Seq2Seq Modeling & Machine Translation

Zhiyuan Liu

liuzy@tsinghua.edu.cn

THUNLP

Outline

- Introduction to Seq2Seq
- Machine Translation
- Attention
- Transformer

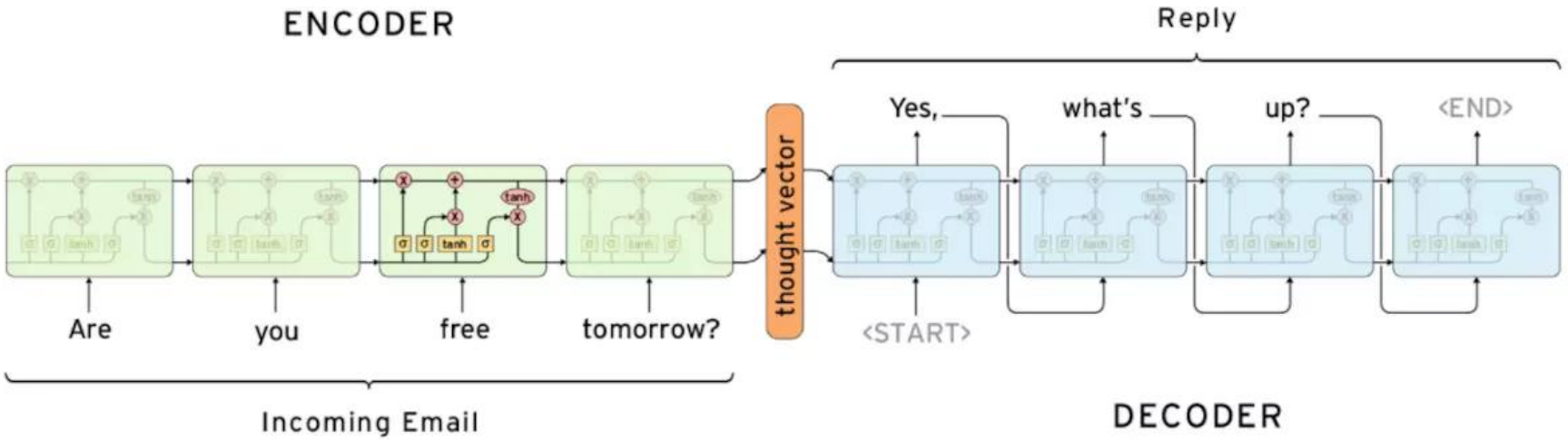
Introduction to Seq2Seq

- Sequence-to-sequence (Seq2Seq): a family of machine learning methods used for language processing

- Architecture:
  - An encoder that produces representations of the source sentence
  - A decoder which is a language model that generates target sentence conditioned on encoding
  - The encoder/decoder can be realized by RNN/GRU/LSTM/Transformer

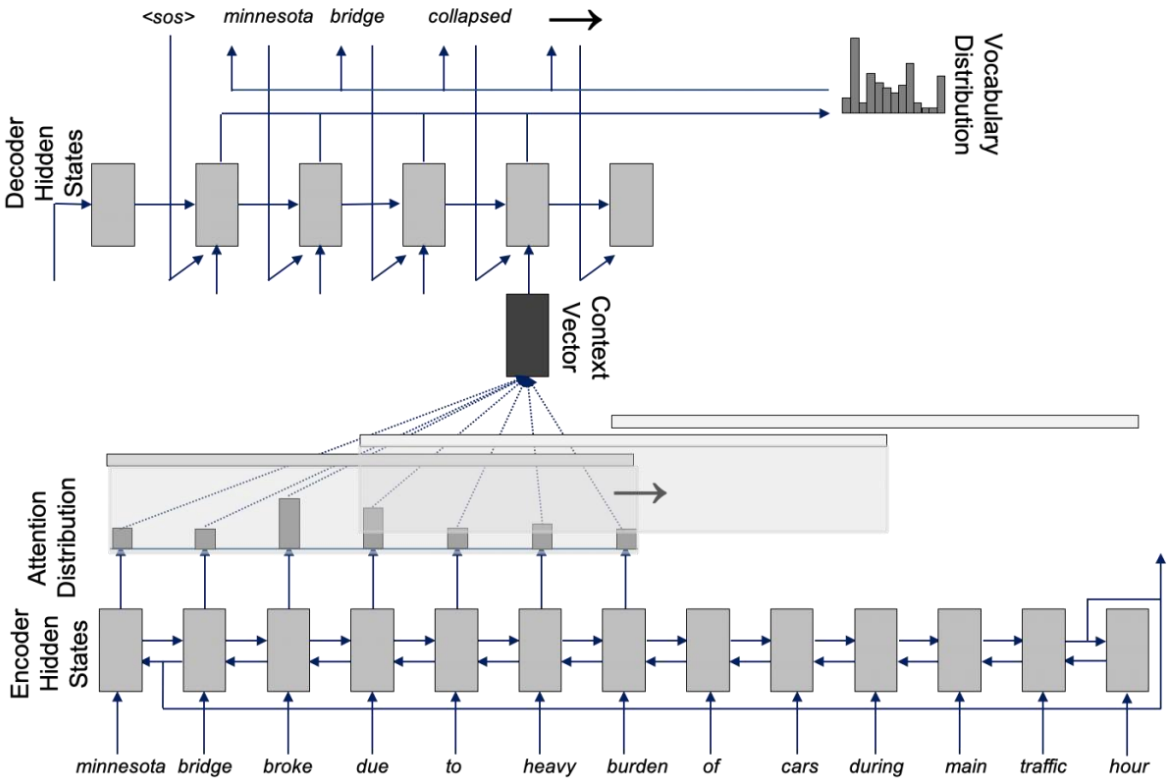
Introduction to Seq2Seq

- Typical Applications:
  - Conversational Models
    - Incoming Email -> Reply



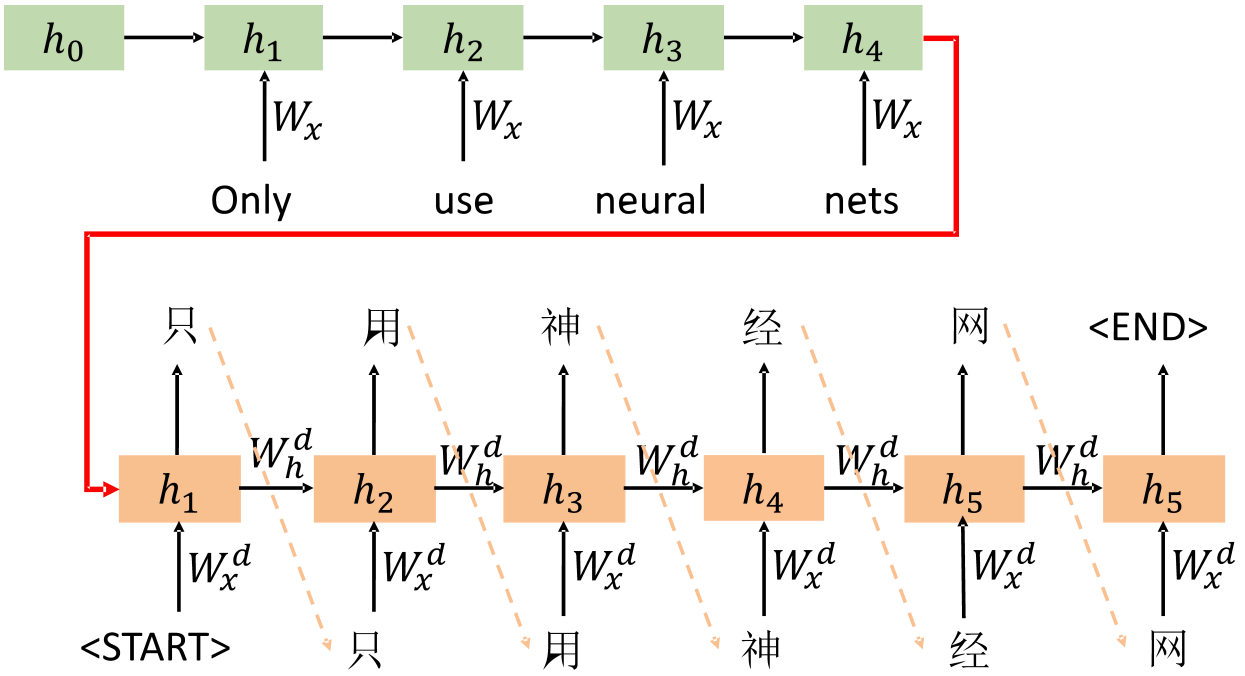
Introduction to Seq2Seq

- Typical Applications:
  - Text Summarization
    - Long text -> Short summary



Introduction to Seq2Seq

- Typical Applications:
  - Machine Translation
    - Language A -> Language B



Introduction to Seq2Seq

- Typical Applications:
  - Conversational Models
    - Incoming Email -> Reply
  - Text Summarization
    - Long text -> Short summary
  - Machine Translation
    - Language A -> Language B
    - We use machine translation as the example in this lecture.

Outline

- Introduction to Seq2Seq
- Machine Translation
  - Introduction
  - Neural Machine Translation
- Attention
- Transformer

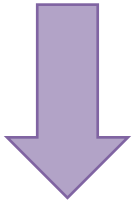
Outline

- Introduction to Seq2Seq
- Machine Translation
  - Introduction
  - Neural Machine Translation
- Attention
- Transformer

Machine Translation

- Machine Translation(MT): the task of translating text from source language to target language

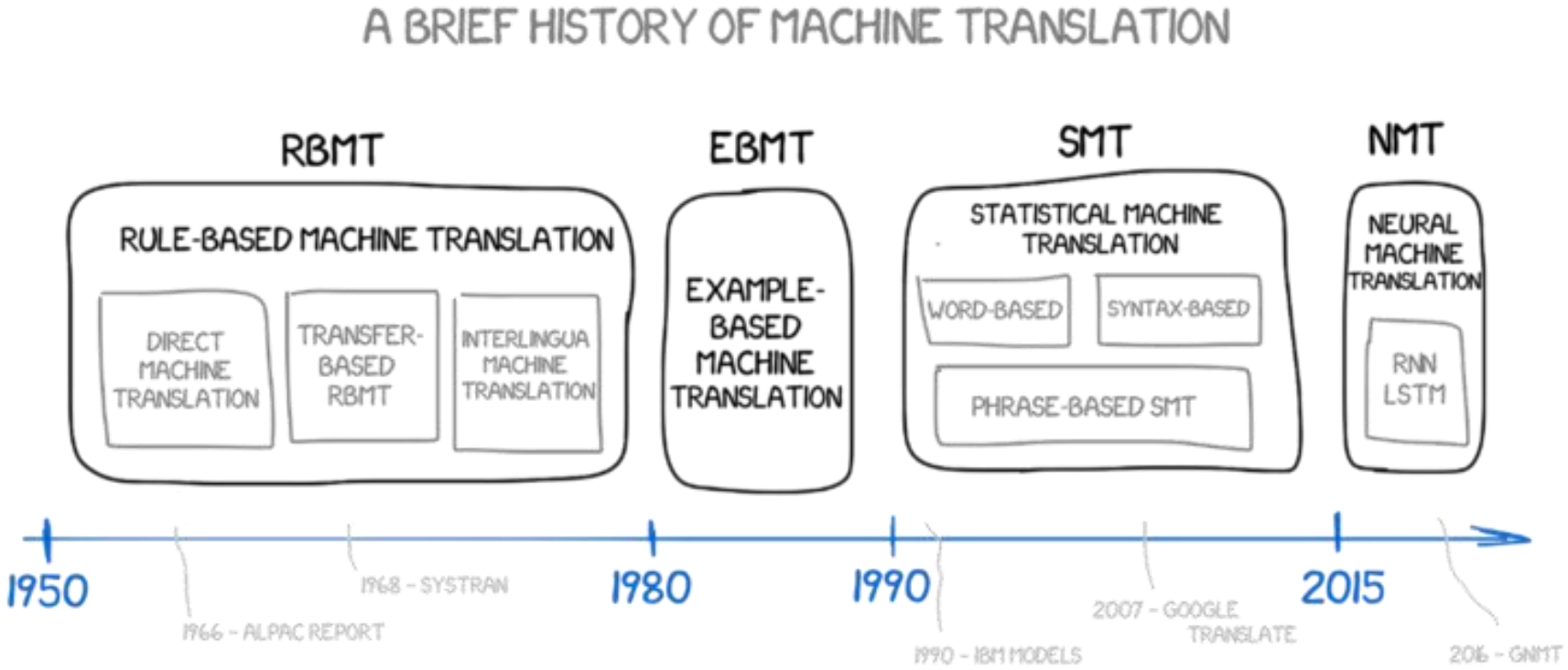
Chinese: 布什与沙龙举行了会谈



English: Bush held a talk with Sharon

Machine Translation

- History (link)



RBMT

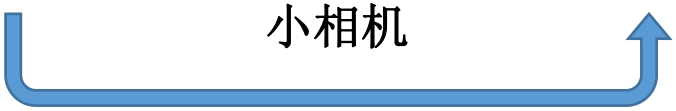
- Rule-based machine translation (RBMT)
- Machine Translation research began in the early 1950s
- Mostly Russian → English

  (motivated by the Cold War!)
- Systems were mostly rule-based, using a bilingual dictionary to map Russian words to their English counterparts
- Extremely complicated

EBMT

- Example-based machine translation (1984)
- Translation of fragmental phrases by analogy
- Extract matching templates from bilingual corpus:

English	Chinese
How much is that **red umbrella**?	那个**红雨伞**多少钱？
How much is that **small camera**?	那个_____多少钱？



- Feed the machine with existing translations and don't need to spend years forming rules and exceptions

Statistical Machine Translation

- Suppose we are translating Chinese → English.
- We want to find best English sentence y, given Chinese sentence x

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

- Use Bayes rule to break this down into two components to be learnt separately:

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

  - where $P(x)$ is a constant

Outline

- Introduction to Seq2Seq
- Machine Translation
  - Introduction
  - Neural Machine Translation
- Attention
- Transformer

Neural Machine Translation

- Neural Machine Translation (NMT): a way to conduct Machine Translation with a single neural network
- No separated language model and translation model (recall SMT)
- Neural network architecture: Seq2Seq architecture which involves two RNNs
- Recall RNN language model first!

RNNs for language modeling

output distribution

$y_4 = \text{softmax}(Uh_4 + b_2) \in \mathbb{R}^{|V|}$

hidden states

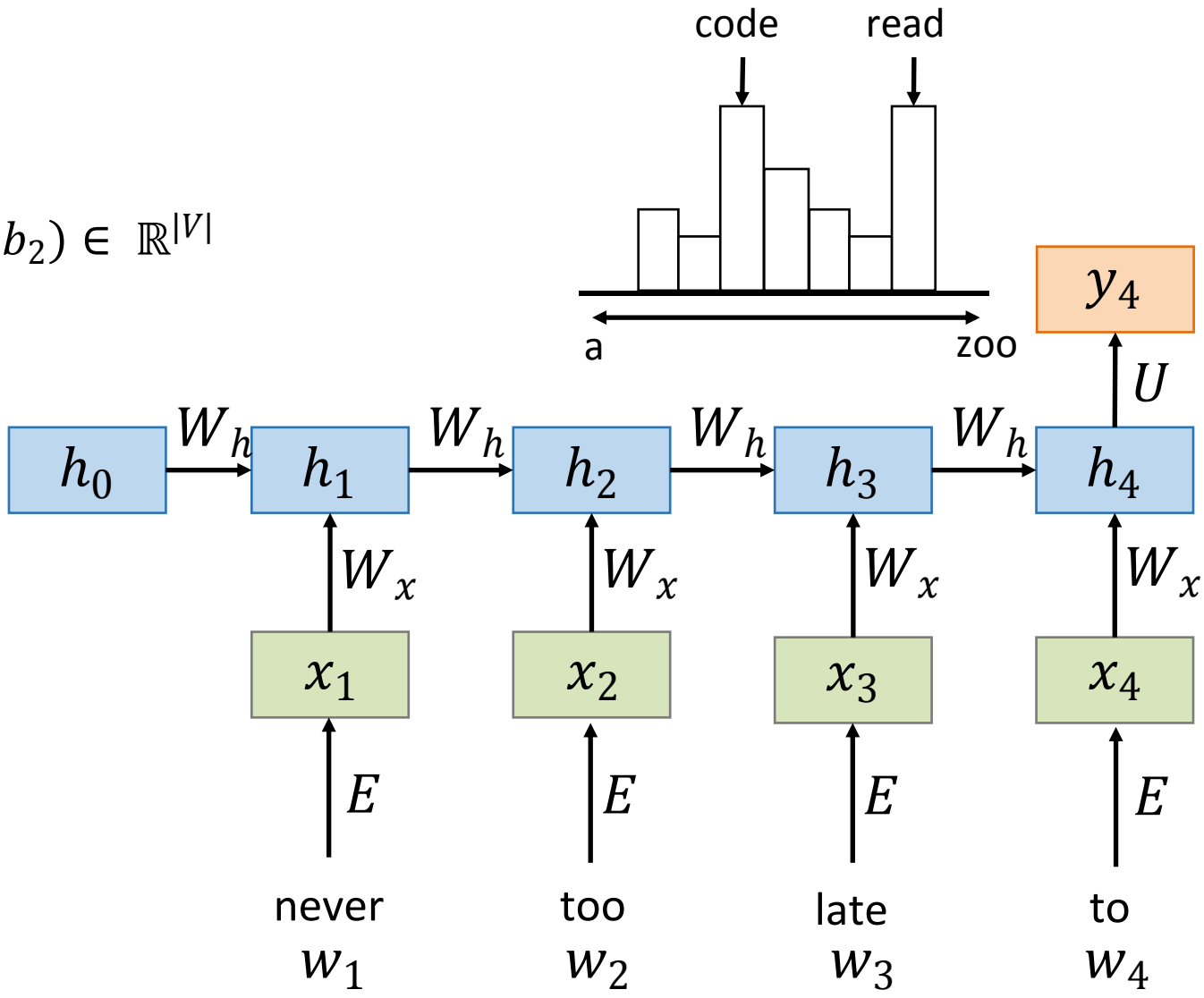
$h_i = \tanh\left(\begin{matrix} W_x x_i + \\ W_h h_{i-1} + b_1 \end{matrix}\right)$

word embeddings

$x_i = Ew_i$

one-hot vectors

$w_i \in \mathbb{R}^{|V|}$



RNNs for language modeling

- How does the RNN cell work?
- RNN cell takes the current RNN state and a word vector and produces a subsequent RNN state that encodes the sentence so far

$$h_i = \tanh\left(W_x x_i + W_h h_{i-1} + b_1\right)$$

- Learned weights represent how to combine past information h_{i-1} and current information x_i

RNNs for language modeling

- How does the output function work?

$$y_4 = \text{softmax}(Uh_4 + b_2) \in \mathbb{R}^{|V|}$$

- y_4 is a probability distribution over the vocab constructed from the RNN memory and the transformation (U, b_2)
- Softmax function turns scores into a probability distribution

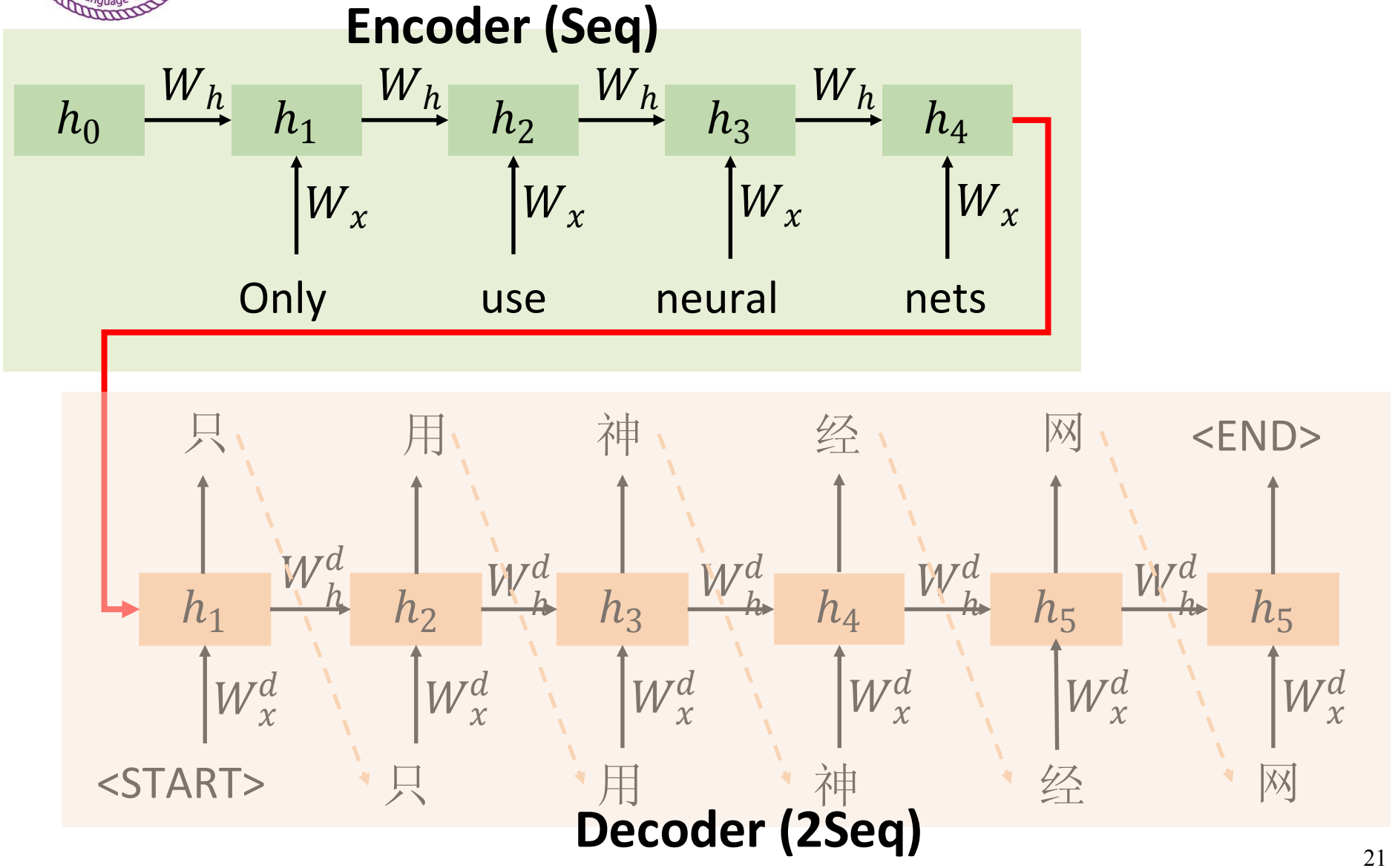
RNNs for Encoding

- Predict things other than next word:
  - POS Tagging
  - Named Entity Recognition
  - Sentiment Classification
  - Relation Classification
- RNNs are good at modeling sequential information
- General idea: Use RNN as an encoder for building the semantic representation of the sentence

