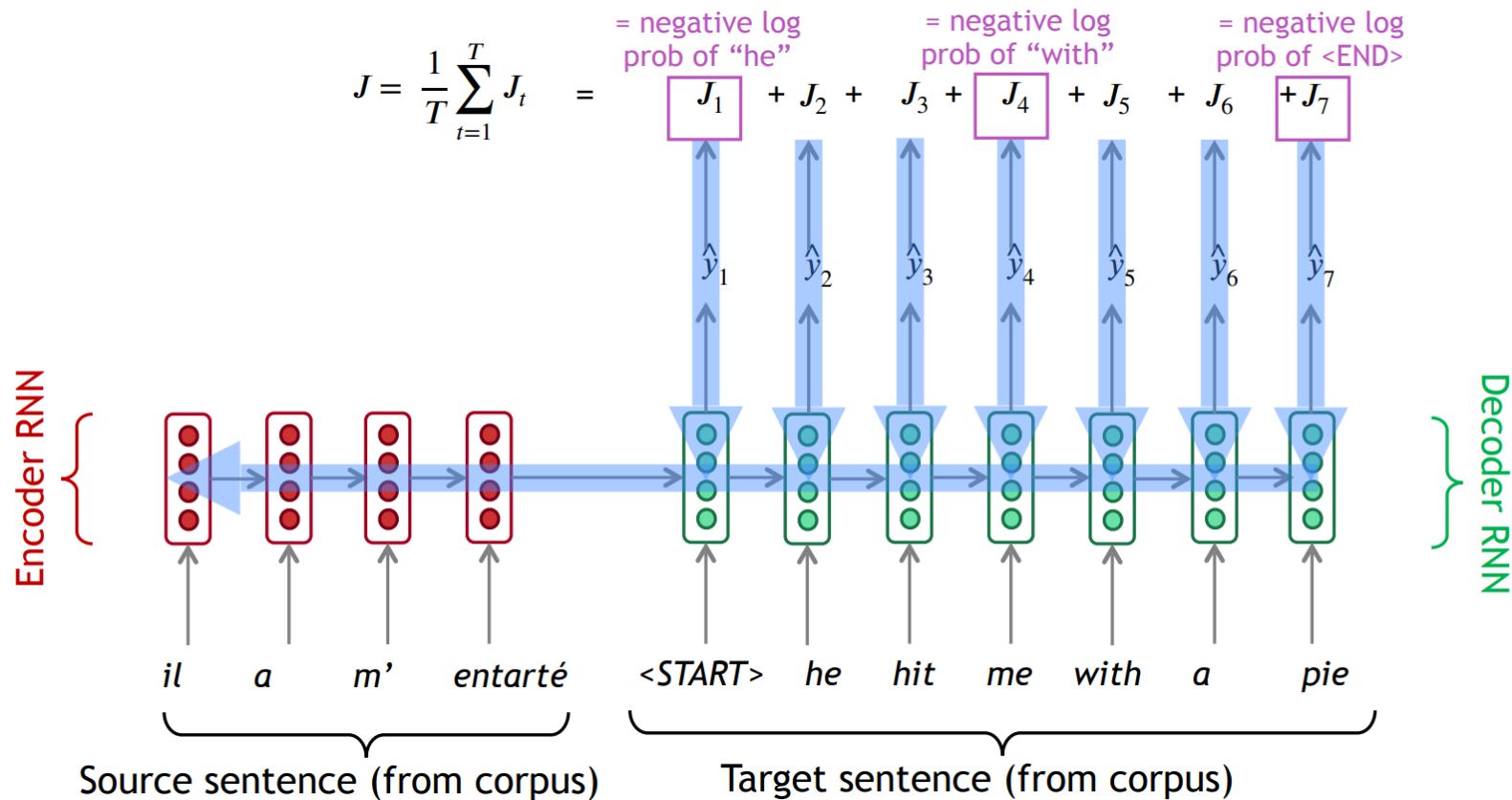
- Seq2seq là một ví dụ của conditional language model
  - Language model: vì decoder dự đoán từ tiếp theo cho target sentence  $y$
  - Conditional: vì dự đoán của decoder được dự đoán dựa trên source sentence  $x$
- NMT tính xác suất  $P(y|x)$ , trong đó:

