Transformer and Machine Translation

王 瑞（Wang, Rui）

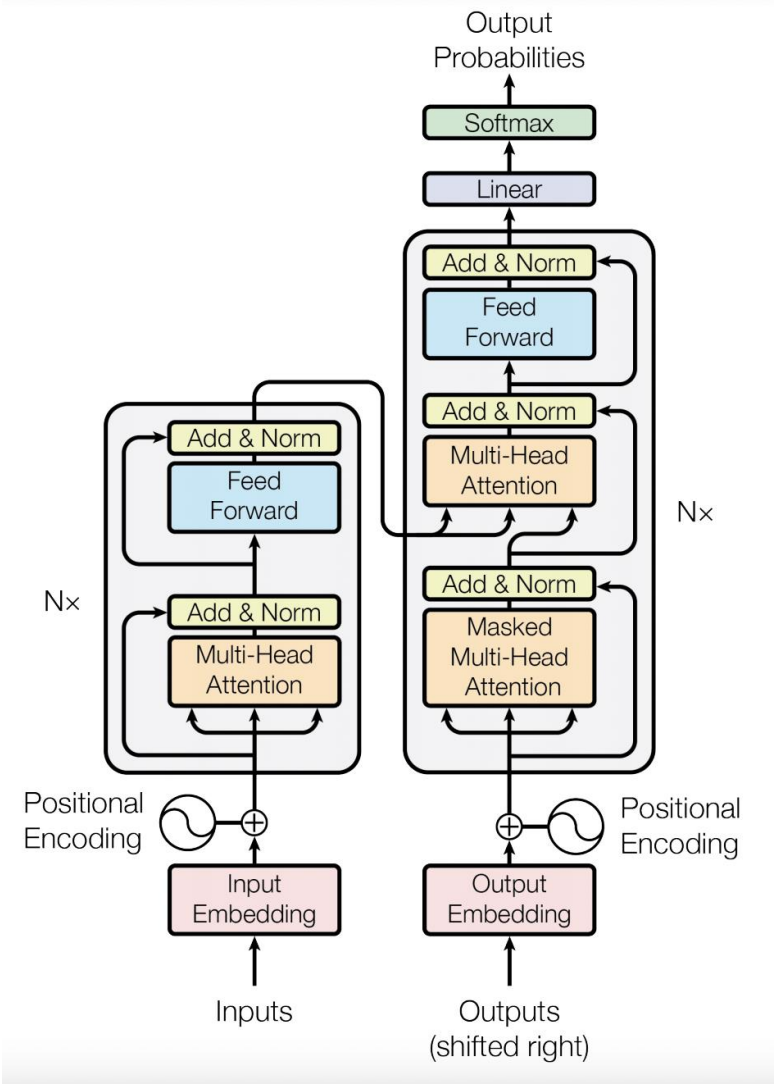
上海交通大学计算机系

Department of Computer Science and Engineering

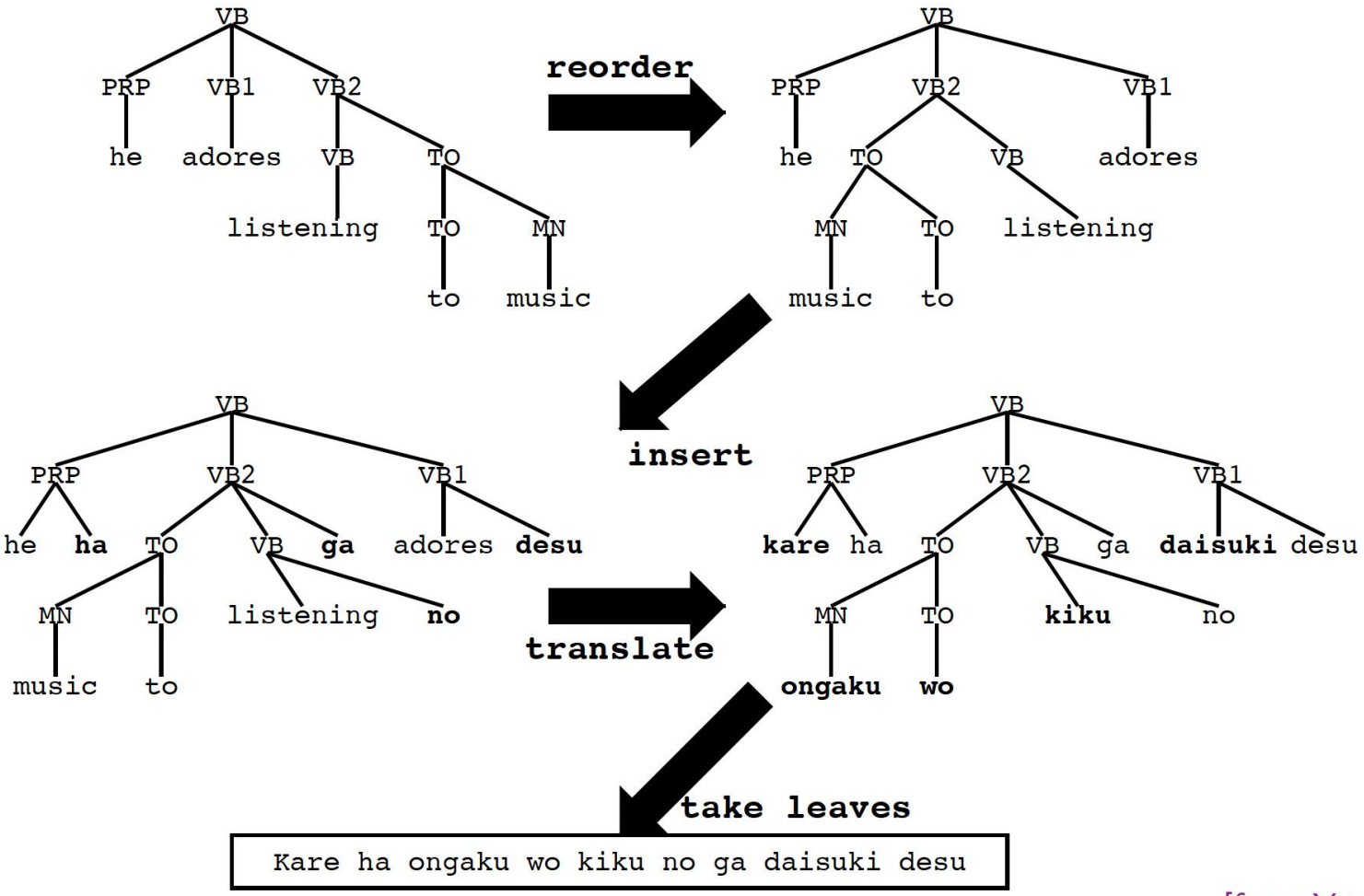
Shanghai Jiao Tong University

Menu

- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



Why I Introduce MT Before Transformer?



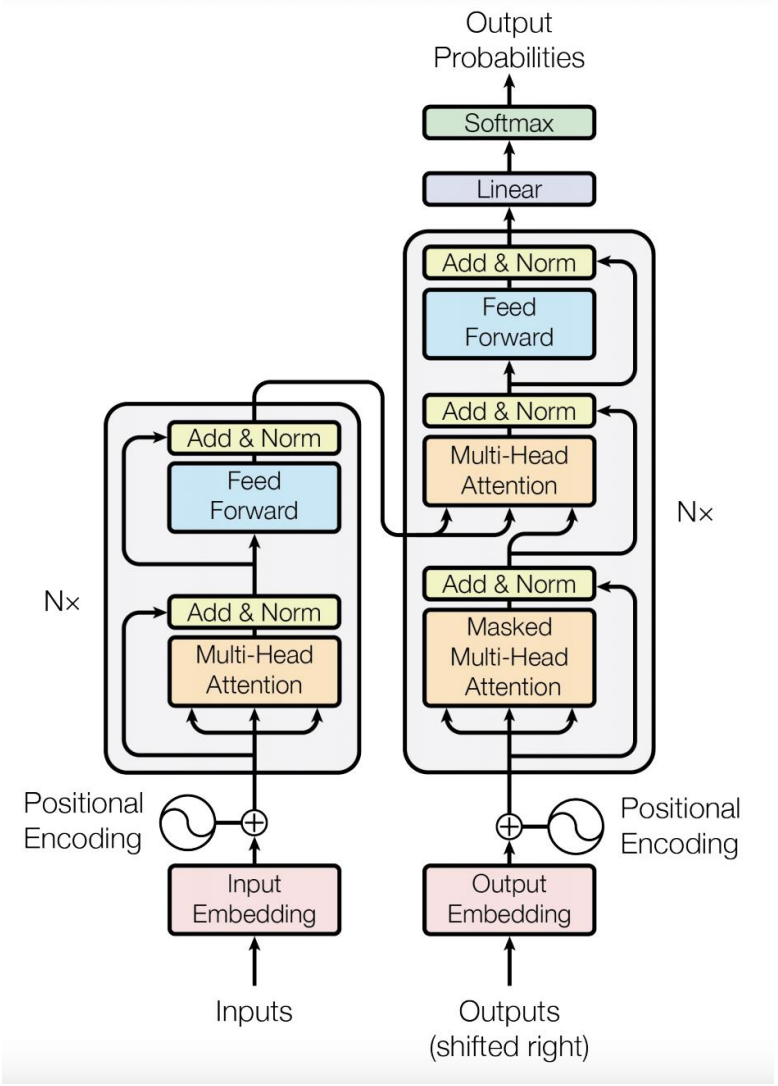
[from Yamada and Knight, 2001]

Why I Introduce MT Before Transformer?