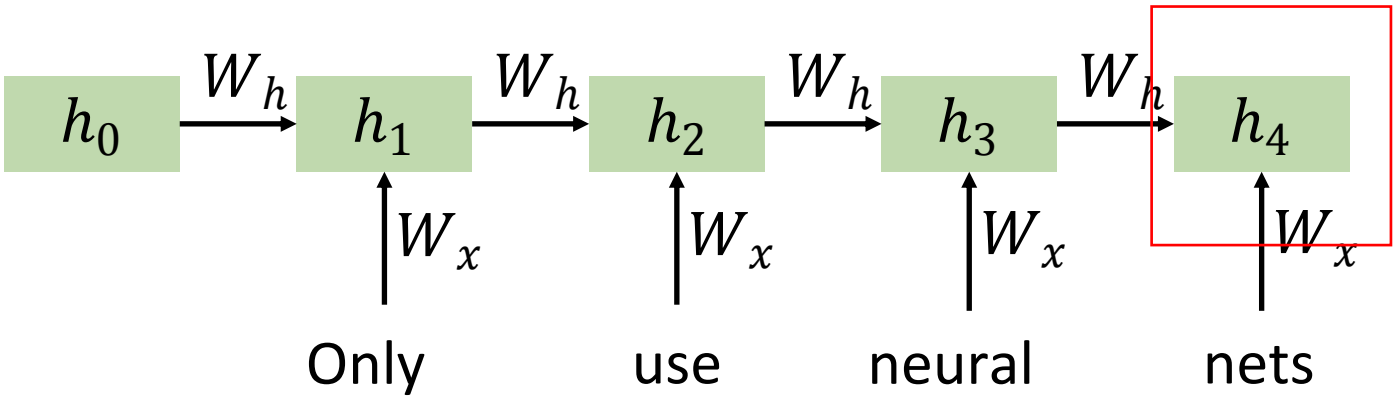
Seq2Seq in MT

- Recall the sequence-to-sequence model
- Two RNNs
  - Encoder RNN
  - Decoder RNN
- Encoder RNN: produces a representation of the source sentence
- Decoder RNN: a language model that generates target sentence conditioned on encoding

Seq2Seq in MT

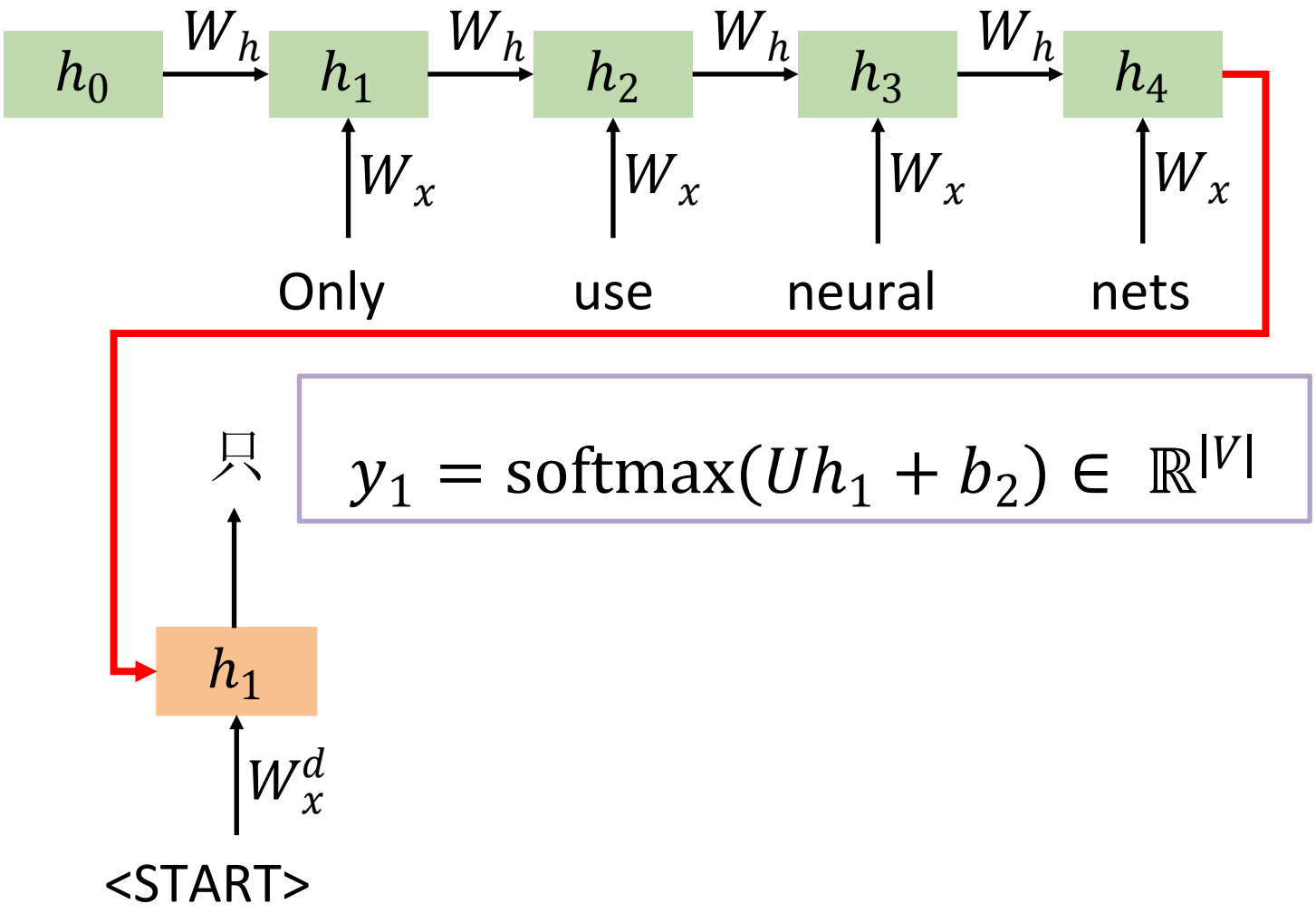


Seq2Seq in MT

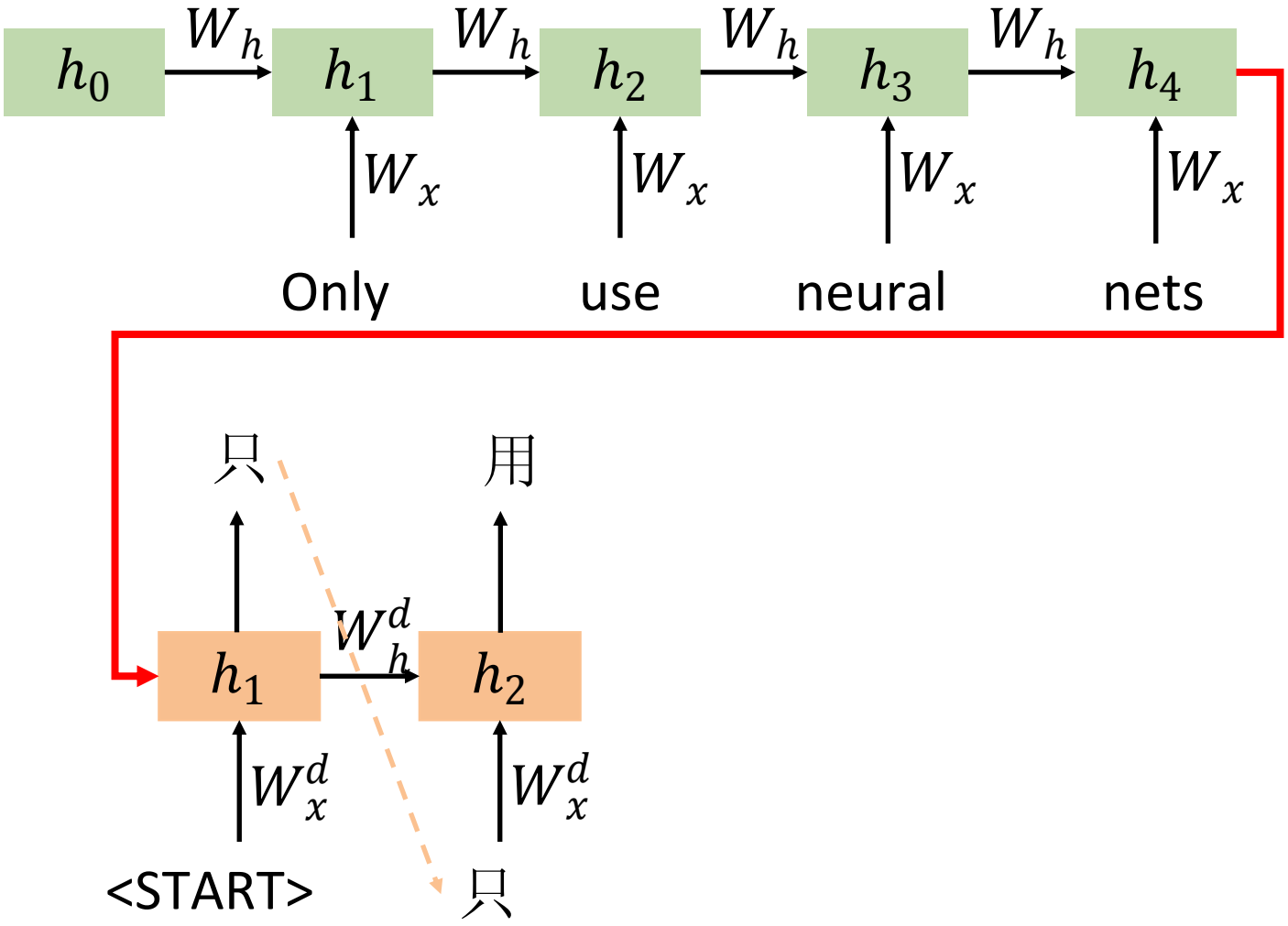


- h_4 is the representation of the source sentence and is provided as the initial hidden state for Decoder RNN

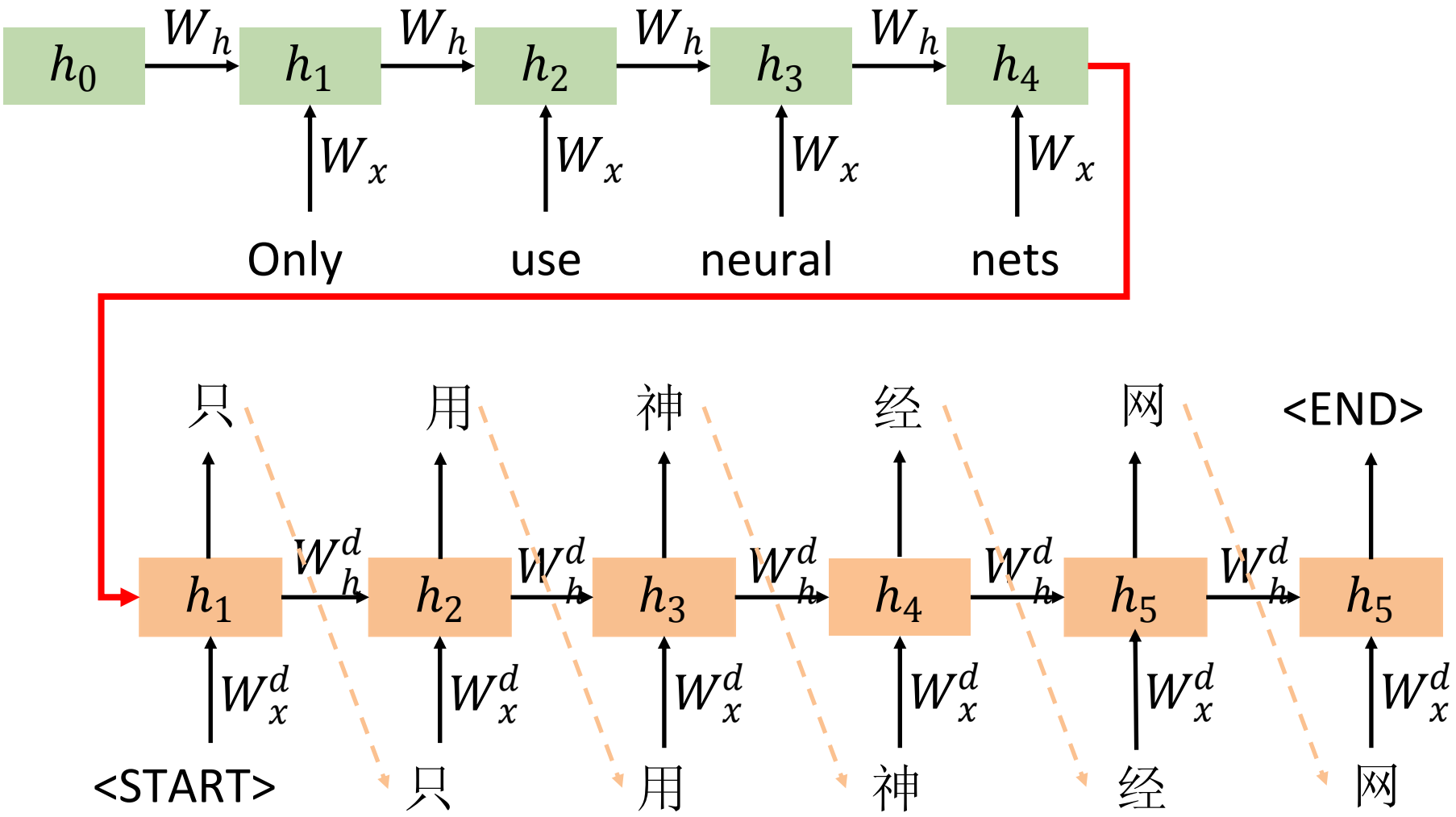
Seq2Seq in MT



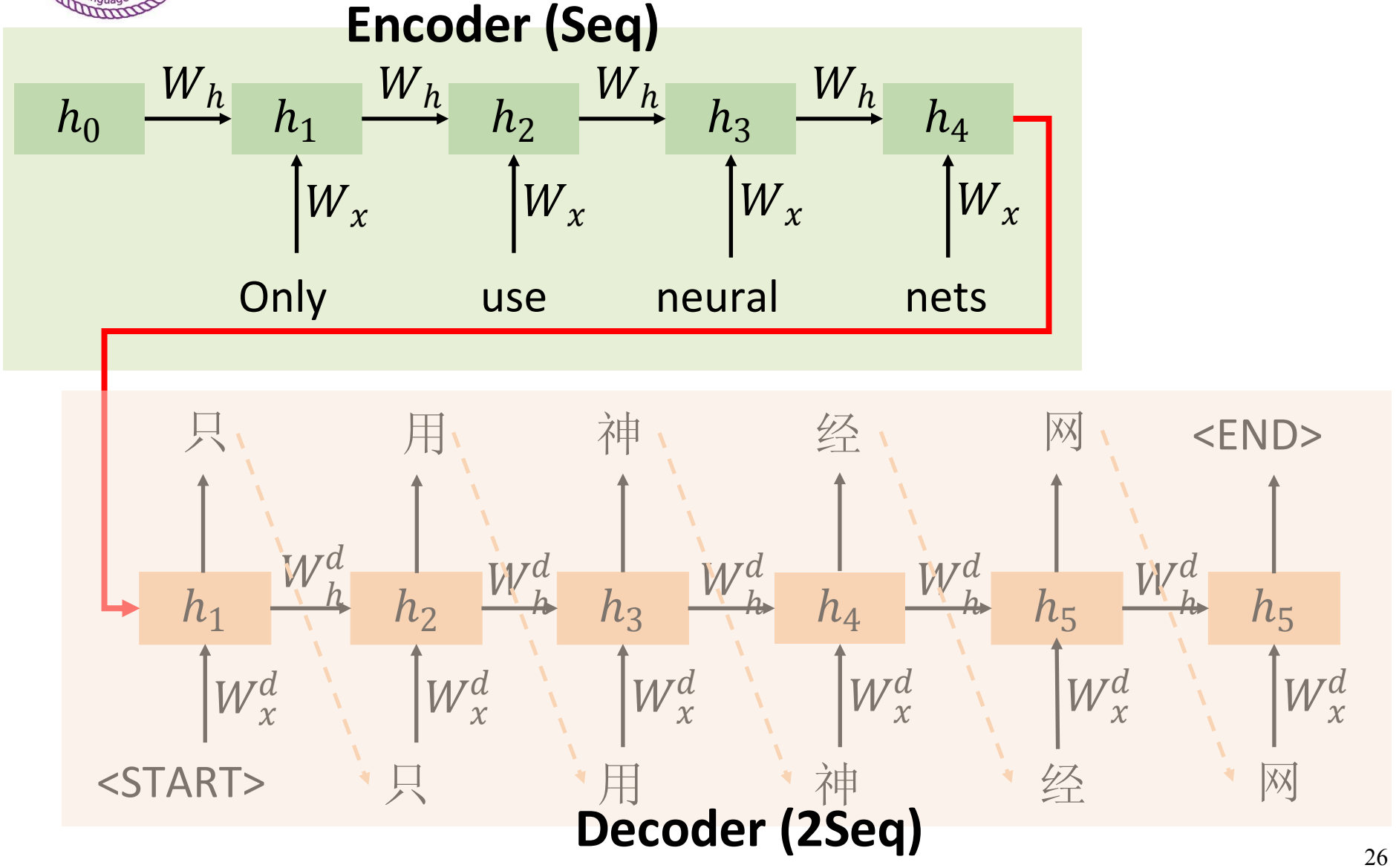
Seq2Seq in MT



Seq2Seq in MT



Seq2Seq in MT



Seq2Seq in MT

- Seq2seq model is an example of a conditional language model
  - Language model: the decoder is predicting the next word of the target sentence y
  - Conditional: its predictions are also conditioned on the source sentence x
- $P(y|x)$ in NMT

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

- Different from SMT: $P(x|y)P(y)$
- More direct!

Outline

- Introduction to Seq2Seq
- Machine Translation
  - Introduction
  - Statistical Machine Translation
  - Neural Machine Translation
    - Decoding Strategy
    - Evaluation of MT
    - Summary

Vanilla Decoder Strategy

- Recall the decoding process

$$\text{argmax}_{y_i} P(y_i | y_1, \ldots, y_{i-1}, x)$$

- Generate the target sentence by taking argmax on each step of the decoder
- Greedy decoding
- Recall the original translation model

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

- Can the greedy decoding always generate the best y?

Vanilla Decoder Strategy

- In this problem, a greedy strategy does not usually produce a globally optimal solution
- Problem: Greedy decoding has no way to undo decisions!

  布什与沙龙举行了会谈 (Bush held a talk with Sharon)

  →Bush

  →Bush and

  →Bush and Sharon

Beam Search Decoding

- Ideally, we want to find y that maximizes

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

- We could try enumerating all y
  - Complexity $O(V^T)$ where V is vocab size and T is target sequence length → too expensive!

- Beam Search: On each step, keep track of the k most probable partial translations

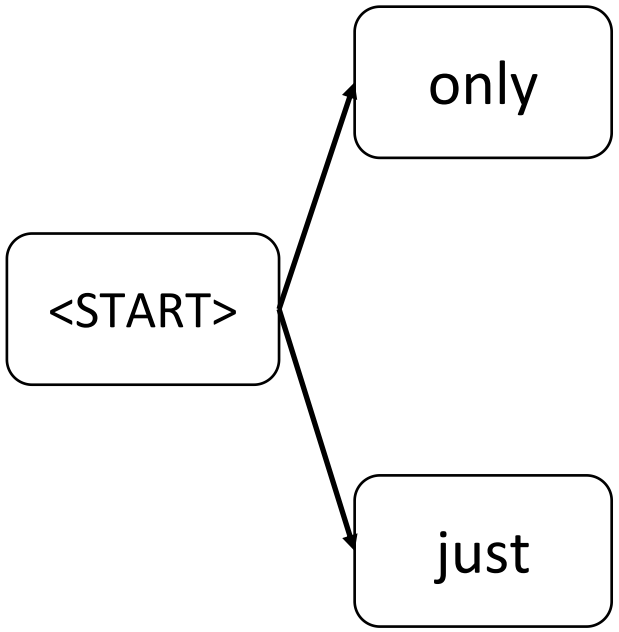
Beam Search Decoding

- Beam Search: On each step, keep track of the k most probable partial translations
- k is the beam size (in practice around 5 to 10)
- Also not guarantee to produce a globally optimal solution
- But results are much more applicable!
- Example:

  只用神经网→ Only use neural nets

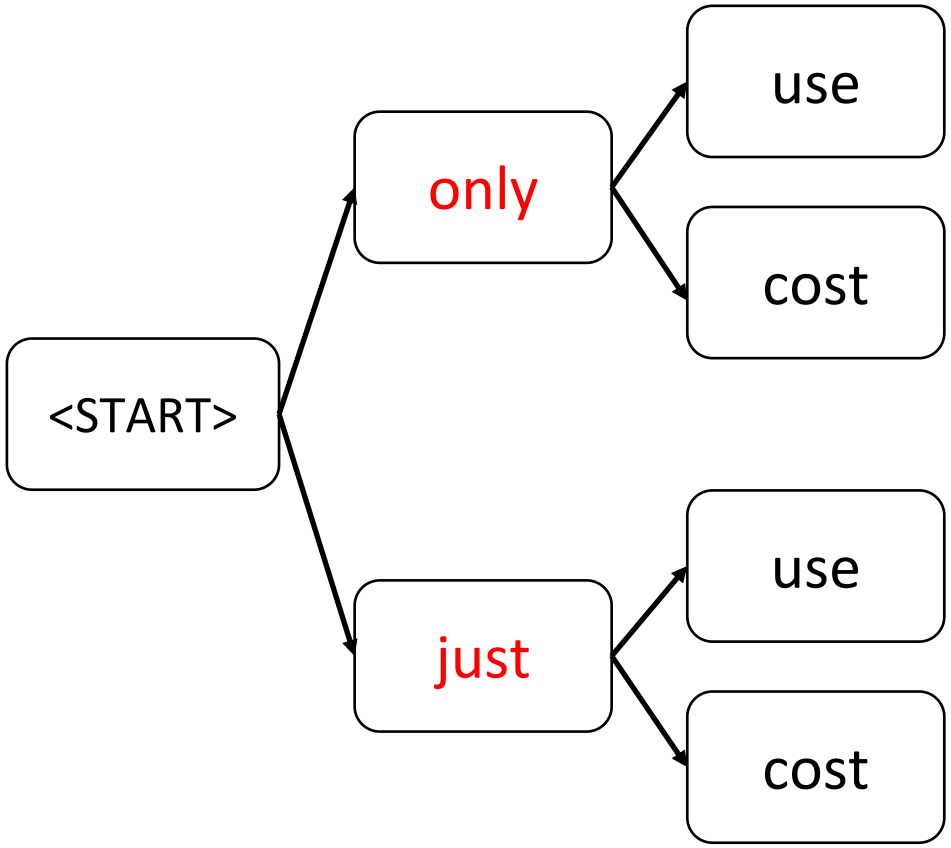
Beam Search Decoding

- Example (beam size = 2)



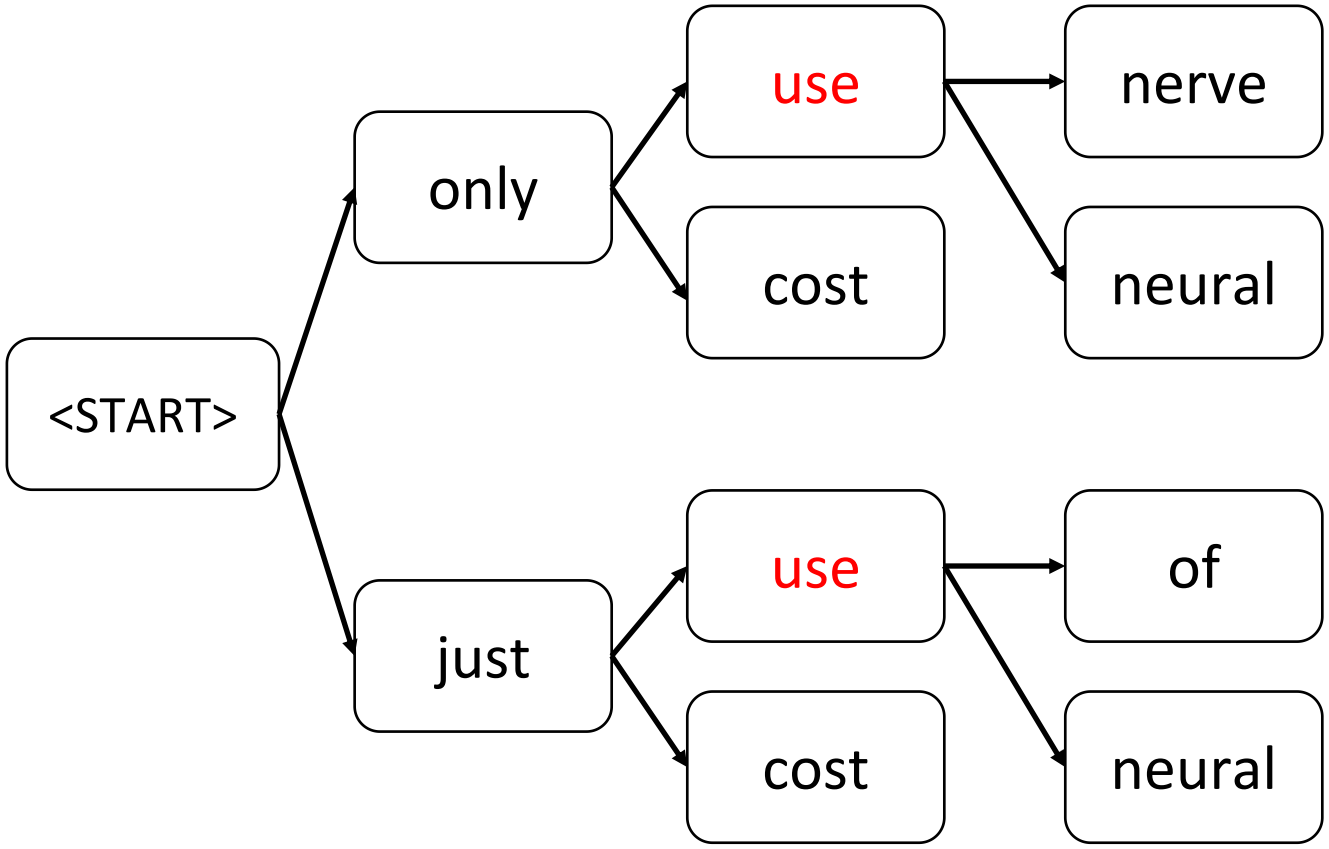
Beam Search Decoding

- Example (beam size = 2)