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

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

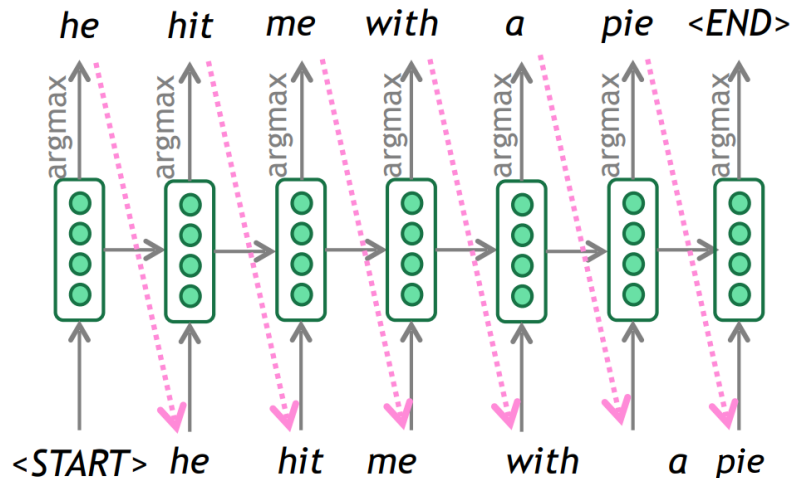
- **Bài toán đặt ra:** Cách huấn luyện một NMT model?

## 2 Training Neural Machine Translation model



## 2 Training Neural Machine Translation model

- Ở mỗi bước decode, target sentence được generate bởi từ có xác suất cao nhất



- Cách generate từ mới này được gọi là **greedy coding** (sử dụng từ xuất hiện với xác suất cao nhất)

## 2 Training Neural Machine Translation model

- Greedy decoding không có cách để quay trở lại bước trước đó
- Ví dụ
  - Input: il a m' entarté (he hit me with a pie)
  - he \_\_\_\_\_
  - he hit \_\_\_\_\_
  - he hit a \_\_\_\_\_

Model generate sai từ tiếp theo  
(no going back issue)

## 2 Exhaustive search decoding

- Cách decoder generate từ mới ở hiện tại: tìm một translation  $y$  với  $y$  có độ dài  $T$  từ và  $P(y|x)$  lớn nhất

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

- Khi đó, ta cần tính xác suất này trên tất cả những chuỗi  $y$  có thể có
  - Ở mỗi bước của decoder (mỗi giá trị  $t$ ), ta cần tính trên  $V^t$  những chuỗi translations  $y$  (câu chưa hoàn chỉnh) có thể có, với  $V$  là vocab size  $\rightarrow$  Độ phức tạp  $O(V^T) \rightarrow$  expensive

## 2 Beam search decoding

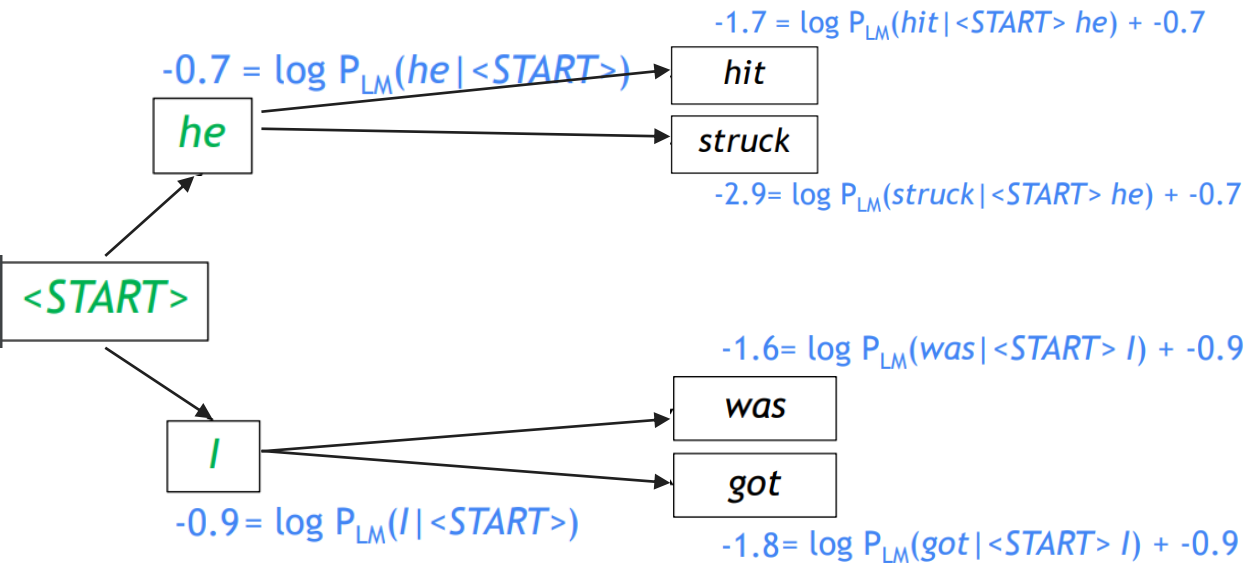
- Ý tưởng chính: Ở mỗi bước decode, chỉ xét k translations y (k hypotheses y)
  - K là beam size (thường được chọn từ 5 đến 10)
- Một hypothesis  $y_1, \dots, y_t$  có score được tính bởi log prob:

