Artificial Neural Networks
人工神经网络

权小军教授
中山大学计算机学院

quanxj3@mail.sysu.edu.cn

2023 年 5 月 18 日

Lecture 10 - Recurrent Neural Network II

循环神经网络II

Some slides adapted from Christopher Manning

Lecture 10.1 Machine Translation

机器翻译

《静夜思》

床前明月光，疑是地上霜。

举头望明月，低头思故乡。

"Thinking at night"

The bright moon in front of the bed is suspected to be frost on the ground.

Raising my head, I see the moon so bright; withdrawing my eyes, my nostalgia comes around.

（百度翻译, 20/08/2021 ）

新华社东京8月19日电（记者王子江、邓敏）中国残奥代表团首批约190人的团队19日顺利抵达东京，首批队伍包括参加自行车、游泳、盲人门球和田径项目的运动员。

Xinhua news agency, Tokyo, August 19 (reporters wang Zijiang and Deng min) - the first group of about 190 members of the Chinese Paralympic delegation successfully arrived in Tokyo on the 19th. The first group includes athletes participating in cycling, swimming, blind gateball and track and field events.

（百度翻译, 20/08/2021）

Xinhua news agency, Tokyo, August 19 (reporters wang Zijiang and Deng min) - the first group of about 190 members of the Chinese Paralympic delegation arrived in Tokyo on the 19th. The first group includes athletes participating in cycling, swimming, blind gateball and track and field events.

（百度翻译, 24/05/2022）

新华社东京8月19日电（记者王子江、邓敏）中国残奥代表团首批约190人的团队19日顺利抵达东京，首批队伍包括参加自行车、游泳、盲人门球和田径项目的运动员。

Xinhua News Agency, Tokyo, August 19 (Reporter Wang Zijiang, Deng Min) The first group of about 190 members of the Chinese Paralympic delegation arrived in Tokyo on August 19. The first group included athletes who participated in cycling, swimming, blind goal kick and track and field events.

（百度翻译, 29/11/2022）

Machine Translation

Machine Translation (MT) is the task of translating a sentence *x* from one language (the source language) to a sentence *y* in another language (the target language).

The early history of MT: 1950s

- Machine translation research began in the early 1950s on machines less powerful than high school calculators.
- MT is basically just simple rule-based systems doing word substitution.
- Human language is more complicated than that and varies across languages!
- Little understanding of natural language syntax (句法), semantics (语义), pragmatics (语用).

1990s-2010s: Statistical Machine Translation

- **Core idea**: Learn a probabilistic model from data
- Suppose we're translating Chinese → English.
- We want to find best English sentence *y*, given Chinese sentence *x*

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

1990s-2010s: Statistical Machine Translation

- Use Bayes Rule to break this down into two components to be learned separately:

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

$$= \text{argmax}_y \underbrace{P(x|y)}_{\text{Translation Model}} \underbrace{P(y)}_{\text{Language Model}}$$

Translation Model

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

Language Model

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

1990s-2010s: Statistical Machine Translation

- Question: How to learn translation model $P(x|y)$?
- First, need large amount of parallel data (平行数据).

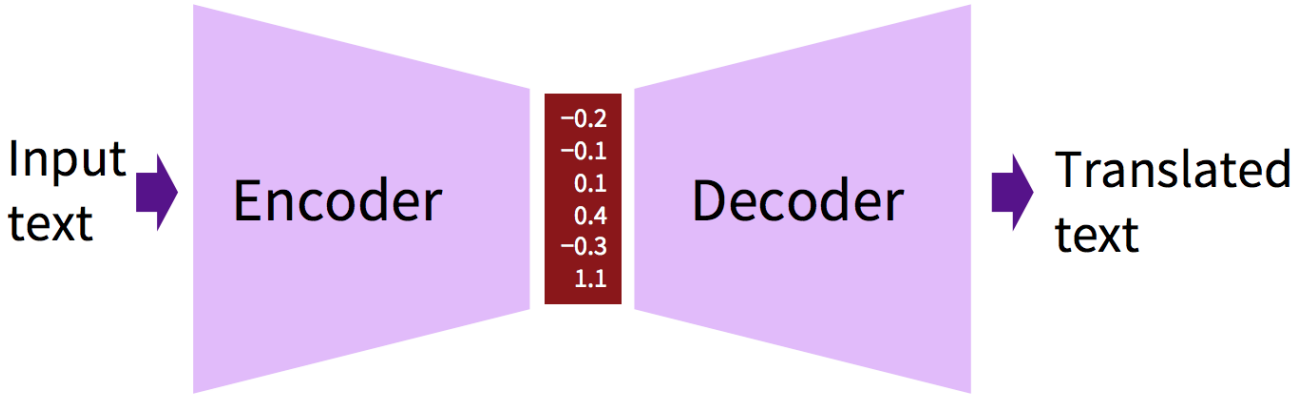
中文	韩文
示威队在撕毁自民党新藤义孝议员的照片。	시위대가 신도 요시타카 일본 자민당 의원 사진을 찢고 있다.
是自由民主主义还是亲北左派？请公开政治立场	자유민주인지 친북좌파인지 정체성을 밝혀라
新国家党访谈节目	새누리당 토크쇼
新右派运动的任务就是改造国民的思维。	뉴 라이트 운동이 할 일은 바로 이러한 국민정신개조 운동이다.
甚至开放的我们党议长郑东泳也表示要向总统提出进行道歉的建议。	열린우리당 정동영 의장까지도 대통령에게 사과를 건의하겠다고 하지 않는가.
双手合十磕头的她身上。	여승은 합장하고 절을 했다.
蚩尤见一时难以取胜，便施起法来。	치우는 일순간 승리가 어렵다고 보고 적당한 방법을 실행하였다.
教育部和全国教职员工会联手反对建立国际中学。	교육부는 전국교직원노동조합과 의기투합해 국제중 설립에 반대해 왔다.
球队核心，具慈哲，金镇炫都患上了感冒。	대표 팀 핵심인 손흥민, 구자철, 김진현은 모두 감기 몸살에 걸렸다.
竟日游乐	하루 종일 놀며 즐기다.
鸡叫声从远处接连不断地传来。	먼 데서 닭 소리가 잇달아 들려왔다.
股价呈现坚挺的涨势。	주가가 견조한 상승세를 보이다.
孤零零的一座房子坐落在山脚下。	외딴 집 한 채 가산 아래 앉아 있다.
怎么保湿都还干燥的时候	아무리 발라도 건조할 때
靠水吃水	물에 인접한 곳에서는 물에 의지하여 먹고산다.
书房里挂了一张郑板桥的单幅	서재에 정판교의 단일 족자 한 폭을 걸었다.
昨天走得太多，所以小腿肚都缩成了一疙瘩。	어제 너무 많이 걸었더니 종아리에 알이 뱄다.
建设业嘛，就有困难。	건설업이요 어렵겠는데요

1990s-2010s: Statistical Machine Translation

- SMT was a huge research field
- The best systems were extremely complex
  - Hundreds of important details we haven't mentioned here
  - 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
    - Like tables of equivalent phrases
  - Lots of human effort to maintain
    - Repeated effort for each language pair!

2014-current: Neural Machine Translation

- Modeling the machine translation using neural networks
  - **Encoder** for language understanding in source language
  - **Decoder** for language generation in target language



Lecture 10.2 Challenges of MT

Challenges

- Ambiguity (歧义) and unknown phenomena in natural language
  - Syntactic ambiguity / lexical ambiguity / pragmatic ambiguity
  - New words, terms, structures, semantics
- The result of machine translation is not unique (唯一)

Progress of MT

原文：Beijing made a third solemn representation to Manila and warned that it is hard to be optimistic about a territorial impasse over an island. Authorities say they have prepared for any escalation of the situation by Manila.

Chinadaily, 8 May 2012

Progress of MT

Baidu 翻译

- 北京做第三严正交涉到马尼拉，并警告说这是很难约领土僵局的一个岛屿乐观。当局说，他们已经准备了马尼拉的情况有任何升级。(2015.4.28)
- 北京由第三严正交涉到马尼拉，并警告说这是很难约了一个岛领土僵局持乐观态度。当局说，他们已经为马尼拉局势的升级准备。(2016.5.1)
- 北京对马尼拉进行了第三次庄严的代表，并警告说，对岛上的领土僵局很难看好。当局表示，他们为马尼拉的情况升级做好了准备。(2017.4.16)
- 北京向马尼拉提出了第三次严正交涉，并警告说很难对一个岛屿的领土僵局持乐观态度。当局表示，他们已经为马尼拉局势升级做好了准备。(2018.5.15)
- 北京向马尼拉提出第三次严正交涉，并警告称，很难对一个岛屿的领土僵局持乐观态度。当局说，他们已经为马尼拉局势的任何升级做好了准备。(2020.12.01)
- 北京向马尼拉提出了第三次郑重交涉，并警告说，很难对一个岛屿的领土僵局感到乐观。当局表示，他们已经为马尼拉局势的任何升级做好了准备。(2022.5.23)

Progress of MT

❖ 北京向马尼拉提出了第三次严正交涉，并警告说，很难对一个岛屿的领土僵局感到乐观。当局表示，他们已经为马尼拉局势的任何升级做好了准备。

(2022.11.29)

❖ 北京向马尼拉作出了第三次严正交涉，并警告说，很难对一个岛屿的领土僵局持乐观态度。当局表示，他们已经为马尼拉局势的任何升级做好了准备。

(2023.05.22)

Lecture 10.3 Neural Machine Translation

神经机器翻译

What is Neural Machine Translation?

- Neural Machine Translation (NMT) is a way to do Machine Translation with a *single* *end-to-end (端到端)* *neural network*

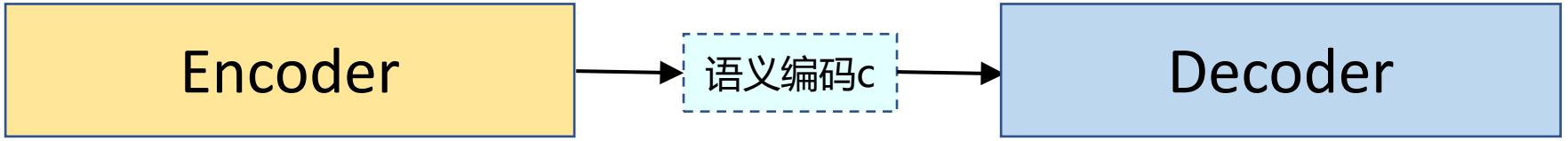
- The neural network architecture is called a sequence-to-sequence model (aka seq2seq), e.g., it can involves *two* RNNs

Sequence-to-sequence is versatile (多用途)!

- Sequence-to-sequence is useful for *more than just MT*
- Many NLP tasks can be phrased as sequence-to-sequence:
  - Summarization (long text → short text)
  - Dialogue (previous utterances → next utterance)
  - Parsing (input text → output parse as sequence)
  - Code generation (natural language → Python code)

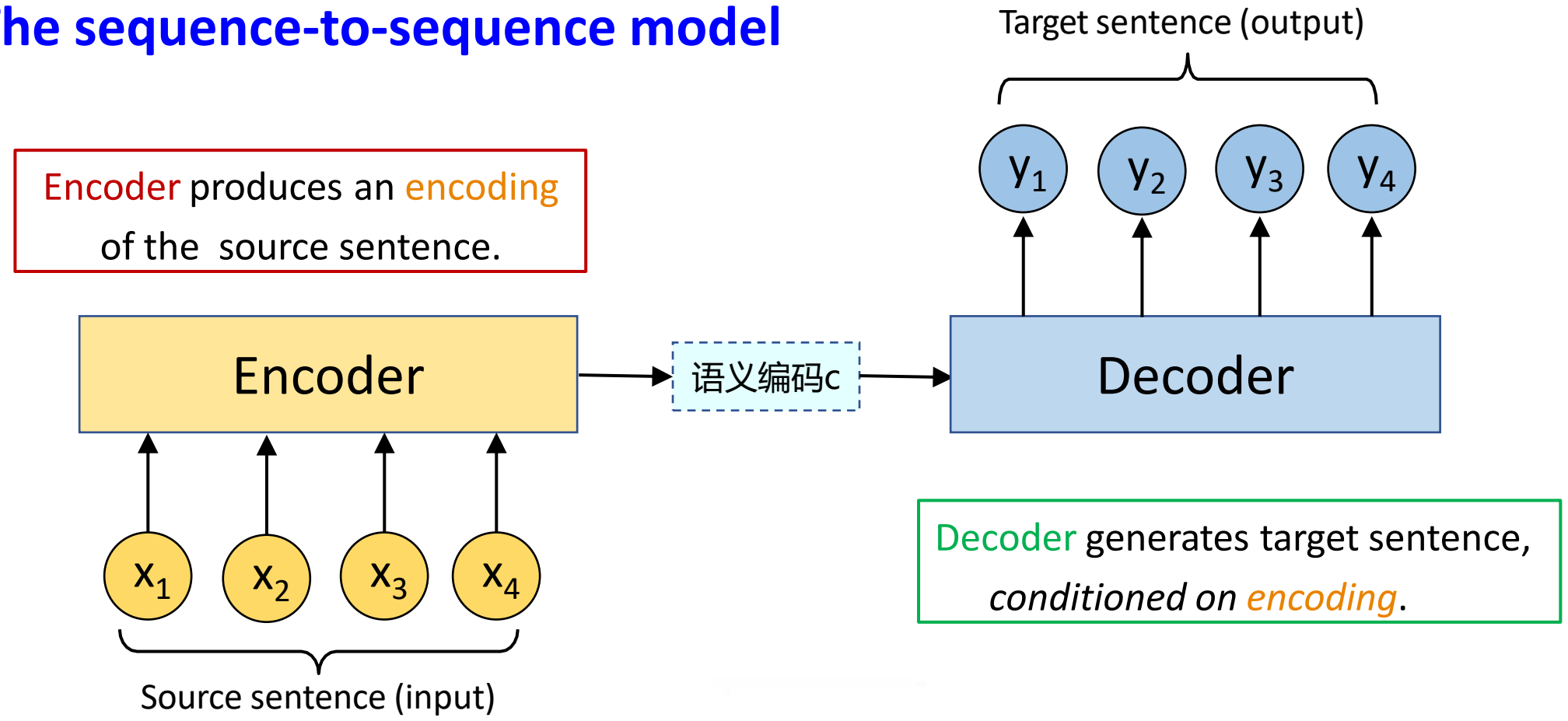
Neural Machine Translation (NMT)