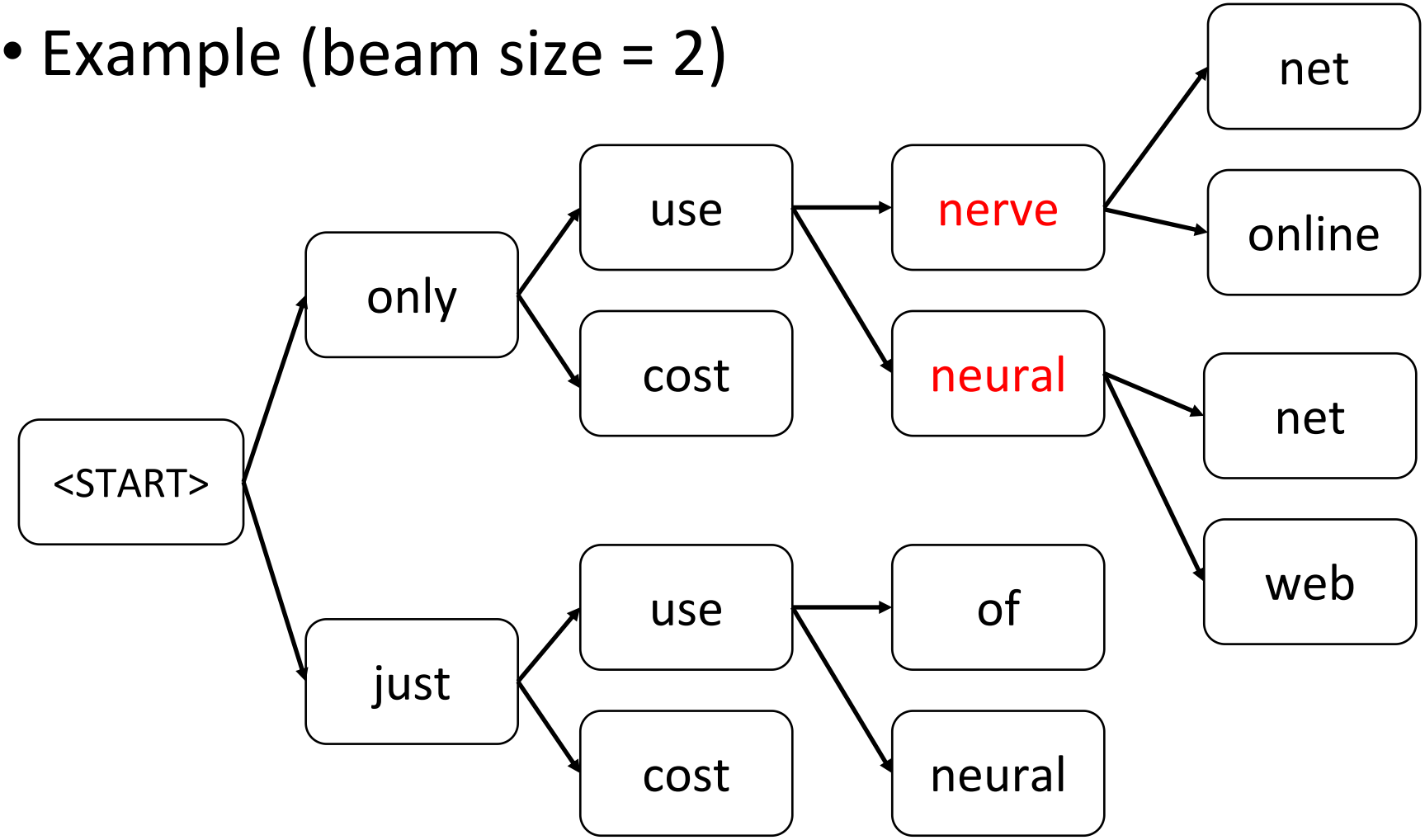
Beam Search Decoding

- Example (beam size = 2)



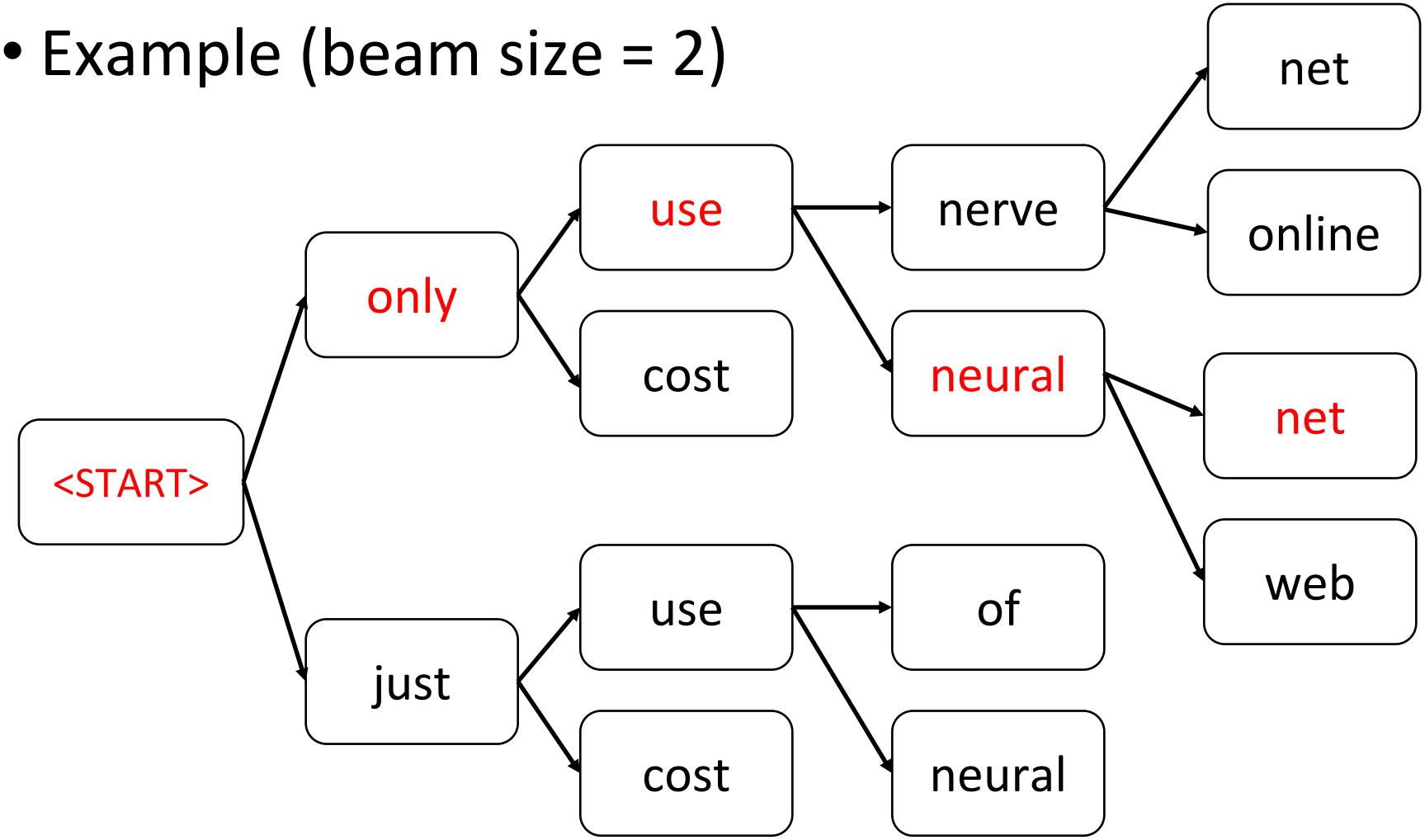
Beam Search Decoding

- Example (beam size = 2)



Beam Search Decoding

- Example (beam size = 2)



Outline

- Introduction to Seq2Seq
- Machine Translation
  - Introduction
  - Statistical Machine Translation
  - Neural Machine Translation
    - Decoding Strategy
    - Evaluation of MT
    - Summary

Evaluation

- **BLEU** (**Bi**lingual **E**valuation **U**nderstudy)
- BLEU compares the machine-written translation to one or several human-written translation(s), and computes a similarity score based on:
  - N-gram precision
  - Penalty for too-short system translations

$$BLEU = \mathrm{BP} * \exp\left(\sum_{i=1}^{N} w_i log P_i\right)$$

Evaluation

- **BLEU** (**Bi**lingual **E**valuation **U**nderstudy)

$$BLEU = \mathrm{BP} * \exp\left(\sum_{i=1}^{N} w_i log P_i\right)$$

  - P_i: The i-gram precision
  - BP: Brevity penalty

$$BP = \begin{cases} 1, & l_c > l_r \\ e^{1-\frac{l_r}{l_c}}, & l_c \leq l_r \end{cases}$$

    - l_c: length of the candidate, l_r: length of the reference.
  - Usually we set $N = 4,\ w_i = \frac{1}{4}$

Evaluation

- BLEU example
- Candidate: Airport security Israeli officials are responsible.
- Reference: Israeli officials are responsible for airport security.
  - 1-gram precision: $P_1 = \frac{6}{6}$
  - 2-gram precision: $P_2 = \frac{4}{5}$
  - 3-gram precision: $P_3 = \frac{2}{4}$
  - 4-gram precision: $P_4 = \frac{1}{3}$

Evaluation

- BLEU example

$$P_1 = \frac{6}{6}, P_2 = \frac{4}{5}, P_3 = \frac{2}{4}, P_4 = \frac{1}{3}$$

- The calculation

$$\exp\left(\frac{1}{4}\ (\log(1) + \log\left(\frac{4}{5}\right) + \log\left(\frac{2}{4}\right) + \log\left(\frac{1}{3}\right))\right) = 0.386$$

- Considering the brevity penalty

$$BLEU = e^{1-7/6} * 0.386 = 0.327$$

Evaluation

- **BLEU** (**Bi**lingual **E**valuation **U**nderstudy)
- BLEU is useful but imperfect
  - There are many valid ways to translate a sentence
  - Sometime a good translation can get a poor BLEU score because it has low n-gram overlap with human translation

Reference: I ate the apple.

Candidates:

1. I consumed the apple.
2. I ate an apple.
3. I ate the potato.

Outline

- Introduction to Seq2Seq
- Machine Translation
  - Introduction
  - Statistical Machine Translation
  - Neural Machine Translation
    - Decoding Strategy
    - Evaluation of MT
    - Summary

Advantages of NMT

- Compared to SMT, NMT has many advantages:
  - 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

Disadvantages of NMT

- Compared to SMT:
- NMT is less interpretable
  - Hard to debug
- NMT is difficult to control
  - For example, cannot easily specify rules or guidelines for translation
  - Safety concerns!

Summary

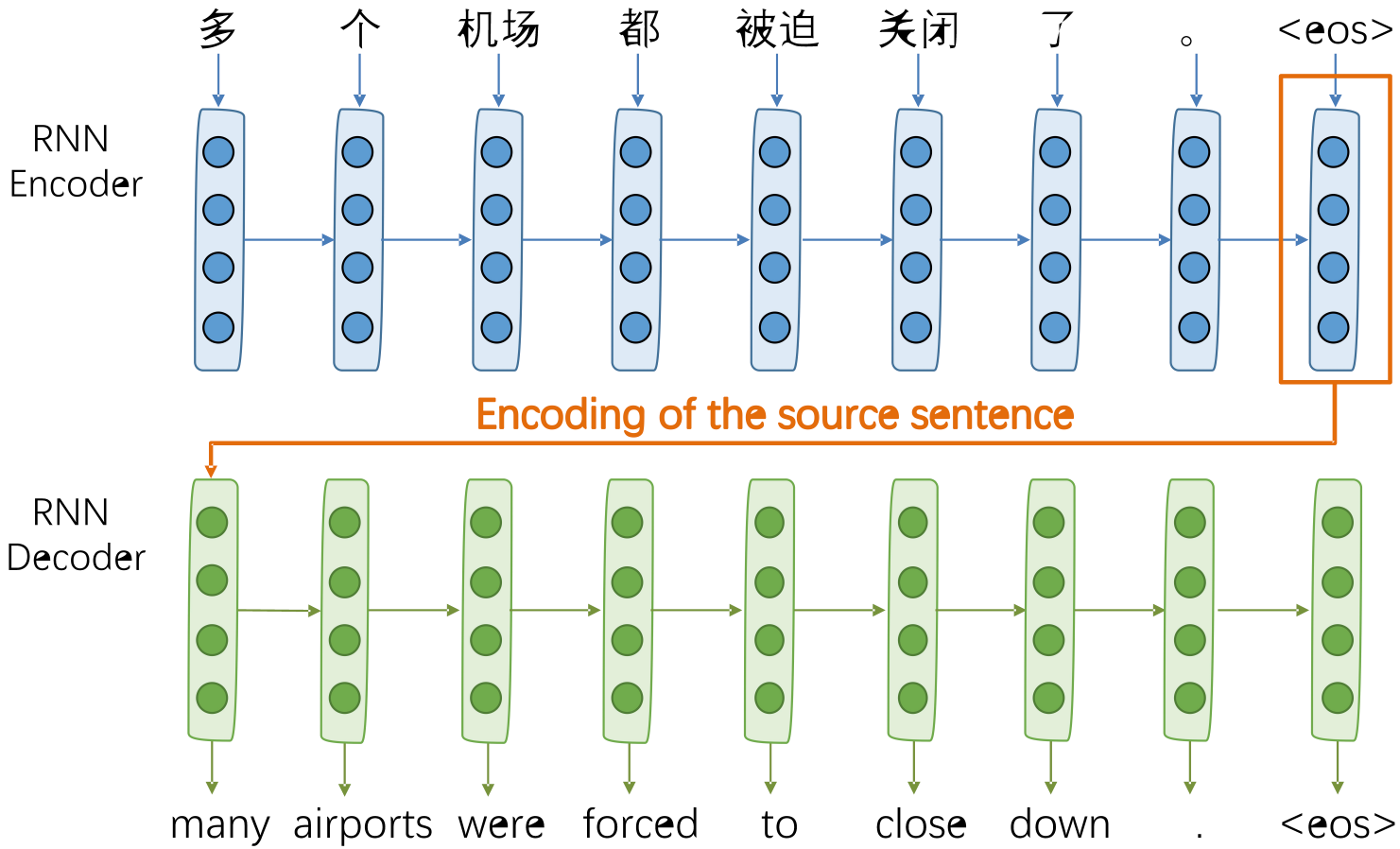
- Seq2Seq can use RNN/GRU/LSTM/Transformer to perform encoder-decoder based tasks
  - Training method, decoding strategy, evaluation
  - Various applications
- Machine translation
  - RBMT, EBMT, SMT, NMT
- Attention is critical for improving seq2seq models
- Transformer is powerful in sequence modeling

Outline

- Introduction to Seq2Seq
- Machine Translation
- Attention
- Transformer

Attention

- Seq2Seq: the bottleneck problem



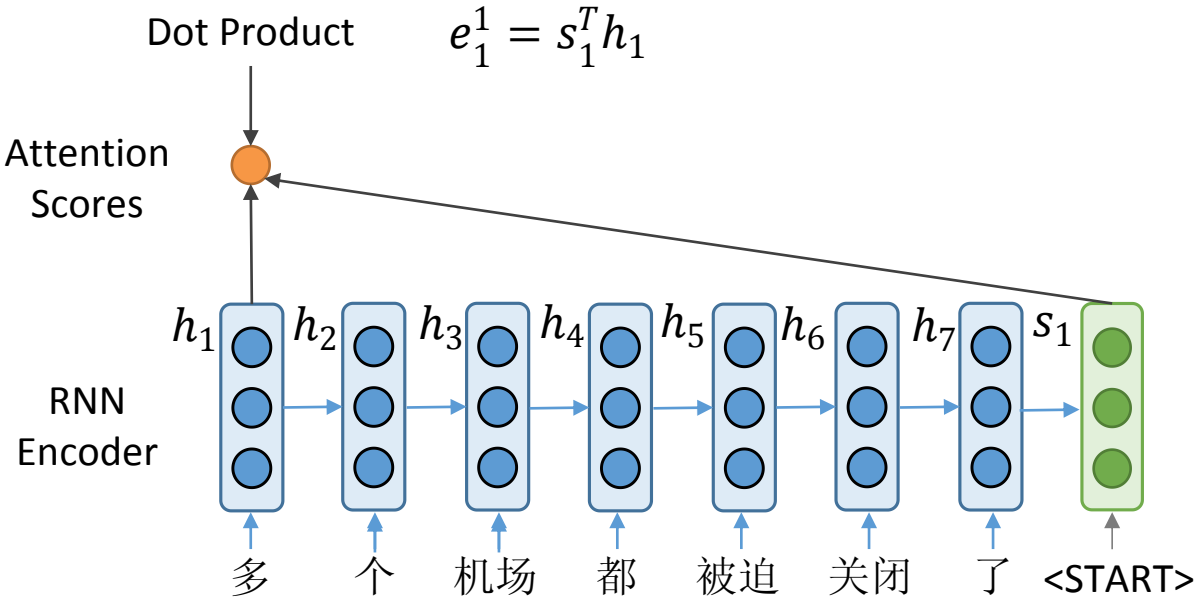
Attention

- The Bottleneck Problem
  - The single vector of source sentence encoding needs to capture all information about the source sentence
  - The single vector limits the representation capacity of the encoder: the information bottleneck

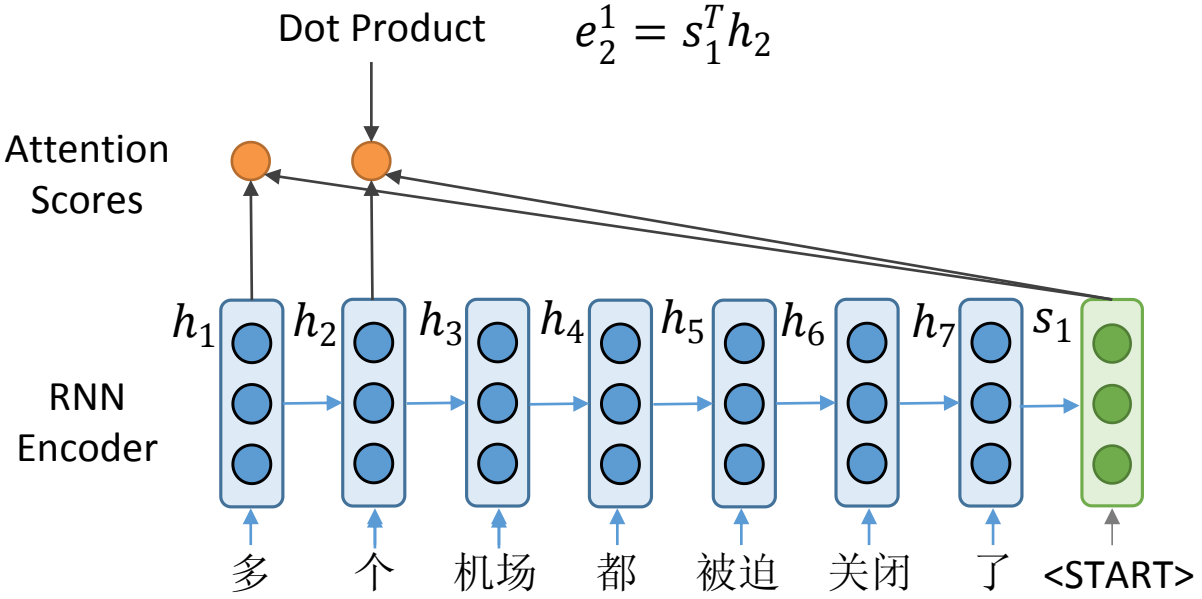
Attention

- Attention
  - Attention provides a solution to the bottleneck problem
  - Core idea: at each step of the decoder, focus on a particular part of the source sequence