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

- Score luôn âm, score càng lớn thì càng tốt
- Beam search không chắc sẽ đưa ra kết quả tối ưu, nhưng hiệu quả hơn nhiều về mặt tính toán so với exhaustive search

## 2 Beam search decoding

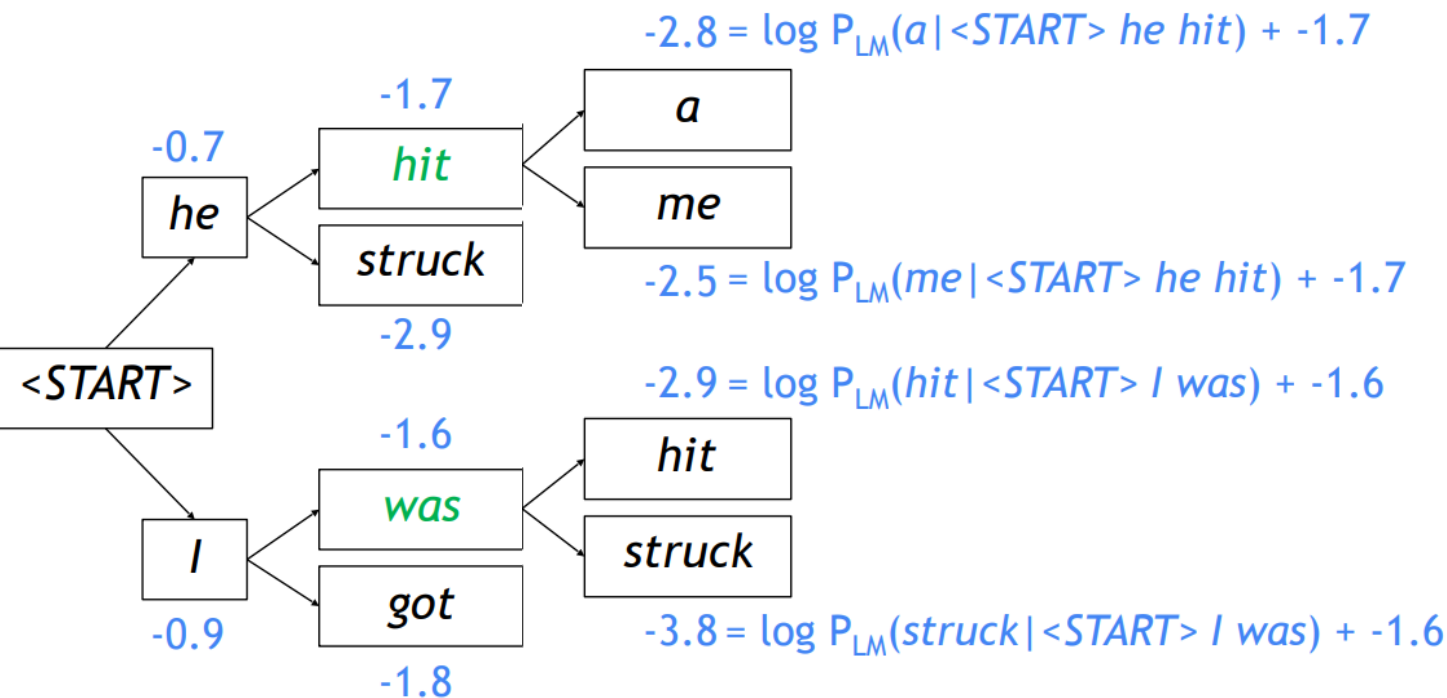
- Ví dụ: Beam size  $k = 2$





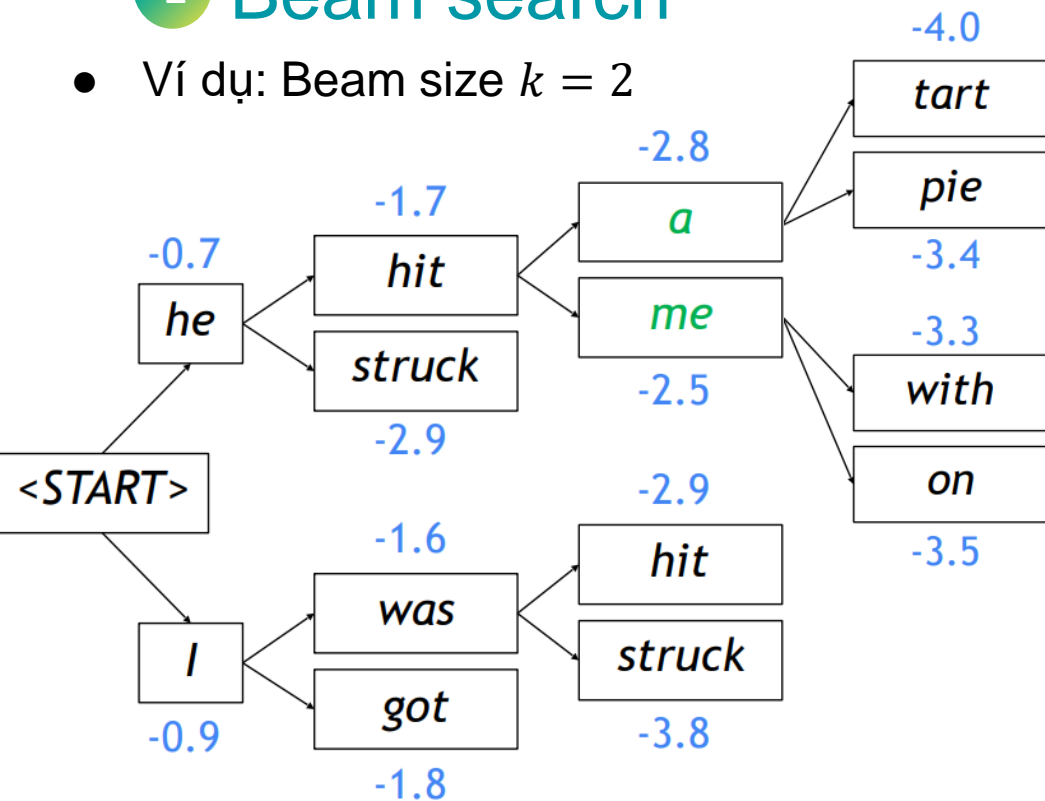
## 2 Beam search decoding

- Ví dụ: Beam size  $k = 2$



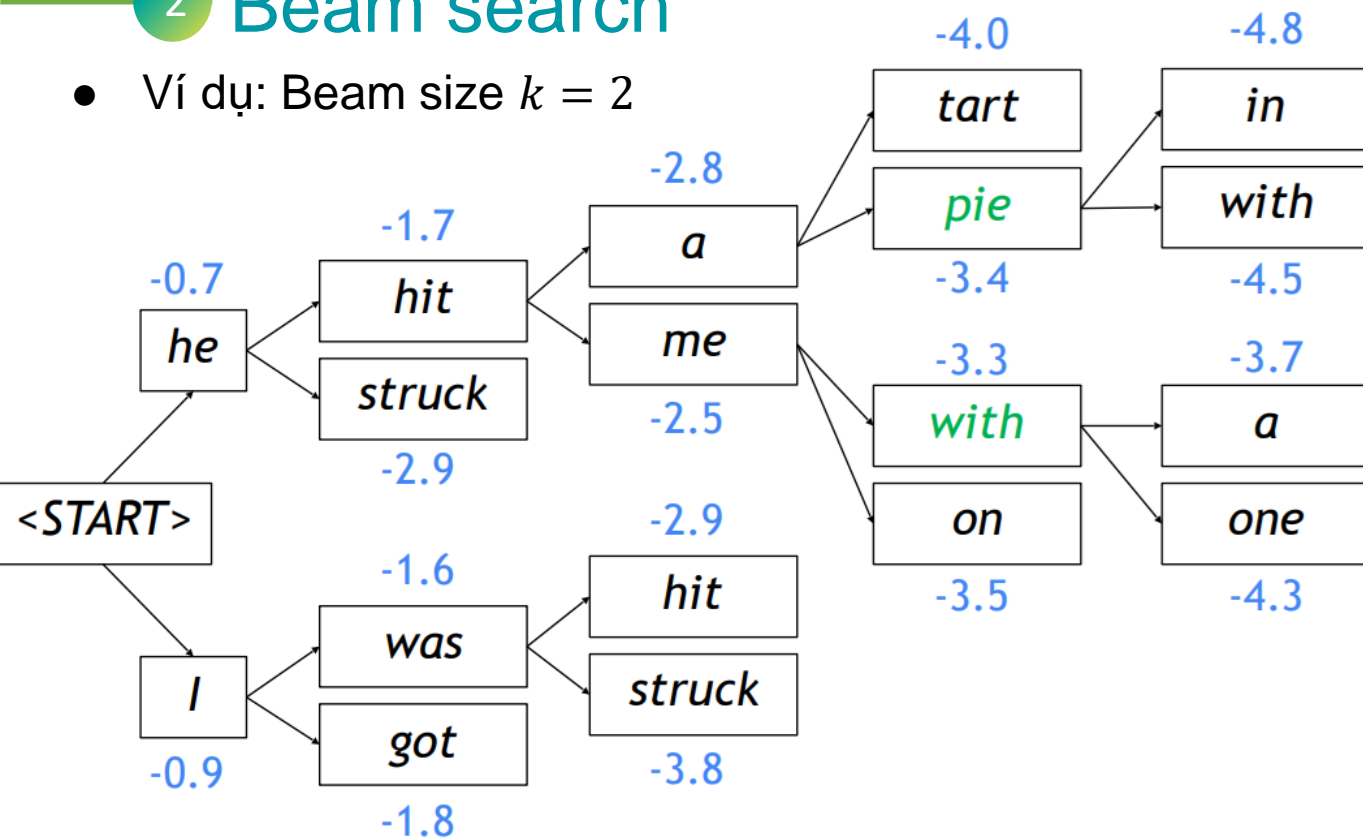
## 2 Beam search

- Ví dụ: Beam size  $k = 2$