The sequence-to-sequence model

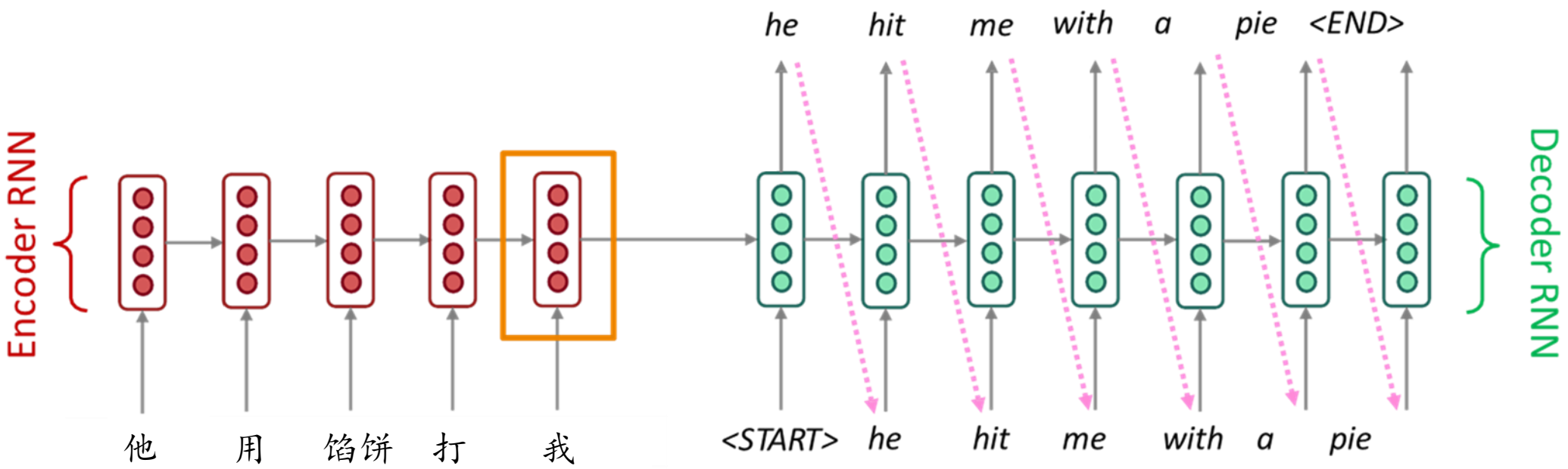


Neural Machine Translation (NMT)

The sequence-to-sequence model



Neural Machine Translation (NMT)



Neural Machine Translation (NMT)

- The sequence-to-sequence model is a **Conditional Language Model**
  - **Language Model**: decoder predicts the next word of the target sentence y
  - **Conditional**: its predictions are also conditioned on the source sentence x

- NMT directly calculates $P(y|x)$:

$$P(y|x) = P(y_1|x)\,P(y_2|y_1,x)\,P(y_3|y_1,y_2,x)\ldots P(\underbrace{y_T|y_1,\ldots,y_{T-1},x}_{\text{Probability of next target word, given target words so far and source sentence x}})$$

- **Question**: How to train a NMT system?
- **Answer**: Get a big parallel corpus…

Training a Neural Machine Translation system

Parallel Corpus

他 用 馅饼 打 我

Source sentence (from corpus)

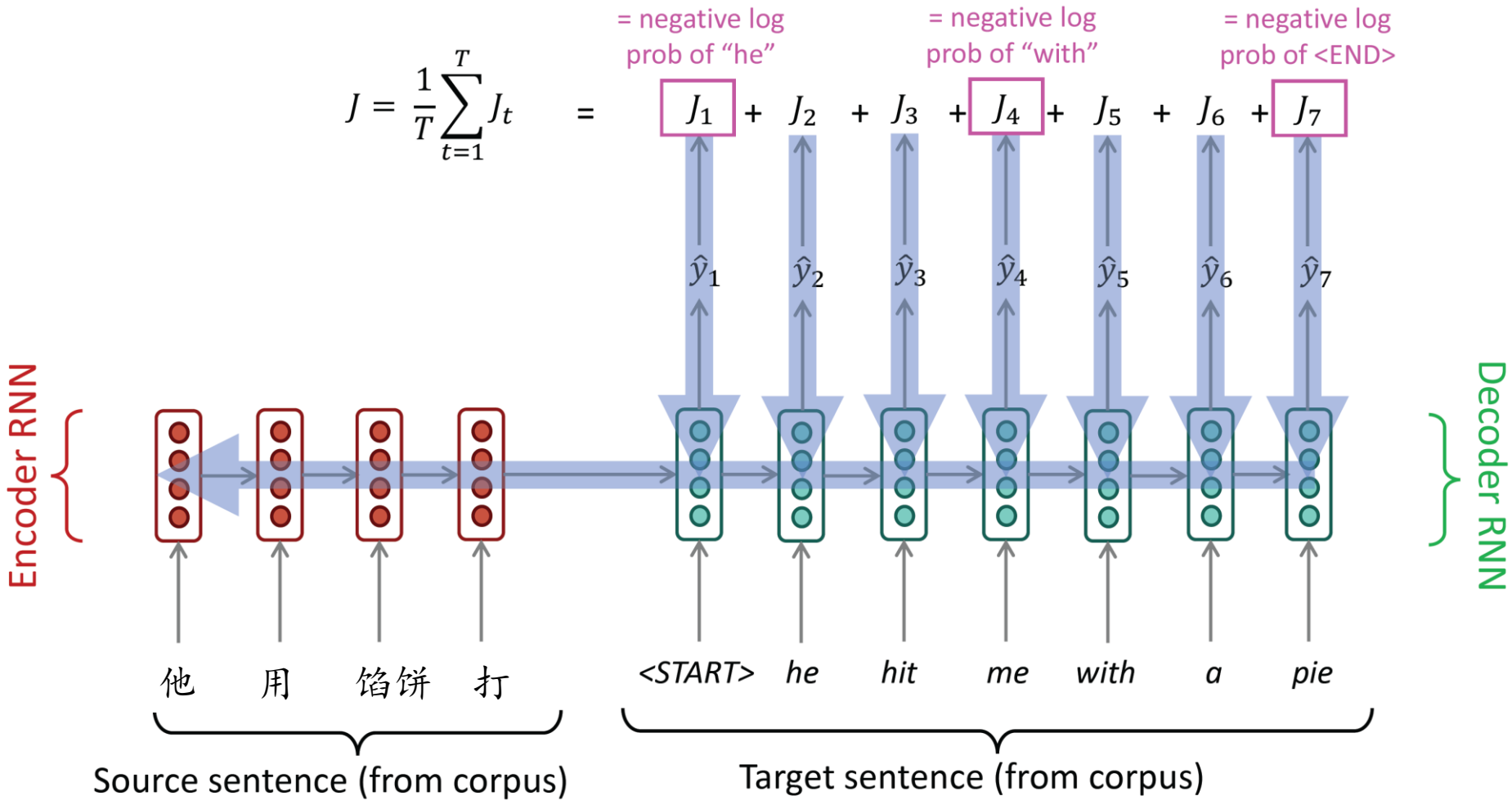
⋮

he hit me with a pie

Target sentence (from corpus)

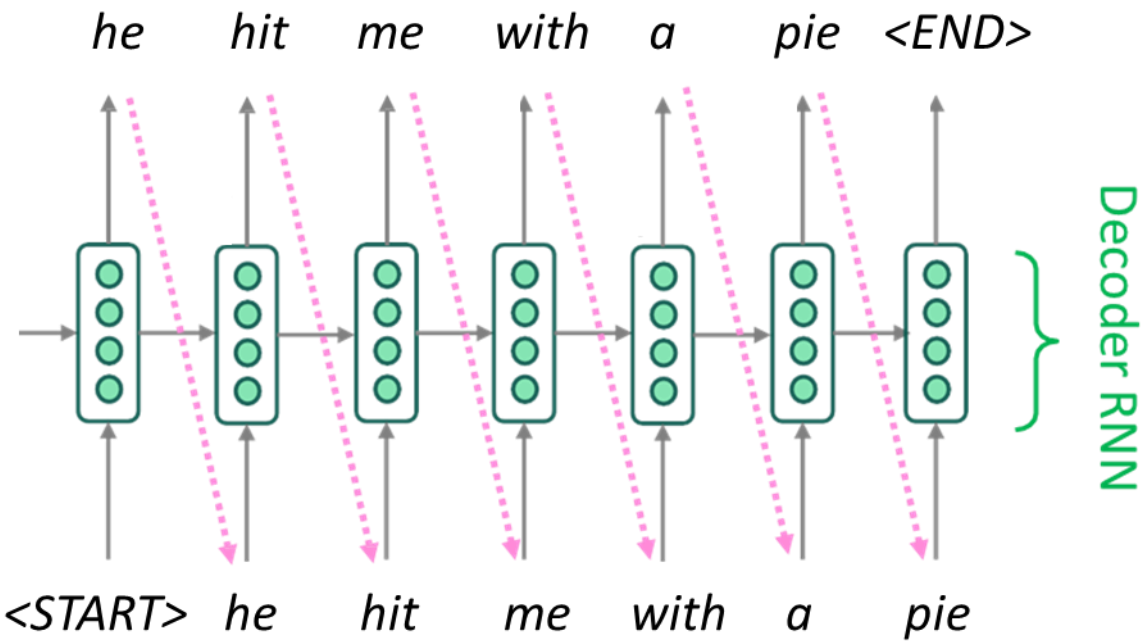
⋮

Training a Neural Machine Translation system



Seq2seq is optimized as a **single system.** Backpropagation operates "*end-to-end*".

Training vs Testing: Free-running Mode



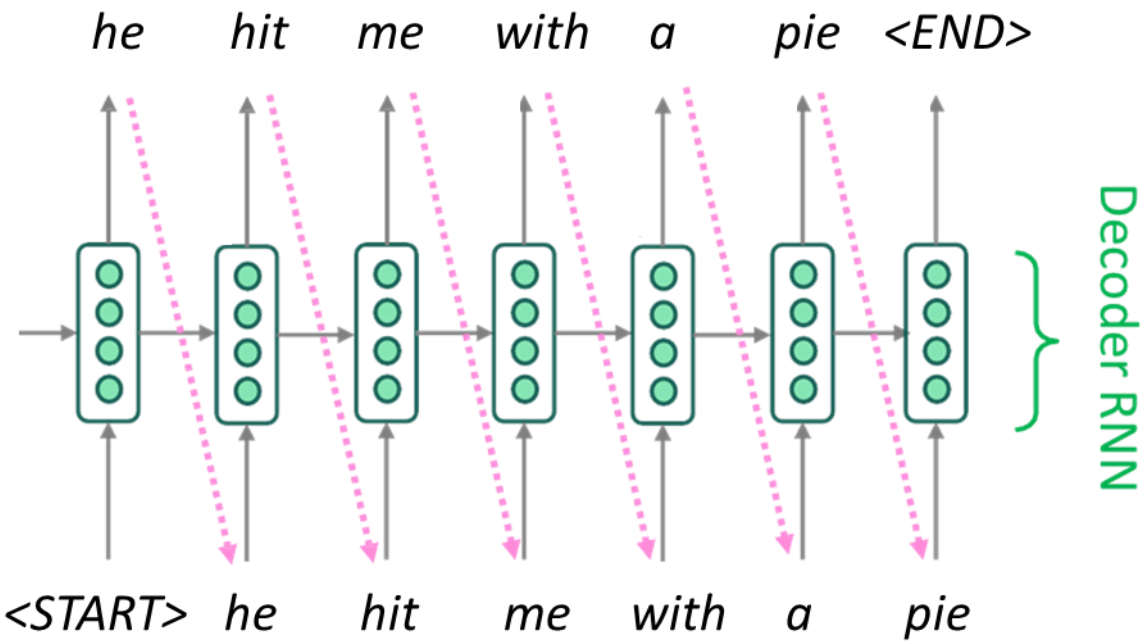
Free-running mode:

- **Training**: pass the *predicted word* to the next step
- **Testing**: pass the *predicted word* to the next time step

Problems:

- Slow convergence
- Model instability

Training vs Testing: Teacher-forcing Mode



Teacher-forcing mode:

- **Training**: pass the *ground truth word* to the next step
- **Testing**: pass the *predicted word* to the next time step

Advantages:

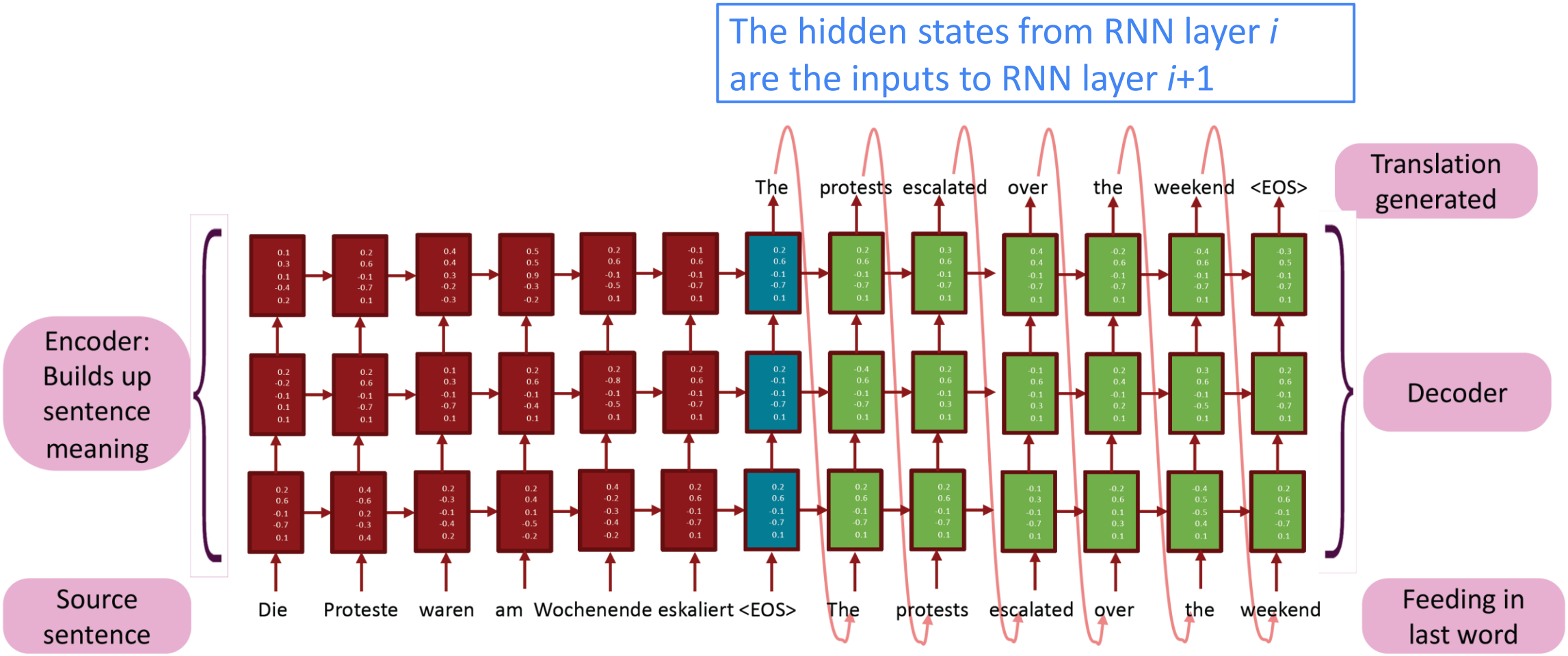
- Fast convergence
- Model stability

Disadvantages: the model has never been trained on its own errors and may not be robust to them.

Multi-layer RNNs

- RNNs are already “deep” on one dimension (unroll over timesteps)
- We can also make them “deep” in another dimension by applying multiple RNNs
- This allows the network to compute more complex representations
  - The lower RNNs should compute lower-level features and the higher RNNs should compute higher-level features.
- Multi-layer RNNs are also called *stacked RNNs*.