- Transformer is designed for MT
- Only language has linguistic structural information
- Translation is a bilingual generation task
  - It is a sequence to sequence task
  - The two sequences have different orders

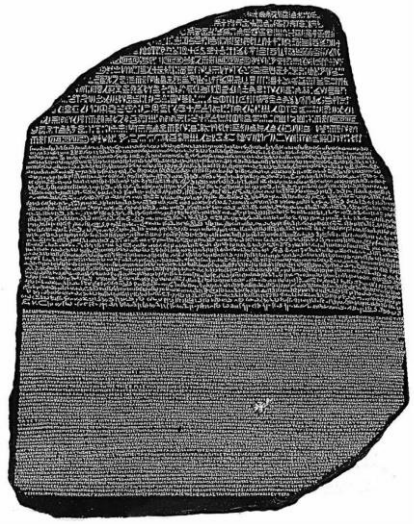
Menu

- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



MT: History

- Human Translation
  - 3rd~1st BC Bible Translation in West
  - 1st AD: Buddhism Translation in China



Ancient Egyptian (hieroglyphic)

Ancient Egyptian (Demotic)

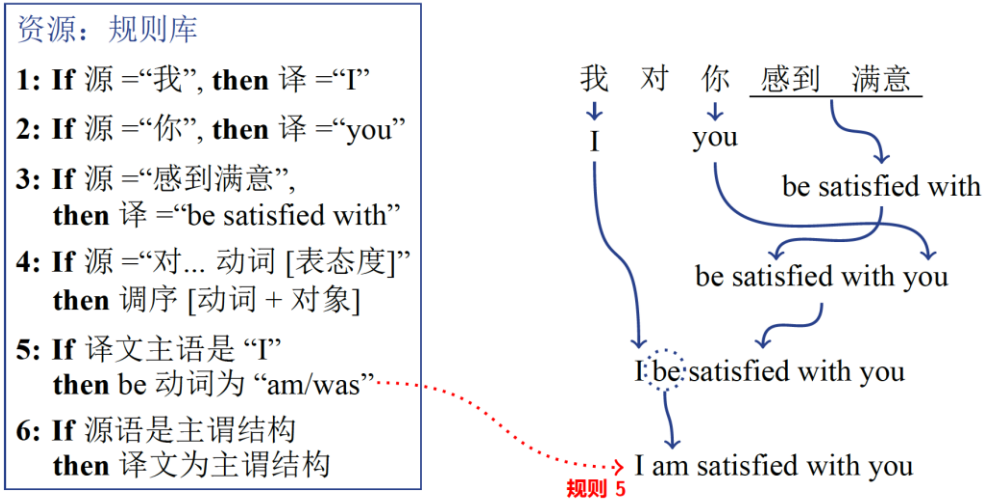
Ancient Greek

Rosetta Stone (196 BC)

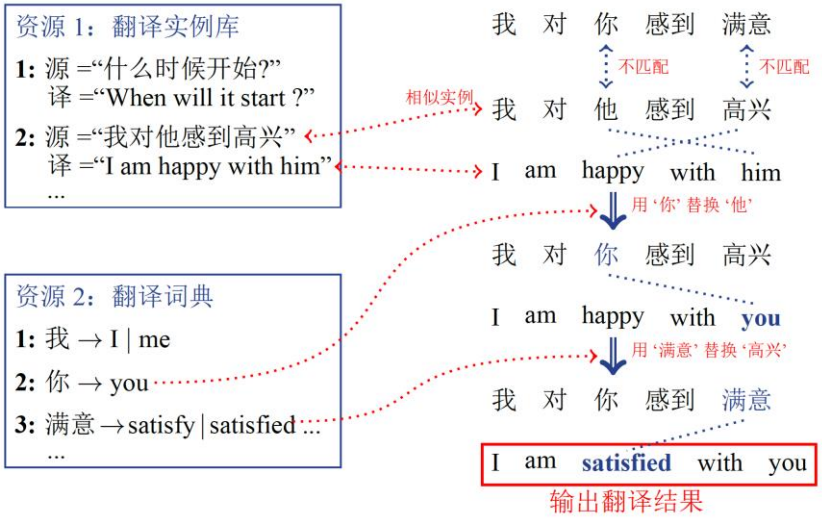
- Machine Translation:
  - Starting from 1949, treat the source language as an *encrypted* target language.
  - 1970s- Rule based MT.
  - 1980s- Example based MT.
  - 1990s- Statistical MT.
  - 2010s- Neural MT.

Rule & Example-based MT

- Rule-based MT:
  - Annotated linguistic rules



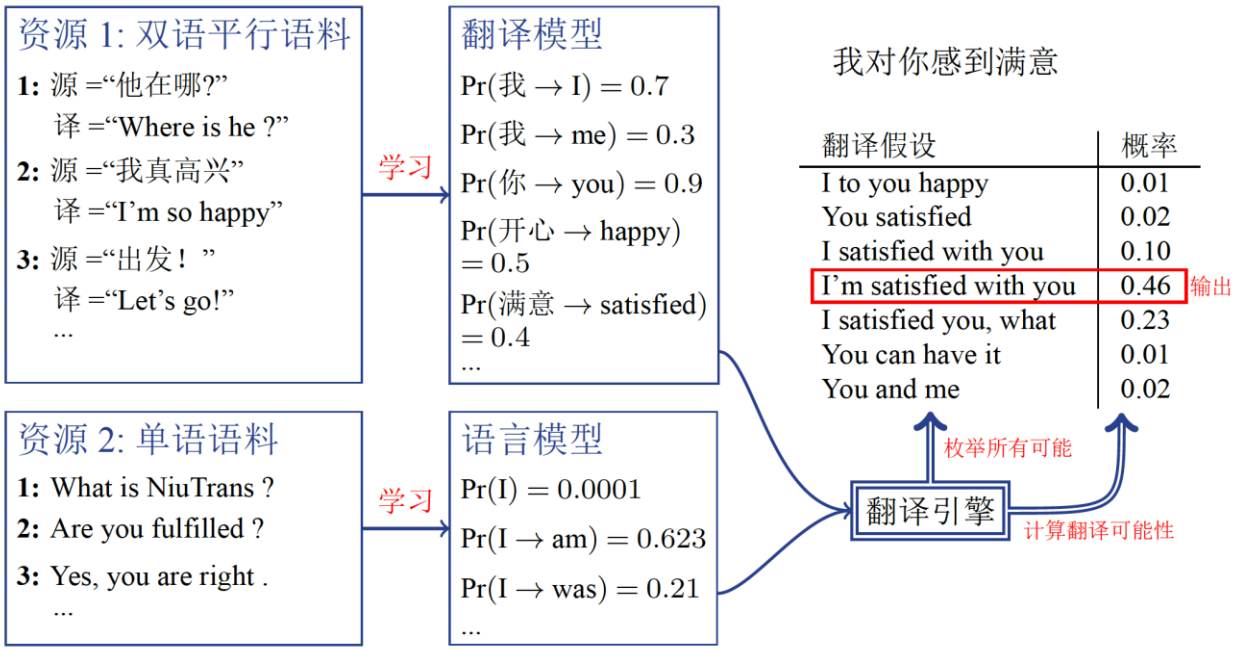
- Example-based MT:
  - Translation examples



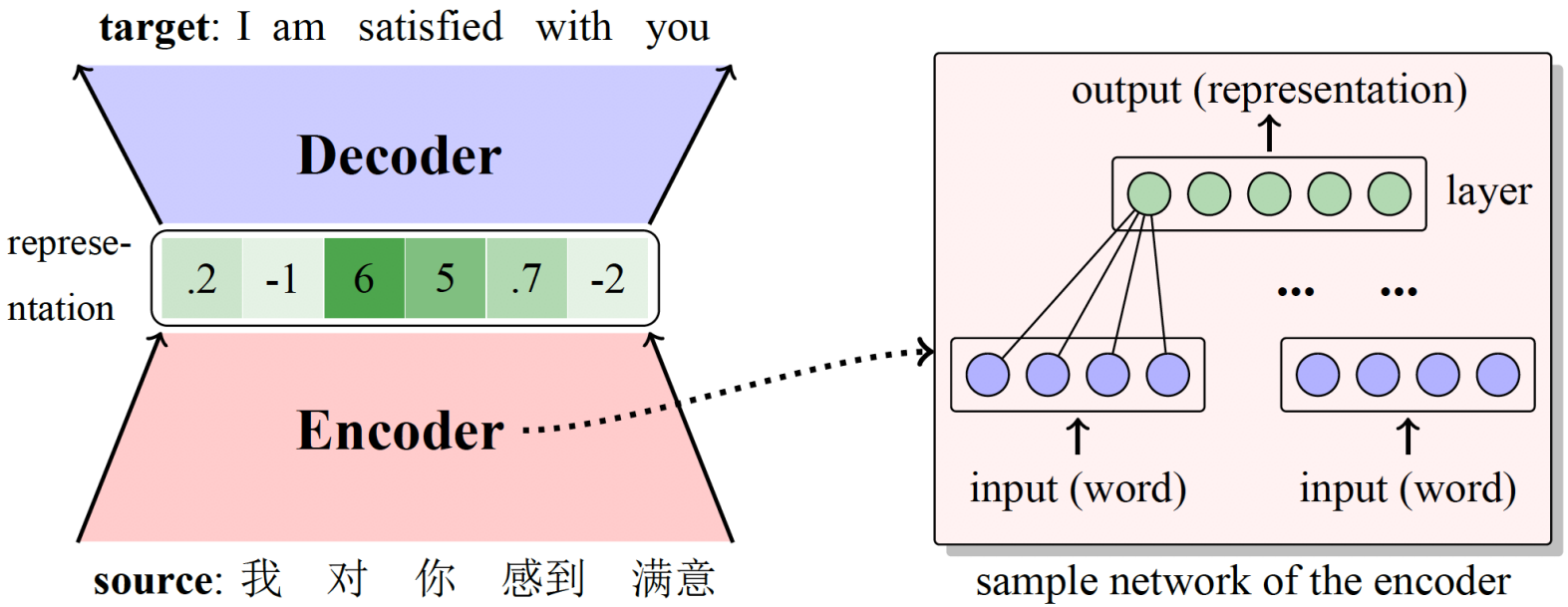
[Examples from Xiao and Zhu, MT-Book]

Statistical MT

- Statistical MT (SMT)
  - Parallel corpus: sentence-level alignment.
  - Monolingual corpus: *n*-grams probability.
  - To learn the translation rules statistically.



Neural MT (NMT)



- MT is a typical text generation task.

$$\mathcal{L}_{\mathbf{MLE}}(\theta) = -\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) = -\sum_{i=1}^{l} \log p_\theta(y_i|\boldsymbol{x}, \boldsymbol{y}_{<i})$$

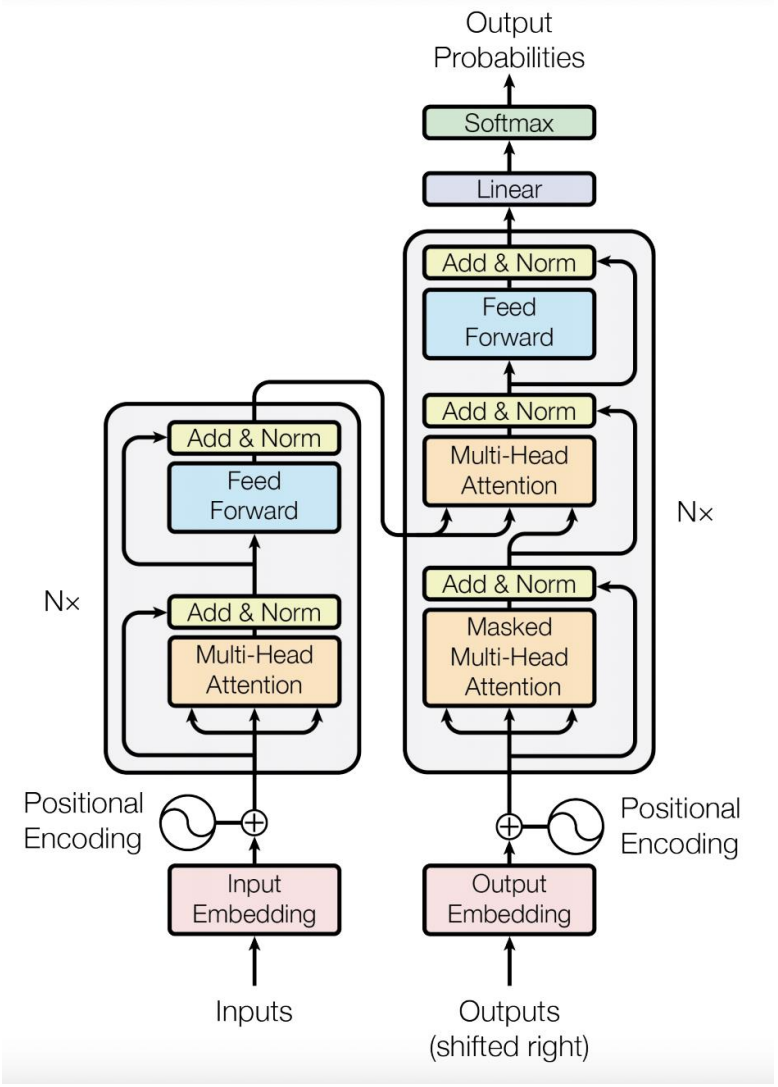
  - $\boldsymbol{x}$: source sentence; $\boldsymbol{y}$: target sentence.
  - maximum likelihood estimation (MLE):
- MT has a standard evaluation metric:

$$BLEU = \frac{\sum ngram_{correct}}{\sum ngram_{in\ \_\ reference}}$$

  - n-gram: contiguous sequence of n words.