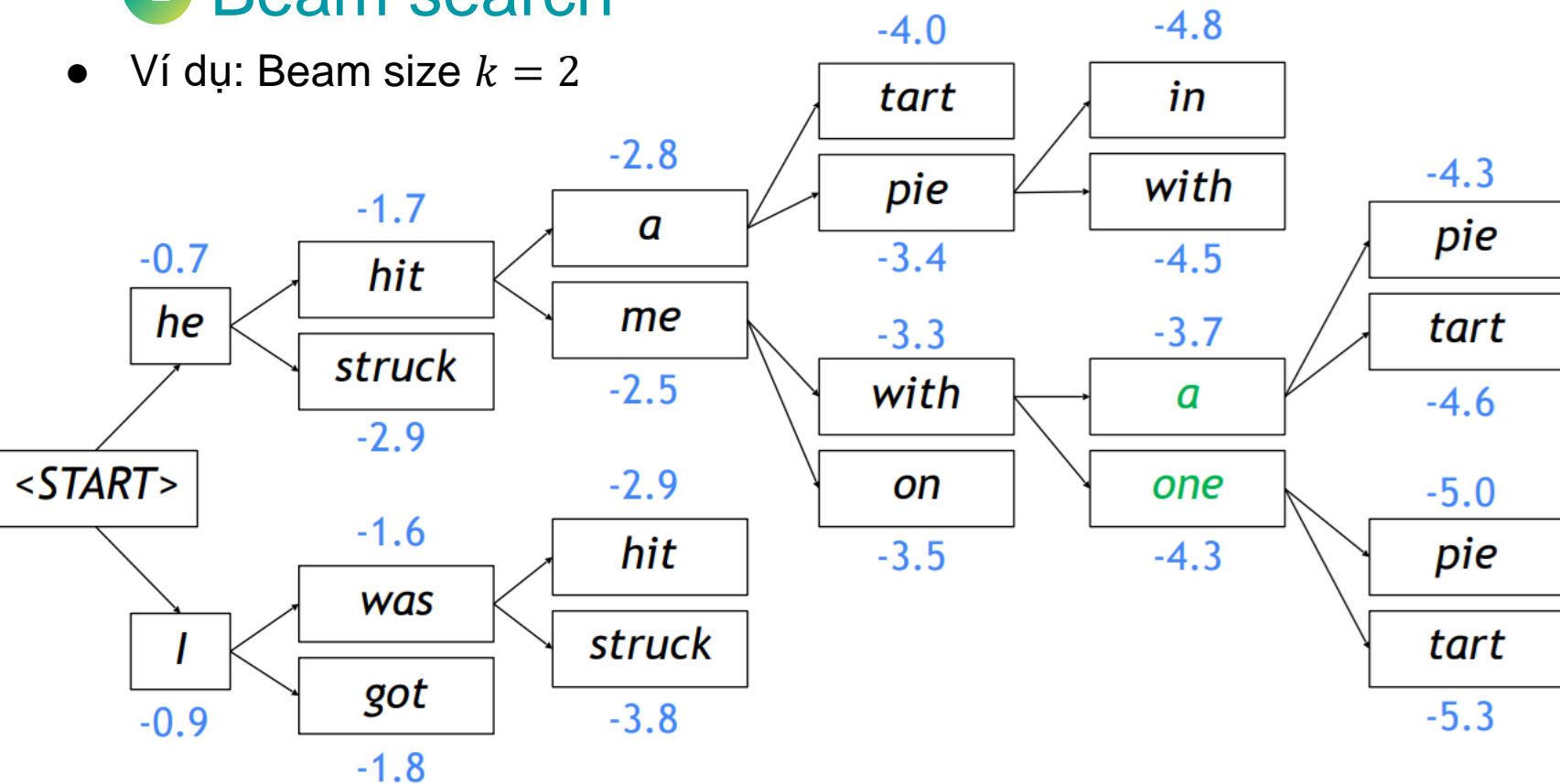
## 2 Beam search

- Ví dụ: Beam size  $k = 2$



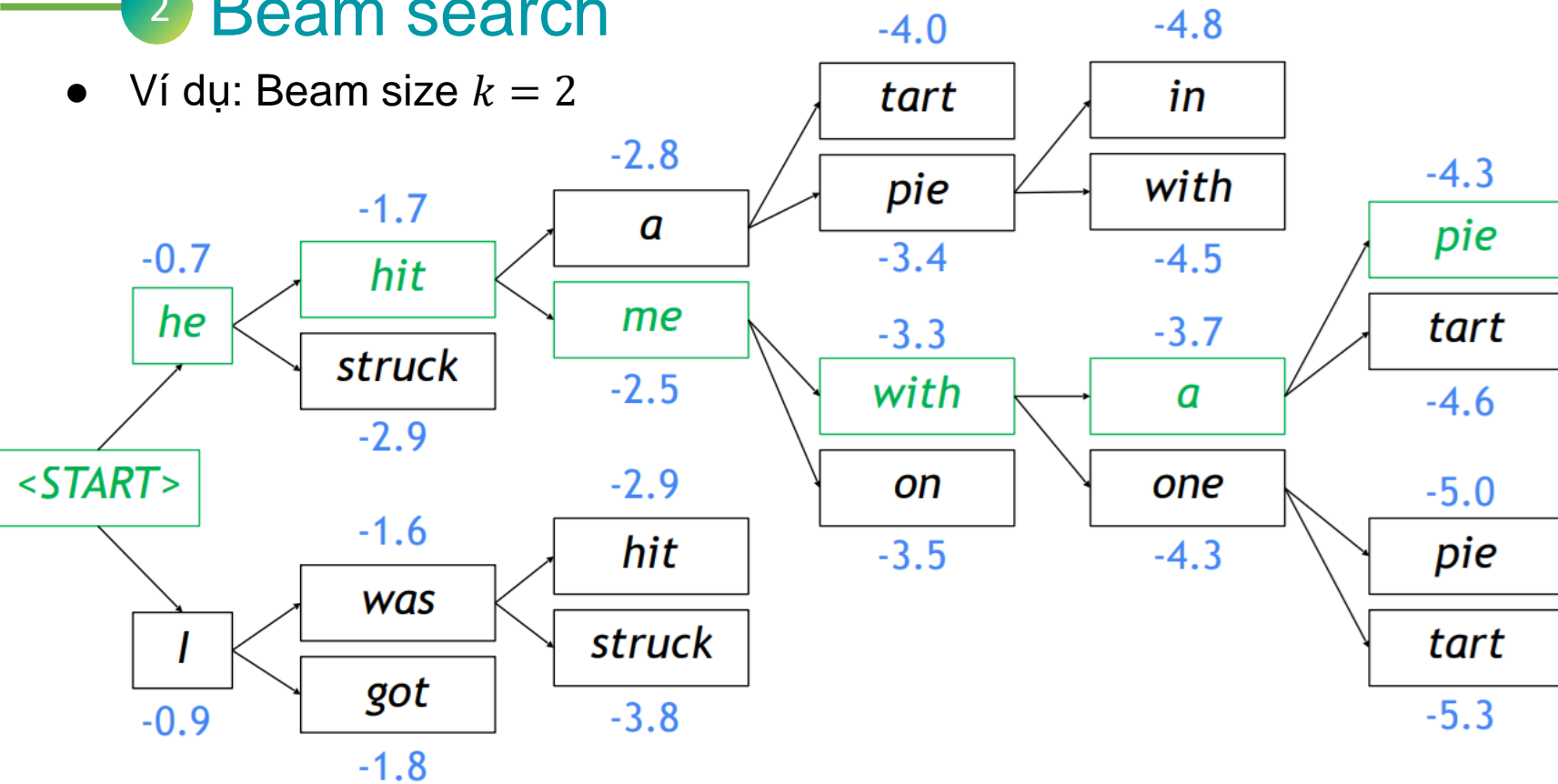
## 2 Beam search

- Ví dụ: Beam size  $k = 2$



## 2 Beam search

- Ví dụ: Beam size  $k = 2$



## 2 Beam search – stopping criterion

- Khi sử dụng **greedy coding**, quá trình decode sẽ dừng lại khi gặp <END> token. Ví dụ: <START> he hit me with a pie <END>
- Khi sử dụng **beam search decoding**, những hypotheses khác nhau sẽ generate <END> token tại những timestamp  $t$  khác nhau