Multi-layer deep encoder-decoder machine translation net



Multi-layer RNNs in practice

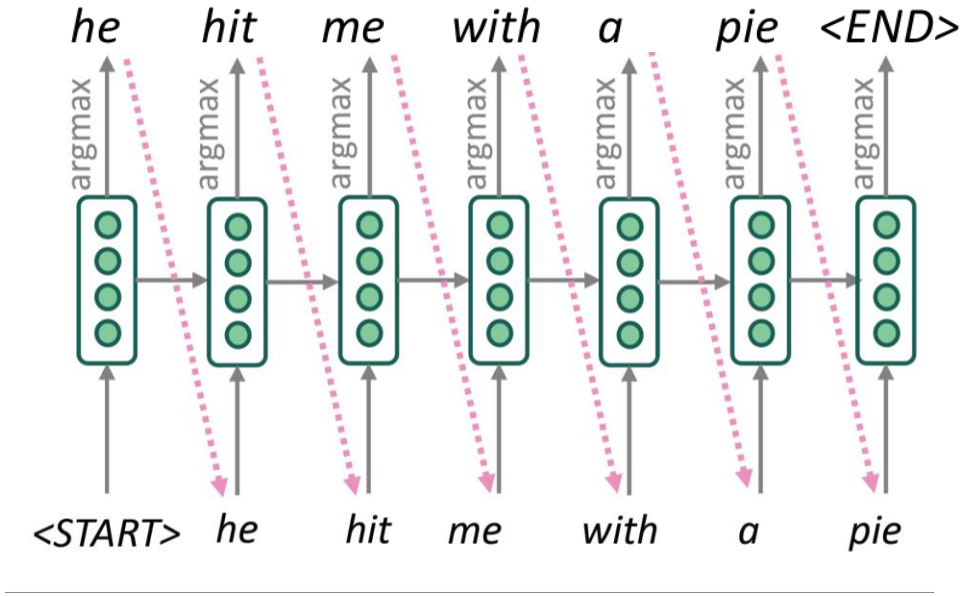
- High-performing RNNs are usually multi-layer
- Neural Machine Translation, 2 to 4 layers is best for the encoder RNN, and 4 layers is best for the decoder RNN

Lecture 10.4 Decoding Strategies

解码策略

Greedy decoding (inference/test)

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



- This is greedy (贪婪) decoding (take most probable word on each step)
- **Problems with this method?**

Problems with greedy decoding

- Greedy decoding has no way to undo decisions!
  - Input: 他 用 馅饼 打 我 *(he hit me with a pie)*
  - → *he* __
  - → *he hit* _____ *(whoops! no going back now…)*
  - → *he hit a* ___

How to fix this?

Beam search (束搜索) decoding (inference/test)

- Core idea: On each step of decoder, keep track of the *k* most probable partial translations
  - *k* is the beam size (in practice around 5 to 10)
- A hypothesis $y_1, \dots, y_t$ has a score which is its log probability:

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

  - Scores are all negative, and higher score is better
  - We search for high-scoring hypotheses, tracking top *k* on each step
- Beam search is not guaranteed to find optimal solution
- But much more efficient than exhaustive search (穷举搜索)!

Beam search decoding: example

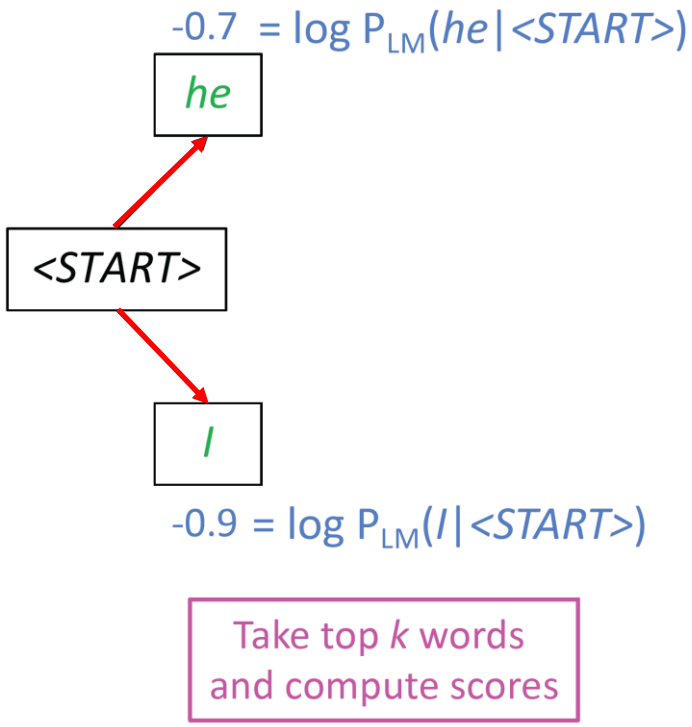
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$

<START>

Calculate prob dist of next word

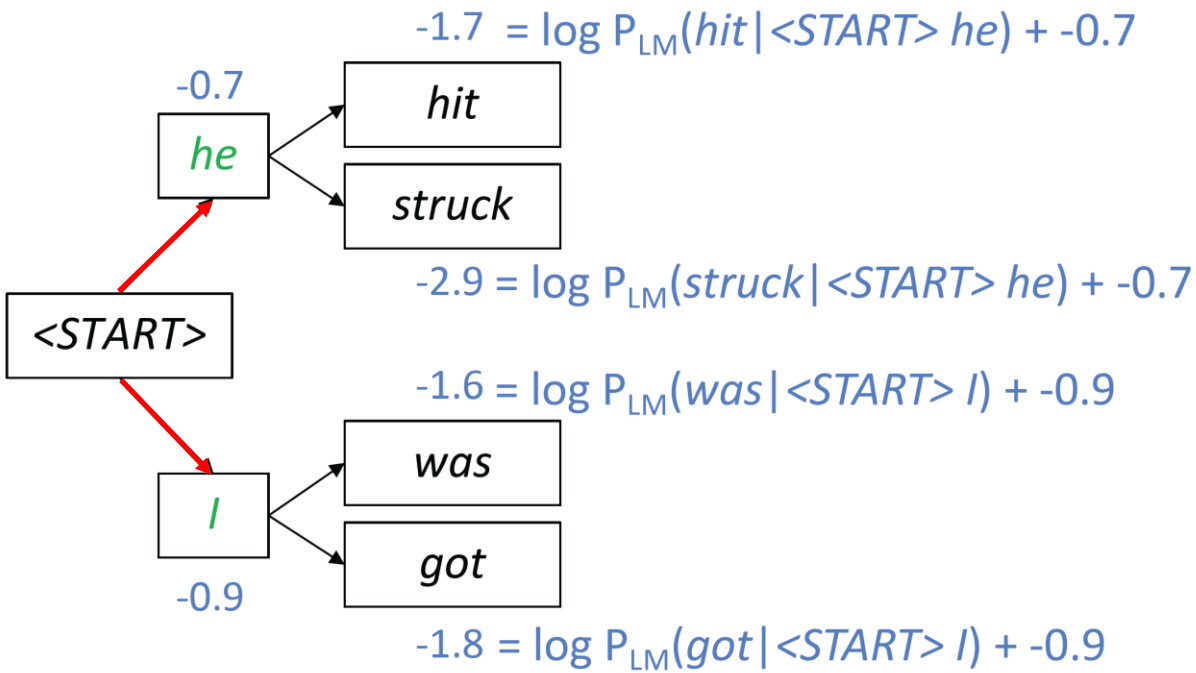
Beam search decoding: example

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



Beam search decoding: example

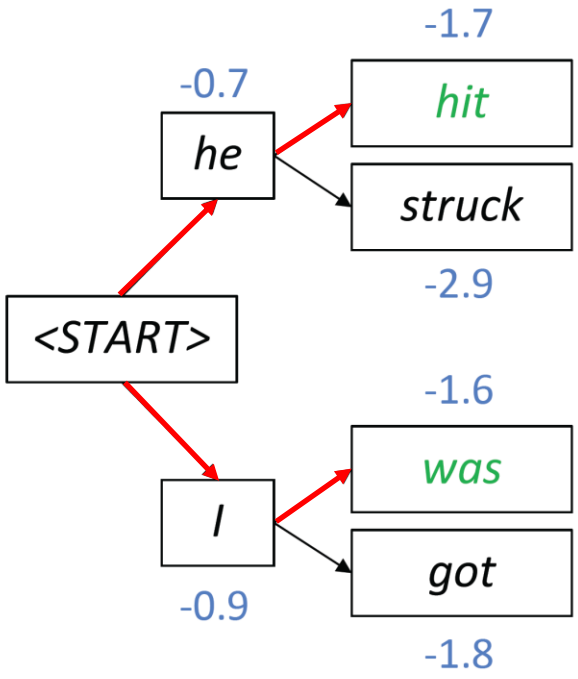
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



For each of the *k* hypotheses, find top *k* next words and calculate scores

Beam search decoding: example

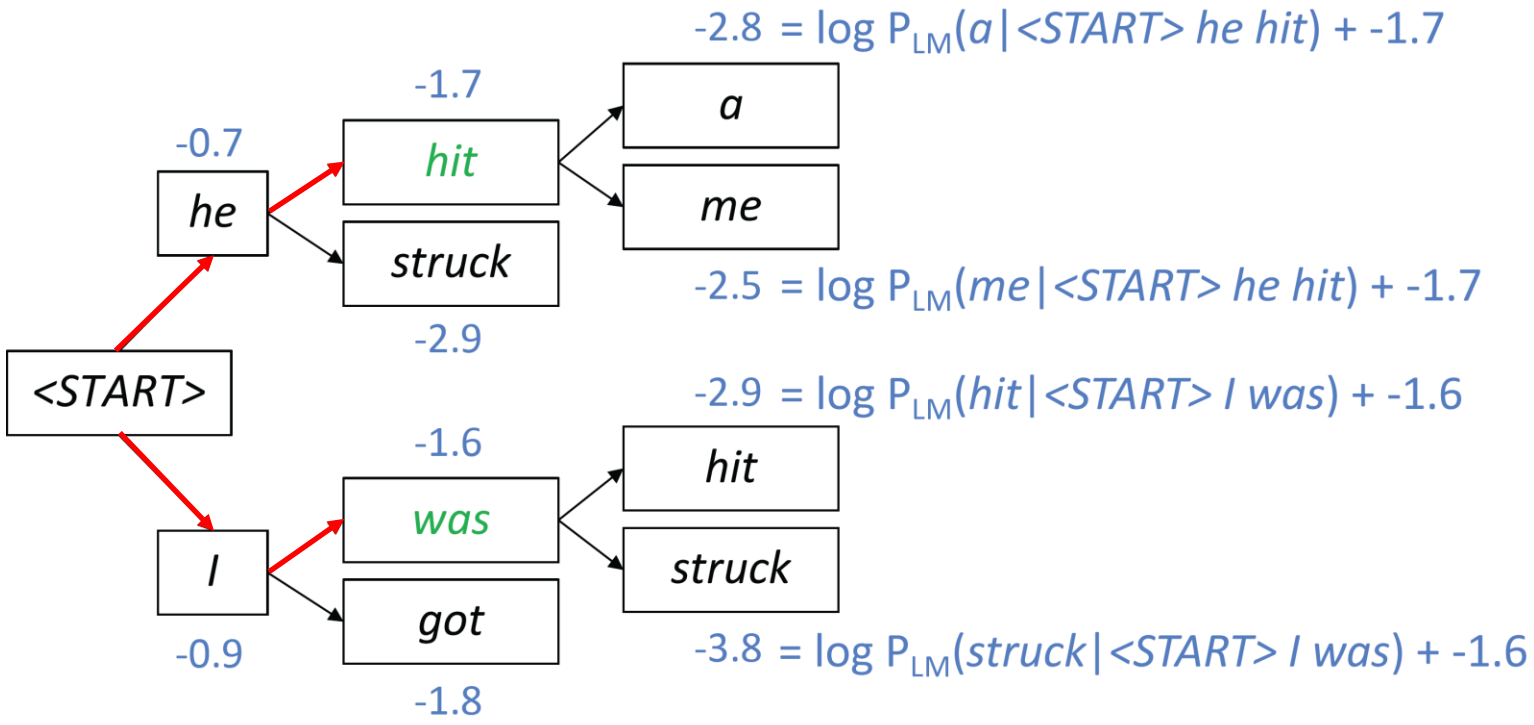
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



Of these k^2 hypotheses, just keep k with highest scores

Beam search decoding: example

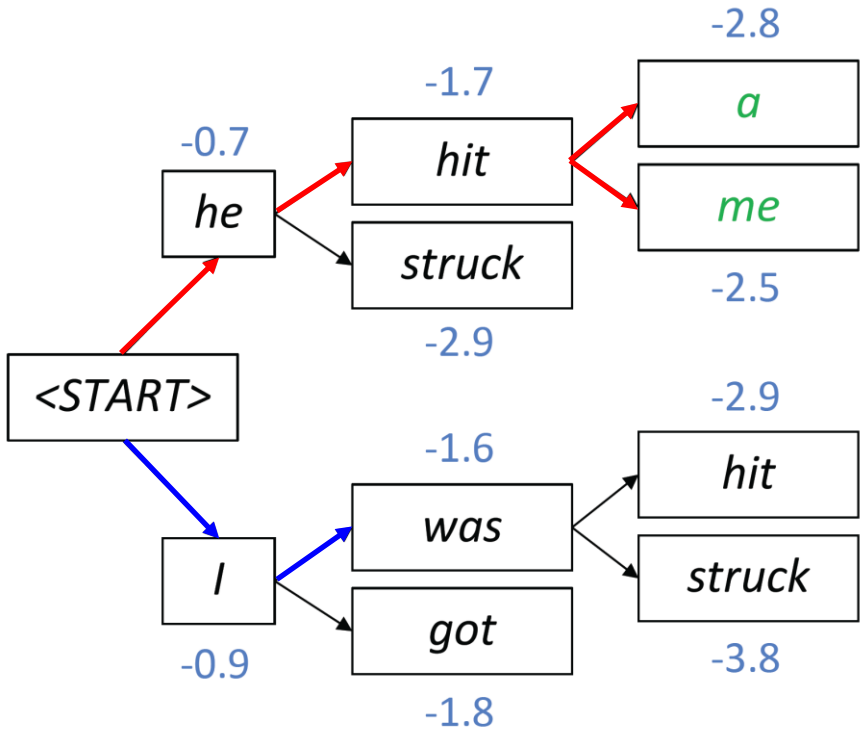
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



For each of the *k* hypotheses, find top *k* next words and calculate scores

Beam search decoding: example

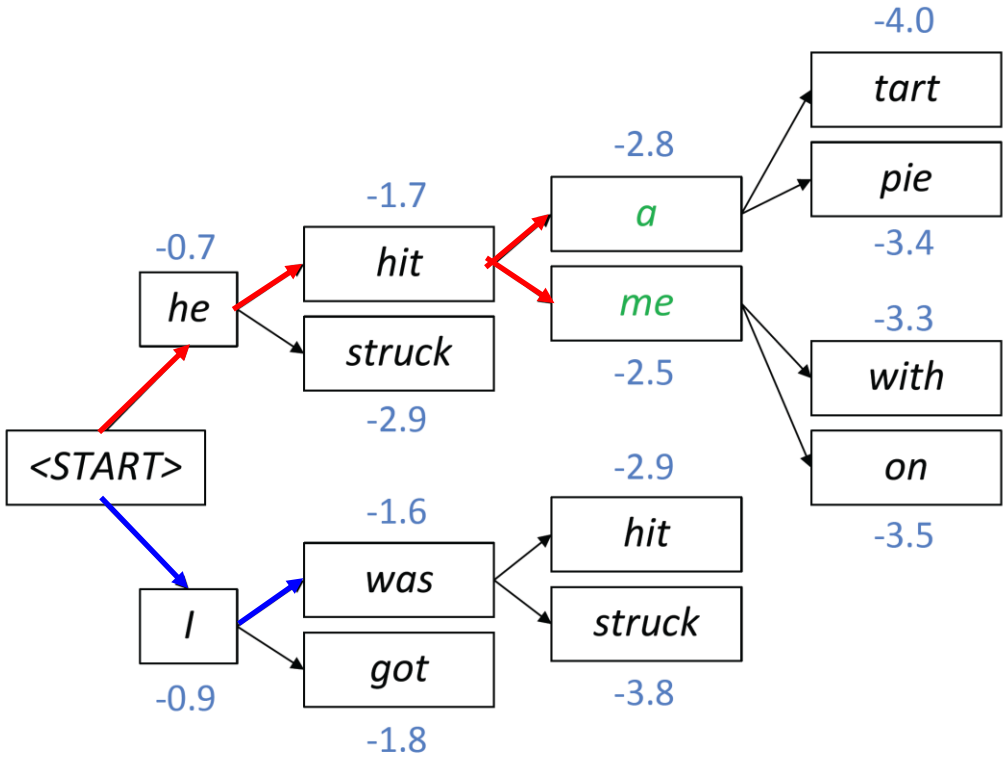
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