Menu

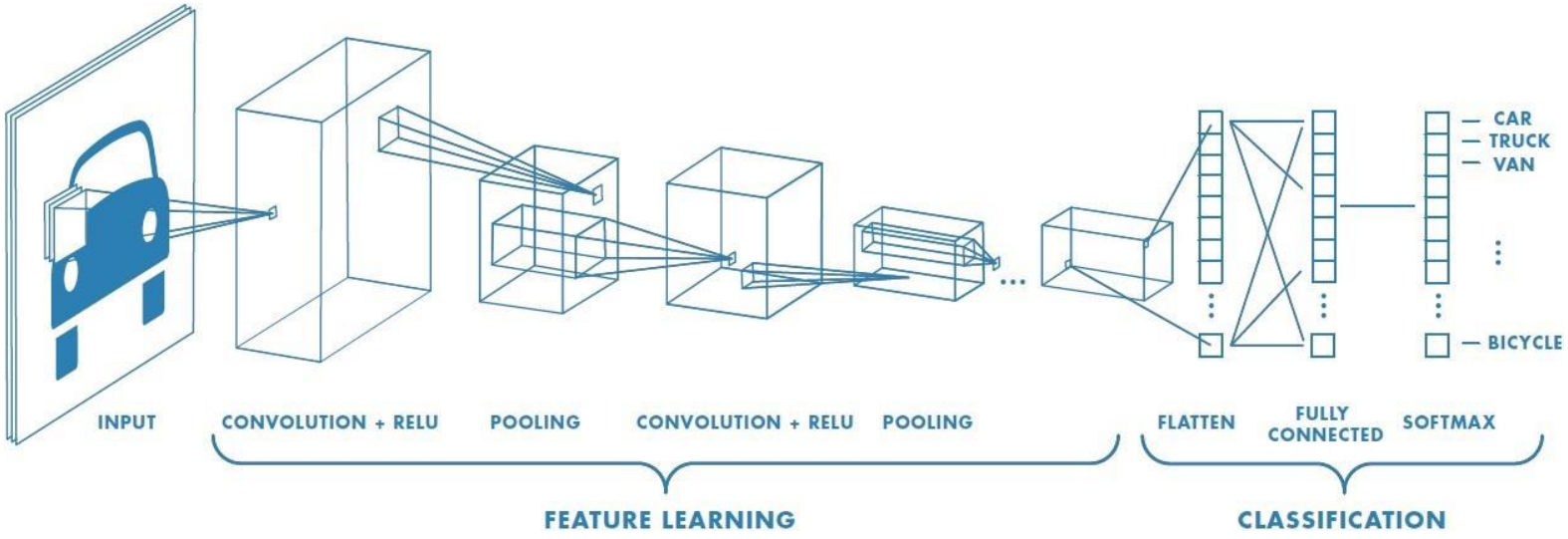
- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



From Shallow to Deep Learning

- Neural Network
  - Artificial neural network: from 1940s, develop not so fast, until 2000s
  - Deep neural network: from 2000s, when the power of GPU satisfies the need of research and industry
- Popular Application (2010s-now)
  - CNN: starting from computer vision
  - RNN: starting from speech recognition
  - Transformer/self-attention: starting from machine translation

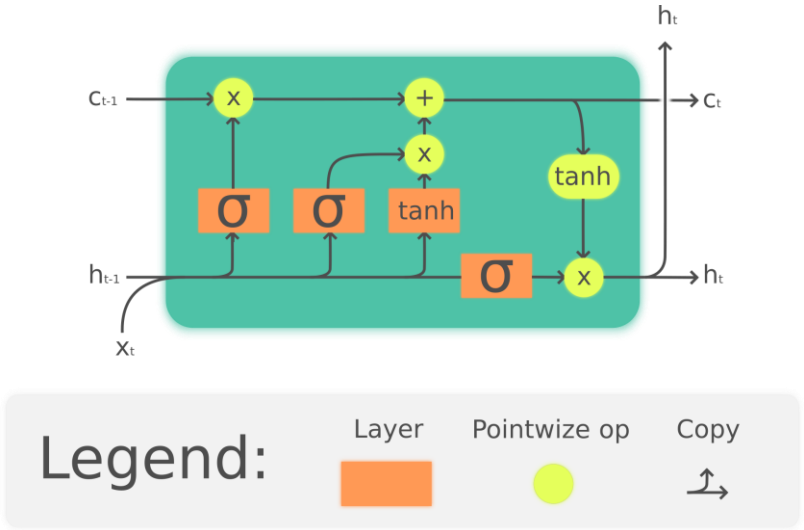
CNN in Computer Vision



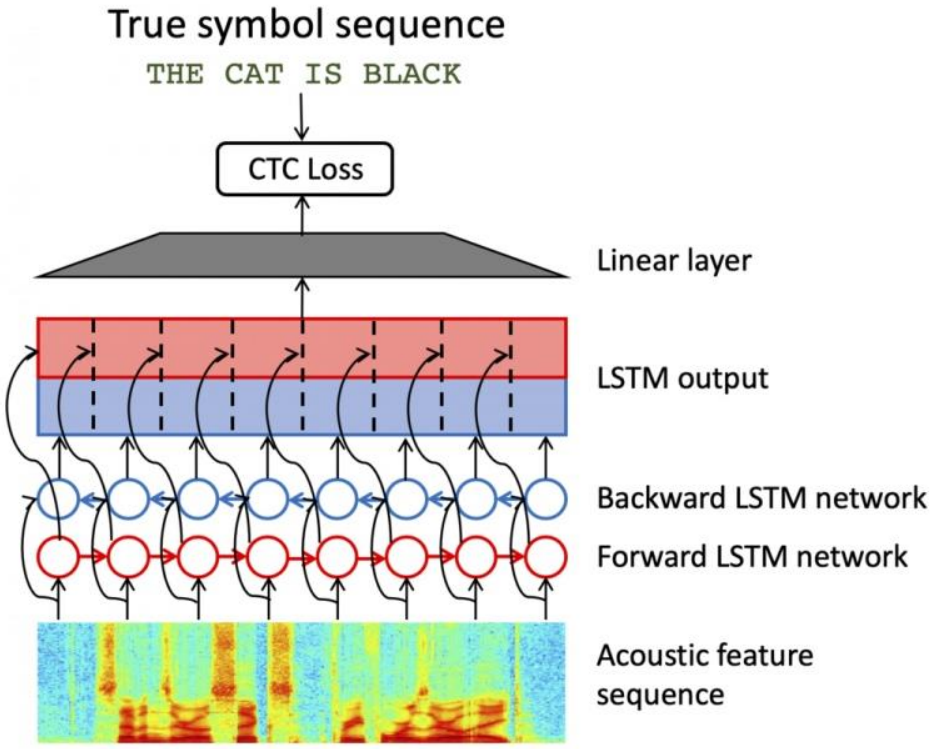
[CNN-based CV, 2010s]

- The region/local information in an image is the very important
- CNN is good at capturing the local informtion

RNN in Automatic Speech Recognition



[LSTM, Schmidhuber 1997]



[RNN-based ASR, 2010s]

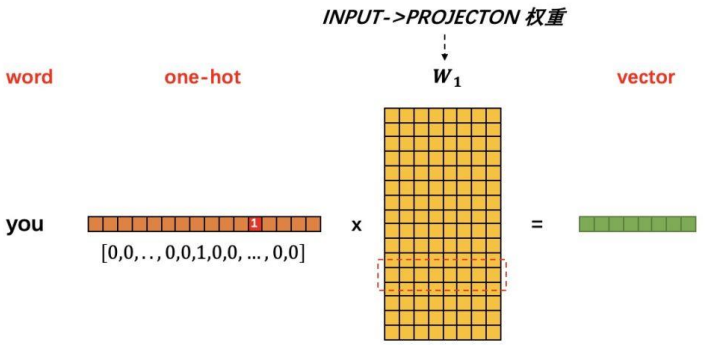
- The sequential/temporal information in c is very important
- RNN (with LSTM) is good at capturing the temporal information.

Monolingual Word Embedding

- As the development of neural network technology in NLP, words can be represented in continuous space.
- However, too sparse…

$$
\begin{aligned}
\text{I} \Leftrightarrow V_{\text{I}} &= [1,0,0,0,0,0,0,\ldots,0] \\
\text{you} \Leftrightarrow V_{\text{you}} &= [0,1,0,0,0,0,0,\ldots,0] \\
\text{is} \Leftrightarrow V_{\text{is}} &= [0,0,1,0,0,0,0,\ldots,0] \\
\text{are} \Leftrightarrow V_{\text{are}} &= [0,0,0,1,0,0,0,\ldots,0] \\
\text{very} \Leftrightarrow V_{\text{very}} &= [0,0,0,0,1,0,0,\ldots,0] \\
\text{wise} \Leftrightarrow V_{\text{wise}} &= [0,0,0,0,0,1,0,\ldots,0] \\
\text{smart} \Leftrightarrow V_{\text{smart}} &= [0,0,0,0,0,0,1,\ldots,0]
\end{aligned}
$$