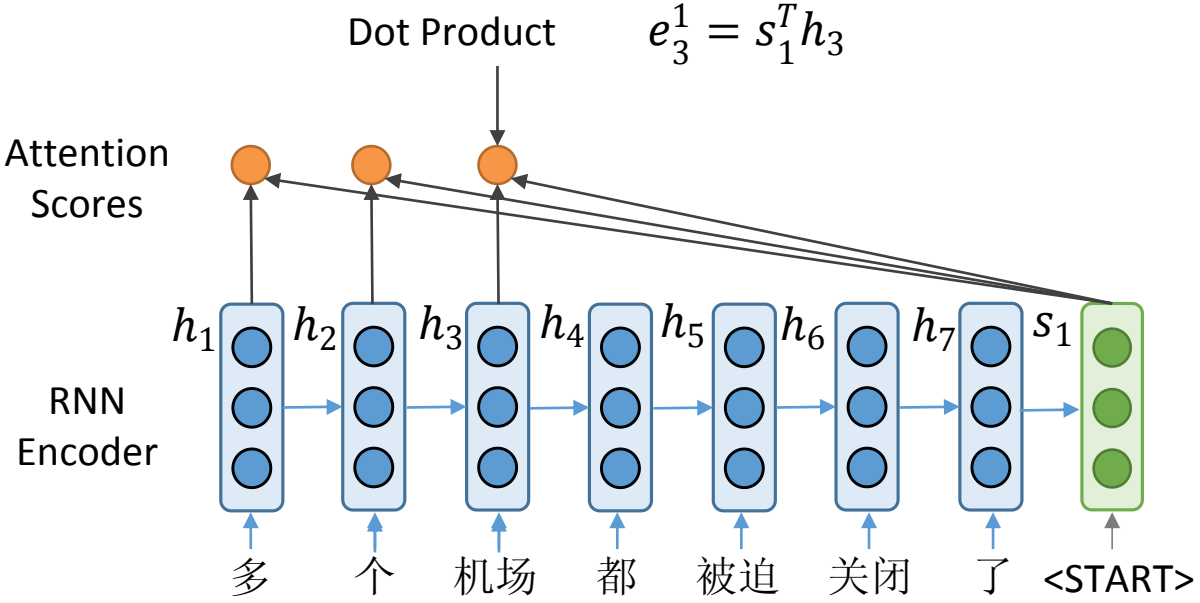
Seq2Seq with Attention



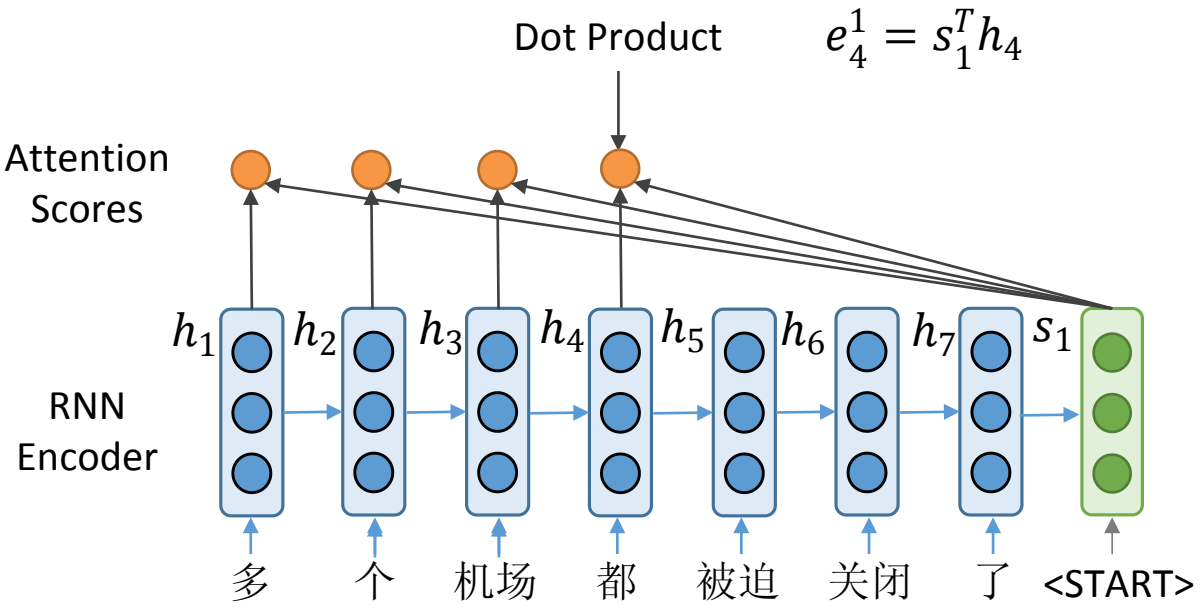
Seq2Seq with Attention



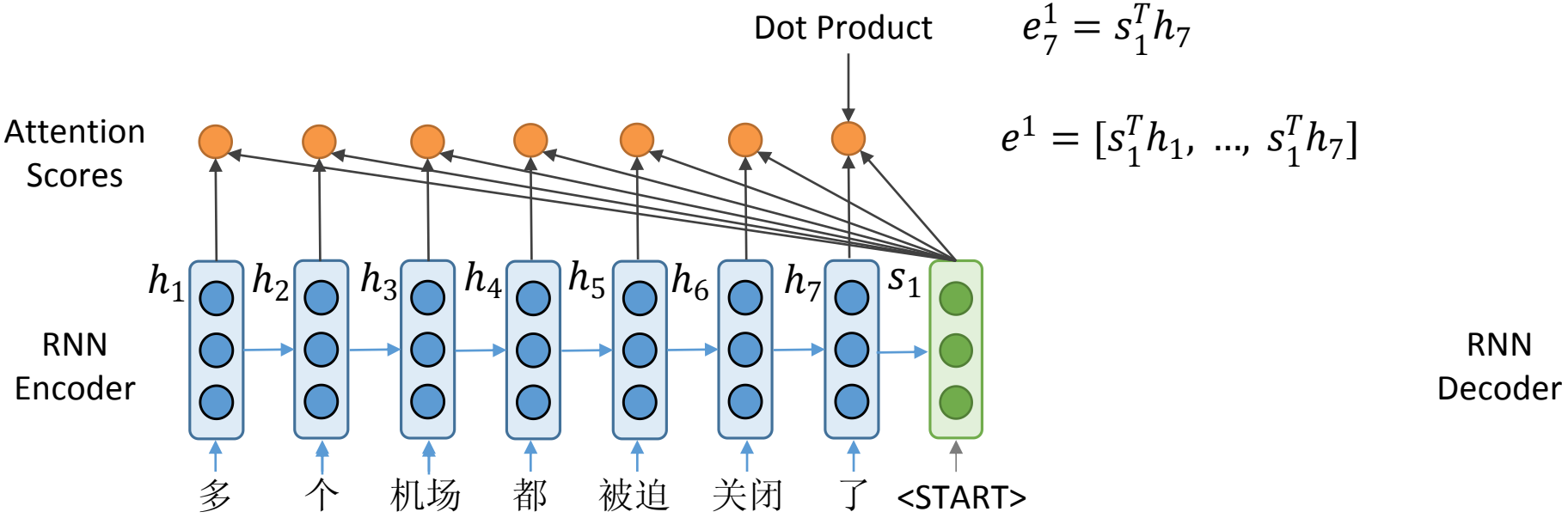
Seq2Seq with Attention



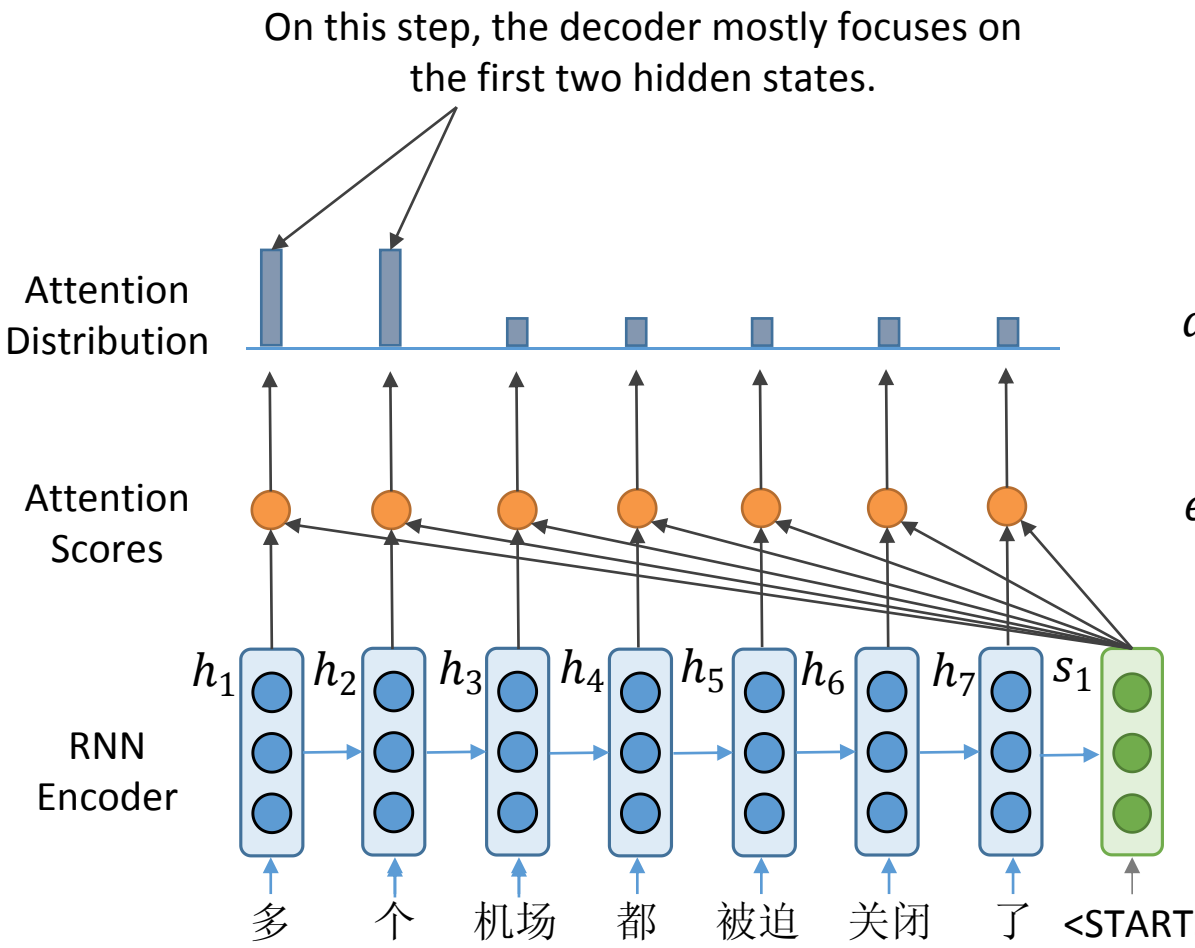
Seq2Seq with Attention



Seq2Seq with Attention



Seq2Seq with Attention



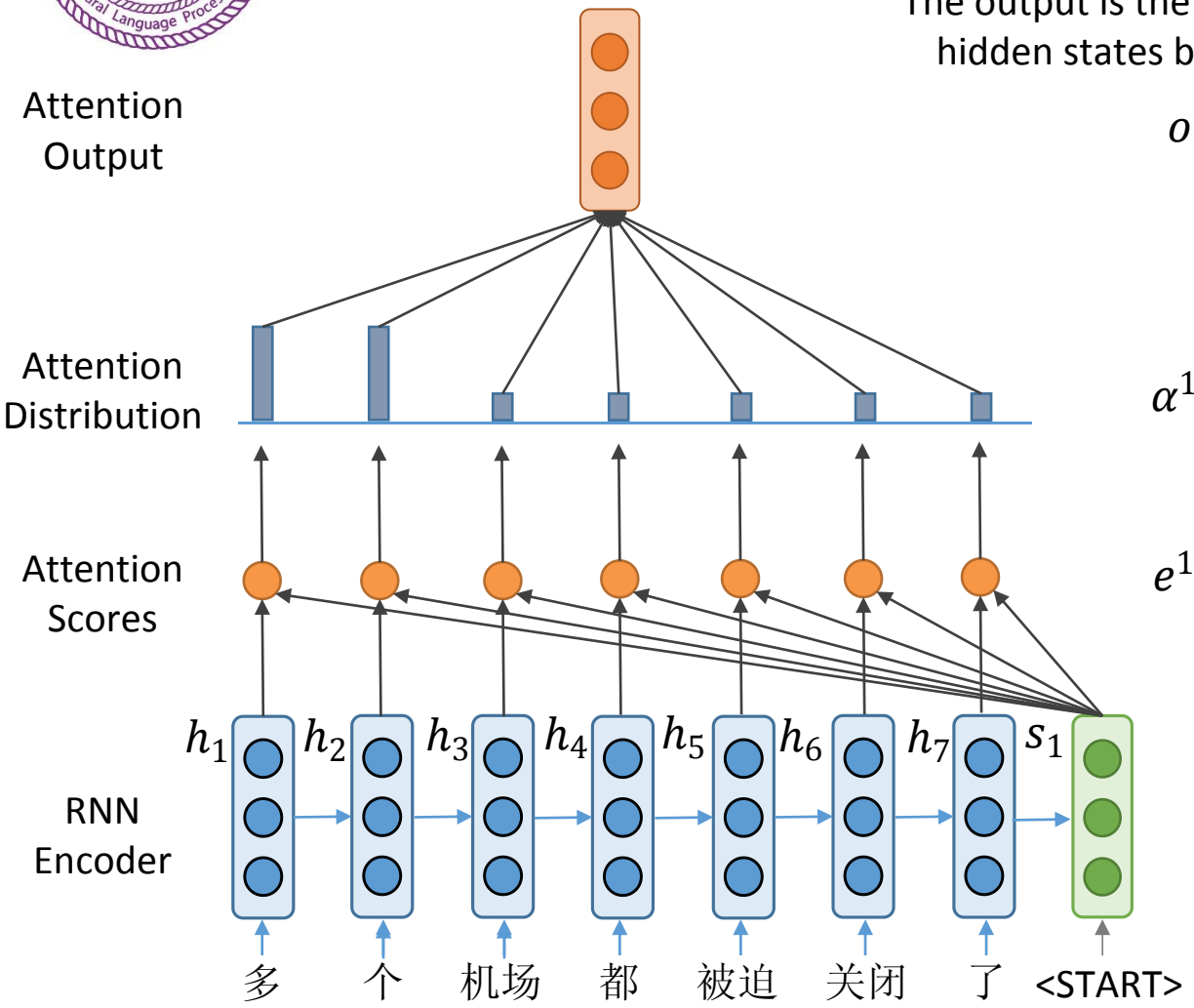
$$\alpha^1 = \text{softmax}(e^1)$$

$$e^1 = [s_1^T h_1, \ \ldots, \ s_1^T h_7]$$

Seq2Seq with Attention

The output is the weighted sum of the encoder hidden states by the attention distribution.

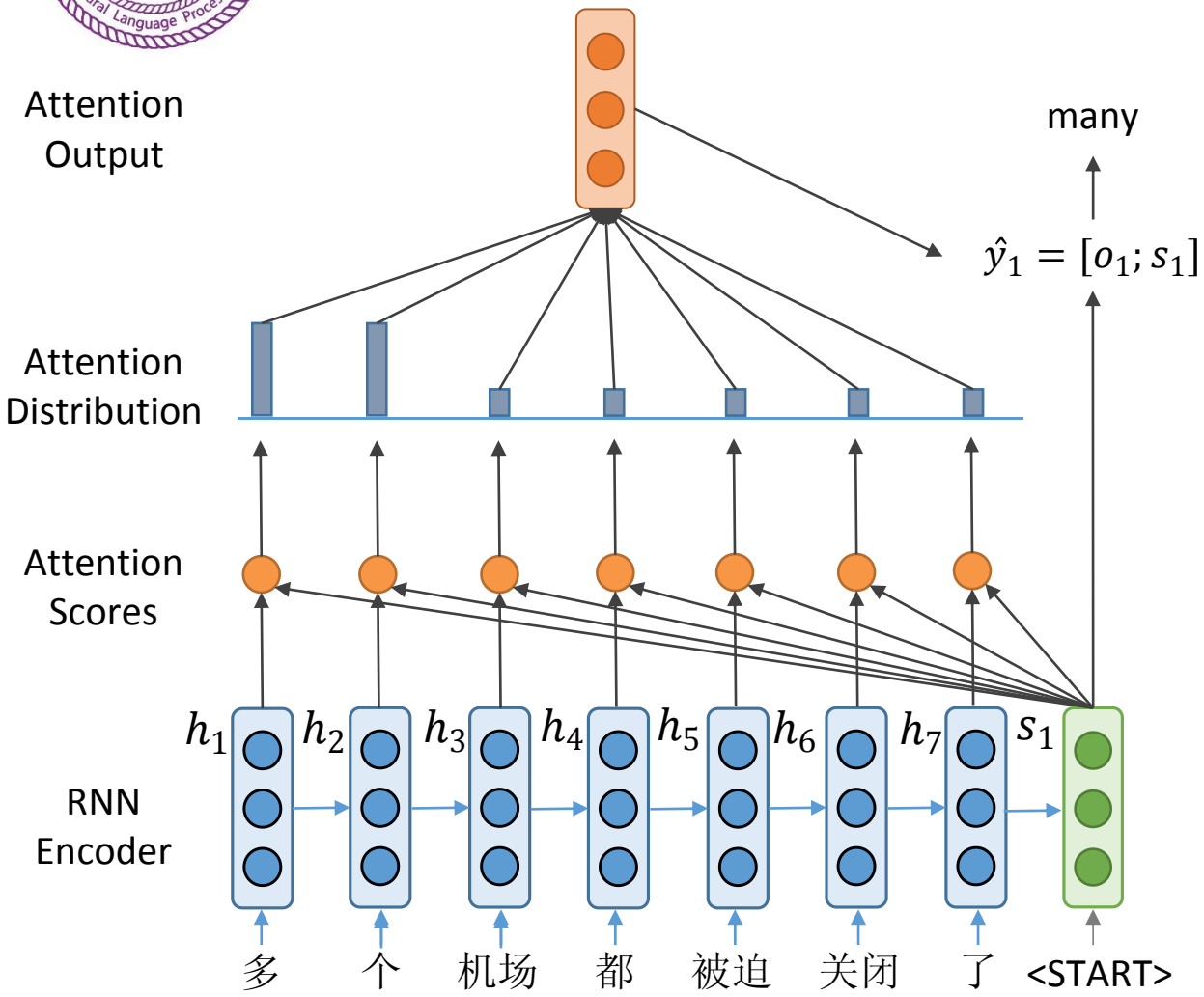
$$o_1 = \sum_{i=1}^{7} \alpha_i^1 h_i$$



$$\alpha^1 = \text{softmax}(e^1)$$

$$e^1 = [s_1^T h_1, \ \ldots, \ s_1^T h_7]$$

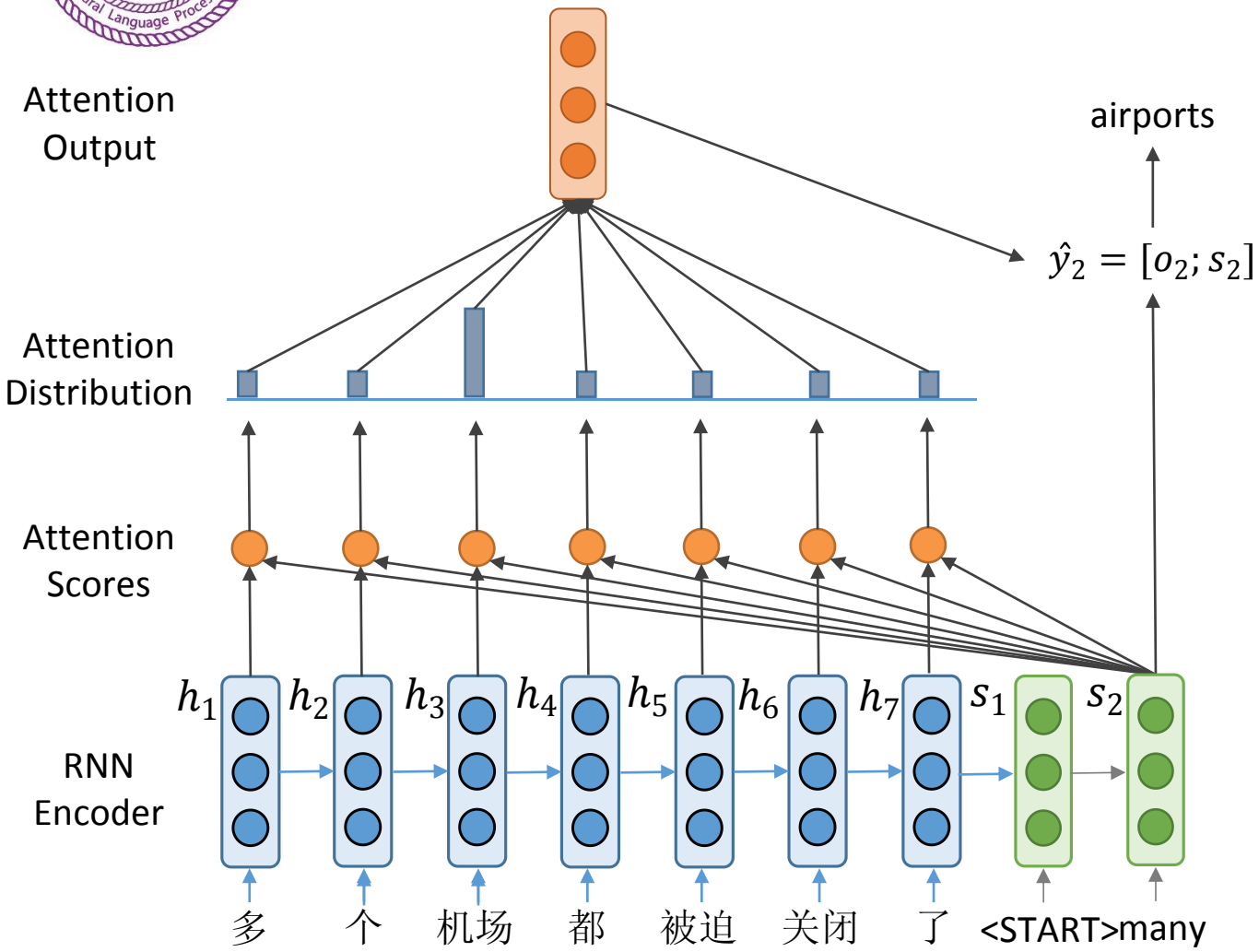
Seq2Seq with Attention



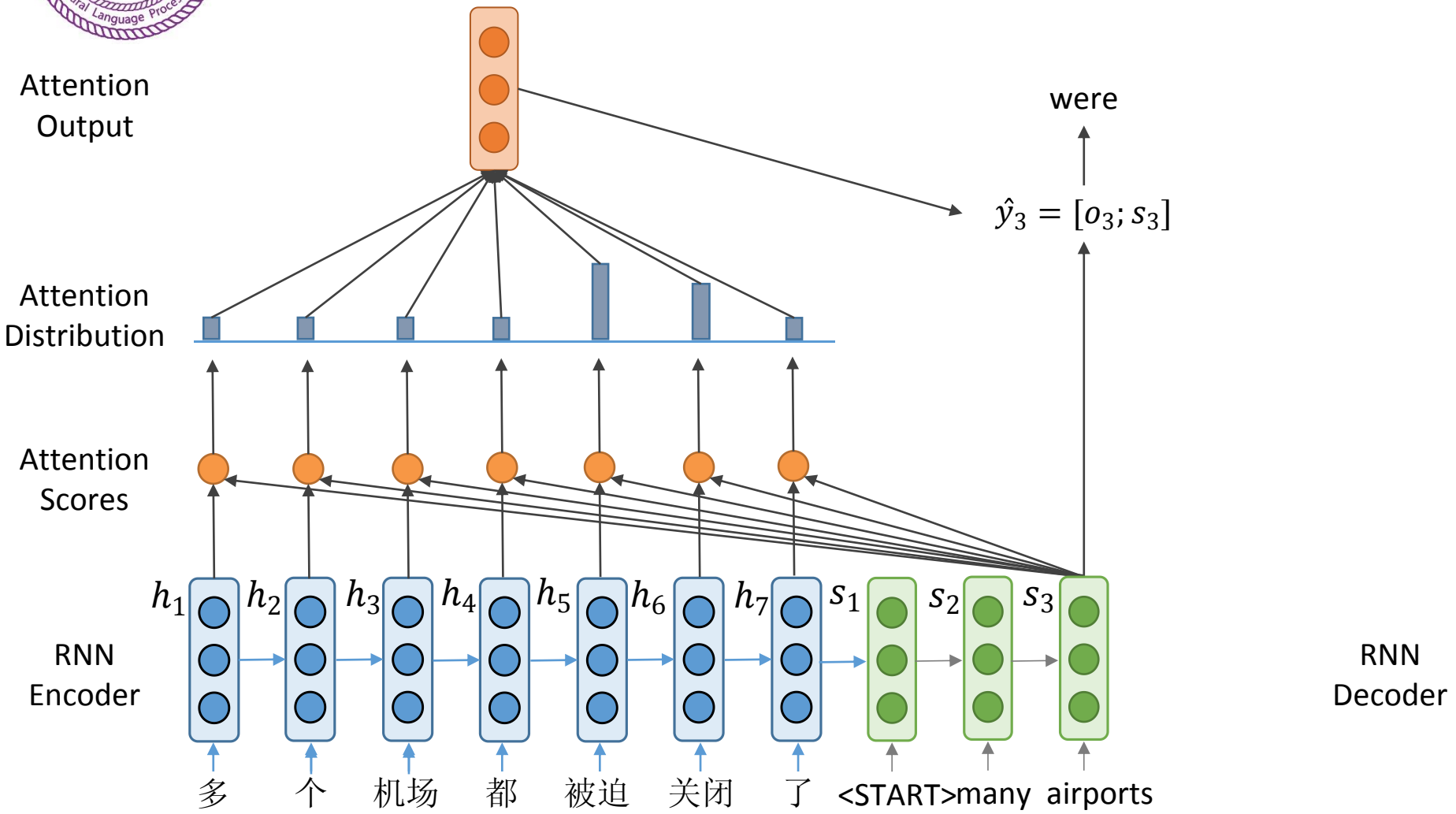
Use the concatenation of the **attention output** and the **hidden state** of the decoder to compute the predicted word.