Of these k^2 hypotheses, just keep k with highest scores

Beam search decoding: example

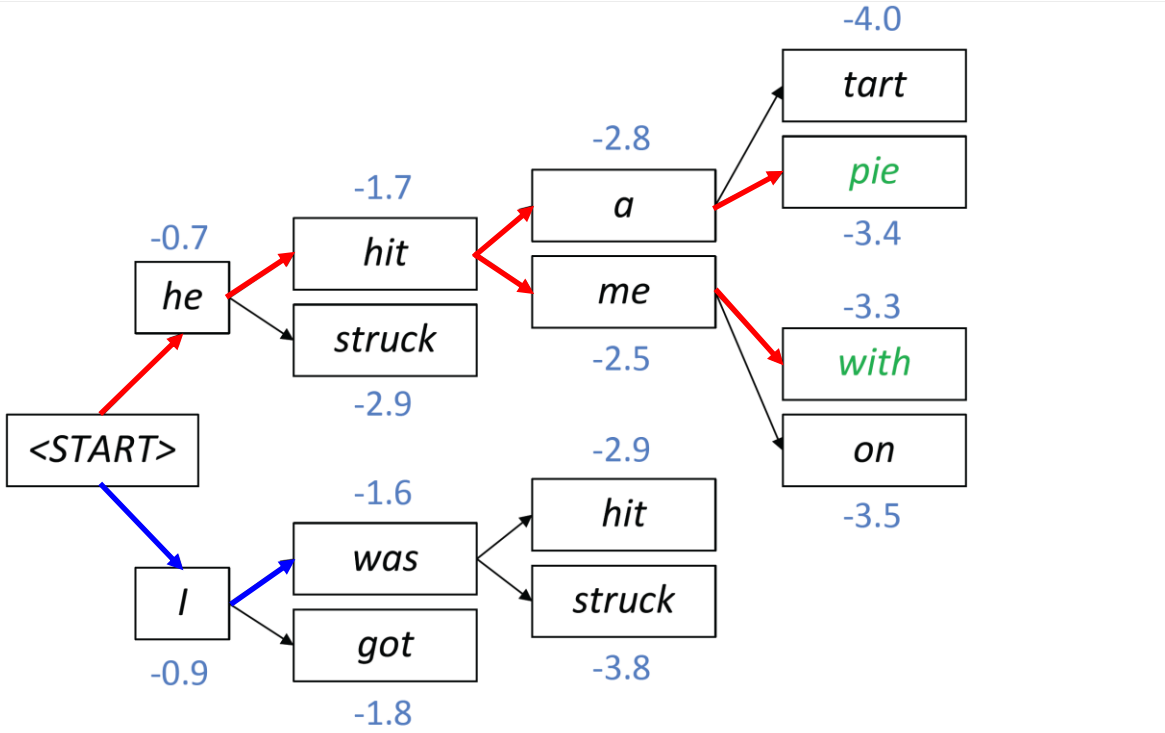
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



For each of the *k* hypotheses, find top *k* next words and calculate scores

Beam search decoding: example

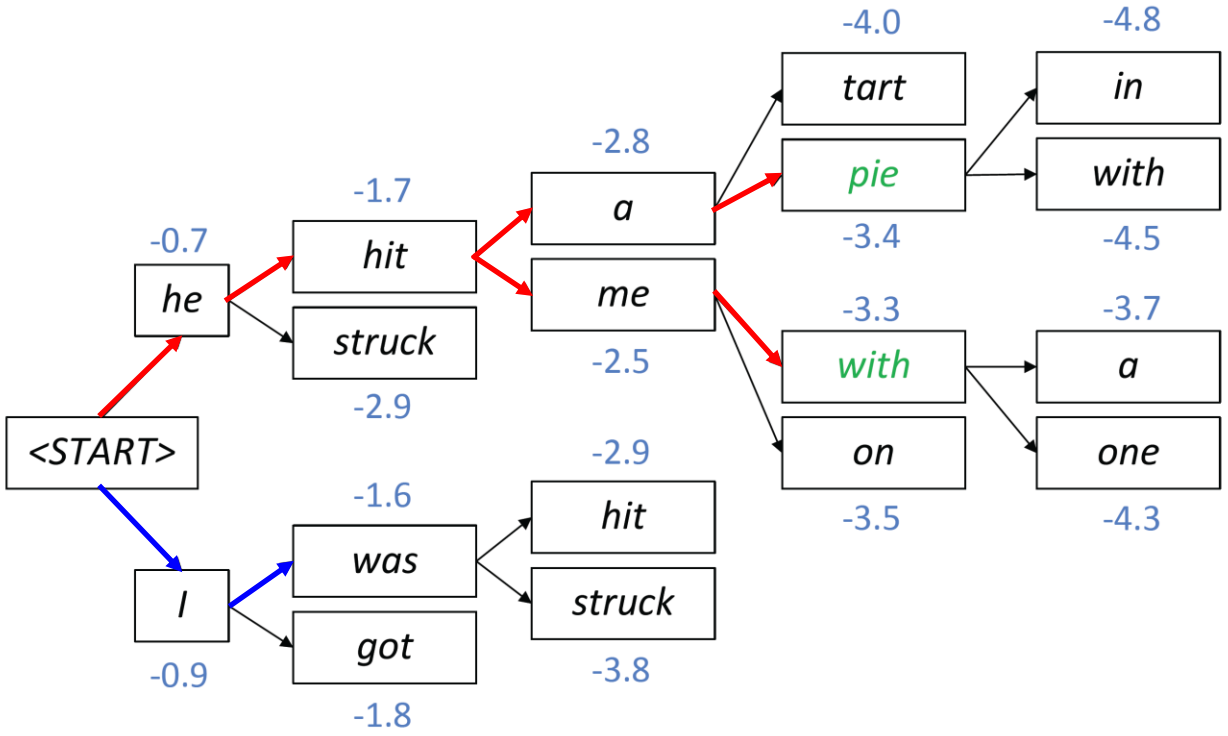
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



Of these k^2 hypotheses, just keep k with highest scores

Beam search decoding: example

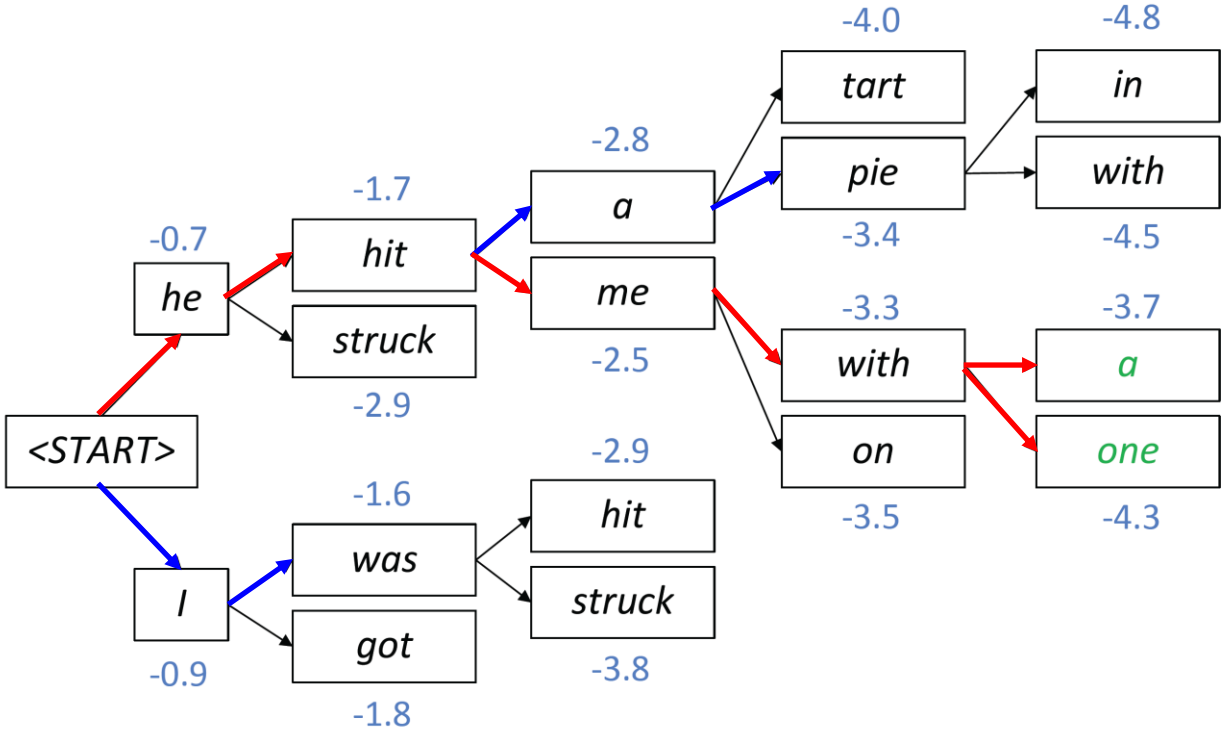
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



For each of the *k* hypotheses, find top *k* next words and calculate scores

Beam search decoding: example

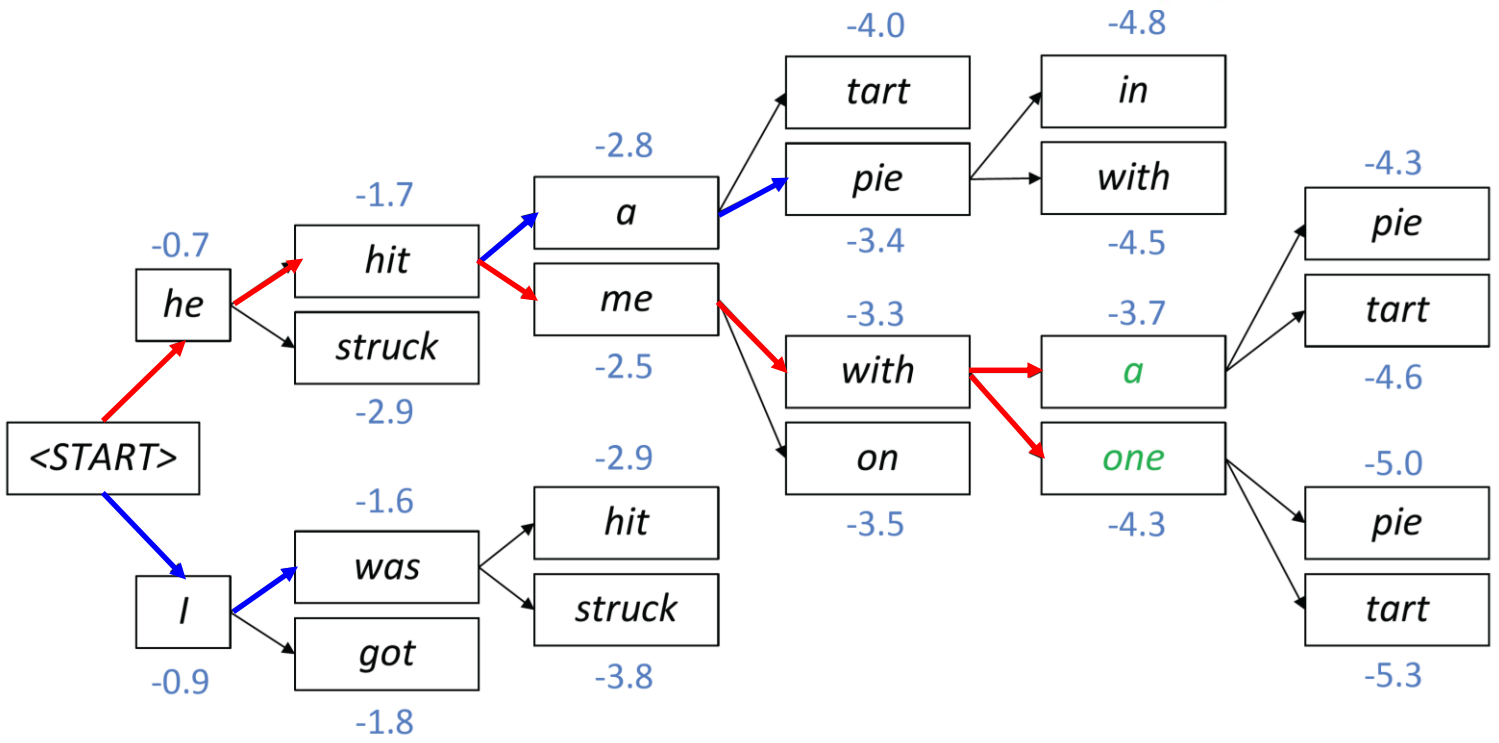
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



Of these k^2 hypotheses, just keep k with highest scores

Beam search decoding: example

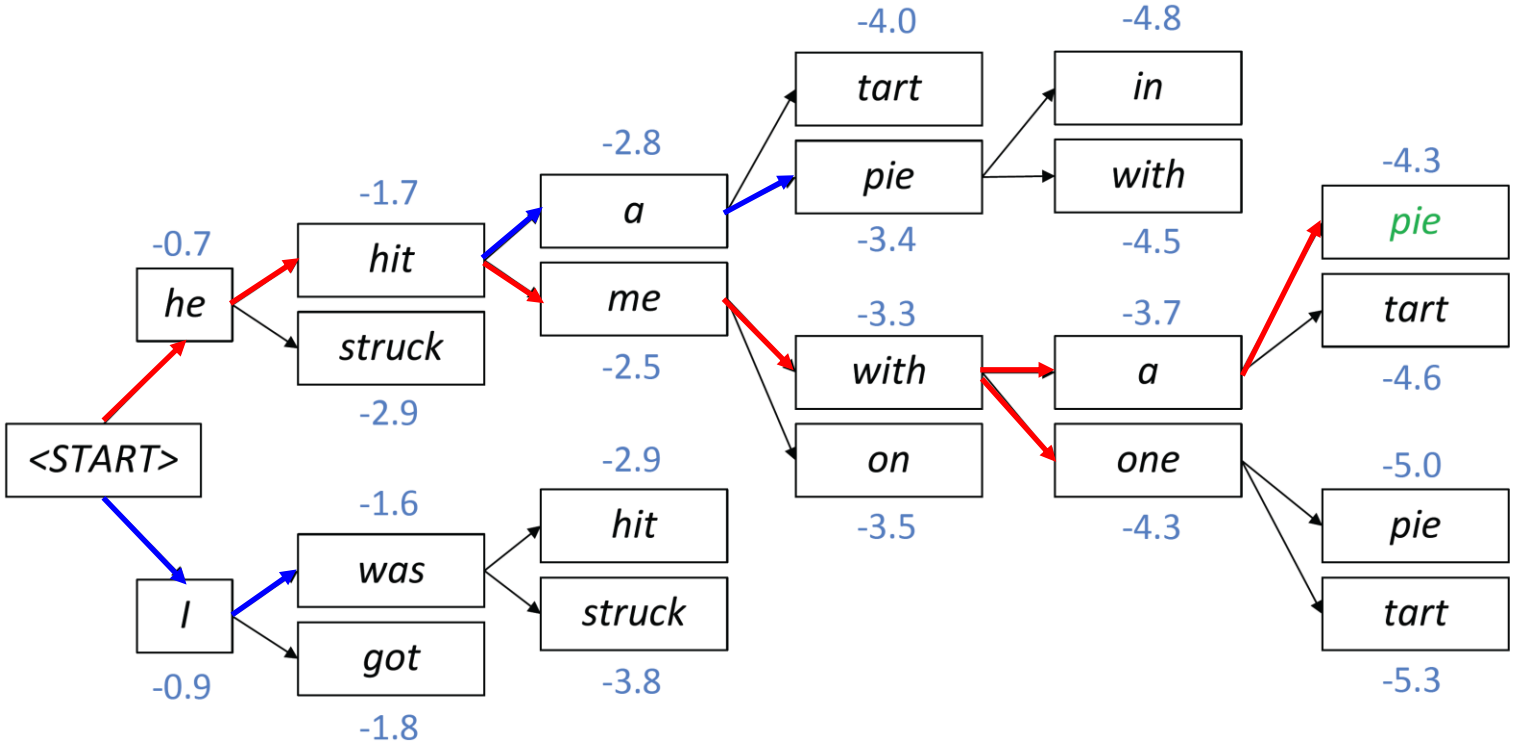
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