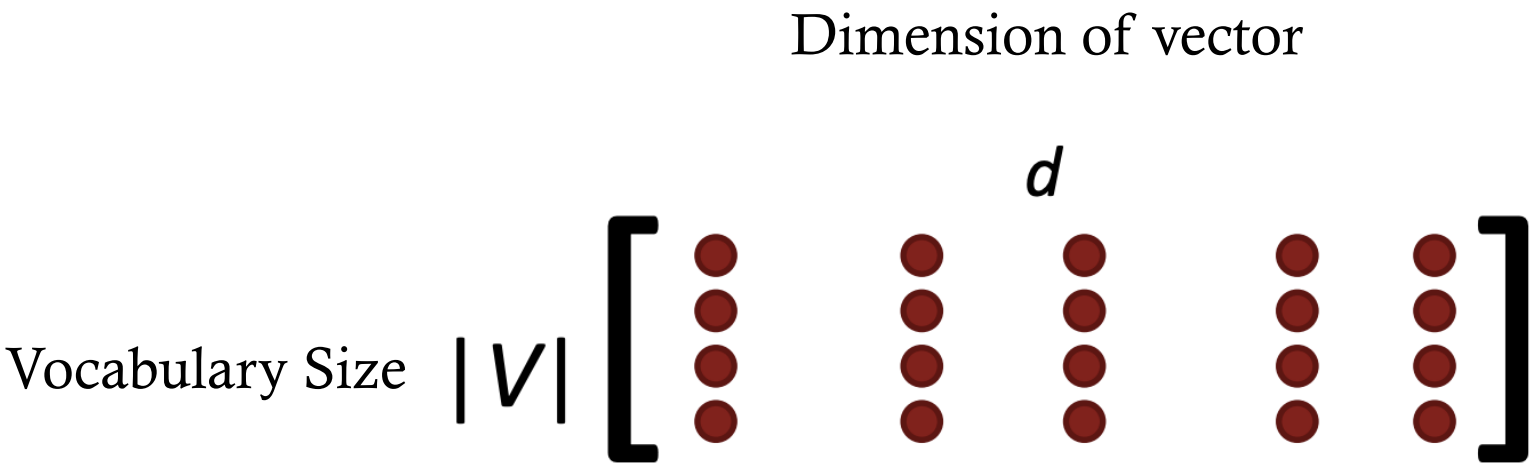
One-hot Representation



Projection

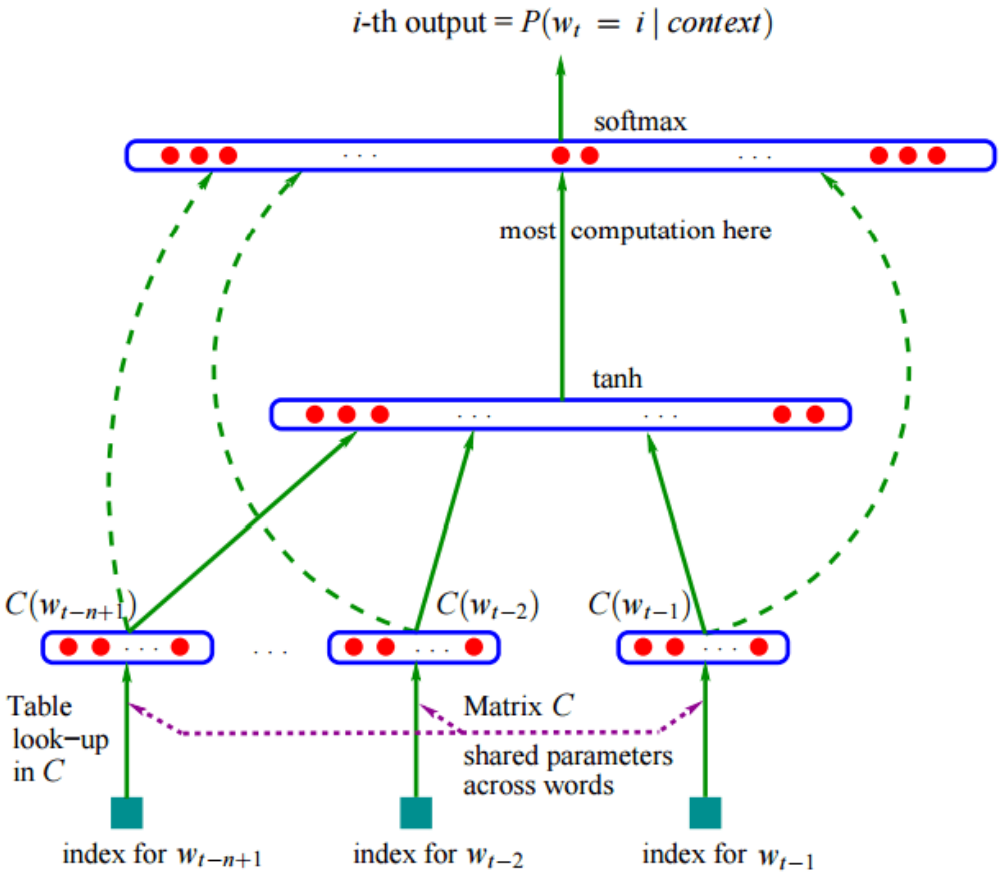
How to Learn the Projection Matrix θ

- The Projection Matrix is viewed as the parameters θ in the neural network



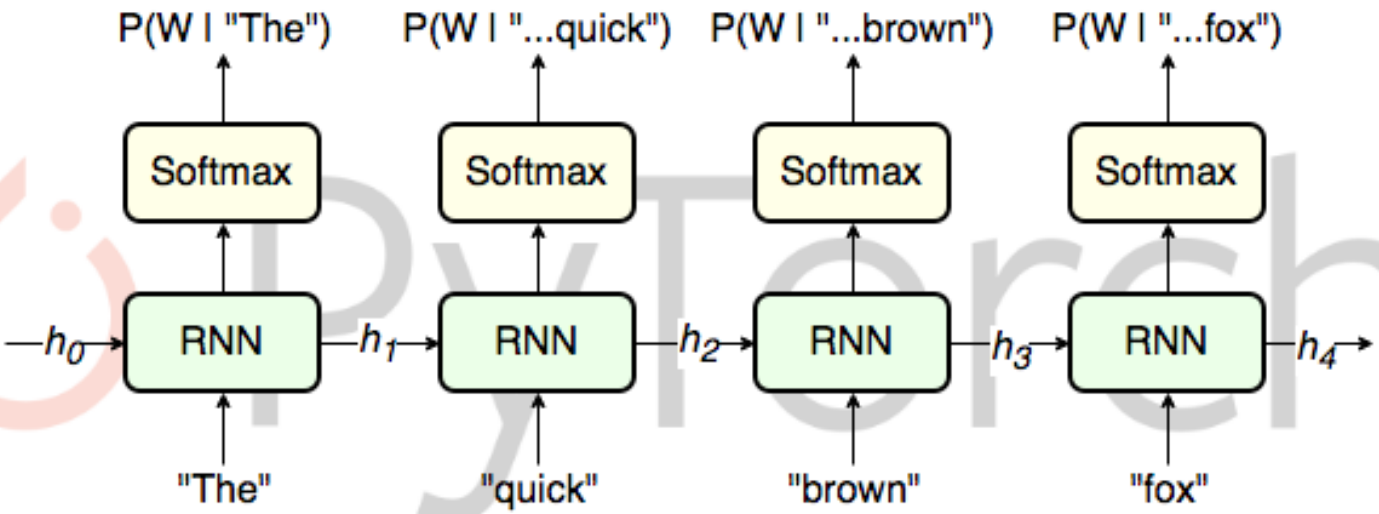
Neural Network Language Model

- The first neural network language model [Bengio et al., 2003]



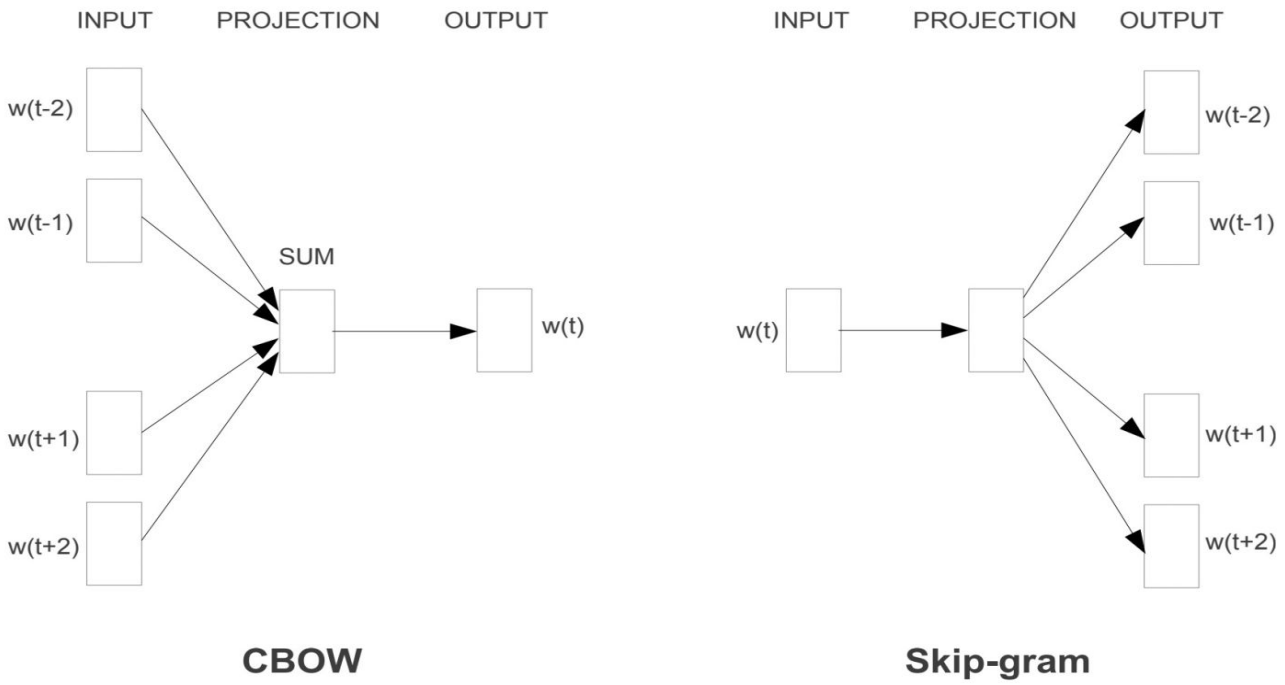
RNN Language Model

- Recurrent neural network language model [Mikolov 2010]



Word Embedding

- Word2Vec



[Mikolov et al., NeurIPS-2013]

Cost Function

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} \log p(w_t \mid w_{t-n}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+n})$$

Gradient Descent

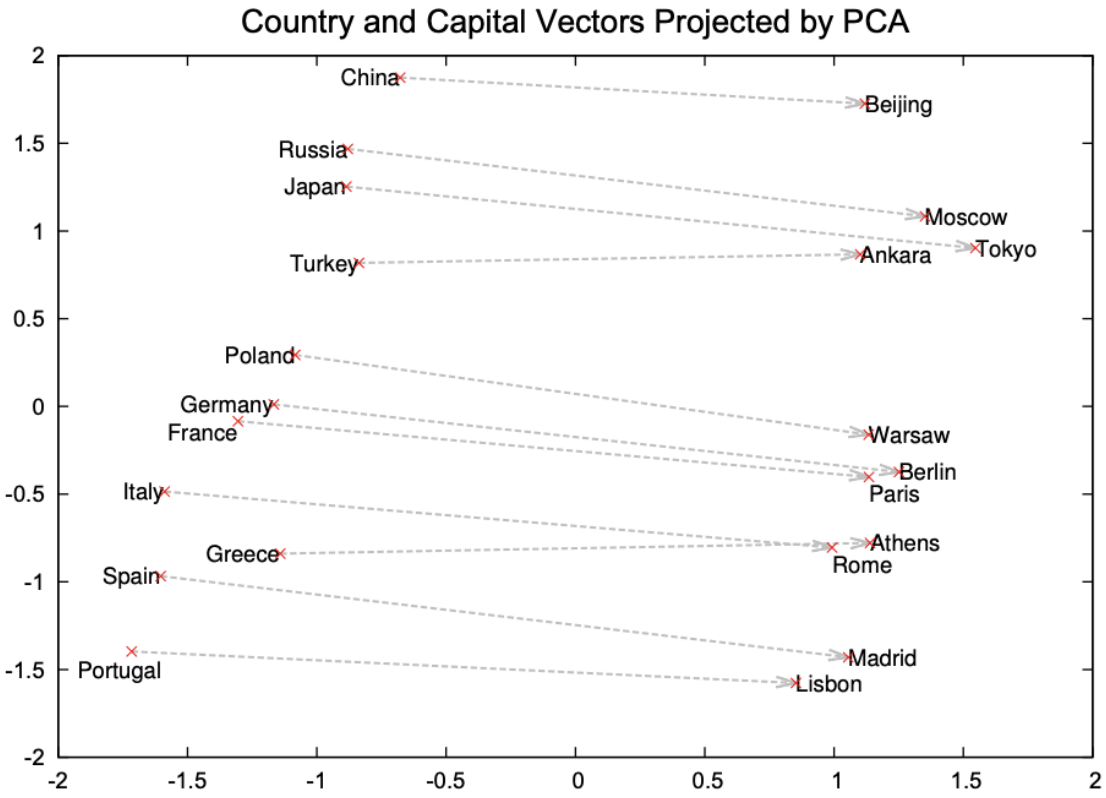
Update equation (in matrix notation):

$$\theta^{new} = \theta^{old} - \alpha \nabla_\theta J(\theta)$$

α = *step size* or *learning rate*

Word Embedding

- Then, there is some interesting findings.