RNN Decoder

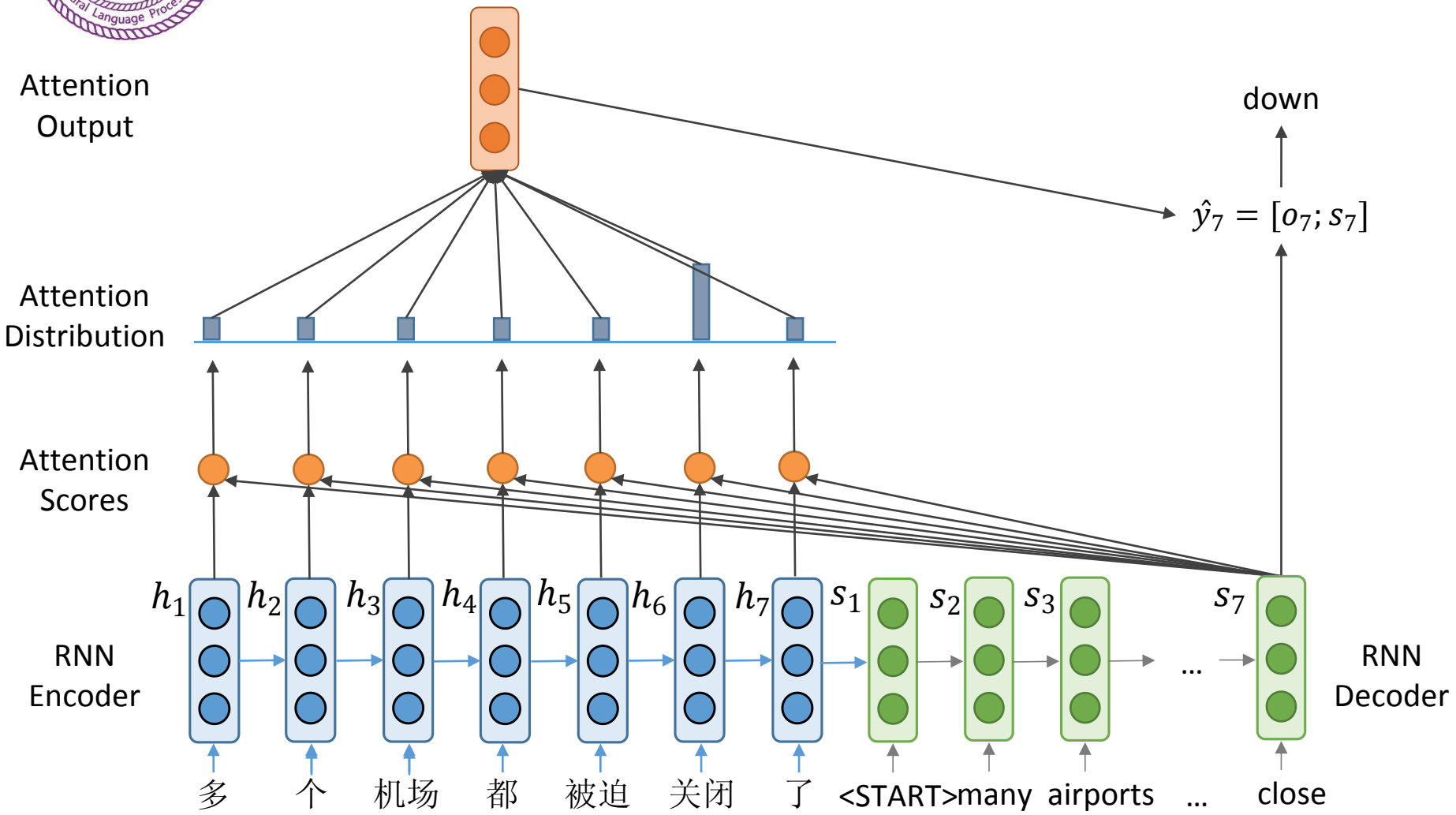
Seq2Seq with Attention



Seq2Seq with Attention



Seq2Seq with Attention



Attention

- Equations (1)
  - Encoder hidden states: $h_1,\ h_2 \ldots, h_N \in \mathbb{R}^h$
  - Decoder hidden state at time step t: $s_t \in \mathbb{R}^h$
  - Compute attention scores e^t for this step

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

  - Use softmax to get the attention distribution α^t

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

Attention

- Equations (2)
  - Use the attention distribution to compute the weighted sum of the encoder hidden states as the attention output

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

  - Concatenate the attention output and the decoder hidden state to predict the word

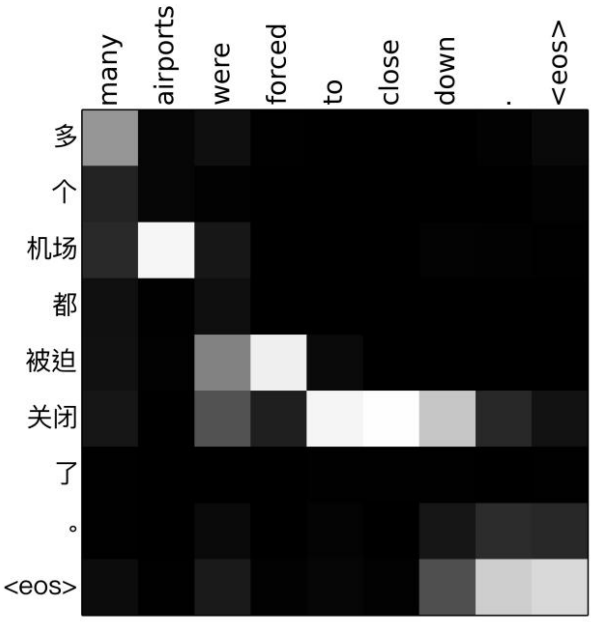
$$[o_t; s_t] \in \mathbb{R}^{2h}$$

Attention

- Insights
  - Attention solves the bottleneck problem
    - The decoder could directly look at source
  - Attention helps with vanishing gradient problem
    - By providing shortcuts to long-distance states

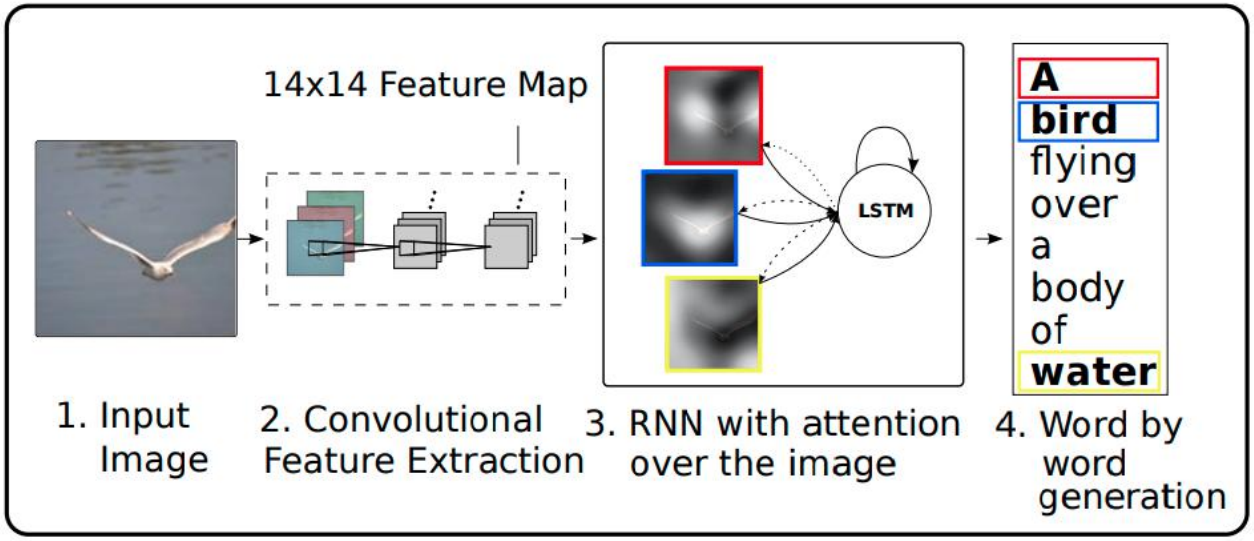
Attention

- Insights
  - Attention provides some interpretability
    - We can find out what the decoder was focusing on by the attention map
    - Attention provides the network the ability to align relevant words
  - Thus, attention significantly improves NMT performance!



Attention Applications

- Attention for image caption
  - Task: Image caption
  - Model:
    - CNN encoder for image processing
    - RNN decoder for text generation



Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, Xu at el. 2015

Attention Applications

- Attention for image caption

Fig. "A woman is throwing a frisbee in a park."

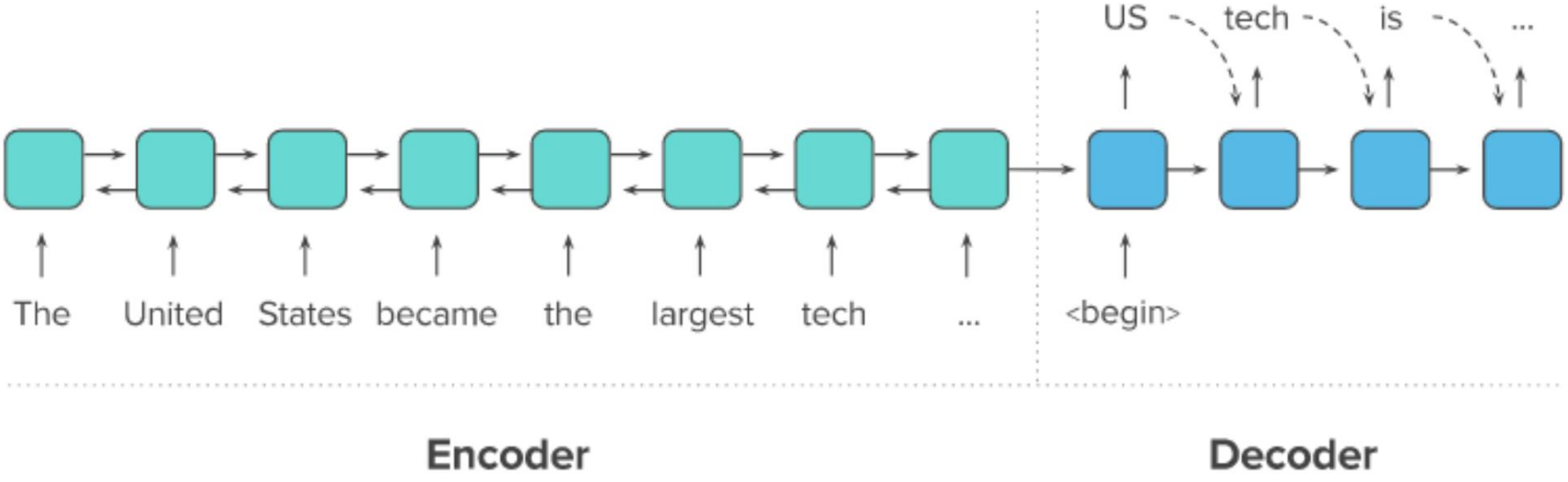
Attention Applications

- Attention for image caption

Fig. "A woman is throwing a frisbee in a park."

Attention Applications

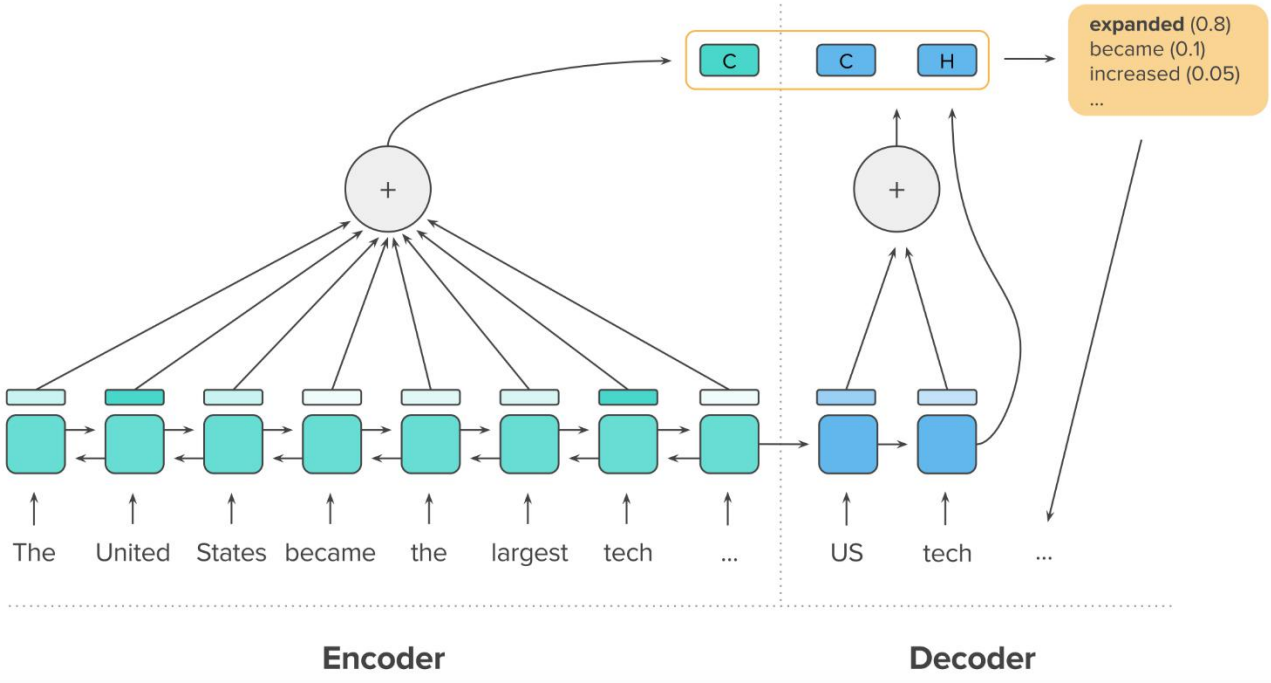
- Intra-decoder attention for summarization
  - Task: Text summarization
  - Problems: For longer outputs, the decoder starts to repeat itself



A deep reinforced model for abstractive summarization. Paulus et al. arXiv 2017.

Attention Applications

- Intra-decoder attention for summarization
  - Solutions
    - More advanced encoder attention
    - Self-attention in decoder

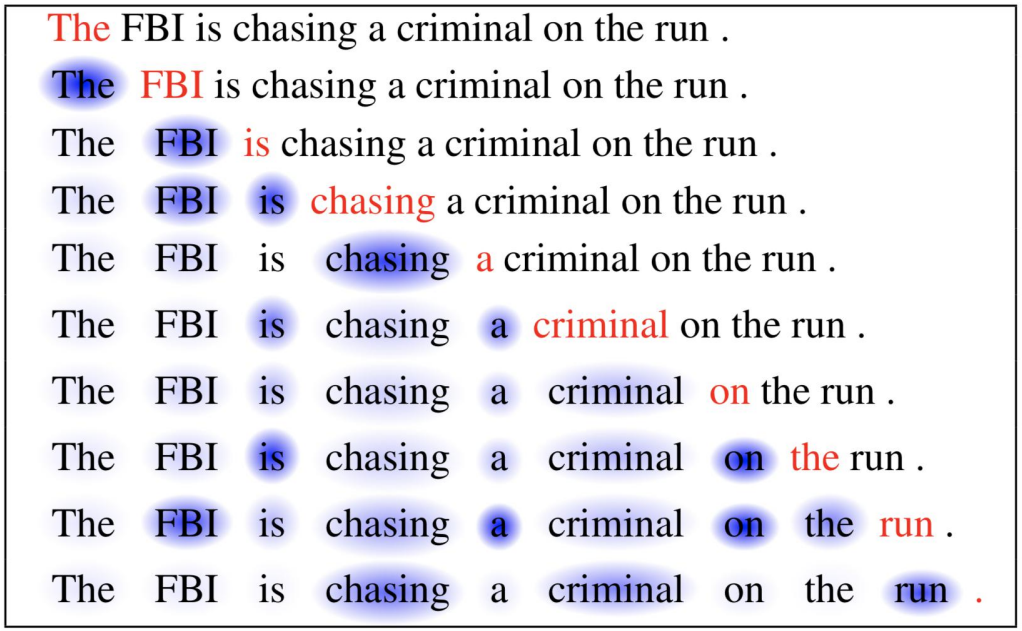


Attention Applications

- Intra-decoder attention for summarization
  - Self-attention in decoder
    - Also called intra-attention.
    - Self-attention is a technique where each value uses itself as a query to attend to itself and other values.
    - It has been shown to be very useful in machine reading, abstractive summarization, or image description generation, etc.

Attention Applications

- Intra-decoder attention for summarization
  - Self-attention – example
    - The self-attention mechanism enables us to learn the correlation between the current words and the previous part of the sentence.



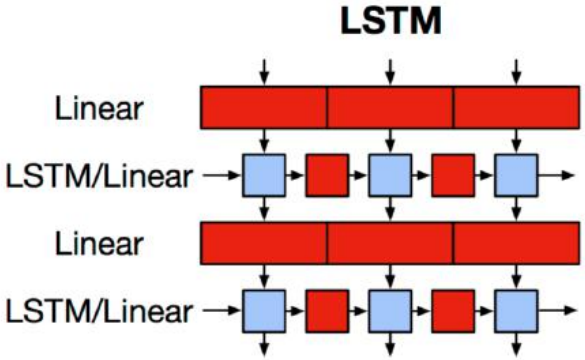
The current word is in red and the size of the blue shade indicates the activation level.

Outline

- Introduction to Seq2Seq
- Machine Translation
- Attention
- Transformer

Transformer

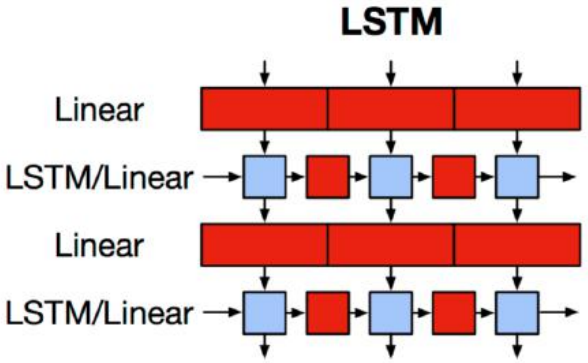
- Motivations
  - Sequential computation in RNNs prevents parallelization



  - Despite using GRU or LSTM, RNNs still need attention mechanism which provides access to any state
  - Maybe we do not need RNN?

Transformer

- Motivations
  - Sequential computation in RNNs prevents parallelization

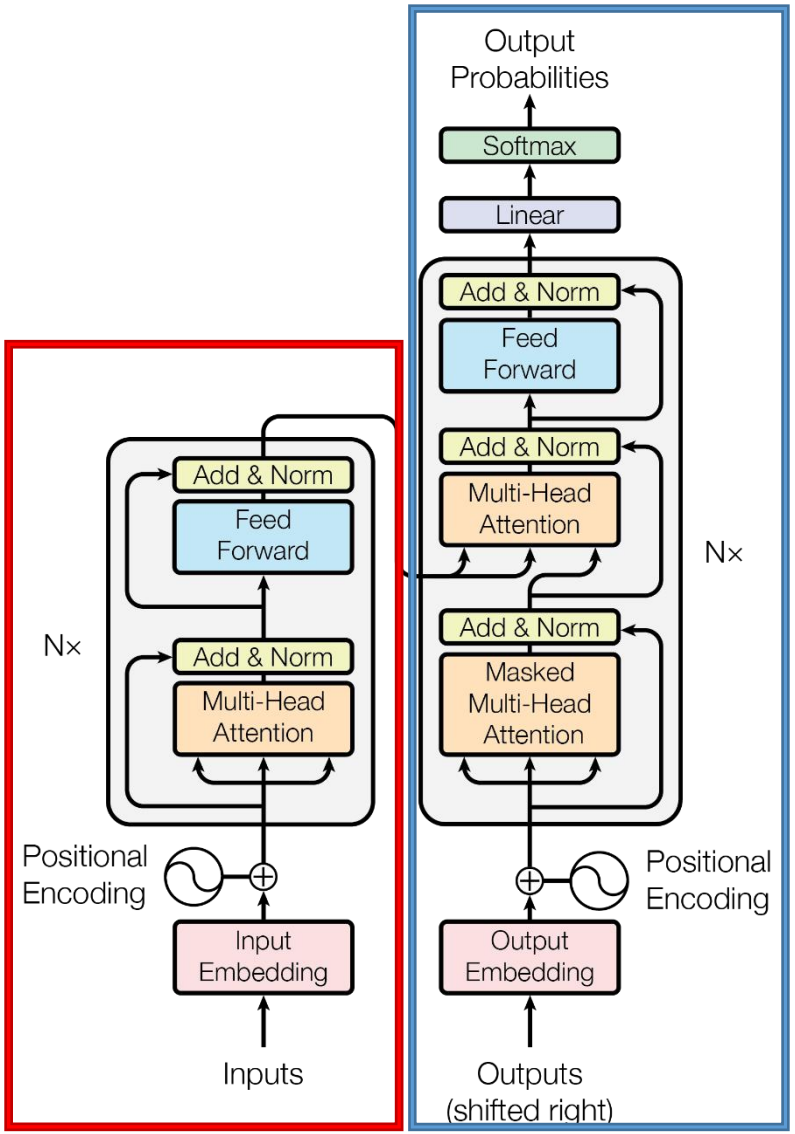


  - Despite using GRU or LSTM, RNNs still need attention mechanism which provides access to any state
  - Maybe we do not need RNN?
  - Attention is all you need (2017)