For each of the *k* hypotheses, find top *k* next words and calculate scores

Beam search decoding: example

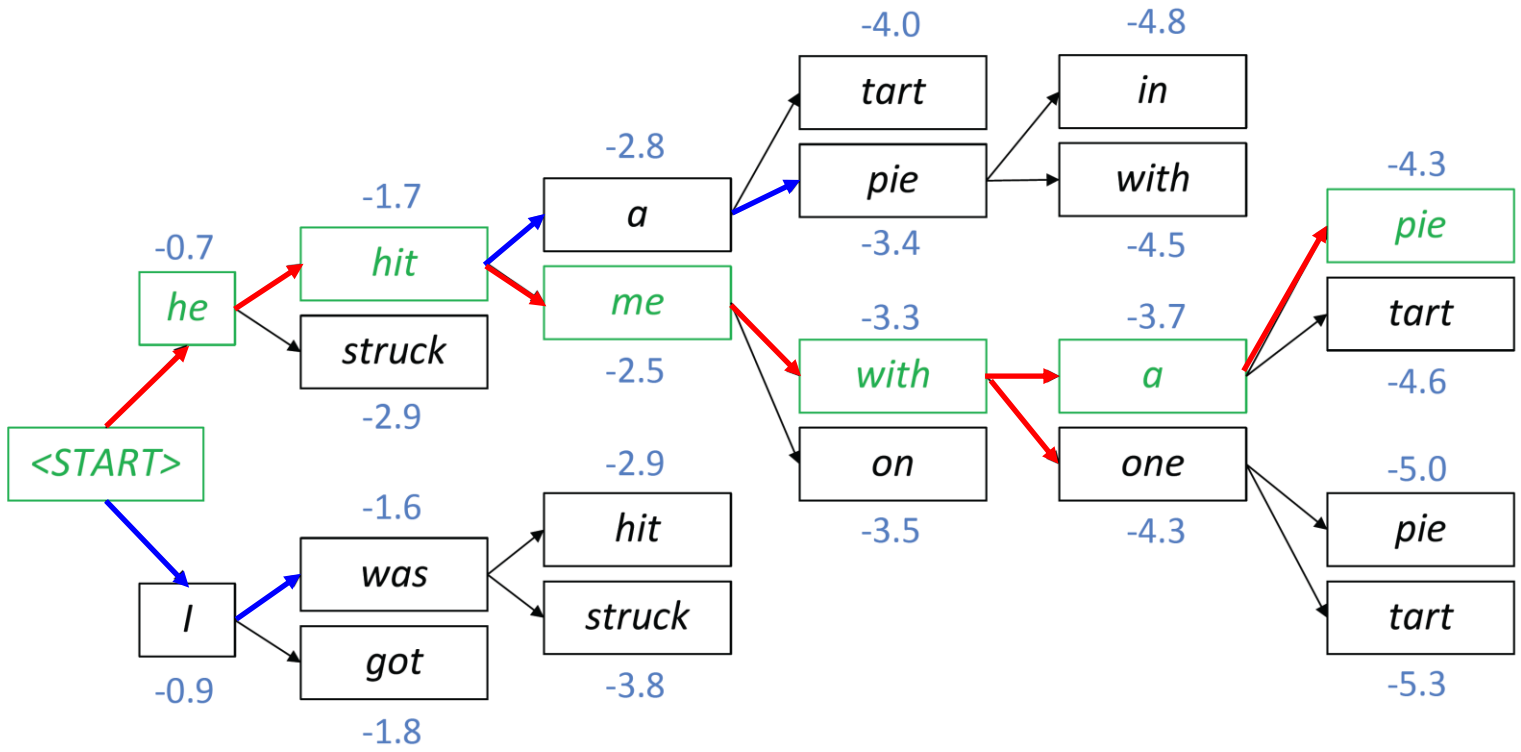
Beam size: k = 2. Blue numbers: $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



This is the top-scoring hypothesis!

Beam search decoding: example

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



Backtrack to obtain the full hypothesis

Beam search decoding: stopping criterion

- In greedy decoding, decode until the model produces an <END> token
  - **For example:** *<START> he hit me with a pie <END>*
- In beam search decoding, different hypotheses may produce <END> tokens at different time steps
  - When a hypothesis produces <END>, that hypothesis is complete.
  - Place it aside and reduce the size of the beam by one.
  - Continue exploring other hypotheses via beam search.

Beam search decoding: finishing up

- We have our list of completed hypotheses.
- How to select top one with highest score?
- Each hypothesis $y_1, \ldots, y_t$ on our list has a score

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

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

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

Advantages of NMT

Compared to SMT, NMT has many advantages:

- Better performance
  - More fluent
  - Better use of context
- 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

Disadvantages of NMT?

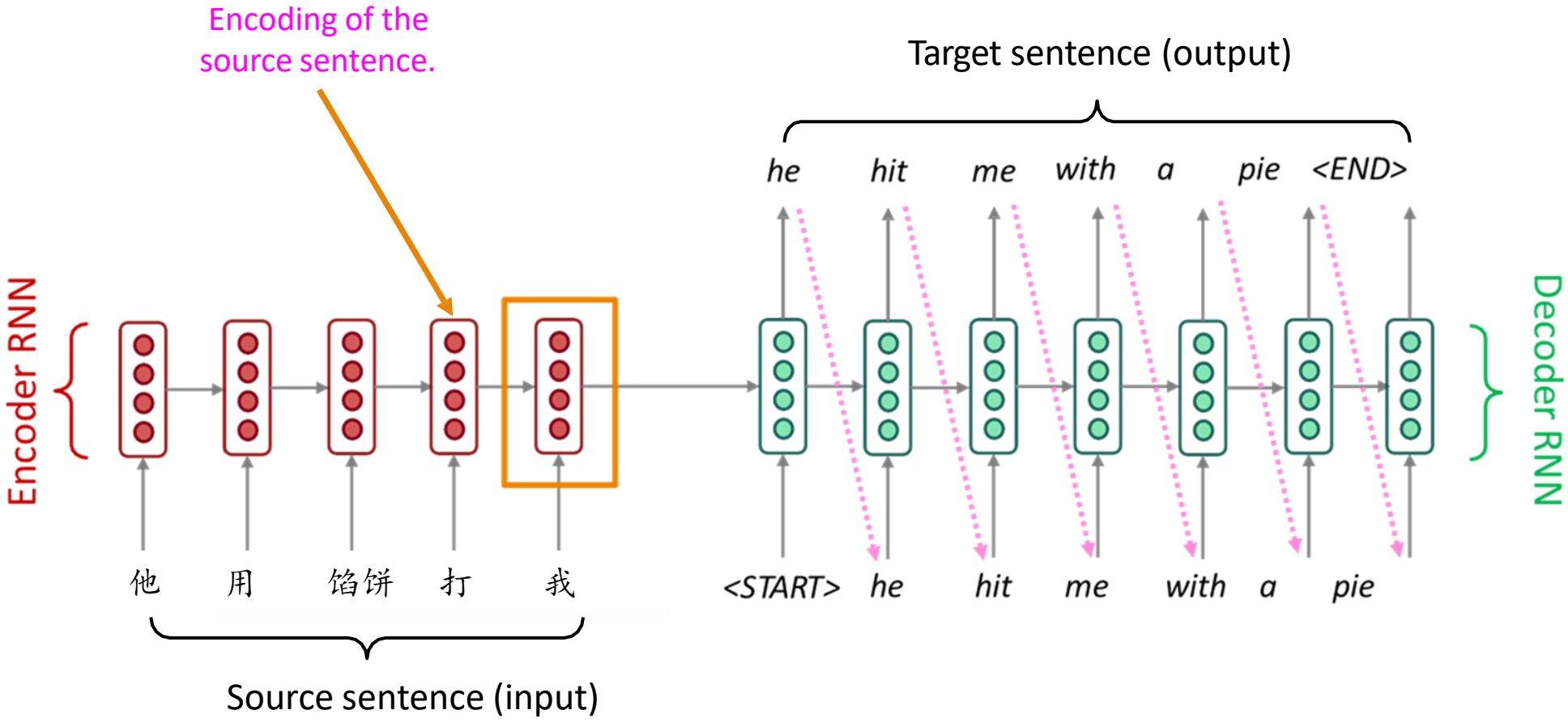
Compared to SMT:

- NMT is less interpretable (可解释性差)
  - Hard to debug
- NMT is difficult to control (不可控)
  - For example, can't easily specify rules or guidelines for translation
  - Safety concerns!

Lecture 10.5 Attention

注意力

Sequence-to-sequence: the bottleneck problem



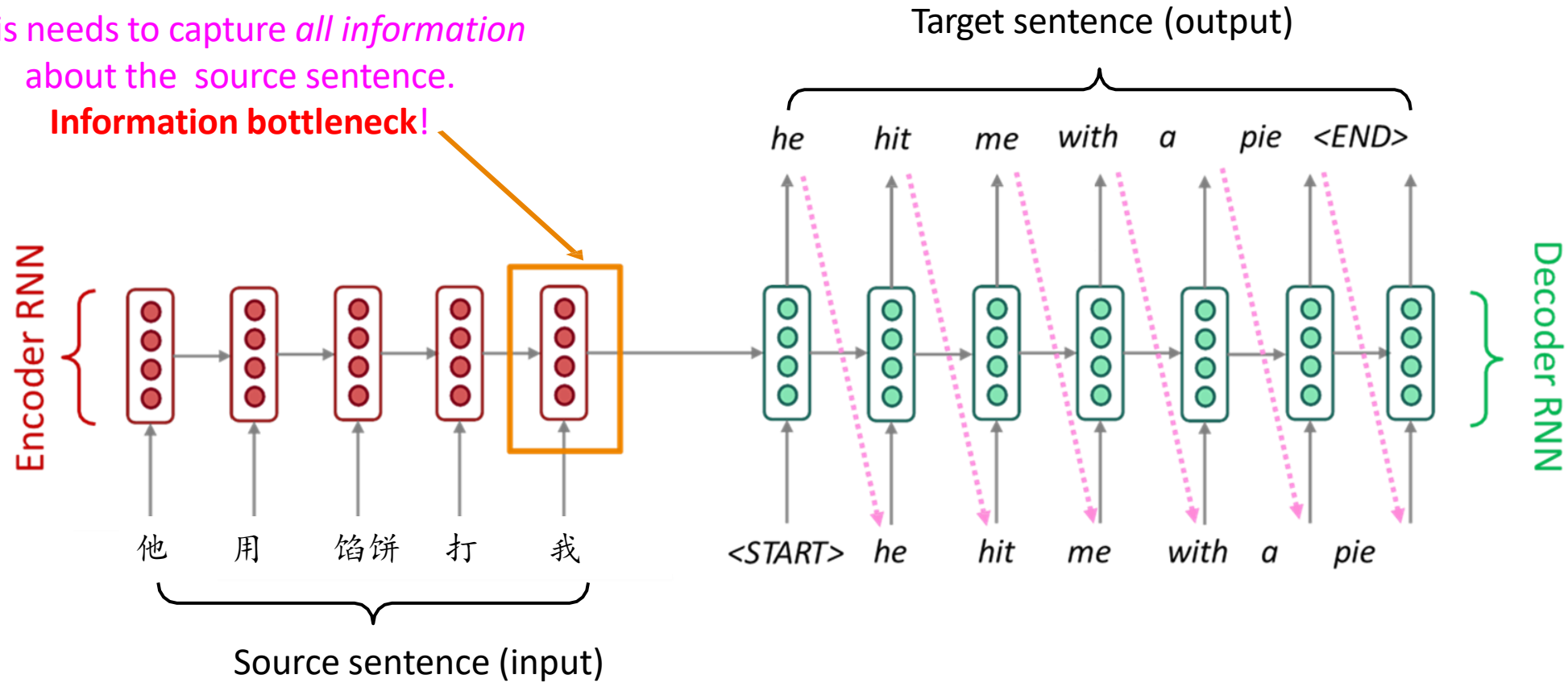
Problems with this architecture?

Sequence-to-sequence: the bottleneck problem

Encoding of the source sentence.

This needs to capture *all information* about the source sentence.

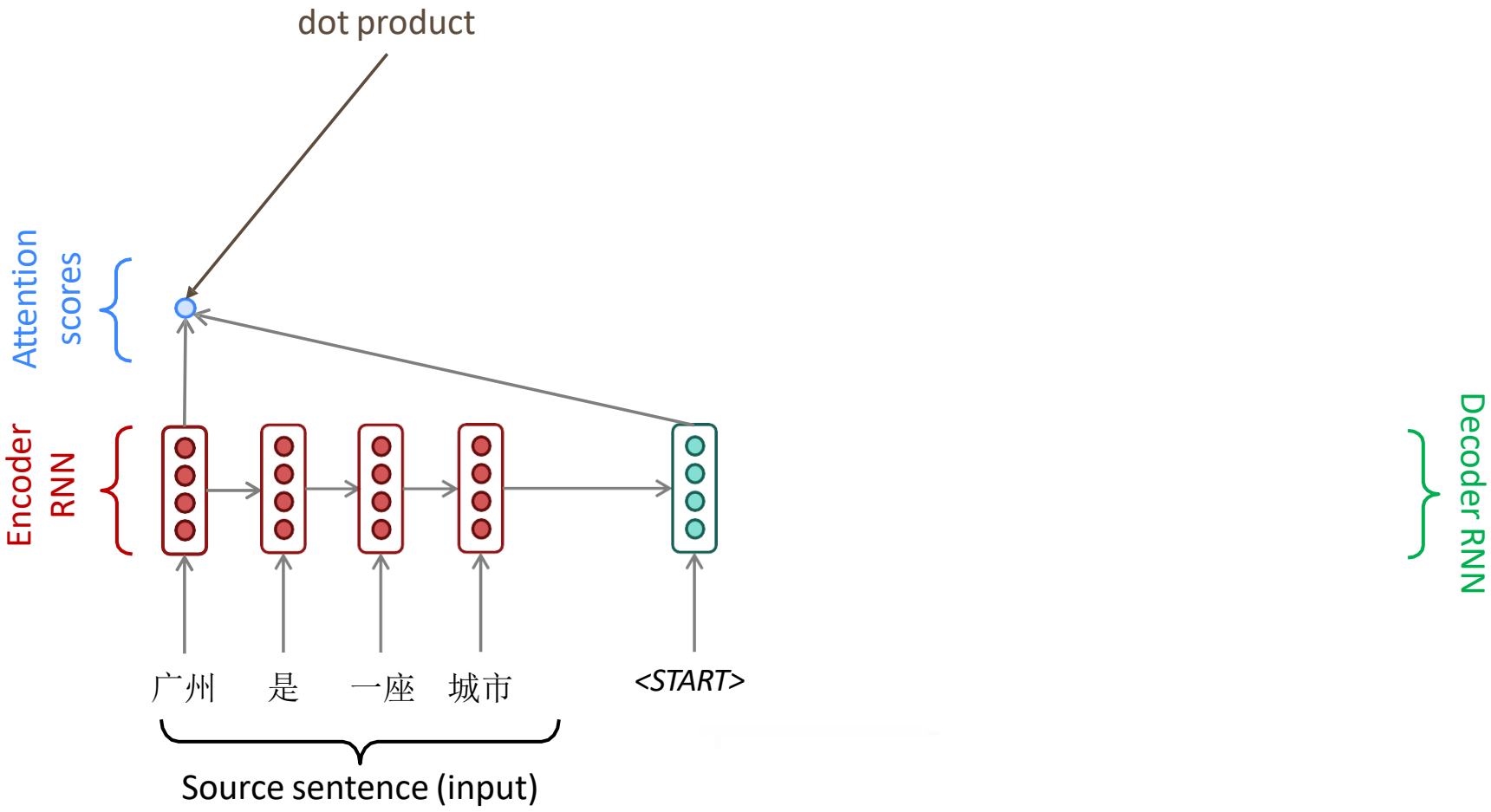
Information bottleneck!