[Mikolov et al., NeurIPS-2013]

What is Word2Vec Used For?

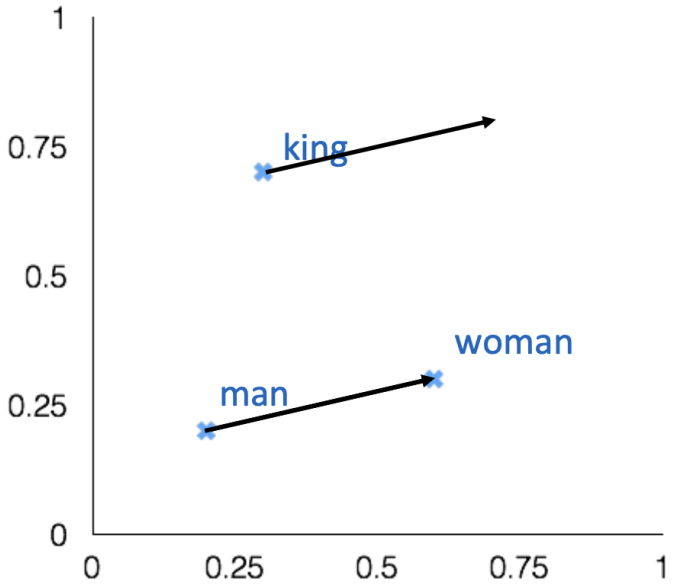
- Word Vector Analogies

a:b :: c:?

man:woman :: king:?

$$d = \arg\max_i \frac{(x_b - x_a + x_c)^T x_i}{||x_b - x_a + x_c||}$$

- Evaluate word vectors by how well their cosine distance after addition captures intuitive semantic and syntactic analogy questions
- Discarding the input words from the search!
- Problem: What if the information is there but not linear?



Initialize Other Models

- For example, the CNN text classification.

- **CNN-rand**: Our baseline model where all words are randomly initialized and then modified during training.
- **CNN-static**: A model with pre-trained vectors from `word2vec`. All words—including the unknown ones that are randomly initialized—are kept static and only the other parameters of the model are learned.
- **CNN-non-static**: Same as above but the pre-trained vectors are fine-tuned for each task.
- **CNN-multichannel**: A model with two sets of word vectors. Each set of vectors is treated as a 'channel' and each filter is applied

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	**89.6**
CNN-non-static	**81.5**	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4

RNN-based MT

- Languages have different order: English/Japanese/Chinese
  - Attention/alignment is necessary.

In a new model architecture, we define each conditional probability in Eq. (2) as:

$$p(y_i|y_1,\dots,y_{i-1},\mathbf{x}) = g(y_{i-1},s_i,c_i), \quad (4)$$

where s_i is an RNN hidden state for time i, computed by

$$s_i = f(s_{i-1},y_{i-1},c_i).$$

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$

The weight α_{ij} of each annotation h_j is computed by

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})},$$

where

$$e_{ij} = a(s_{i-1},h_j)$$

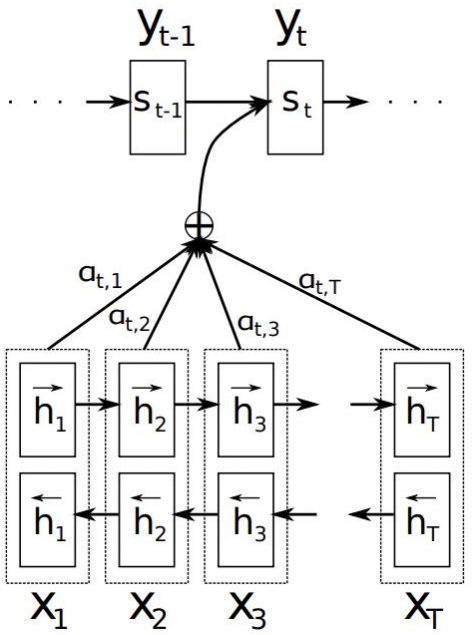
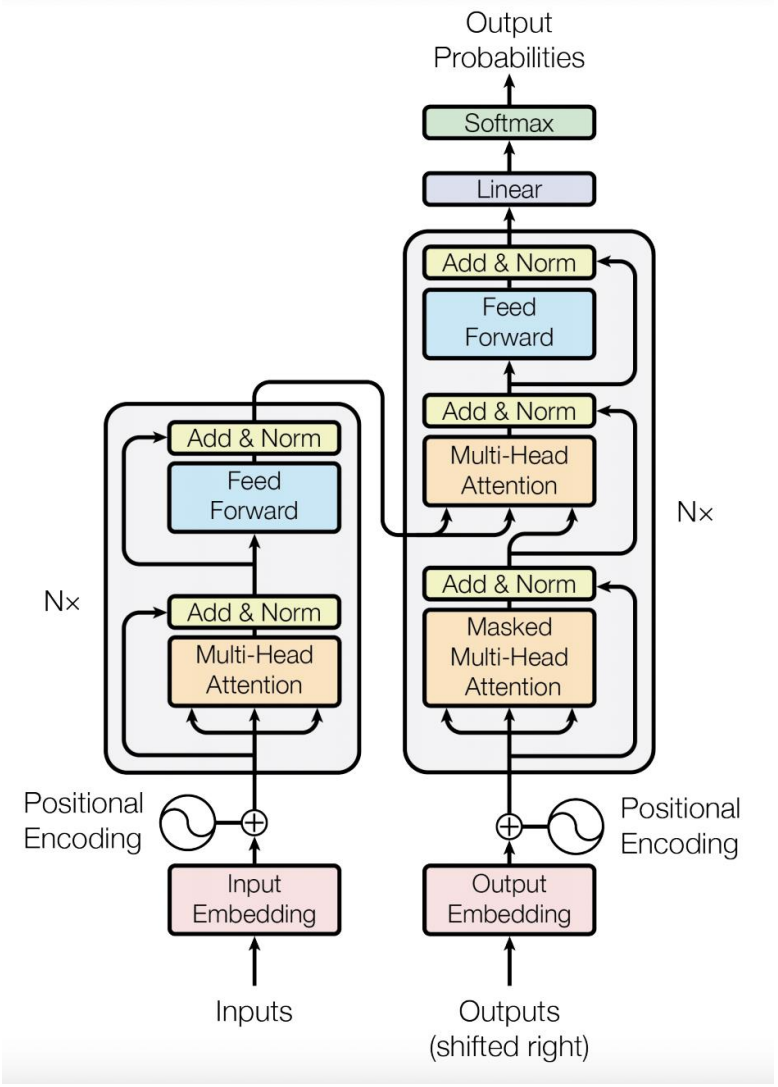


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \dots, x_T)$.

[RNN-based MT, Bahdanau, 2014]