Attention is all you need, Vaswani at el. 2017

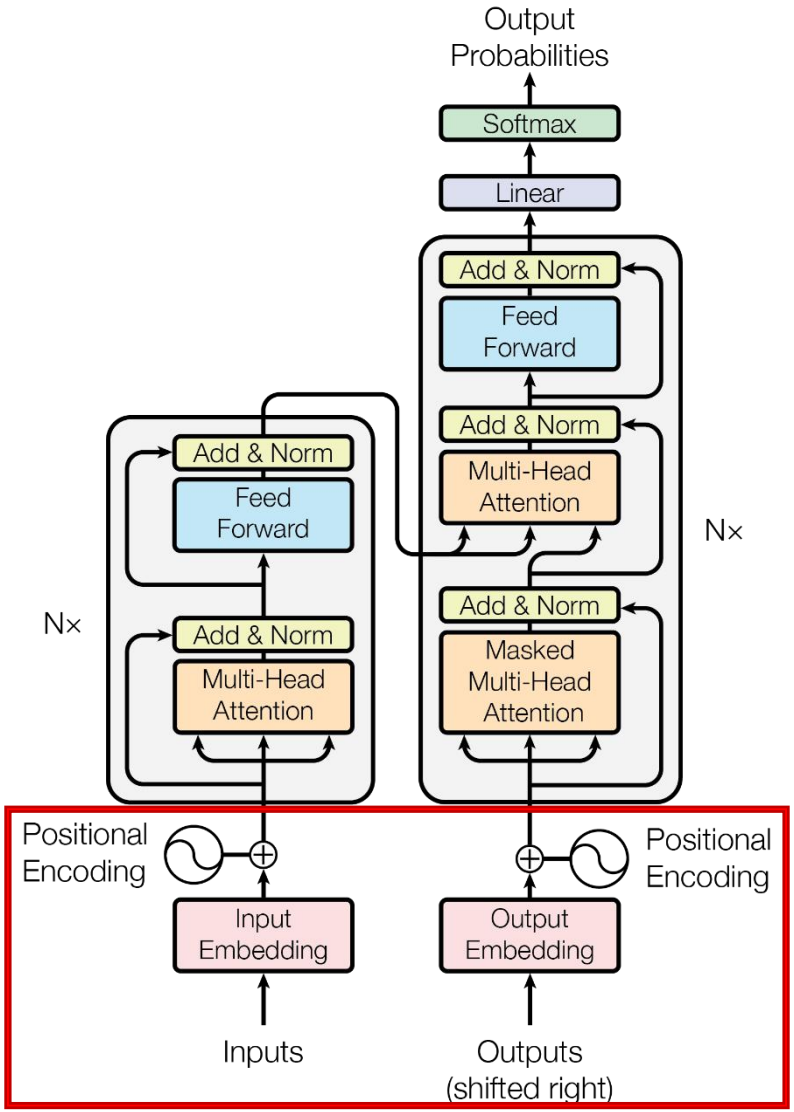
Transformer

- Overview
  - Architecture: encoder-decoder



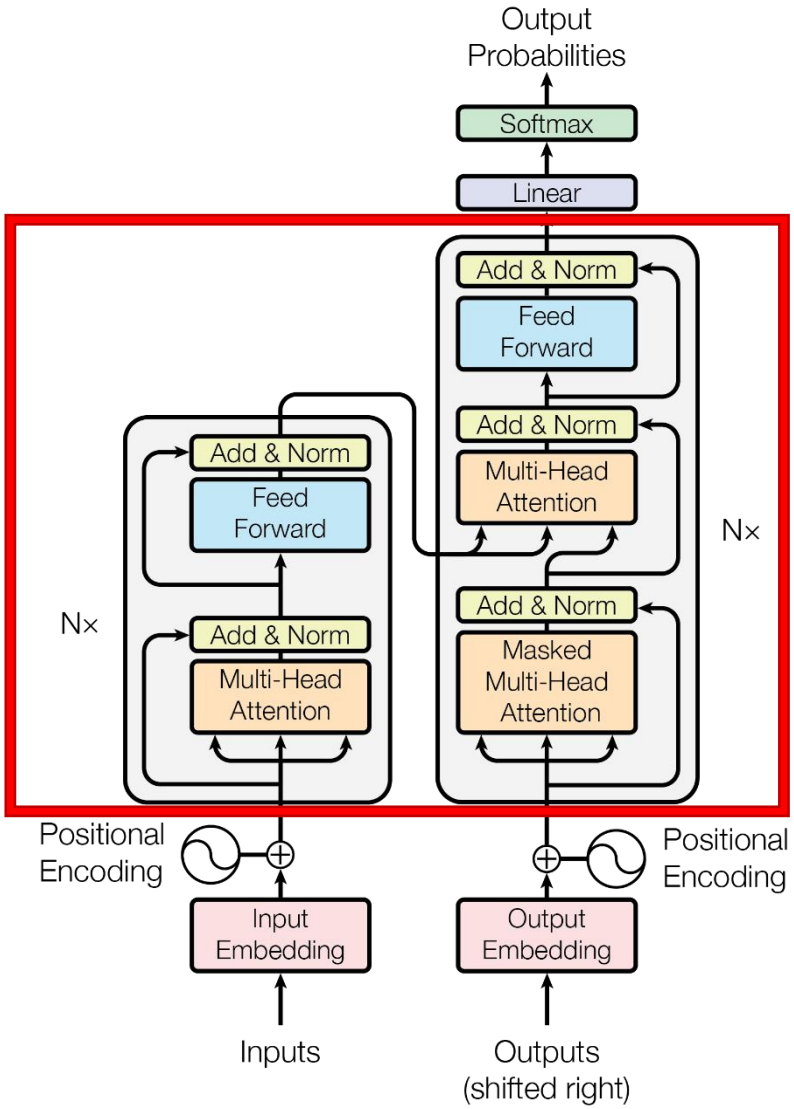
Transformer

- Overview
  - Architecture: encoder-decoder
  - Input: byte pair encoding + positional encoding



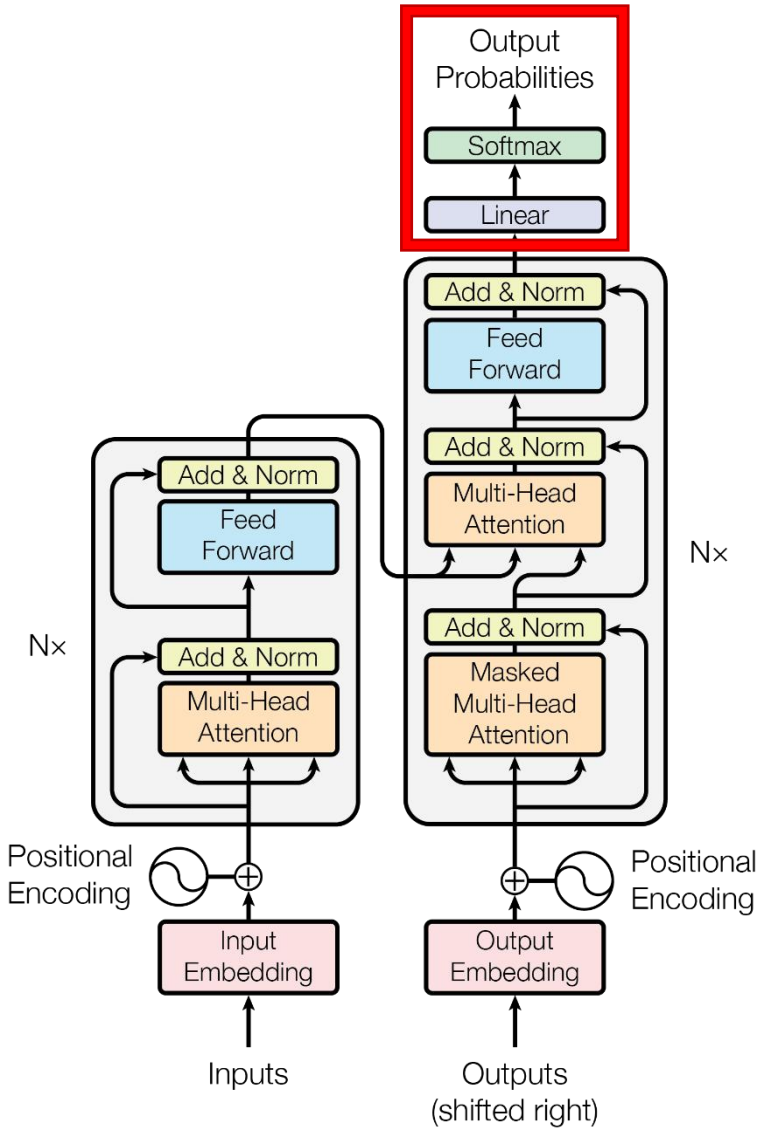
Transformer

- Overview
  - Architecture: encoder-decoder
  - Input: byte pair encoding + positional encoding
  - Model: stack of several encoder/decoder blocks



Transformer

- Overview
  - Architecture: encoder-decoder
  - Input: byte pair encoding + positional encoding
  - Model: stack of several encoder/decoder blocks
  - Output: probability of the translated word
  - Loss function: standard cross-entropy loss over a softmax layer



Outline

- Introduction to Seq2Seq
- Machine Translation
- Attention
- Transformer
  - Input Layer
  - Transformer Block
  - Performance

Input Encoding

- Byte Pair Encoding (BPE)
  - A word segmentation algorithm
    - Start with a vocabulary of characters
    - Turn the most frequent n-gram to a new n-gram

Dictionary

Freq	Word
5	l o w
2	l o w e r
6	n e w e s t
3	w i d e s t

Vocabulary

l, o, w, e, r, n, w, s, t, i, d

Start with all characters.

Input Encoding

- Byte Pair Encoding (BPE)
  - A word segmentation algorithm
    - Start with a vocabulary of characters
    - Turn the most frequent n-gram to a new n-gram

Dictionary

Freq	Word
5	l o w
2	l o w e r
6	n e w **es** t
3	w i d **es** t

Vocabulary

l, o, w, e, r, n, w, s, t, i, d, **es**

Add a pair (e, s) with freq 9.

Input Encoding

- Byte Pair Encoding (BPE)
  - A word segmentation algorithm
    - Start with a vocabulary of characters
    - Turn the most frequent n-gram to a new n-gram

Dictionary

Freq	Word
5	l o w
2	l o w e r
6	n e w **est**
3	w i d **est**

Vocabulary

l, o, w, e, r, n, w, s, t, i, d, es, **est**

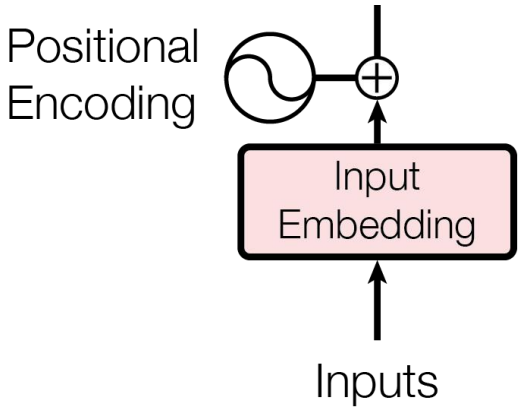
Add a pair (es, t) with freq 9.

Input Encoding

- Byte Pair Encoding (BPE)
  - Solve the OOV (out of vocabulary) problem by encoding rare and unknown words as sequences of subword units.
  - In the example above, the OOV word "lowest" would be segmented into "low est".
  - The model which used BPE got top places in WMT 2016.

Neural Machine Translation of Rare Words with Subword Units, Sennrich at el. 2016

Input Encoding

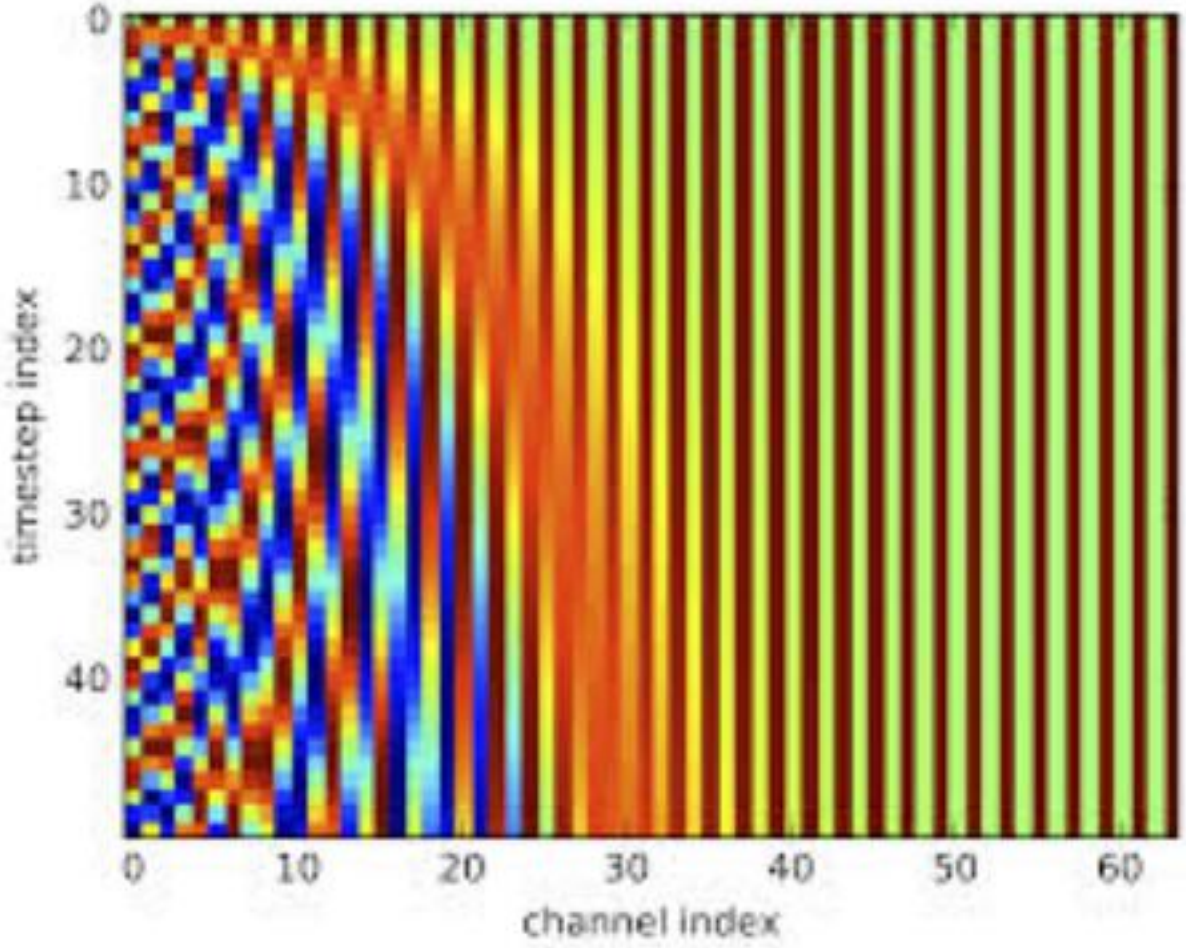


- Input Encoding
  - Byte Pair Encoding (BPE)
    - Dimension: d
  - Positional Encoding (PE)
    - The Transformer block is not sensitive to the same words with different positions.
    - The positional encoding is added so that the same words at different locations have different representations.

$$PE_{(pos,\ 2i)} = \sin(pos/10000^{2i/d})$$
$$PE_{(pos,\ 2i+1)} = \cos(pos/10000^{2i/d})$$

    - i is the index of embedding, ranges from 0 to d/2
  - Input = BPE + PE

Visualization of PE



From Stanford CS224N-2018

Outline

- Introduction to Seq2Seq
- Machine Translation
- Attention
- Transformer
  - Input Layer
  - Transformer Block
  - Performance

Attention Layer

- General Dot-Product Attention
  - Inputs
    - A query **q** and a set of key-value (**k**, **v**) pairs
    - Queries and keys are vectors with dimension d_k
    - Values are vectors with dimension d_v
  - Output
    - Weighted sum of values
    - Weight of each value is computed by the dot product of the query and corresponding key

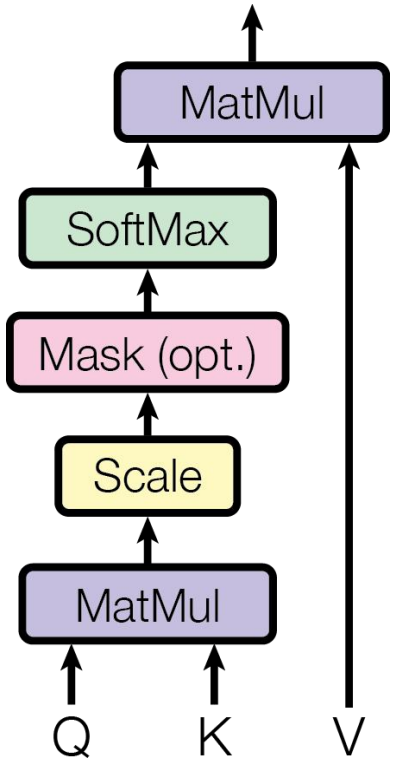
$$A(q,\ K,\ V) = \sum_i \frac{e^{q \cdot k_i}}{\sum_j e^{q \cdot k_j}} v_i$$

    - stack multiple queries q in a matrix Q

$$A(Q,\ K,\ V) = \text{softmax}(QK^T)V$$

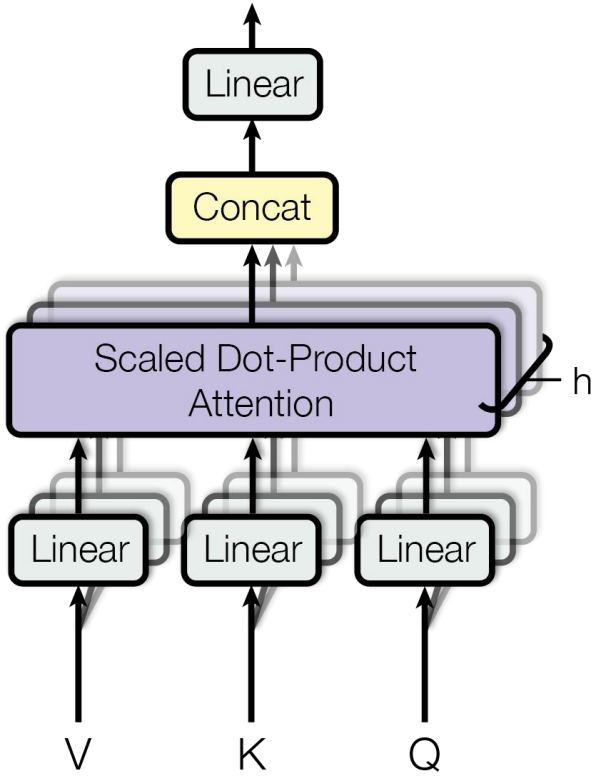
Attention Layer

- Scaled Dot-Product Attention
  - Problem
    - As d_k gets large, the variance of $q^T \cdot k$ increases
    - The softmax gets very peaked; Gradient gets smaller
  - Solution
    - Scale by the length of the query/key vectors
    - $A(Q,\ K,\ V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$



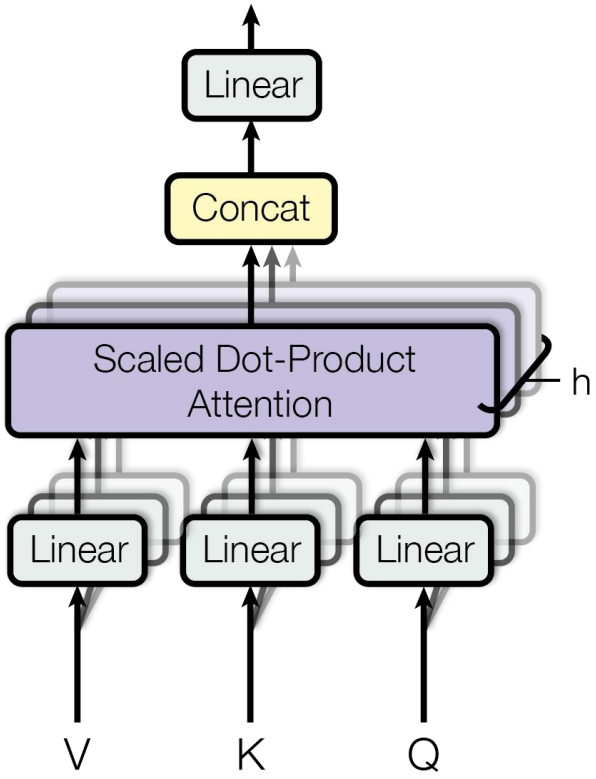
Attention Layer

- Self-attention
  - Let the word vectors themselves select each other
  - Stack word vectors of a sentence as Q, K, V



Attention Layer

- Multi-head Attention
  - Different head: same computation component & different parameters
  - Concatenate all outputs and feed into the linear layer
  - For h heads, we have:

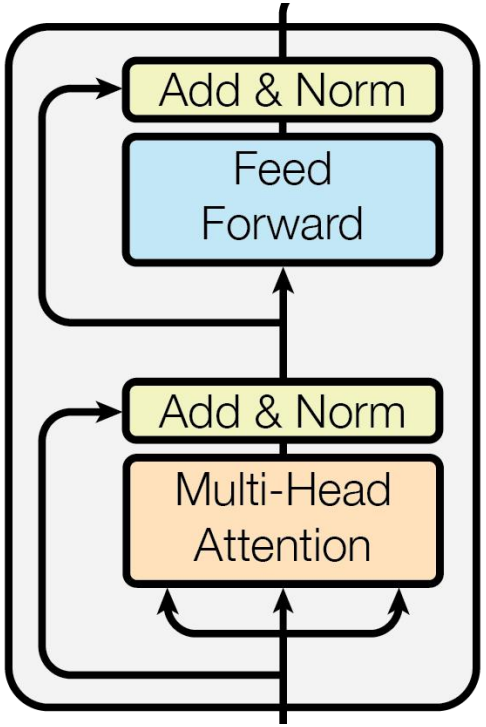


$$head_i = \mathrm{A}\left(QW_i^Q,\ KW_i^K, VW_i^V\right)$$

$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(head_1,\ \ldots, head_h)W^O$$

Transformer Block

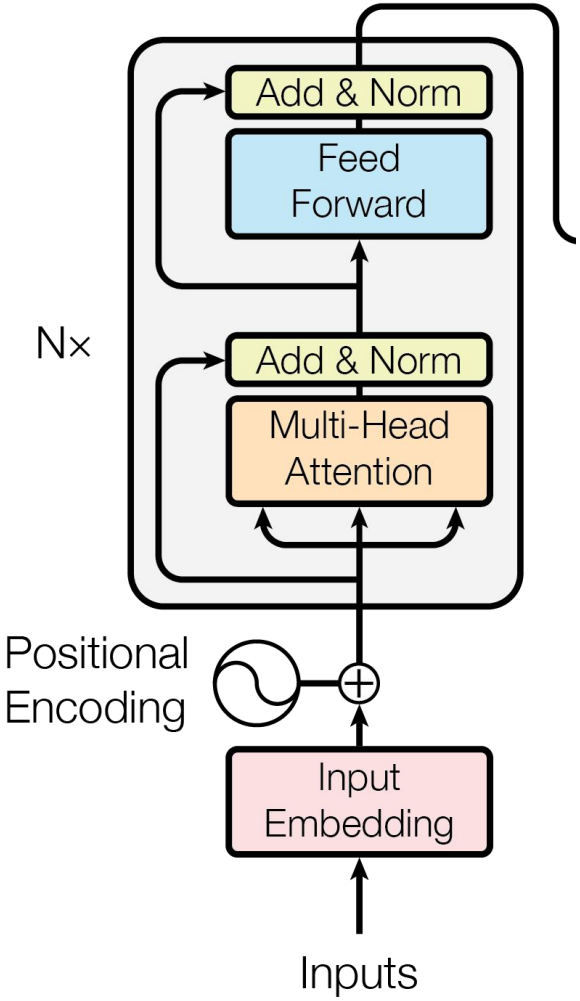
- Transformer block
  - Two sublayers
    - Multi-head attention
    - 2-layer feed-forward network
  - Two tricks
    - Residual connection
    - Layer normalization
      - Changes input to have mean 0 and variance 1



Layer normalization. Ba et al. arXiv 2016.
Deep residual learning for image recognition. He et al. CVPR 2016.