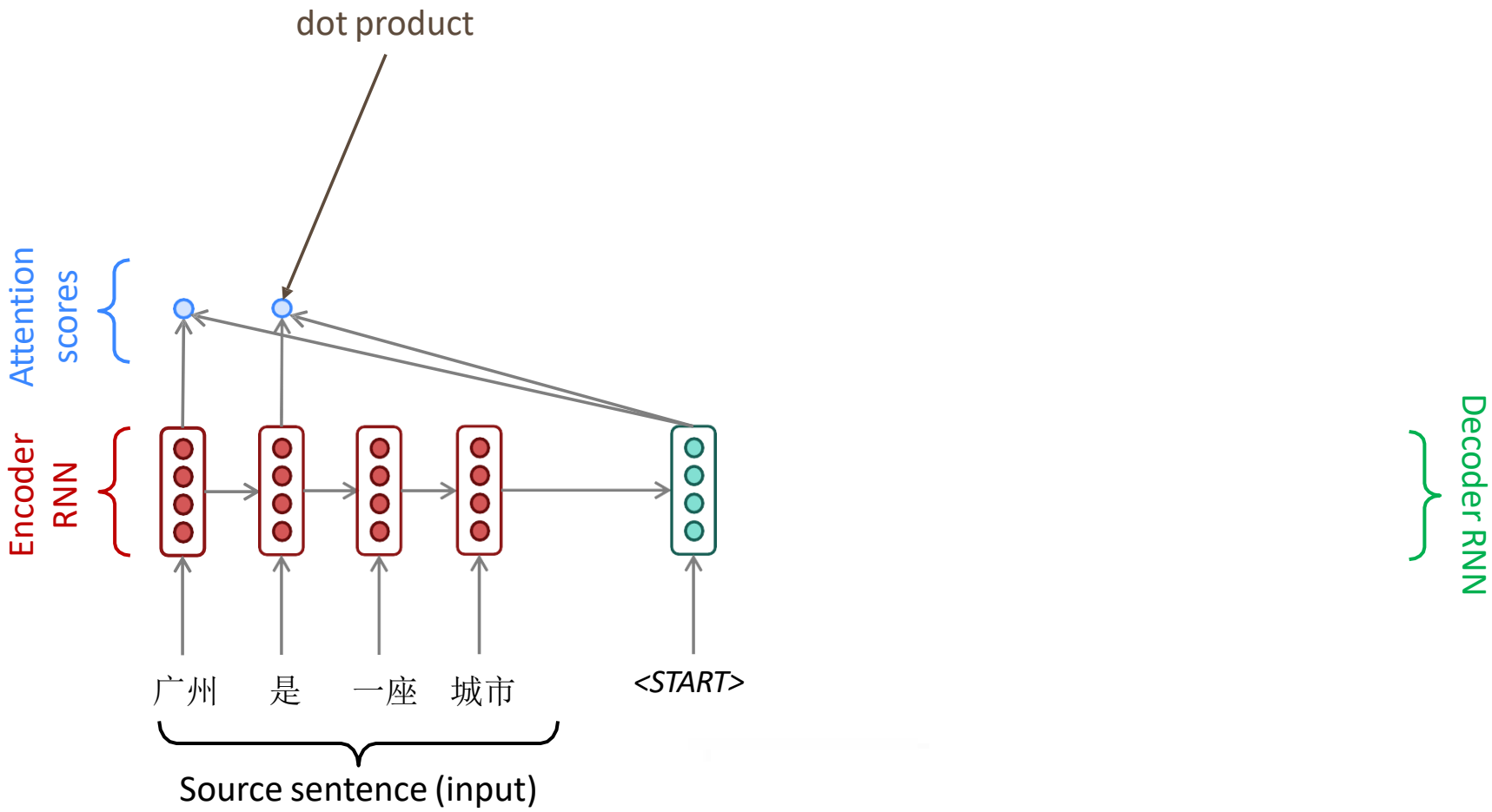
Attention 注意力

- **Attention** provides a solution to the bottleneck problem.
- Core idea: at each step of the decoder, use *direct connection to the encoder* to *focus on a particular part* of the source sequence
- First, we will show via diagram, then we will show with equations

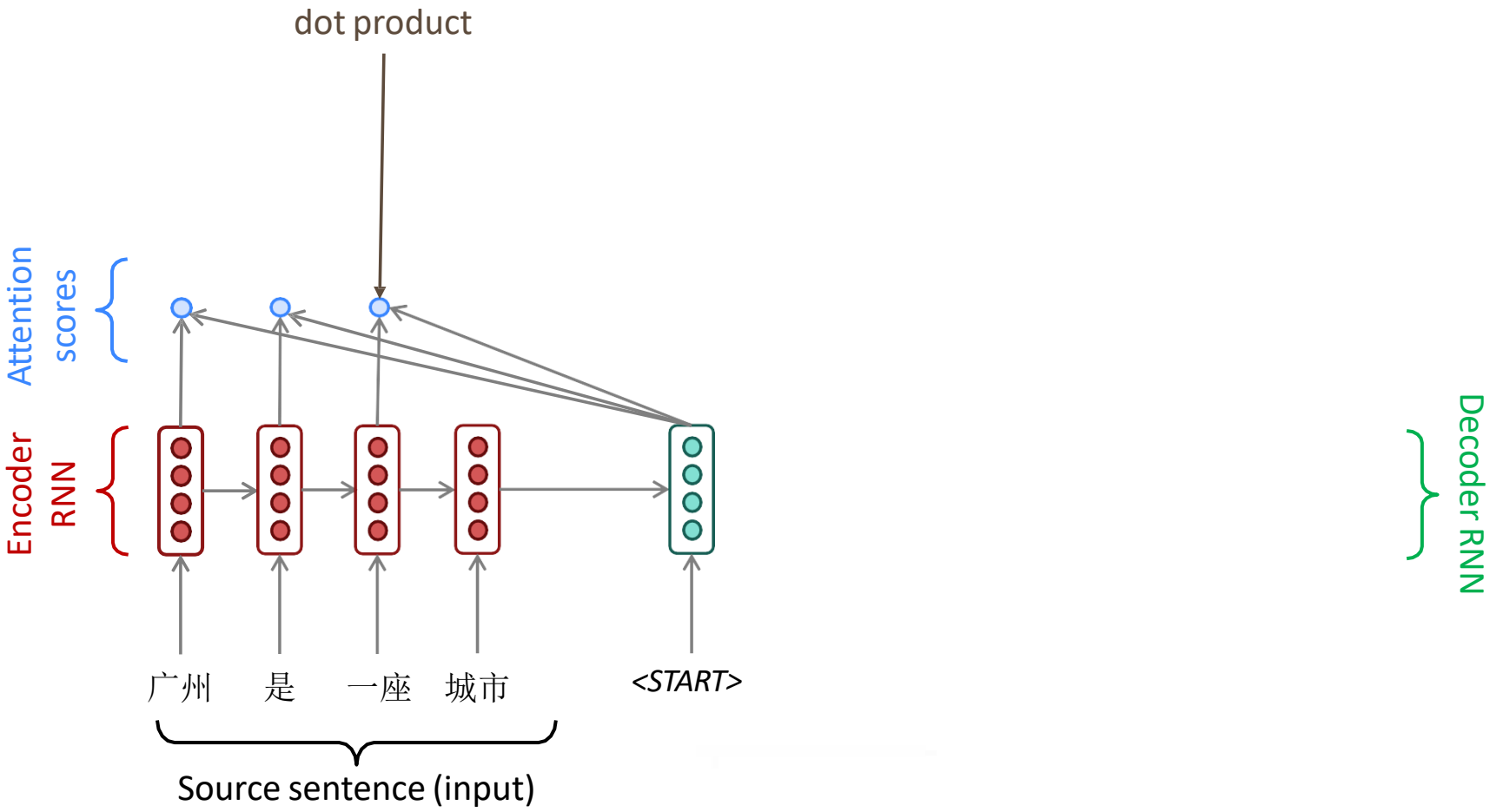
Sequence-to-sequence with attention



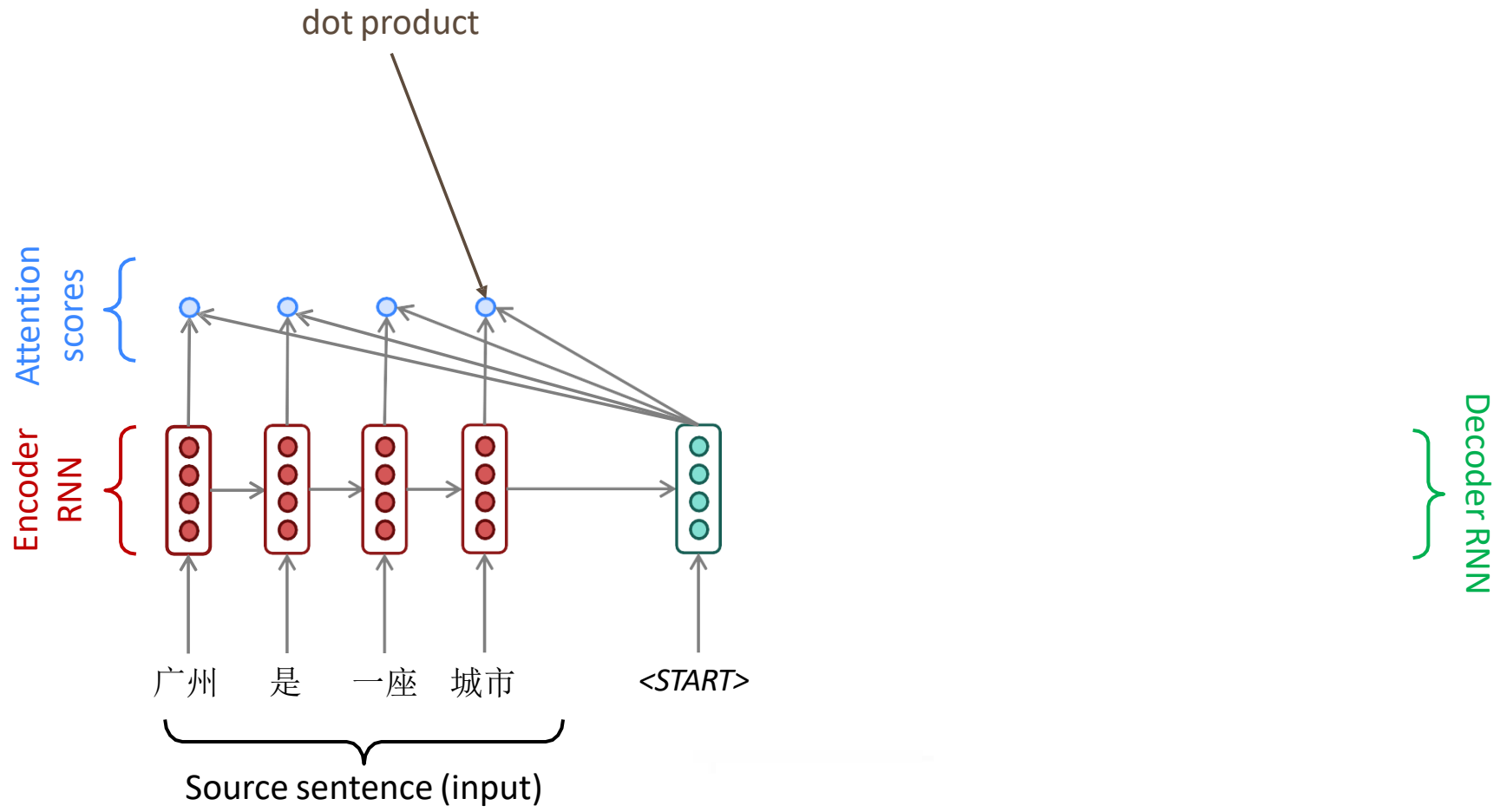
Sequence-to-sequence with attention



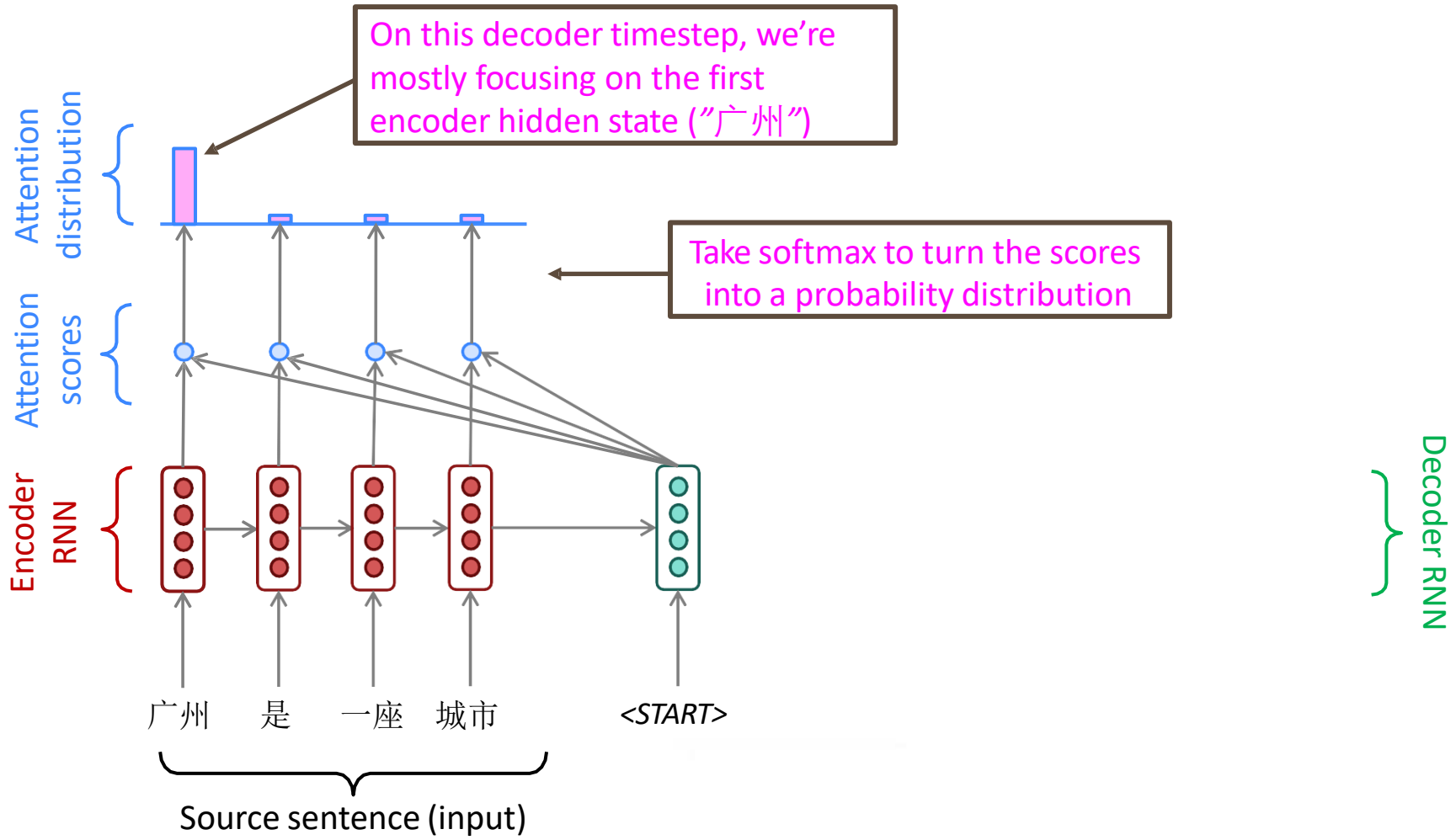
Sequence-to-sequence with attention



Sequence-to-sequence with attention



Sequence-to-sequence with attention



Reminder: softmax: a generalization of sigmoid

- For a vector *z* of dimensionality *k*, the softmax is:

$$\text{softmax}(z) = \left[\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)}, \ldots, \frac{\exp(z_k)}{\sum_{i=1}^{k} \exp(z_i)} \right]$$