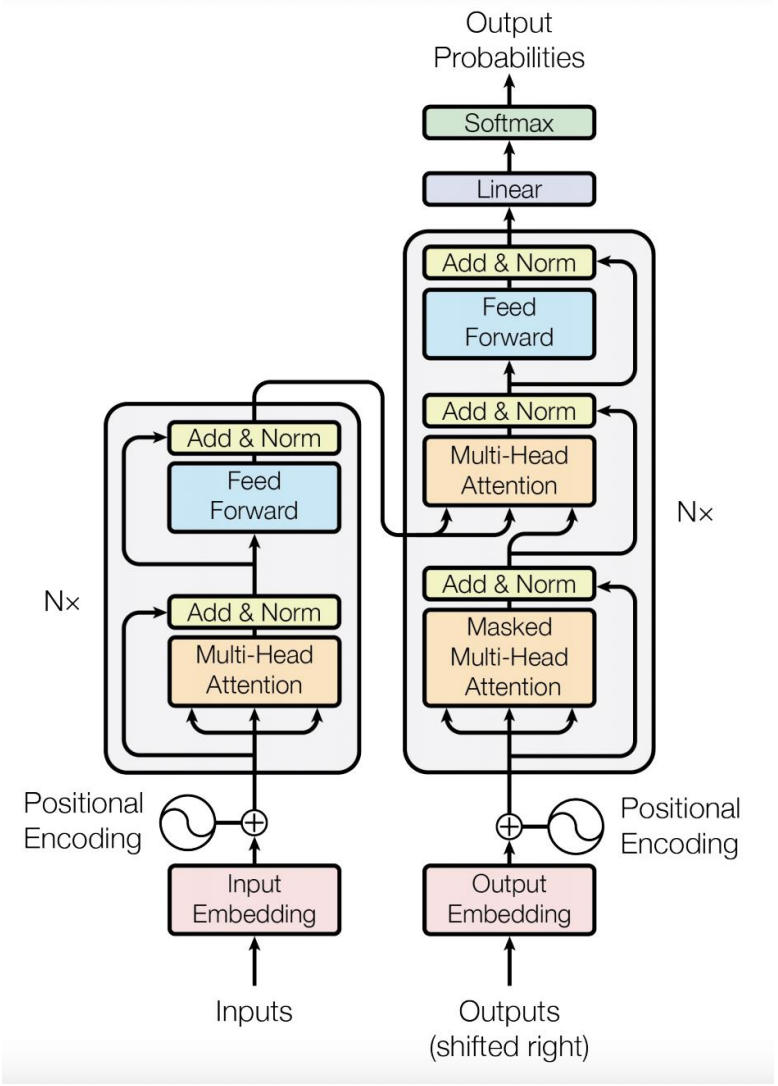
Menu

- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



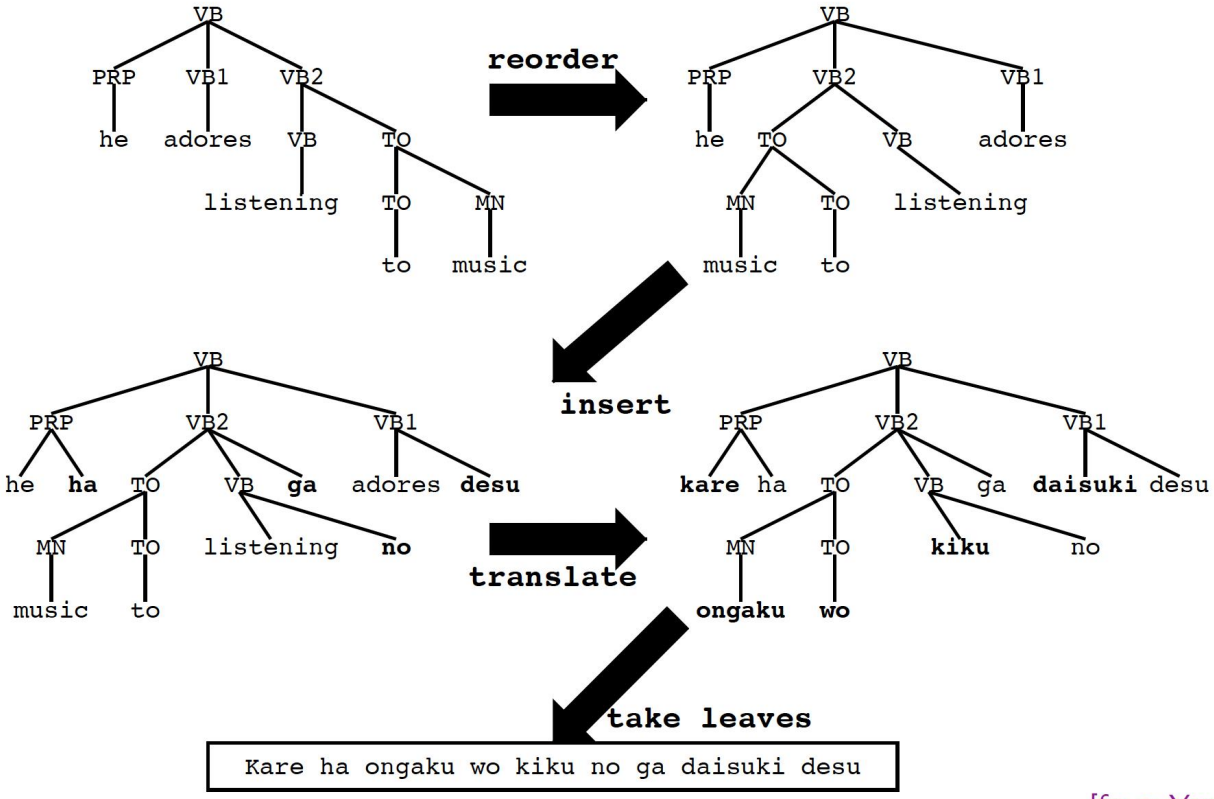
Menu

- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



Self-attention

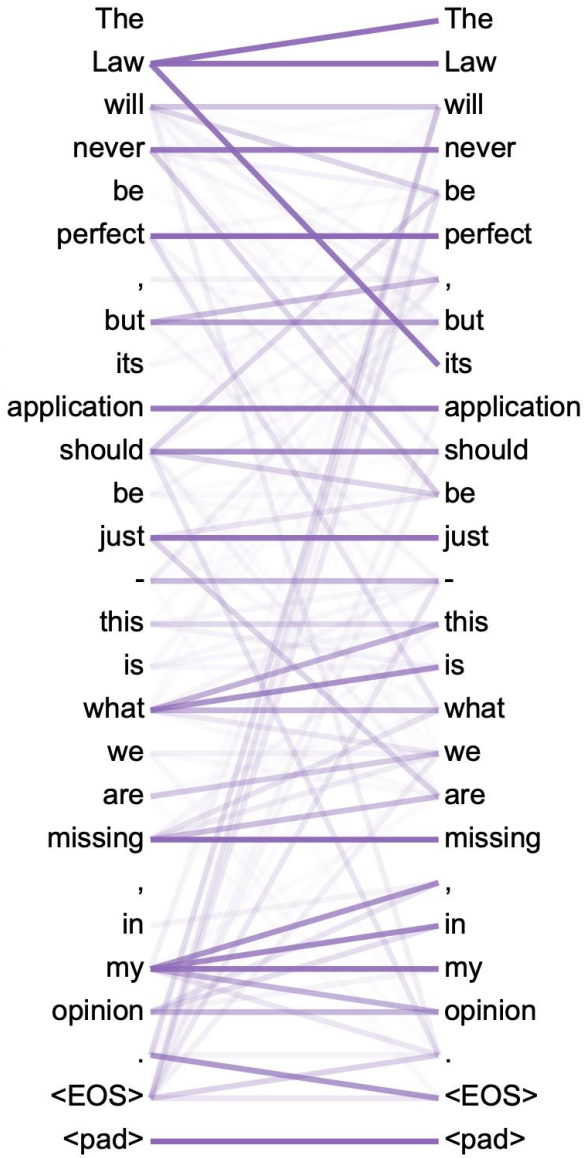
- Languages have syntactic structure
  - Syntactic structure is fully-connected and contains high-order relationship
  - This cannot be solved by RNN or other NN



[from Yamada and Knight, 2001]

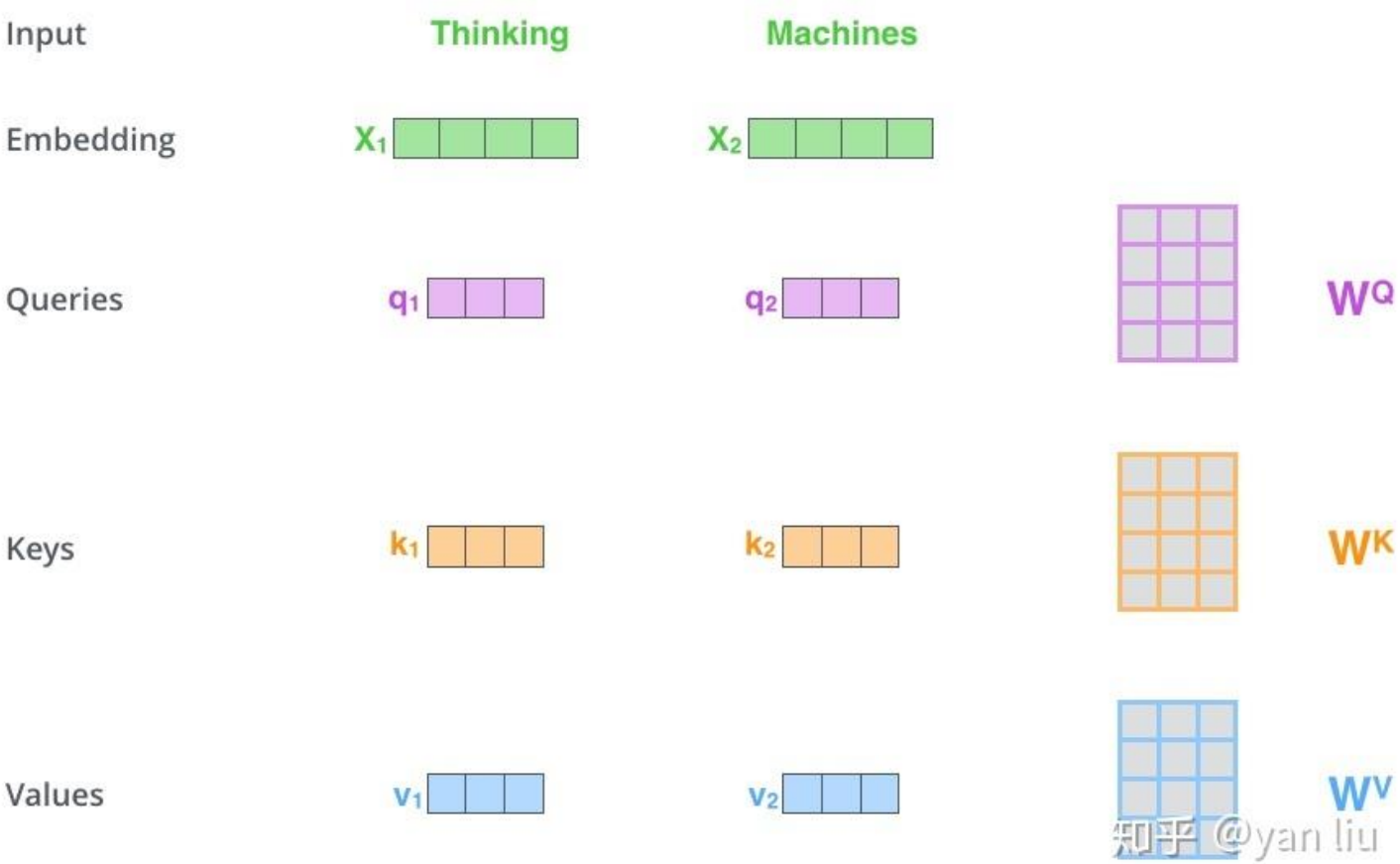
Fully-connected Network

- How to define the weight of each edge?

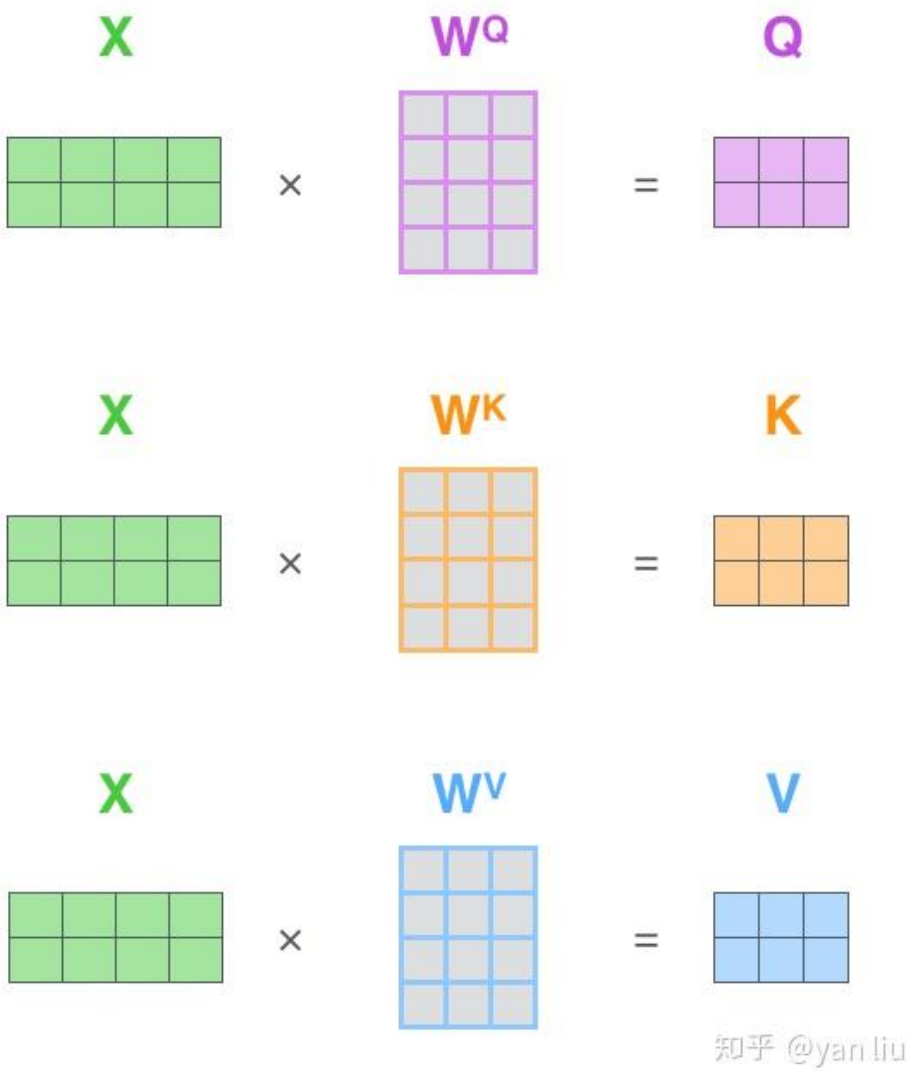


Self-attention

- Fully-connected self-attention is necessary
  - Each word has its query (Q), key (K), value (V).