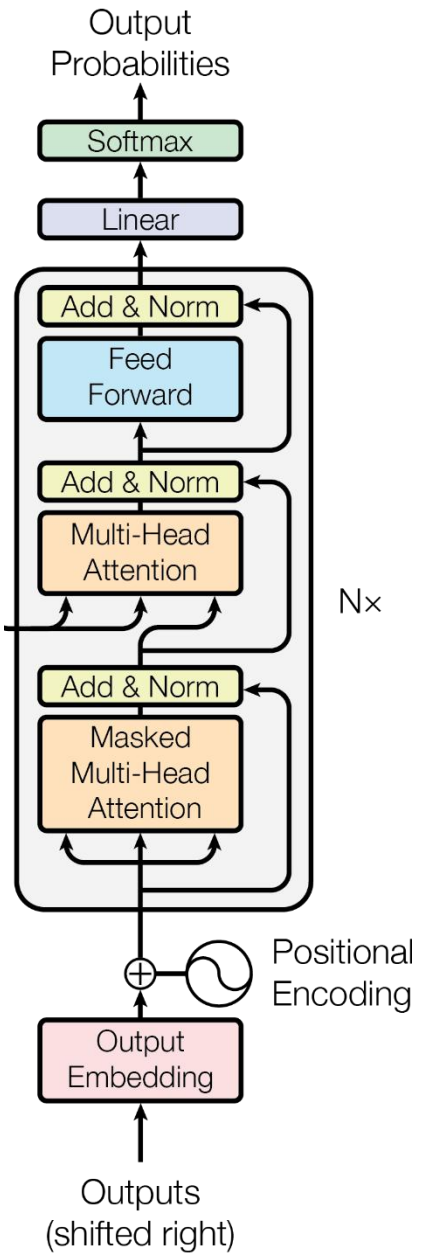
Transformer

- Complete Encoder
  - We use the same Q, K, V from the previous layer
  - The number of blocks N is set to 6 according to experimental results



Transformer

- Complete Decoder
  - Two changes:
    - Masked self-attention
      - The word can only look at previous words
    - Encoder-decoder attention
      - Queries come from the decoder while keys and values come from the encoder
  - Blocks are also repeated 6 times



Transformer

- Other tricks
  - Checkpoint averaging
  - ADAM optimizer
  - Dropout during training at every layer just before adding residual
  - Label smoothing
  - Auto-regressive decoding with beam search and length penalties

Outline

- Introduction to Seq2Seq
- Machine Translation
- Attention
- Transformer
  - Input Layer
  - Transformer Block
  - Performance

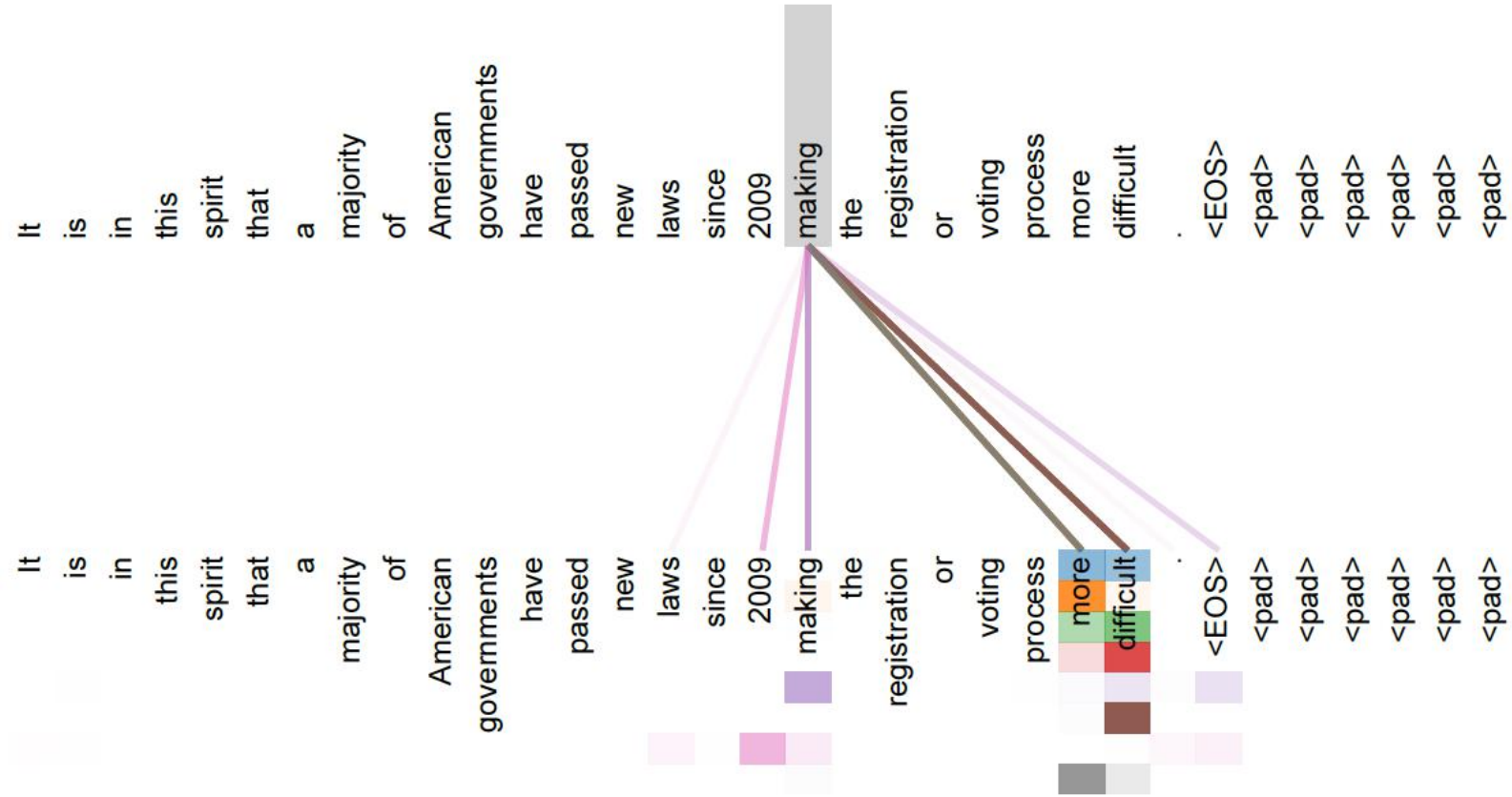
Transformer

- Experimental Results for Machine Translation

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

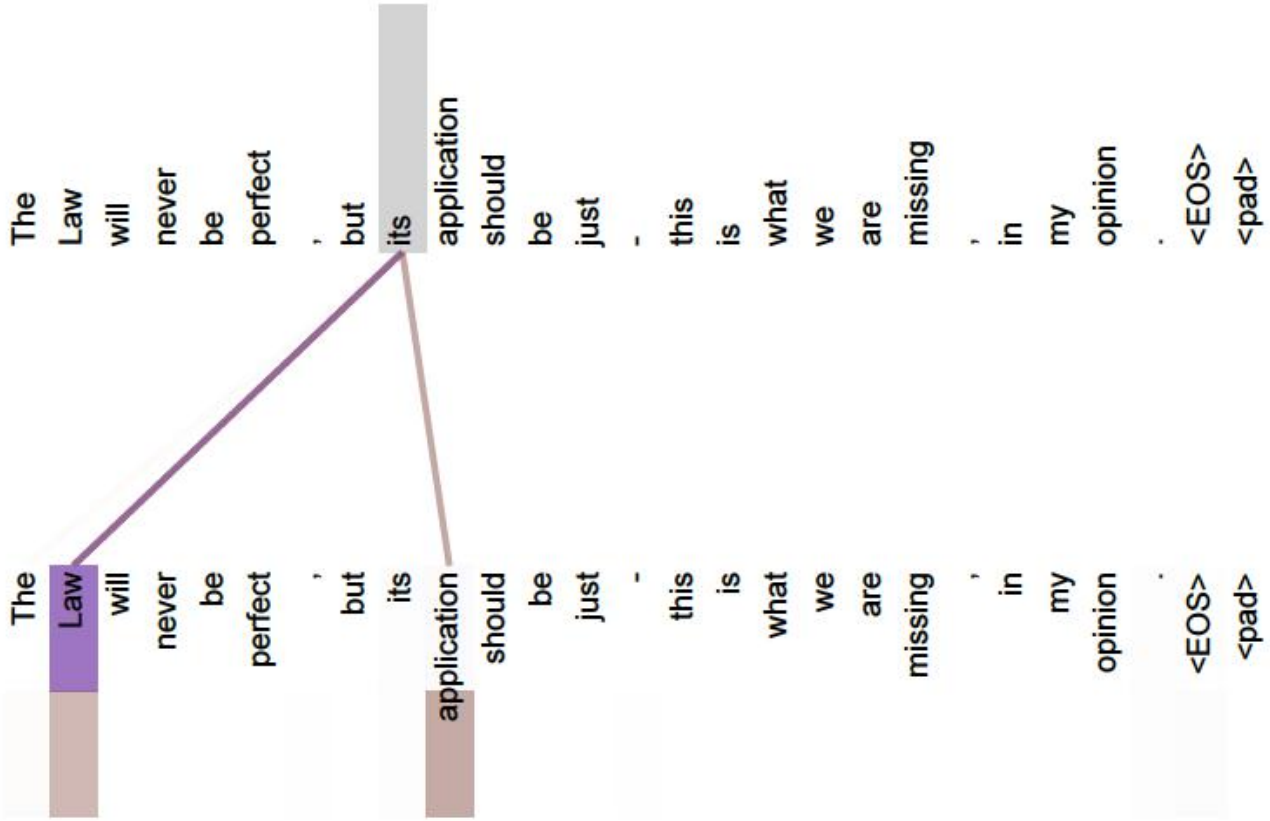
Transformer

- Attention Visualization in Layer 5
  - Words start to focus on other words accordingly



Transformer

- Attention Visualization
  - Implicit co-reference resolution



Transformer

- Conclusion
  - The Transformer is a powerful model and proven to be effective in many NLP tasks
  - It proves the effectiveness of the attention mechanism
  - It also gives insights to recent NLP advancements such as BERT and XLNet
  - However, the architecture is hard to optimize and sensitive to model modifications

Summary

- Seq2Seq: use RNN/GRU/LSTM/Transformer to perform encoder-decoder based tasks
  - Decoding strategy, evaluation
  - Various applications
- Machine translation
  - RBMT, EBMT, SMT, NMT
- Attention is critical for improving seq2seq models.
- Transformer is powerful in sequence modeling.

Reading Material

a. Machine Translation

- **Must-read Papers**

· The Mathematics of Statistical Machine Translation: Parameter Estimation. Peter EBrown, Stephen ADella Pietra, Vincent JDella Pietra, and Robert LMercer. Computational Linguistics 1993 [link]

· (Seq2seq) Sequence to Sequence Learning with Neural Networks. Ilya Sutskever, Oriol Vinyals, and Quoc VLe. NIPS 2014 [link]

· (BLEU) BLEU: a Method for Automatic Evaluation of Machine Translation. Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. ACL 2002 [link]

For reading material recommendation of this course, please refer to our github.

Reading Material

a. Machine Translation

- **Further Reading**

· Statistical Phrase-Based Translation. Philipp Koehn, Franz JOch, and Daniel Marcu. NAACL 2003 [link]

· Hierarchical Phrase-Based Translation. David Chiang. Computational Linguistics 2007 [link]

· (Beam Search) Beam Search Strategies for Neural Machine Translation. Markus Freitag and Yaser Al-Onaizan. 2017 [link]

· MT paper list. [link]

· THUMT toolkit. [link]

b. Attention

· Introduction to attention. [link]

· Neural Machine Translation by Jointly Learning to Align and Translate. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. ICLR 2015 [link]

For reading material recommendation of this course, please refer to our github.

Reading Material

c. Transformer

- Must-read Papers

· (Transformer) Attention is All You Need. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan NGomez, Lukasz Kaiser, and Illia Polosukhin. NIPS 2017 [link]

· (BPE) Neural Machine Translation of Rare Words with Subword Units. Rico Sennrich, Barry Haddow, and Alexandra Birch. ACL 2016 [link]

- Further Reading

· Illustrated Transformer. [link]

· Layer normalization. Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016 [link]

· Deep residual learning for image recognition. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. CVPR 2016 [link]

For reading material recommendation of this course, please refer to our github.


TSINGHUA UNIVERSITY
-1911-


Tsinghua University
NLP
Natural Language Processing

Q&A

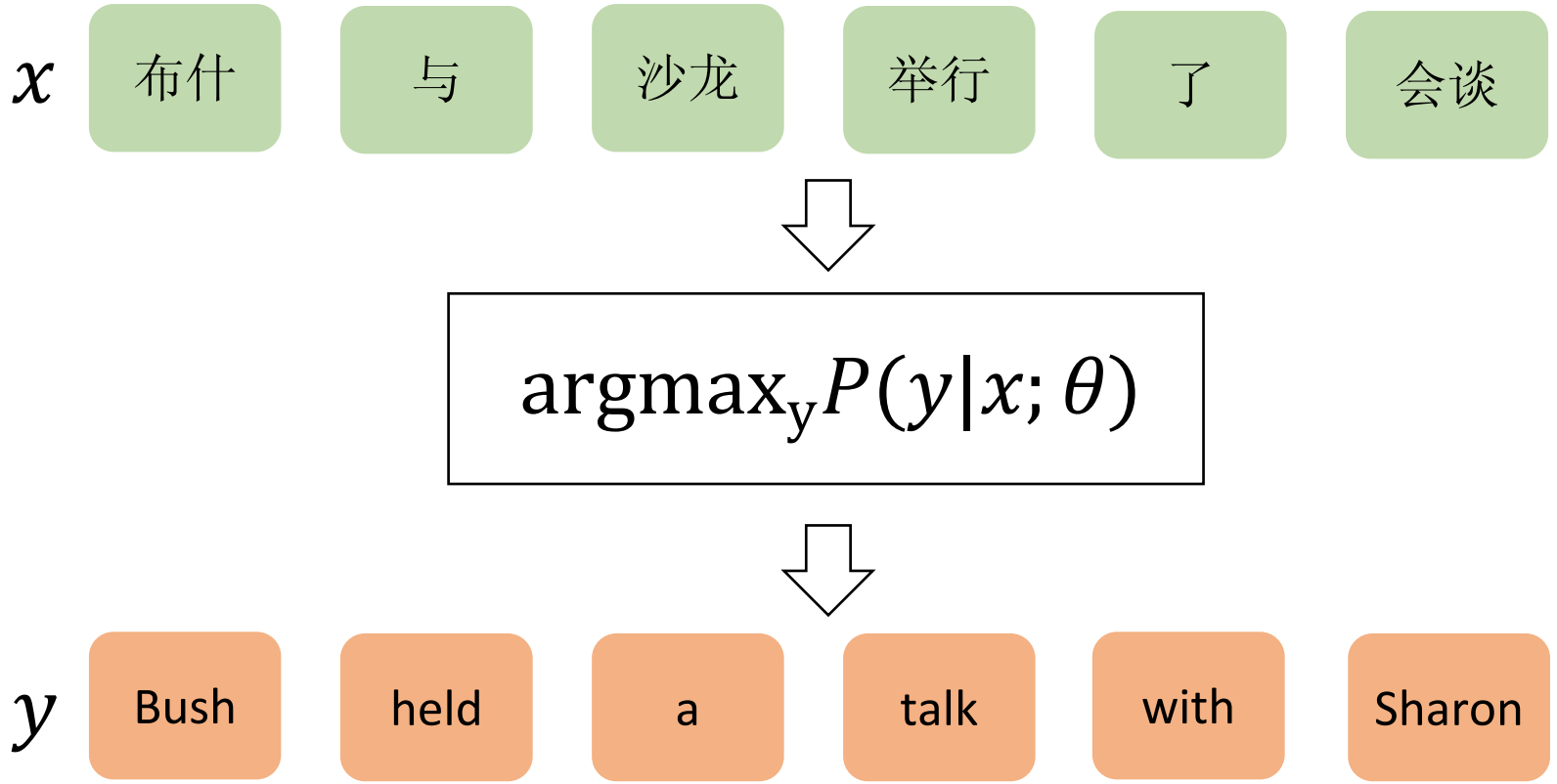
THUNLP

Appendix

THUNLP

Statistical Machine Translation

- Core idea: Learn a probabilistic model from data



Statistical Machine Translation

- Translation function:

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

- Meaning of two components:
  - $P(x|y)$
    - Translation model
    - How words and phrases should be translated (Learned from parallel data)
  - $P(y)$
    - Language model
    - How to write good English (Learned from monolingual data)

Statistical Machine Translation

- Translation function

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

- How to compute the argmax?
- Enumerate every possible y and calculate the probability
  - Too expensive!
- Answer: Use a heuristic search algorithm to gradually build up the the translation y

Heuristic Search Algorithm

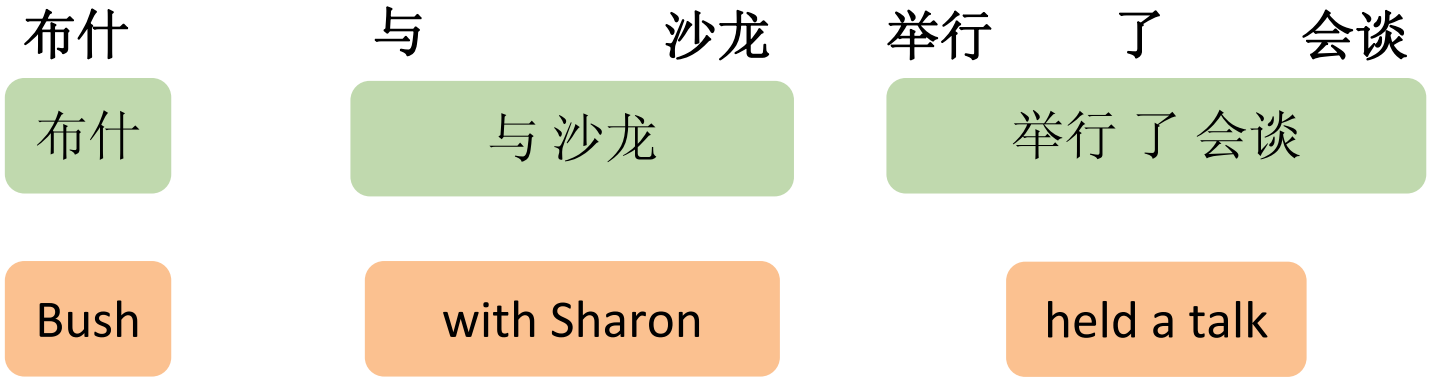
- Example of translation from Chinese to English

布什　　与　　沙龙　　举行　　了　　会谈

- Steps:

Heuristic Search Algorithm

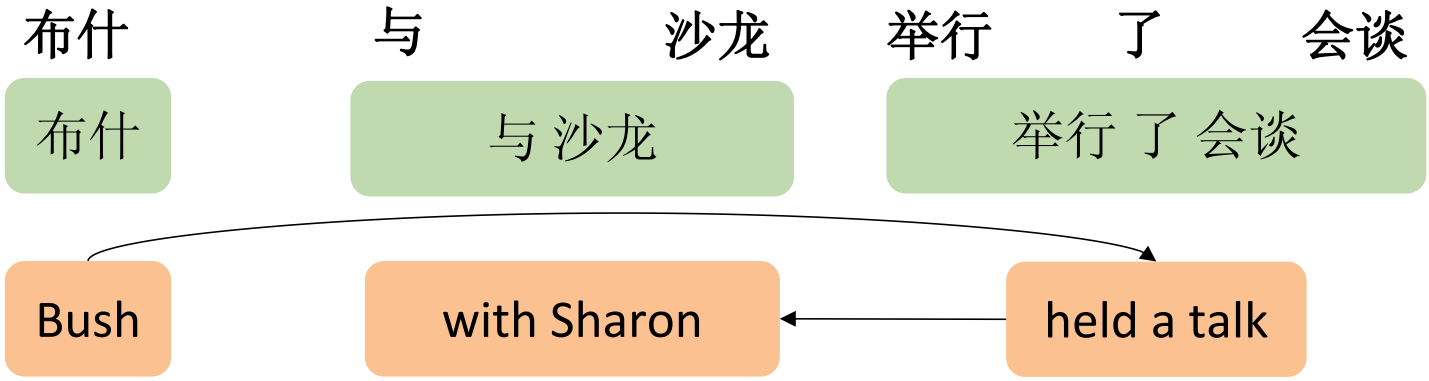
- Example of translation from Chinese to English



- Steps:
  - Pick phrases in input and translate. Phrases may have multiple words.

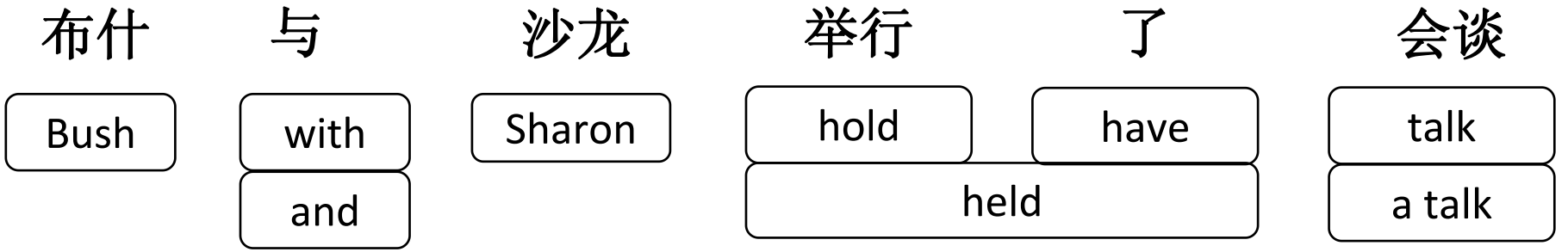
Heuristic Search Algorithm

- Example of translation from Chinese to English



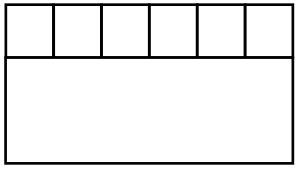
- Steps:
  - Pick phrases in input and translate. Phrases may have multiple words.
  - Allowed to pick phrases regardless of the original order.
  - Sentences with low probabilities are discarded.