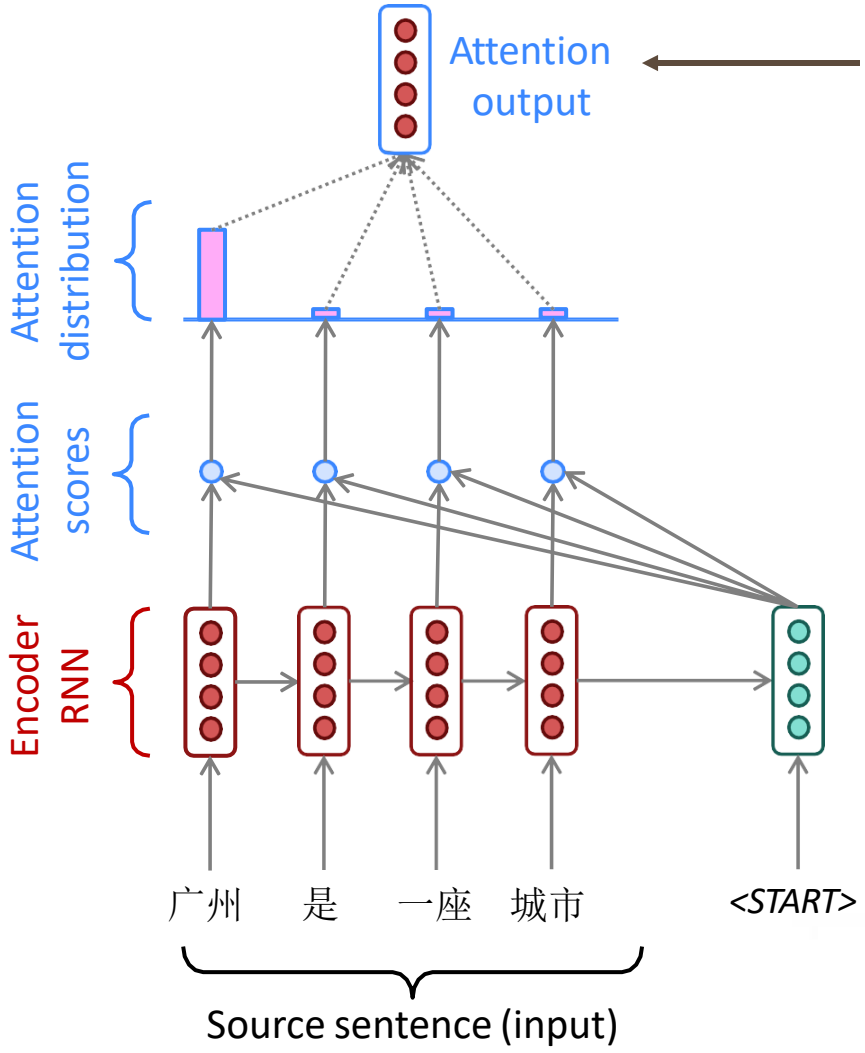
- Example:

$$z = [0.6, 1.1, -1.5, 1.2, 3.2, -1.1]$$

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad 1 \leq i \leq k$$

$$\text{softmax}(z) = [0.055, 0.090, 0.006, 0.099, 0.74, 0.010]$$

Sequence-to-sequence with attention

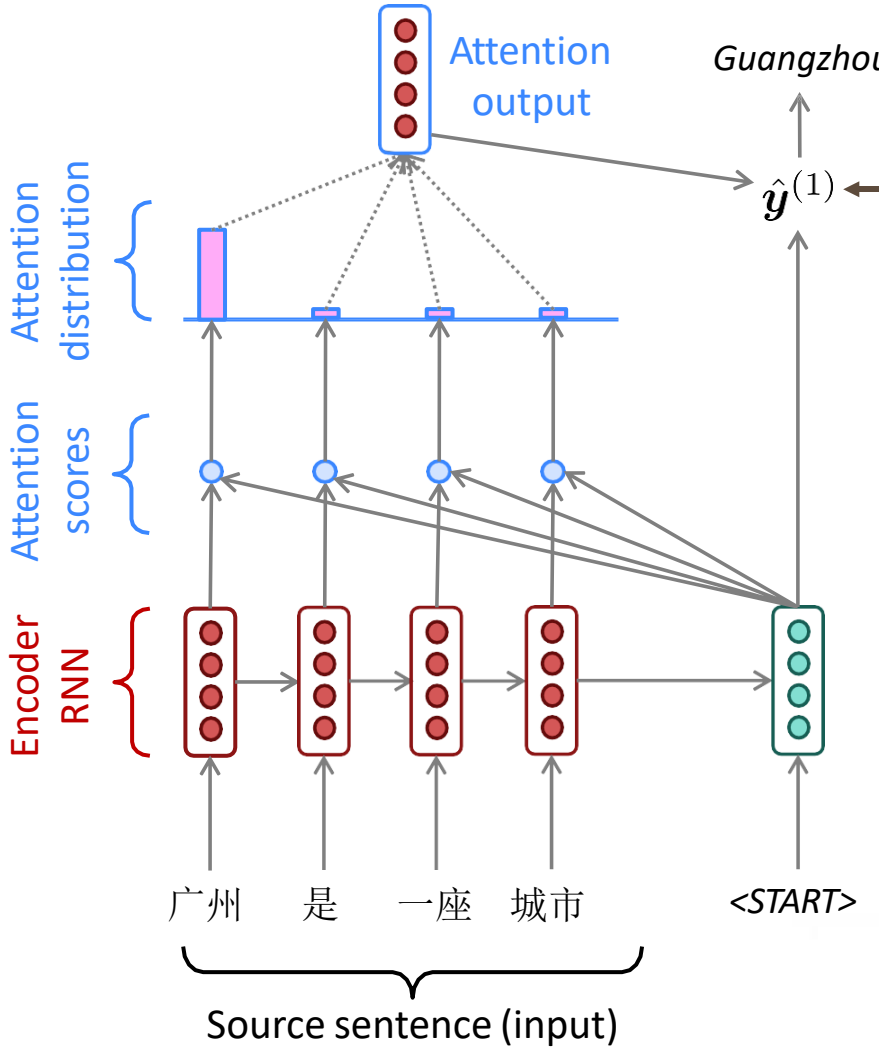


Use the attention distribution to take a **weighted sum** of the encoder hidden states.

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

Decoder RNN

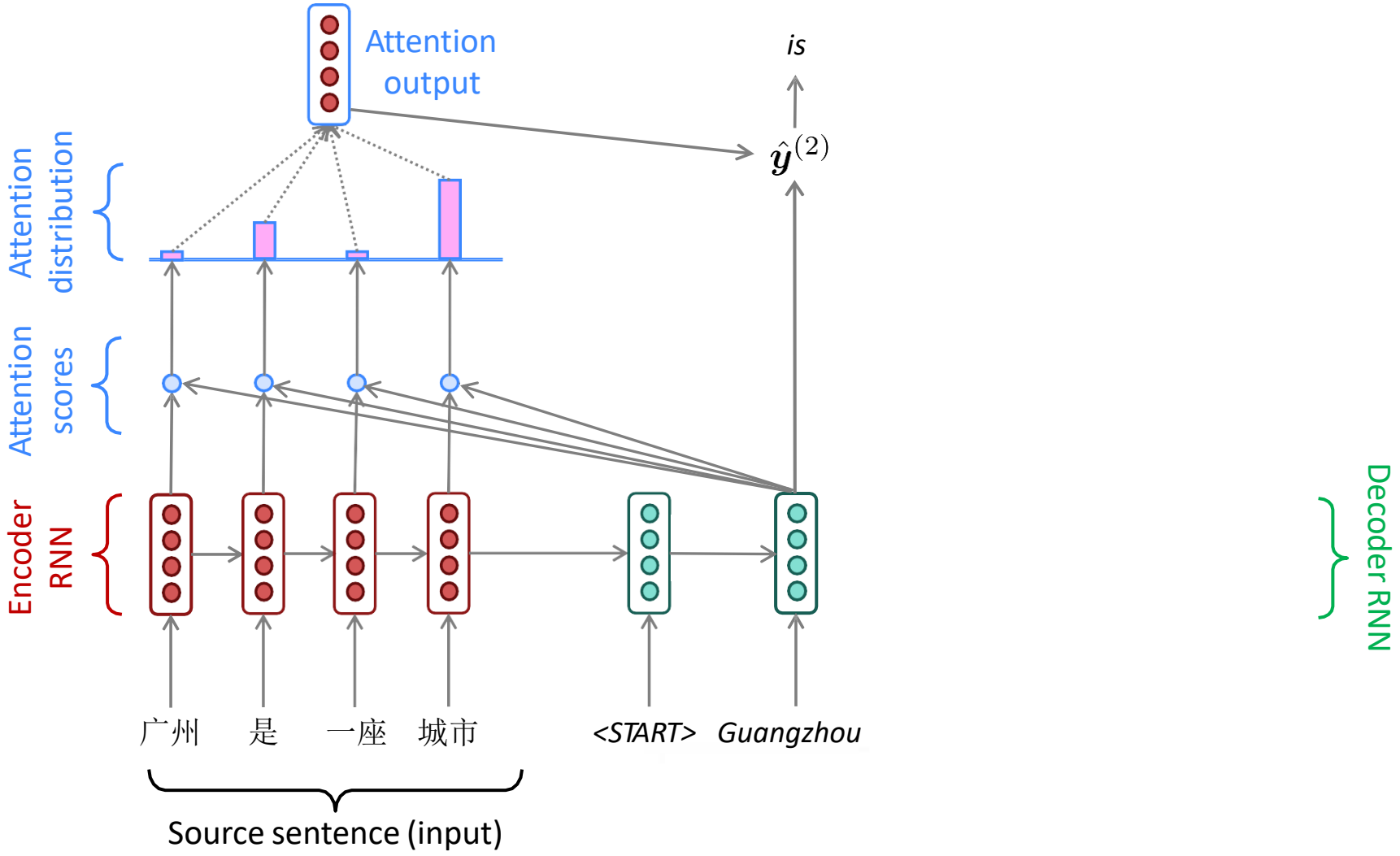
Sequence-to-sequence with attention



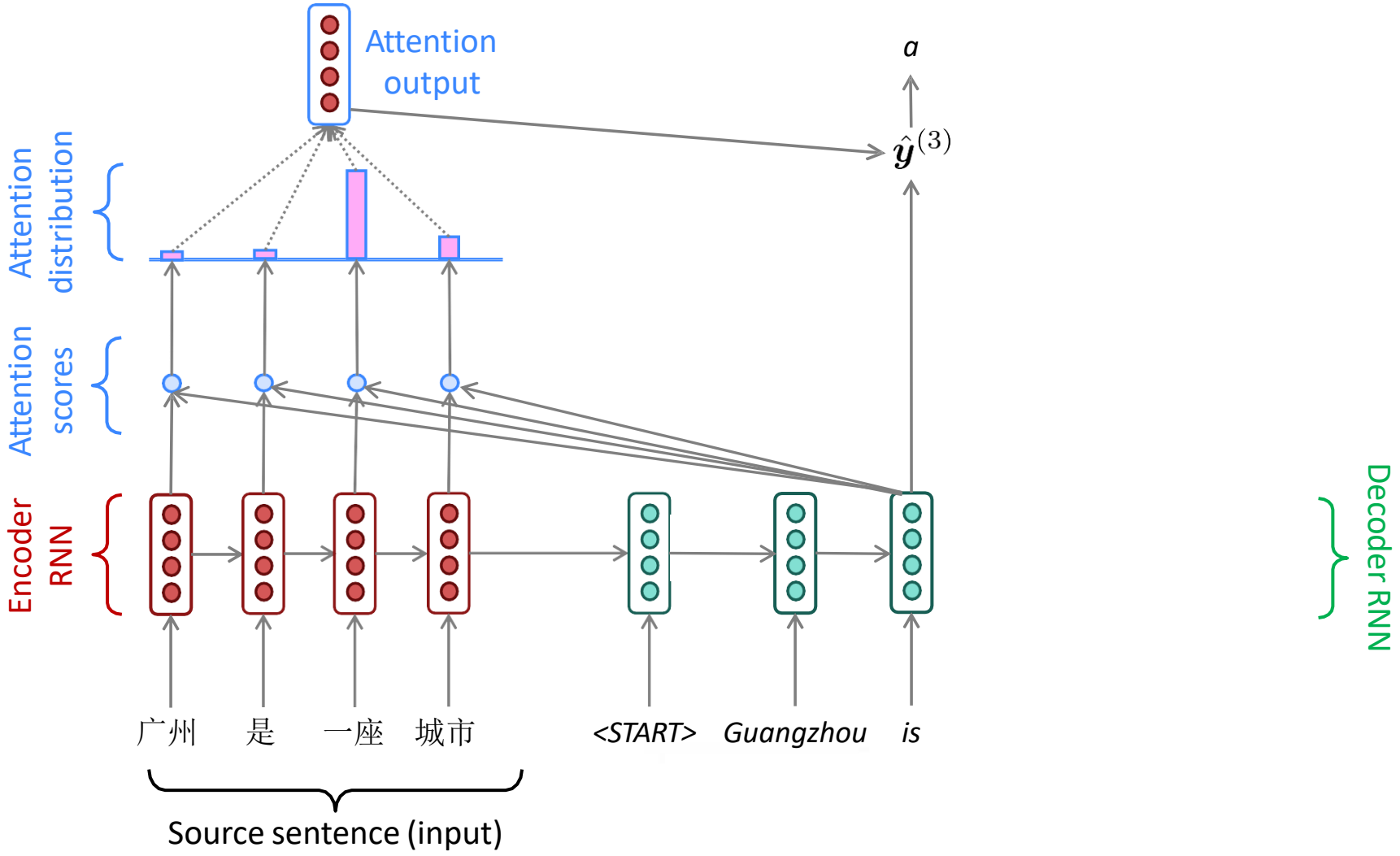
Concatenate attention output with decoder hidden state, then use to compute $\hat{y}^{(1)}$ as before