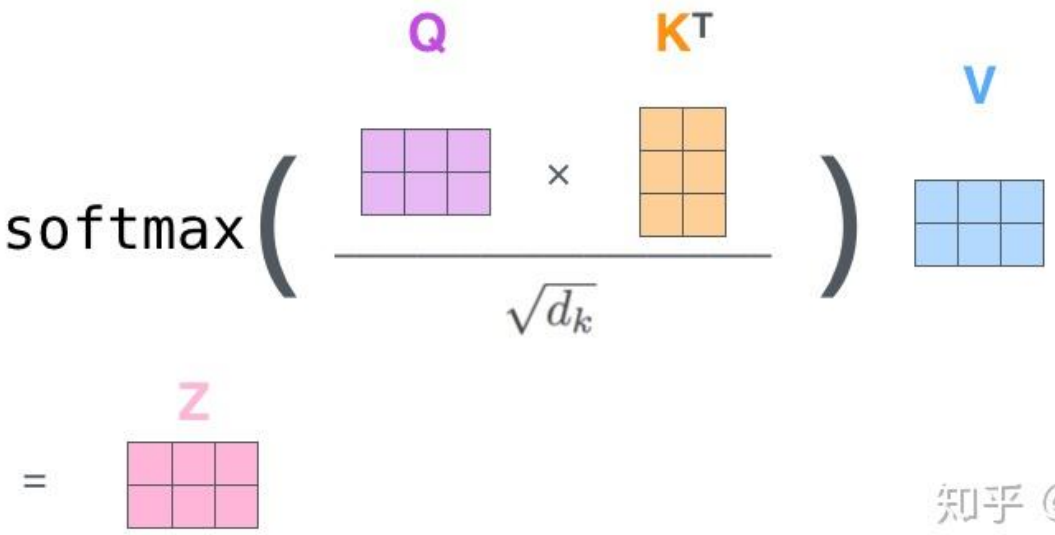


{Q, K, V}

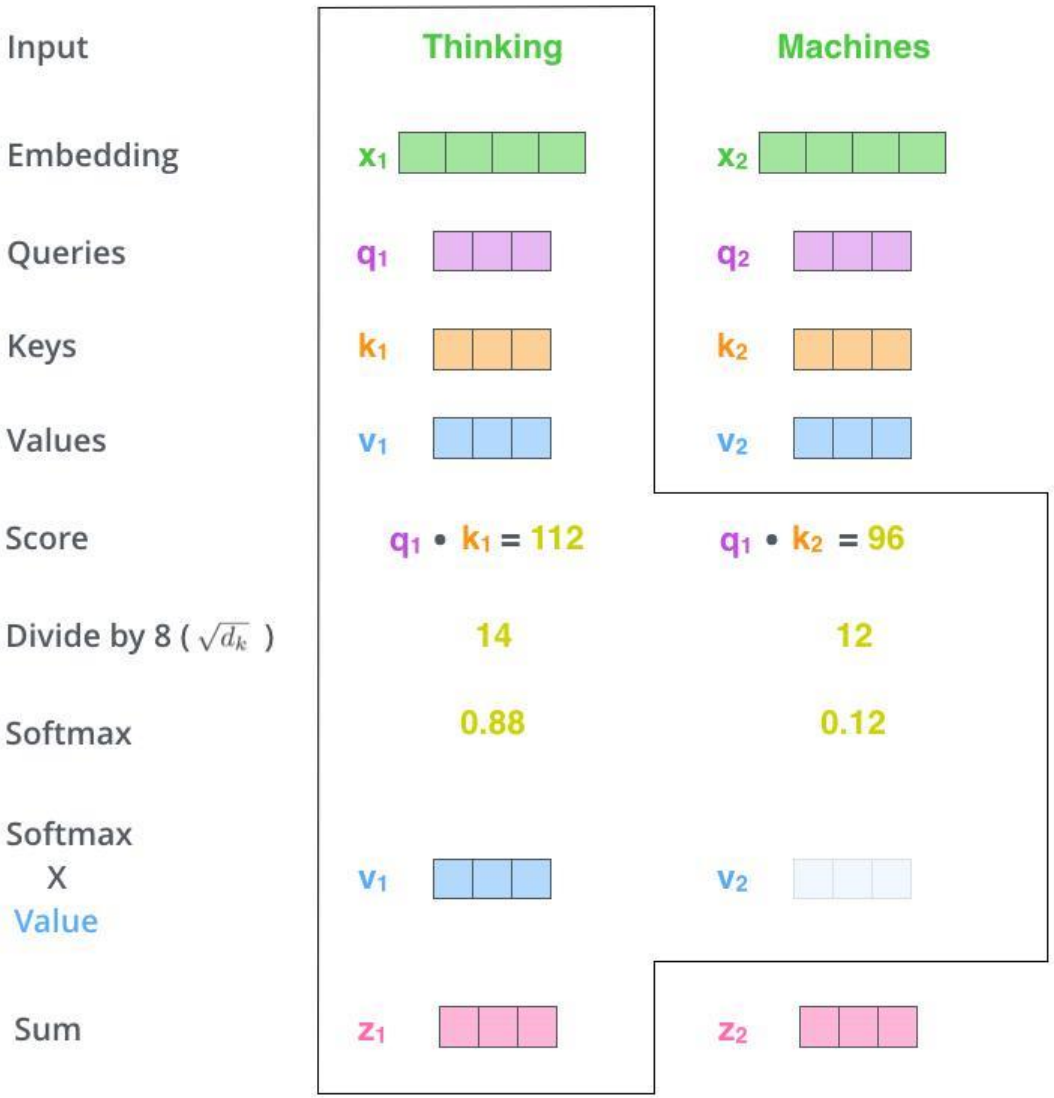


Attention

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

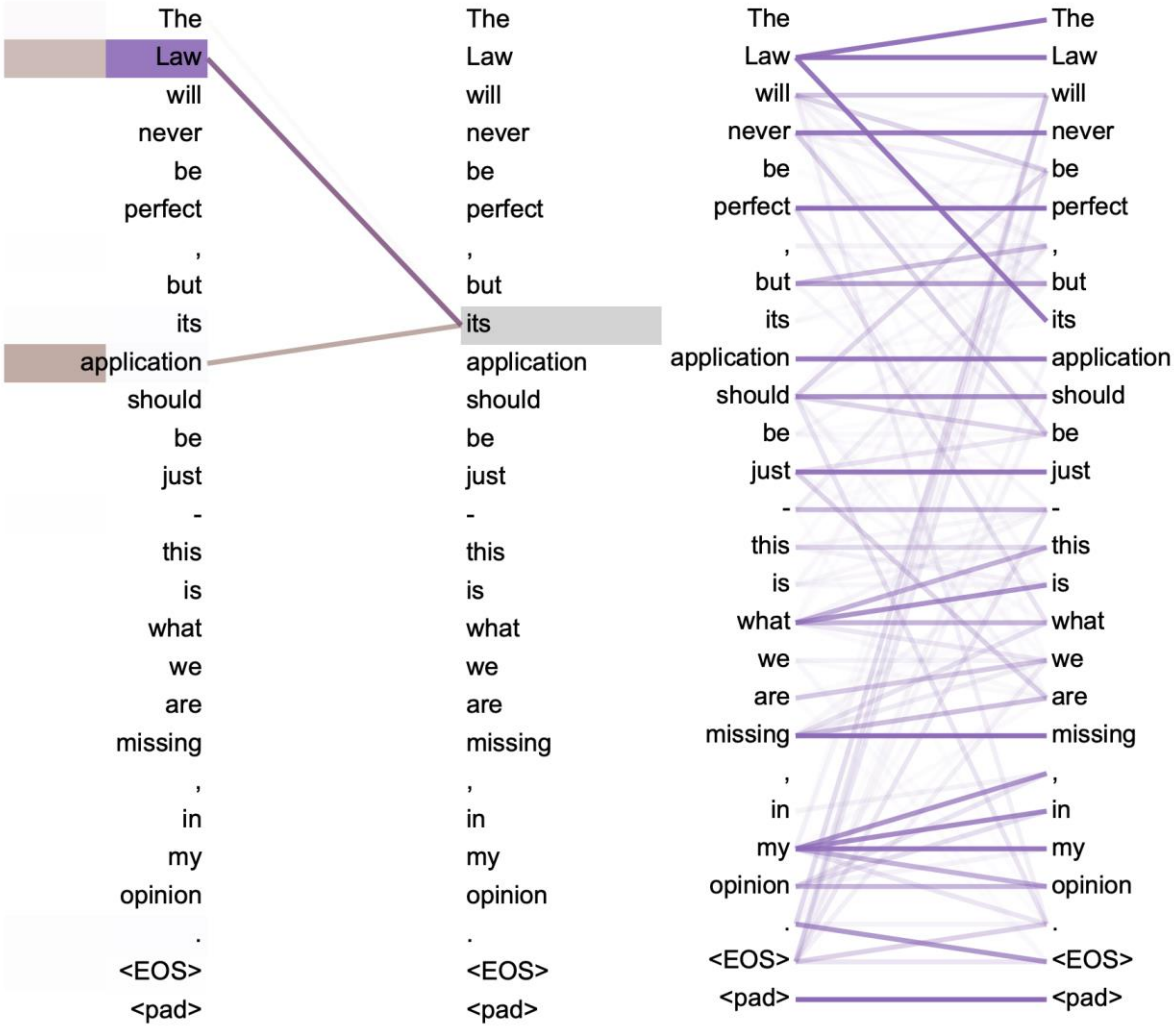


Attention


Input
Thinking
Machines
Embedding
x1
x2
Queries
q1
q2
Keys
k1
k2
Values
v1
v2
Score
q1 • k1 = 112
q1 • k2 = 96
Divide by 8 (√dk)
14
12
Softmax
0.88
0.12
Softmax X Value
v1
v2
Sum
z1
z2

Attention

- If we prune some edges, we can obtain a syntactic tree.