Heuristic Search Algorithm



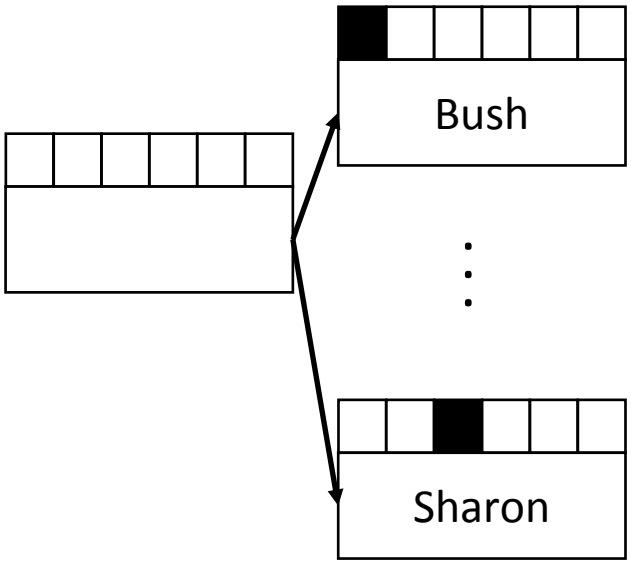
Heuristic Search Algorithm

布什	与	沙龙	举行	了	会谈
Bush	with	Sharon	hold	have	talk
	and		held		a talk

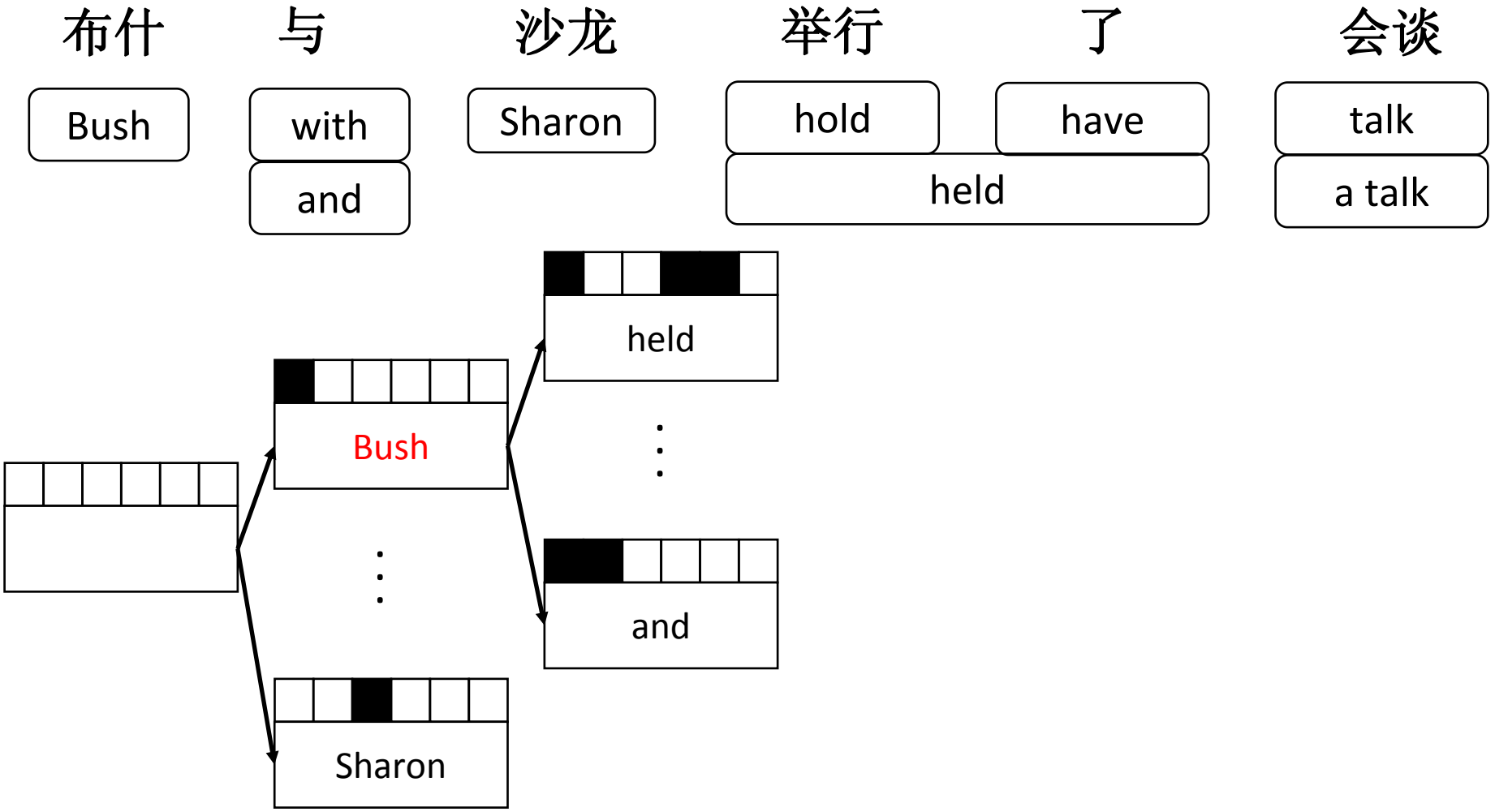


Heuristic Search Algorithm

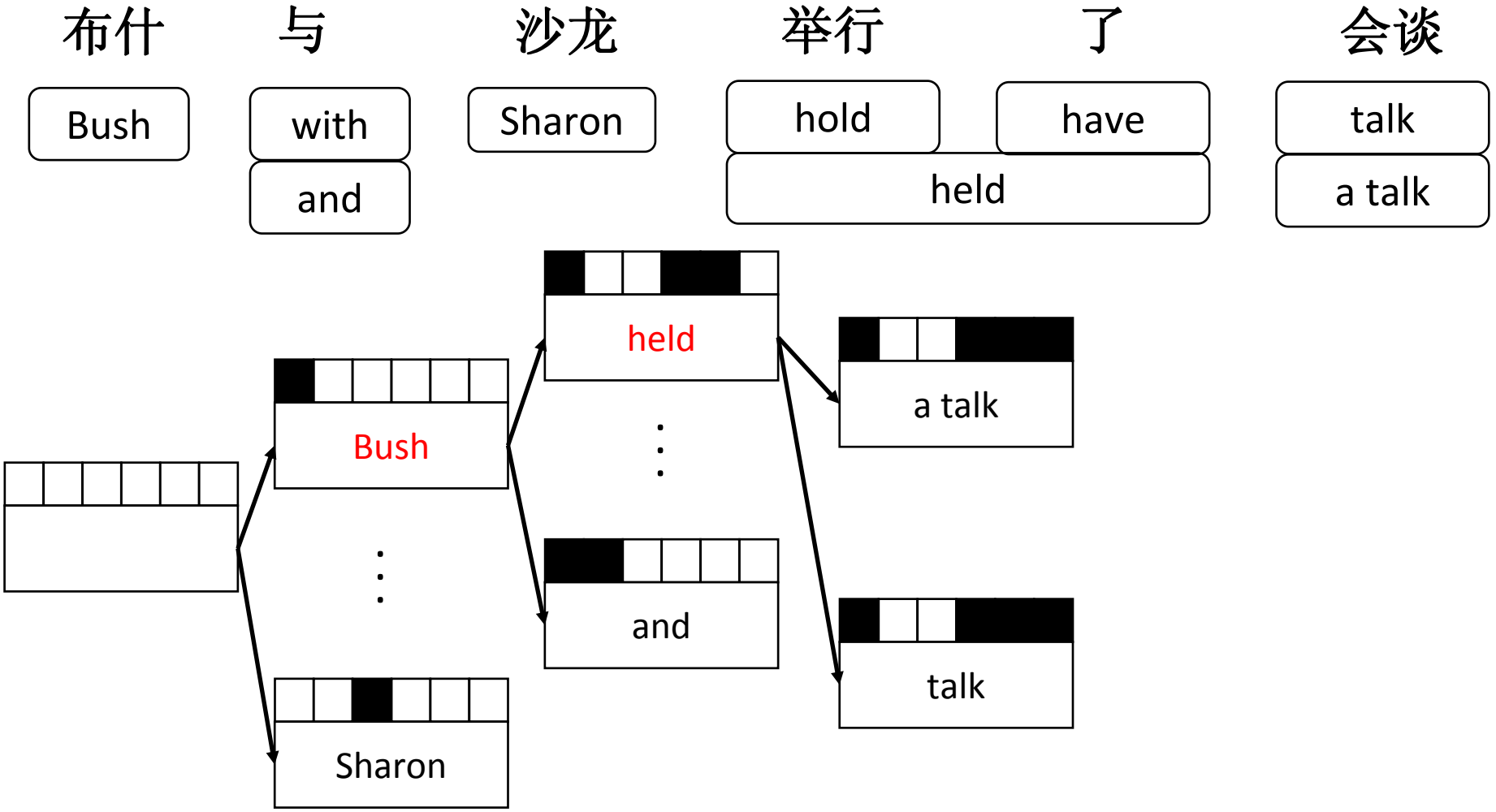
布什	与	沙龙	举行	了	会谈
Bush	with	Sharon	hold	have	talk
	and		held		a talk



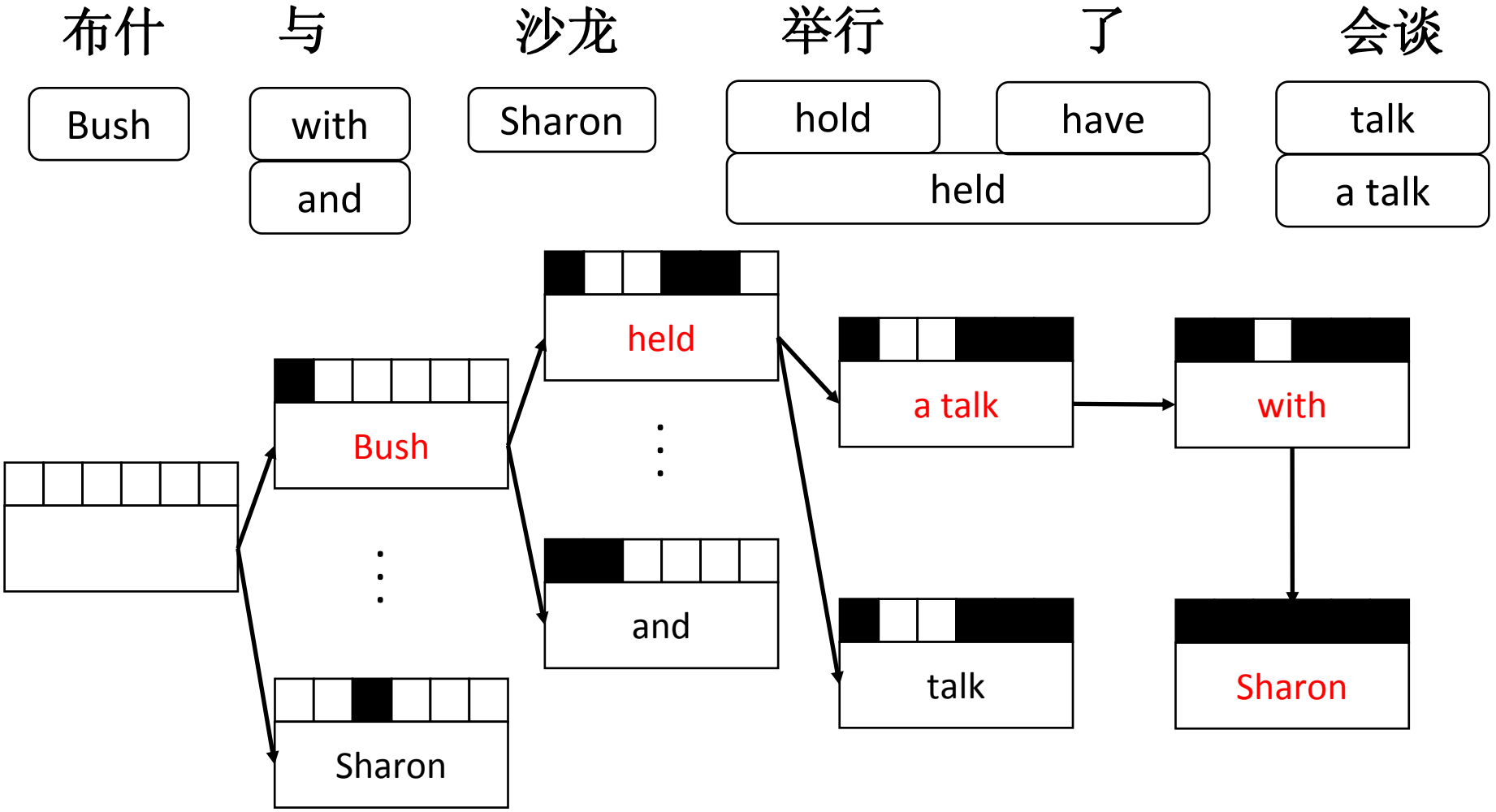
Heuristic Search Algorithm



Heuristic Search Algorithm



Heuristic Search Algorithm

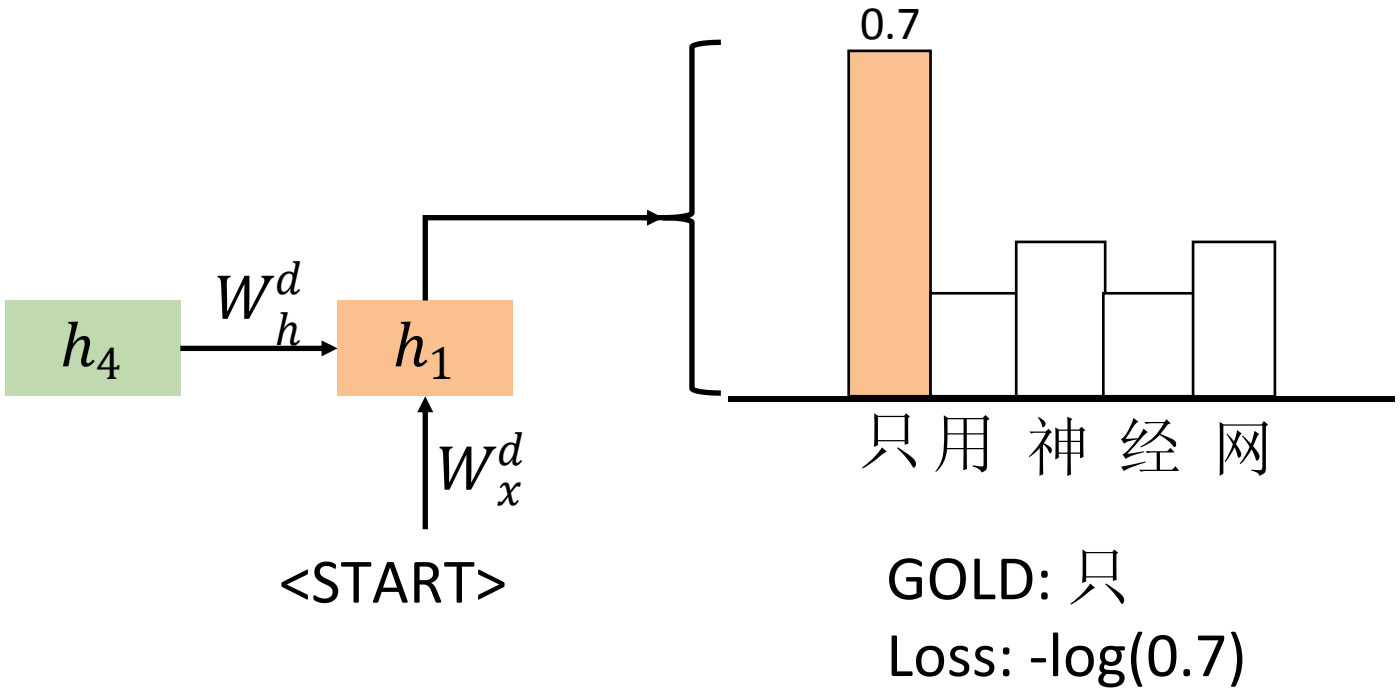


Statistical Machine Translation

- The best systems are extremely complex
  - Hundreds of important details we have not mentioned here
  - Systems have many separately-designed sub-components
  - Lots of feature engineering
    - Need to design features to capture particular language phenomena
  - Lots of human efforts to maintain

Training Seq2Seq

- Force the decoder to generate gold sequence
- Sum of losses for each token as the objective function
- Use cross-entropy loss

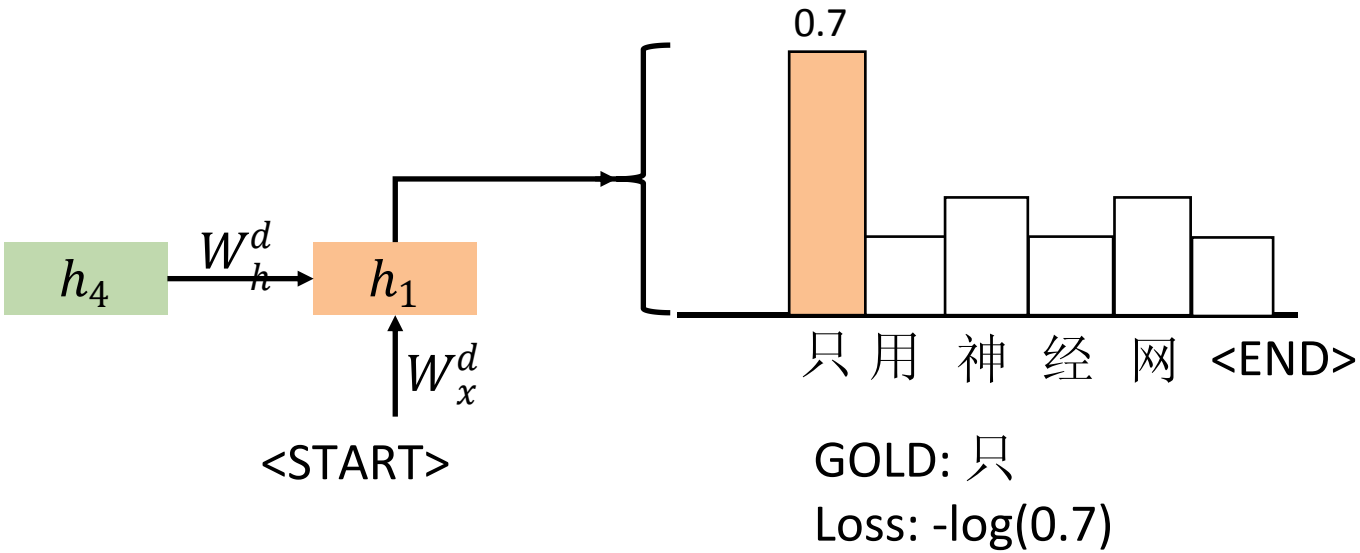


Training Seq2Seq

- Cross-entropy loss

$$-\sum_x p(x)\log q(x)$$

- $q(x)$: the distribution produced by the network
- $p(x)$: the true distribution (1 on the actual next token)

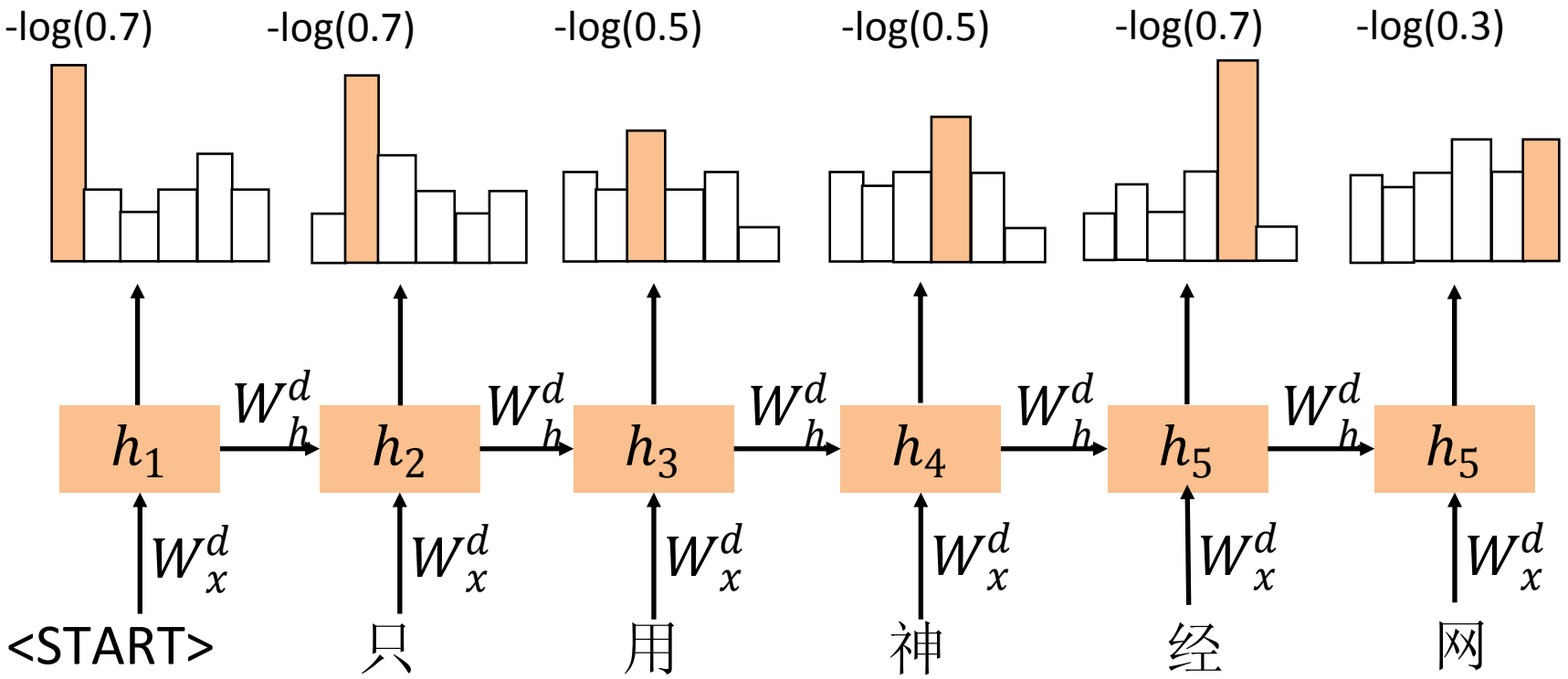


Training Seq2Seq

- Sum losses of each token for sentence loss J

$$\mathrm{J} = \mathrm{sum}(-3 * \log(0.7) - 2 * \log(0.5) - \log(0.3))$$

- Minimize the loss J for the given sentence pair



Training Seq2Seq

- Notice!
- Seq2seq is optimized as a unified system
- Backpropagation operates end-to-end
- Update two RNNs simultaneously
- Model parameters include word embeddings!

General Definition of Attention

- A more general definition of attention:
  - Given a query vector and a set of value vectors, the attention technique computes a weighted sum of the values according to the query.

- Intuition:
  - Based on the query, the weighted sum is a selective summary of the values.
  - We can obtain a fixed-size representation of an arbitrary set of representations via the attention mechanism.

General Definition of Attention

- Math Formulations
  - If we have values $\boldsymbol{h}_1, \boldsymbol{h}_2 \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and the query $\mathbf{s} \in \mathbb{R}^{d_2}$
  - The attention technique is used to compute the attention output $\mathbf{o} \in \mathbb{R}^{d_1}$ from the attention scores $\mathbf{e} \in \mathbb{R}^N$

$$\boldsymbol{\alpha} = \mathrm{softmax}(\boldsymbol{e}) \in \mathbb{R}^N$$
$$\boldsymbol{o} = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^h$$

  - There are several attention variants by different methods to compute $\mathbf{e} \in \mathbb{R}^N$

Attention Variants

- There are several attention variants by different methods to compute $e \in \mathbb{R}^N$
  - Basic dot-product attention

$$e_i = s^T h_i \in \mathbb{R}$$

    - This assumes the vector size $d_1 = d_2$

  - Multiplicative attention

$$e_i = s^T W h_i \in \mathbb{R}$$

    - $W \in \mathbb{R}^{d_2 \times d_1}$

Attention Variants

- There are several attention variants by different methods to compute $e \in \mathbb{R}^N$
  - Additive attention

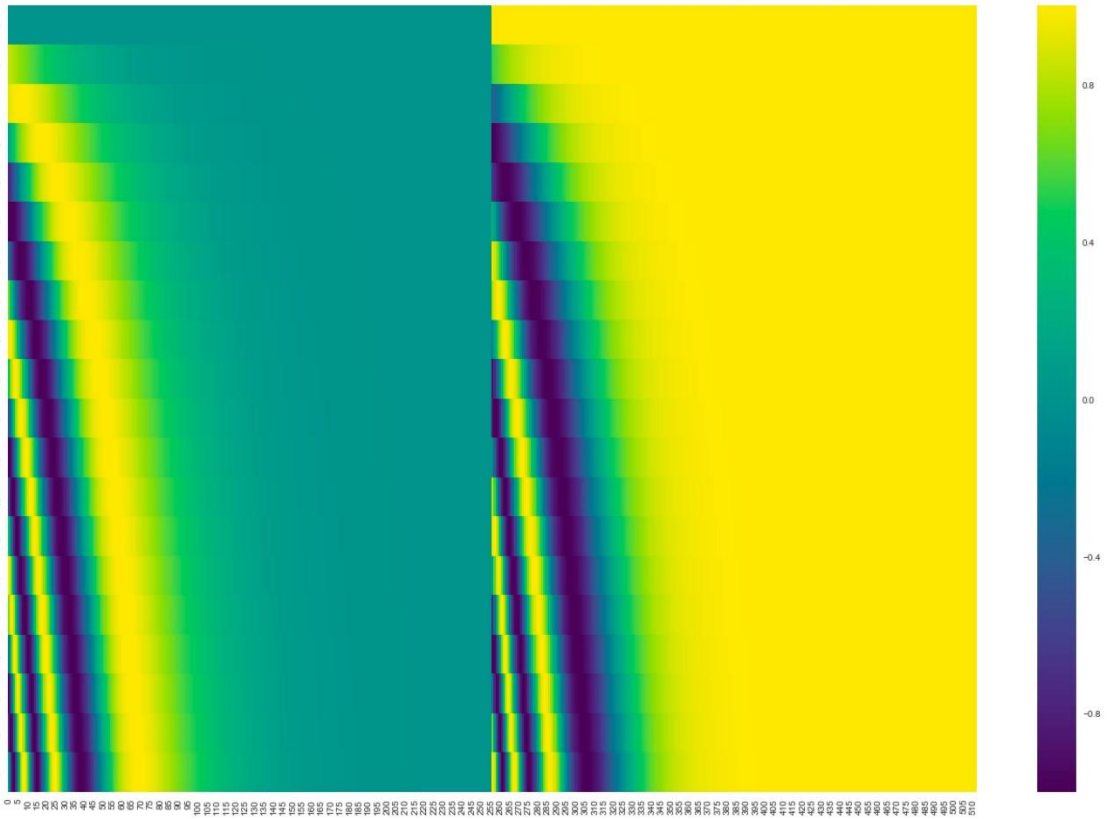
$$e_i = v^T \tanh(W_1 h_i + W_2 s) \in \mathbb{R}$$

    - Where $W_1 \in \mathbb{R}^{d_3 \times d_1}, W_2 \in \mathbb{R}^{d_3 \times d_2}$ are weight matrices and $v \in \mathbb{R}^{d_3}$ is a weight vector

  - More variants [link]

Neural Machine Translation by Jointly Learning to Align and Translate, Bahdanau at el. ICLR 2015

Visualization of PE



A real example of positional encoding for 20 words (rows) with an embedding size of 512 (columns). You can see that it appears to split in half down the center. That's because the values of the left half are generated by one function (which uses sine), and the right half is generated by another function (which uses cosine). They're then concatenated to form each of the positional encoding vectors.

http://jalammar.github.io/illustrated-transformer/