Decoder RNN

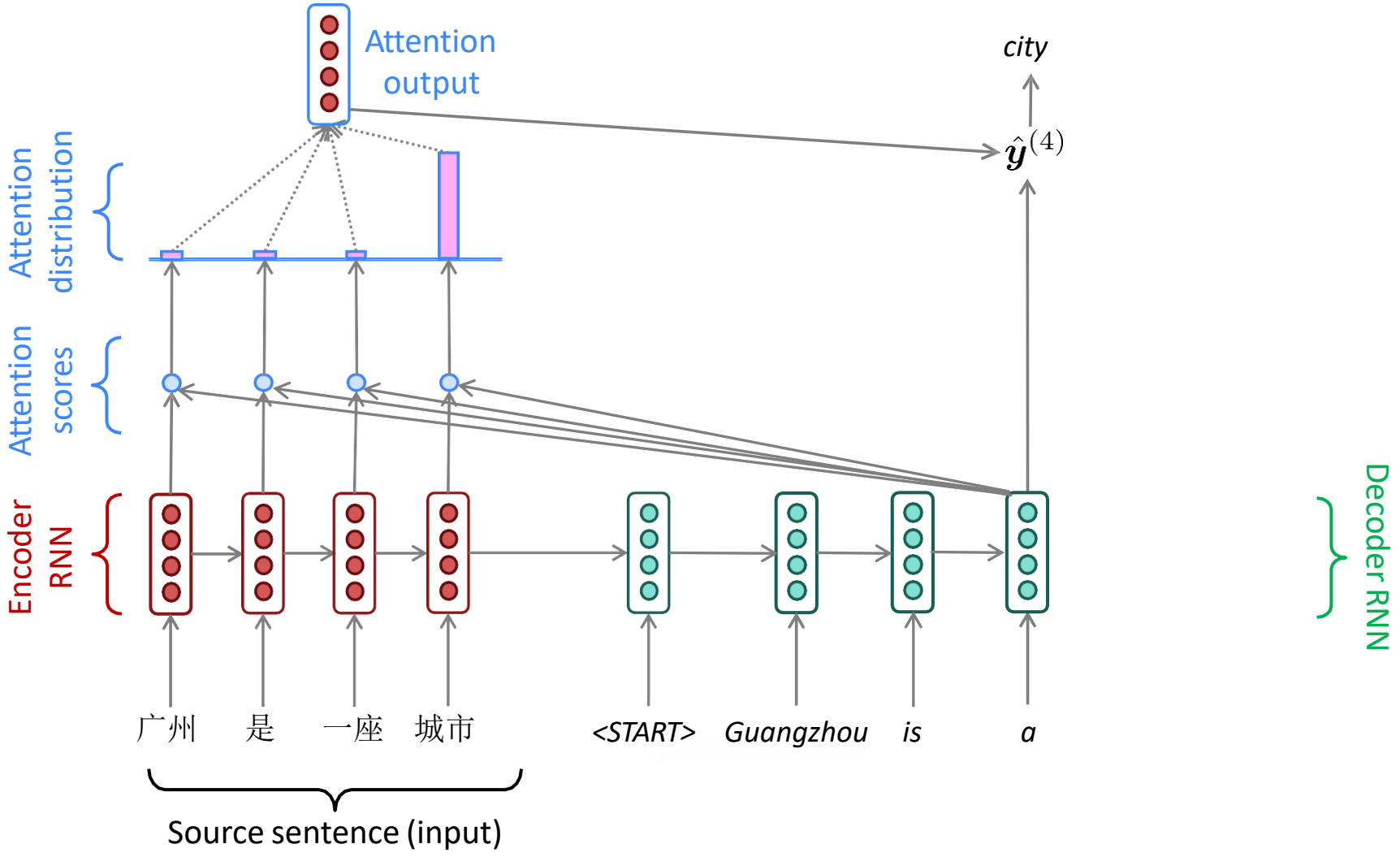
Sequence-to-sequence with attention



Sequence-to-sequence with attention



Sequence-to-sequence with attention



Attention: in equations

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- At time step *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores 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)

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

Attention: in equations

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

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

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state $[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$ and proceed as in the non-attention seq2seq model

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

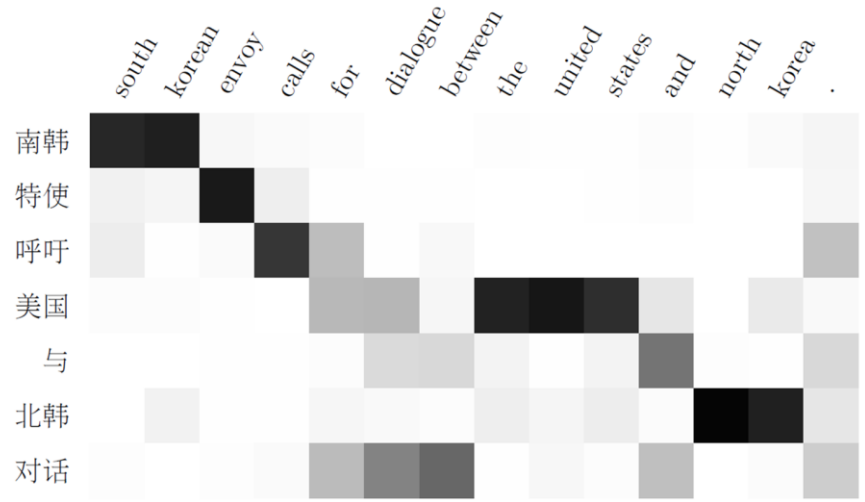
There are different ways to compute attention for machine translation!

Attention is great

- 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 is great

- Attention provides some interpretability (可解释性)
  - By inspecting attention distribution, we can see what the decoder was focusing on
  - This is cool because we never explicitly trained an alignment system
  - The network just learned alignment by itself



Attention is a *general* Deep Learning technique

- We've seen that attention is a great way to improve the sequence-to-sequence model for Machine Translation.
- However: You can use attention in many architectures (not just seq2seq) and many tasks (not just MT)
- **More general definition of attention:**
  - Given a set of vector *values*, and a vector *query*, attention is a technique to compute a weighted sum of the *values*, dependent on the *query*.
- We sometimes say that the query *attends to* the values.

There are *several* attention variants

- We have some *values* $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and a *query* $\boldsymbol{s} \in \mathbb{R}^{d_2}$
- Attention always involves:

There are multiple ways to do this

1. Computing the *attention scores* $\boldsymbol{e} \in \mathbb{R}^N$
2. Taking softmax to get *attention distribution* α :

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

3. Using attention distribution to take weighted sum of values:

$$\boldsymbol{a} = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^{d_1}$$

thus obtaining the *attention output* $\boldsymbol{a}$ (sometimes called the *context vector*)

Attention variants

- Several ways can compute $\boldsymbol{e} \in \mathbb{R}^N$ from $\boldsymbol{s} \in \mathbb{R}^{d_2}$ and $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$:
  - Basic dot-product attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$
    - Note: this assumes $d_1 = d_2$
    - This is the version we saw earlier
  - Multiplicative attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$
    - Where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d_1}$ is a weight matrix
  - Additive attention: $\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$
    - Where $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times d_1}, \boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times d_2}$ are weight matrices and $\boldsymbol{v} \in \mathbb{R}^{d_3}$ is a vector
    - d_3 (the attention dimensionality) is a hyperparameter

Summary

- We learned some history of Machine Translation (MT)
- Since 2014, Neural MT rapidly replaced intricate Statistical MT
- Sequence-to-sequence is the architecture for NMT (uses 2 models: encoder and decoder)
- Attention is a way to *focus on particular parts* of the input
  - Improves sequence-to-sequence a lot!

Lecture 10.6
Evaluating MT: BLEU

Evaluating MT: Using human evaluators

- **Fluency**: How intelligible, clear, readable, or natural in the target language is the translation?
- **Fidelity (保真度)**: Does the translation have the same meaning as the source?
  - **Adequacy**: Does the translation convey the same information as source?
    - Bilingual judges given source and target language, assign a score
  - **Informativeness**: Does the translation convey enough information as the source to perform a task?

Automatic Evaluation of MT

- Human evaluation is expensive and very slow
- Need an evaluation metric that takes seconds, not months
- Intuition: MT is good if it looks like a human translation

1. Collect one or more human ***reference translations*** of the source.
2. Score MT output based on its similarity to the reference translations.
   - BLEU

BLEU (Bilingual Evaluation Understudy)

- "n-gram precision"
- Ratio of correct n-grams to the total number of output n-grams
  - Correct: Number of *n*-grams (unigram, bigram, etc.) the MT output shares with the reference translations.
  - Total: Number of *n*-grams in the MT result.
- The higher the precision, the better the translation
- Recall is ignored

Why recall is ignored?

- BLEU considers *multiple reference translations*, each of which may use a different word choice to translate the same source word.
- A *good* candidate translation only recalls *one* of these possible choices.
- Recalling all choices leads to a bad translation:

Example

Candidate 1: `I always invariably perpetually do.`
Candidate 2: `I always do.`

Reference 1: `I always do.`
Reference 2: `I invariably do.`
Reference 3: `I perpetually do.`

Papineni, et al. BLEU: a Method for Automatic Evaluation of Machine Translation. 2002.

Multiple Reference Translations

Reference translation 1:
The U.S. island of Guam is maintaining a high state of alert after the Guam airport and its offices both received an e-mail from someone calling himself the Saudi Arabian Osama bin Laden and threatening a biological/chemical attack against public places such as the airport .

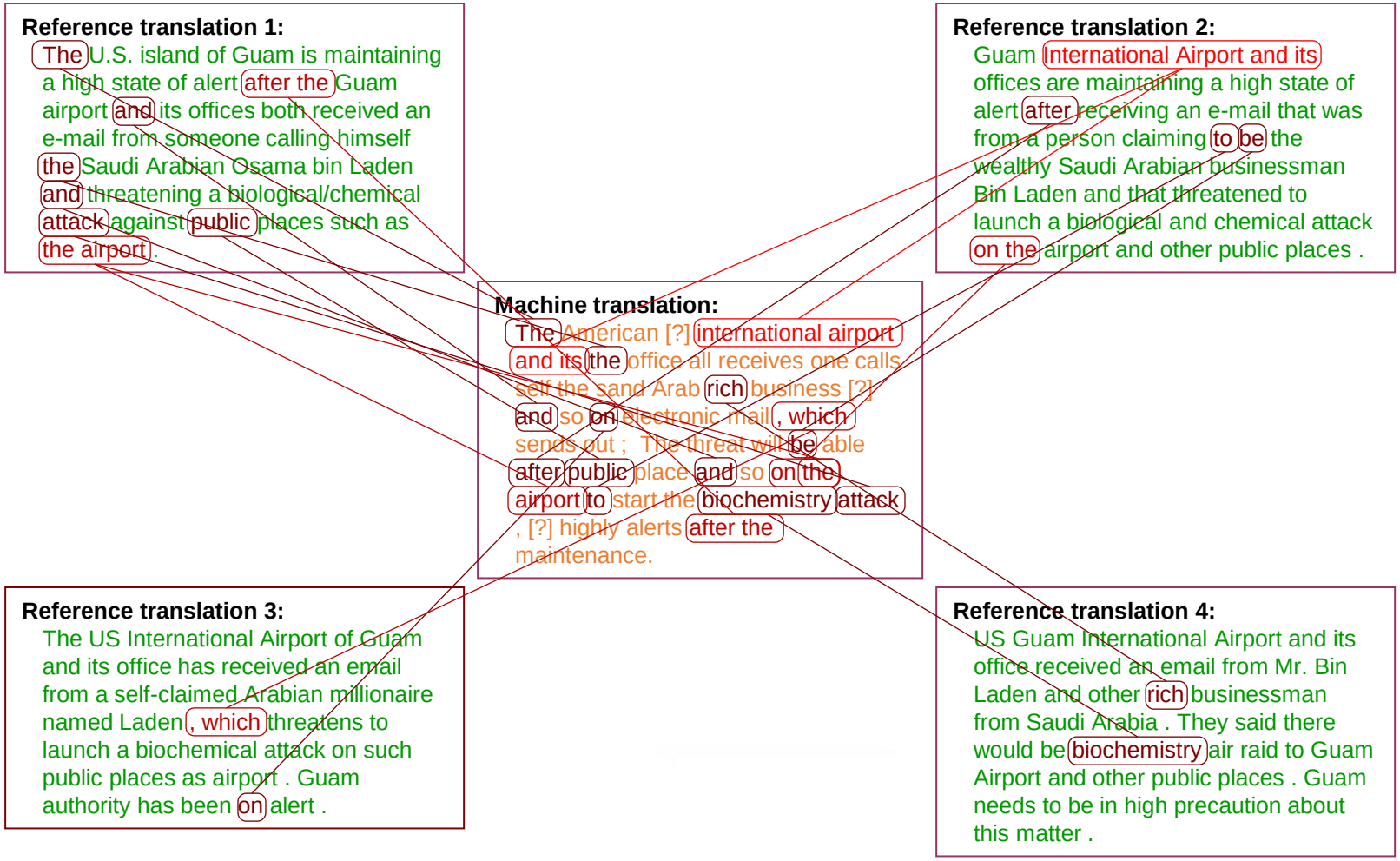
Reference translation 2:
Guam International Airport and its offices are maintaining a high state of alert after receiving an e-mail that was from a person claiming to be the wealthy Saudi Arabian businessman Bin Laden and that threatened to launch a biological and chemical attack on the airport and other public places .

Machine translation:
The American [?] international airport and its the office all receives one calls self the sand Arab rich business [?] and so on electronic mail , which sends out ; The threat will be able after public place and so on the airport to start the biochemistry attack , [?] highly alerts after the maintenance.

Reference translation 3:
The US International Airport of Guam and its office has received an email from a self-claimed Arabian millionaire named Laden , which threatens to launch a biochemical attack on such public places as airport . Guam authority has been on alert .

Reference translation 4:
US Guam International Airport and its office received an email from Mr. Bin Laden and other rich businessman from Saudi Arabia . They said there would be biochemistry air raid to Guam Airport and other public places . Guam needs to be in high precaution about this matter .

Multiple Reference Translations



Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Candidate 1 Unigram Precision: 5/6

Computing BLEU: Unigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Clip the count of each *n*-gram to the maximum count of the *n*-gram in any single reference

Candidate 2 Unigram Precision: 7/10

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Computing BLEU: Bigram precision

Cand 1: Mary no slap the witch green

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Candidate 2 Bigram Precision: 4/9

Brevity Penalty

- BLEU is precision-based: no penalty for dropping words
- Instead, we use a brevity penalty for translations that are shorter than the reference translations.

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

Computing BLEU

- Precision$_1$, precision$_2$, etc., are computed over all candidate sentences C in the test set

$$\text{precision}_n = \frac{\sum_{C \in \textit{corpus}} \sum_{\text{n-gram} \in C} \text{count-in-reference}_{\text{clip}}(\text{n-gram})}{\sum_{C \in \textit{corpus}} \sum_{\text{n-gram} \in C} \text{count}(\text{n-gram})}$$

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

- output-length: the overall length of all output sentences
- reference-length: the overall length of all reference sentences

Thank you!