  - Khi một hypothesis generate <END>, hypothesis đó hoàn thành  $y$
  - Tiếp tục beam search cho những hypotheses khác
  - Tiếp tục beam search cho đến khi
    - Đến timestamp  $S$  nhất định
    - Có được  $N$  hypotheses hoàn thành

## 2 Beam search – stopping criterion

- Khi sử dụng **greedy coding**, quá trình decode sẽ dừng lại khi gặp <END> token. Ví dụ: <START> he hit me with a pie <END>
- Khi sử dụng **beam search decoding**, những hypotheses khác nhau sẽ generate <END> token tại những timestamp  $t$  khác nhau
  - Khi một hypothesis generate <END>, hypothesis đó hoàn thành  $y$
  - Tiếp tục beam search cho những hypotheses khác
  - Tiếp tục beam search cho đến khi
    - Đến timestamp  $S$  nhất định
    - Có được  $N$  hypotheses hoàn thành

## 2 Beam search – normalization

- Mỗi hypothesis  $y_1, \dots, y_k$  sẽ có một score tương ứng

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

- Khi đó, hypothesis tạo chuỗi dài sẽ có score thấp hơn
- Solution: Normalize dựa trên độ dài chuỗi

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



## 2 Advantages of NMT

- Performance tốt hơn
  - Fluency
  - Context
  - Phrase similarities
- Single neural network – sử dụng gradient descent để optimize end-to-end
  - No sub-component to be optimized individually
- Cần ít nhân lực
  - Không cần feature engineering
  - Sử dụng chung một phương pháp cho tất cả cặp ngôn ngữ

## 2

## Disadvantages of NMT

- Hard to debug
- Hard to control