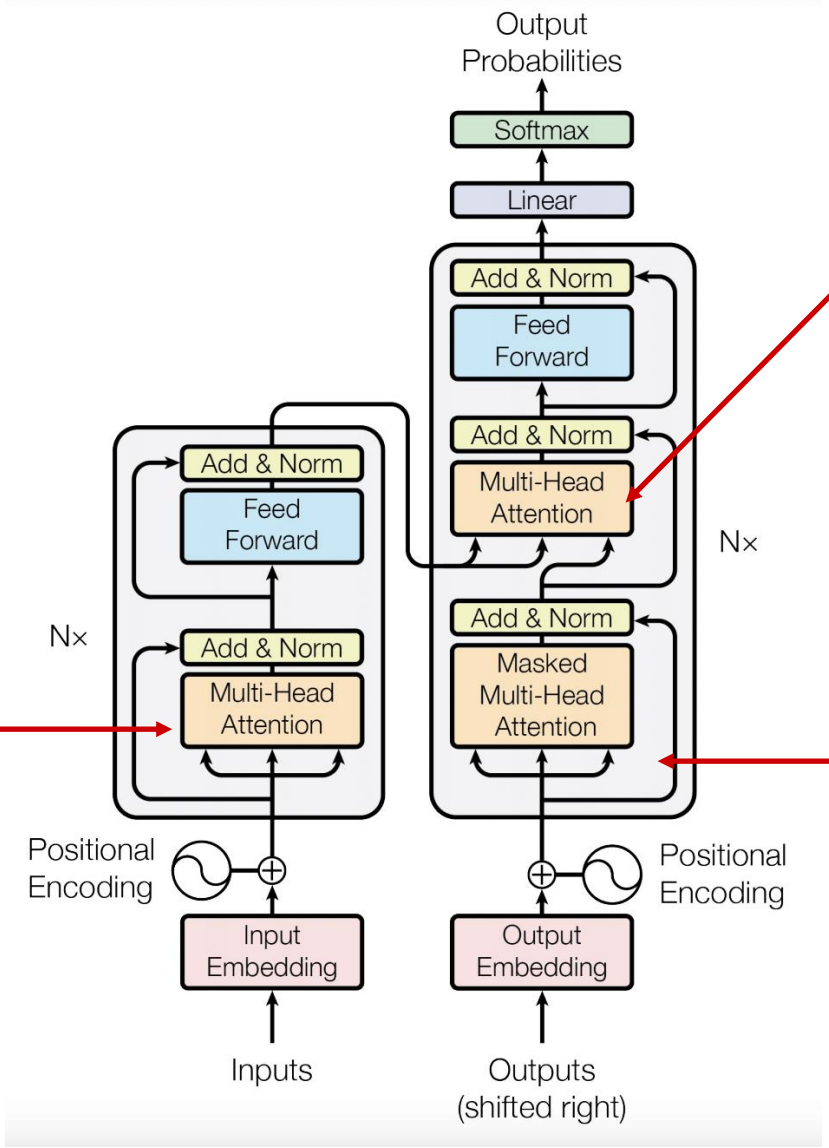


There Are Two Attentions

Cross-Lingual Attention

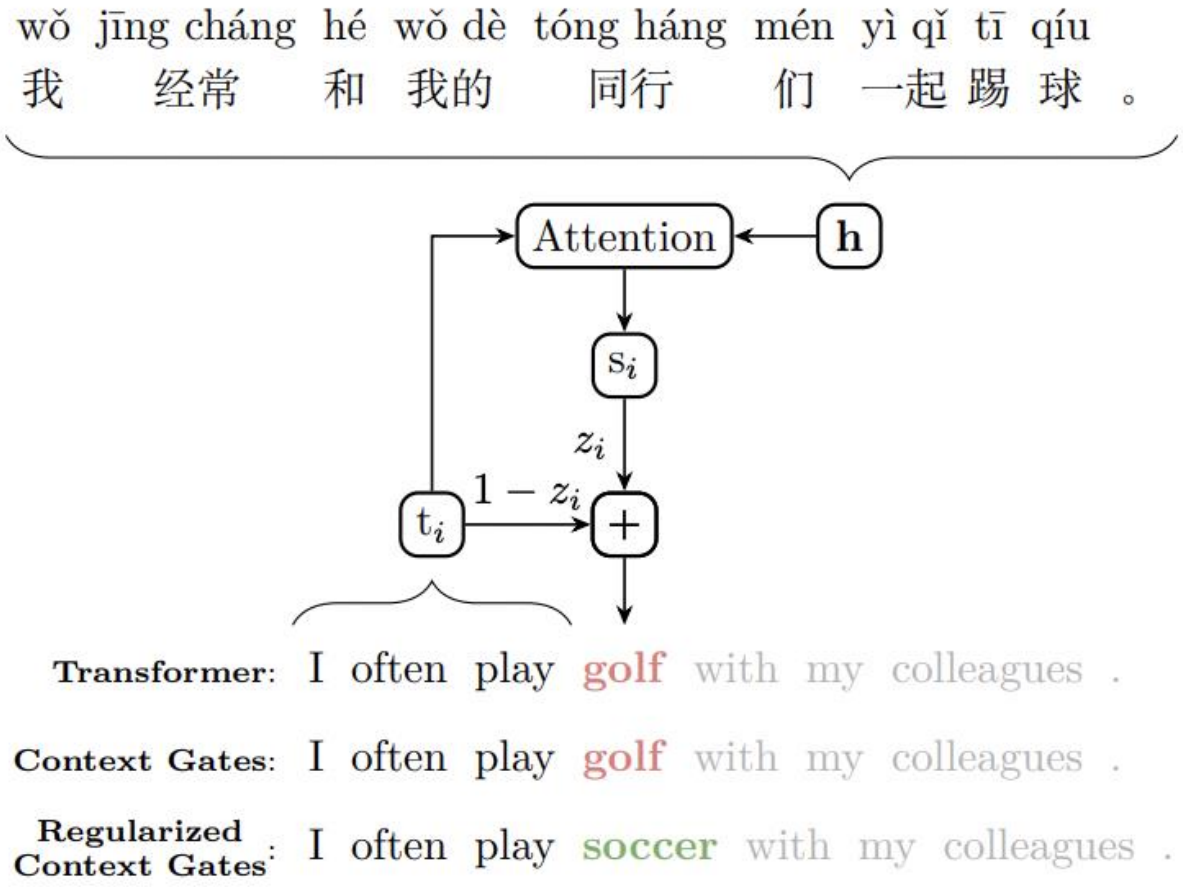
Monolingual Attention

Monolingual Attention



What Will Happen?

- We need to control the contribution between source and target attention.

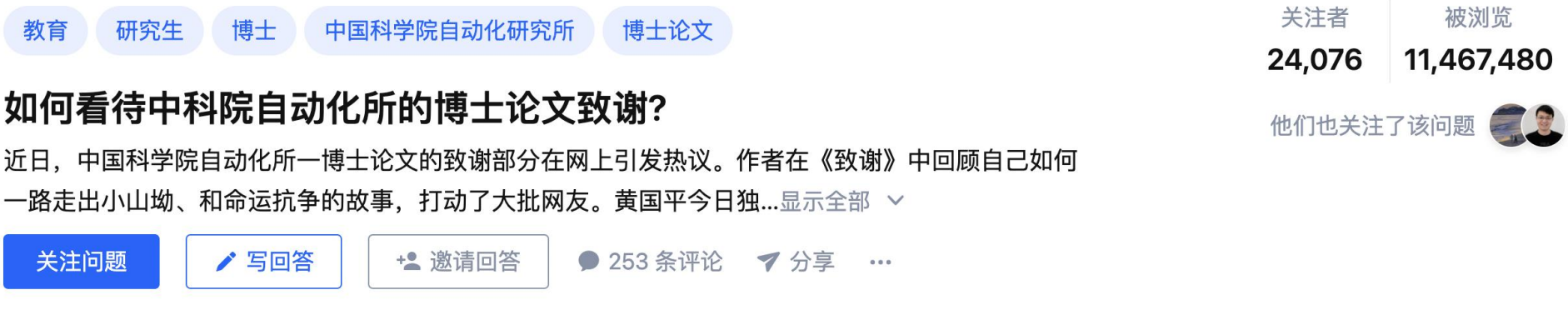


ACL-2020

Regularized Context Gates on Transformer for Machine Translation

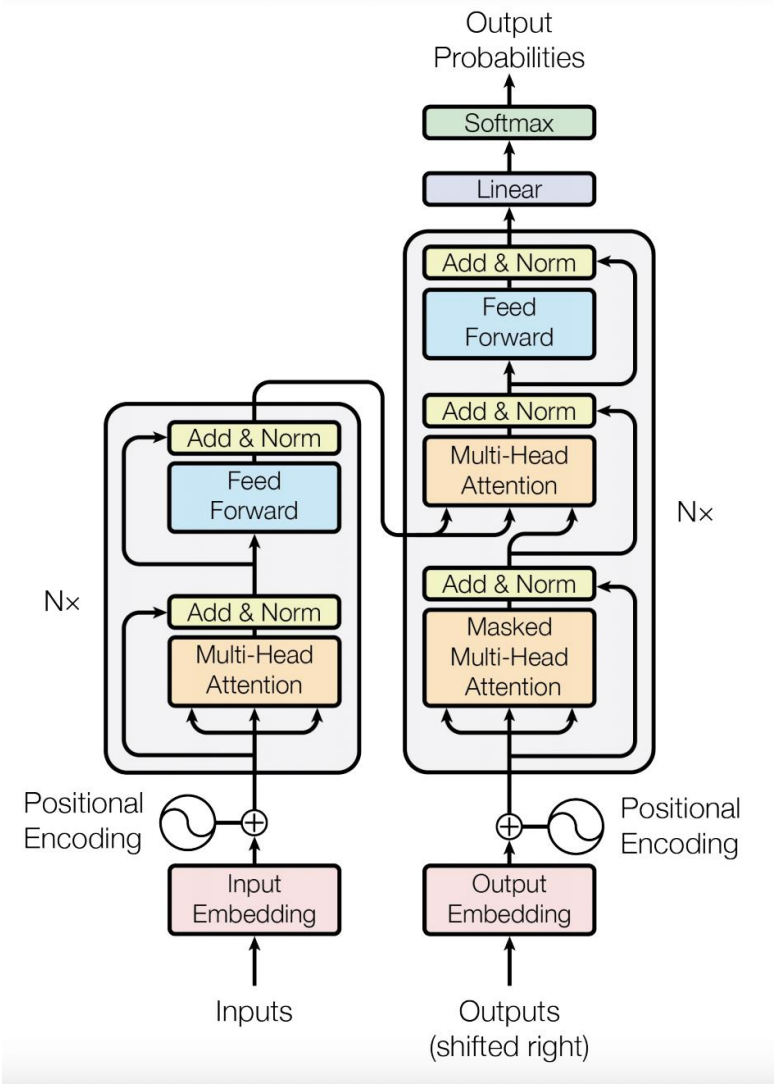
Xintong Li[1], Lemao Liu[2], Rui Wang[3], Guoping Huang[2], Max Meng[1]

[1]The Chinese University of Hong Kong [2]Tencent AI Lab
[3]National Institute of Information and Communications Technology



Menu

- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



If The Story Stops Here

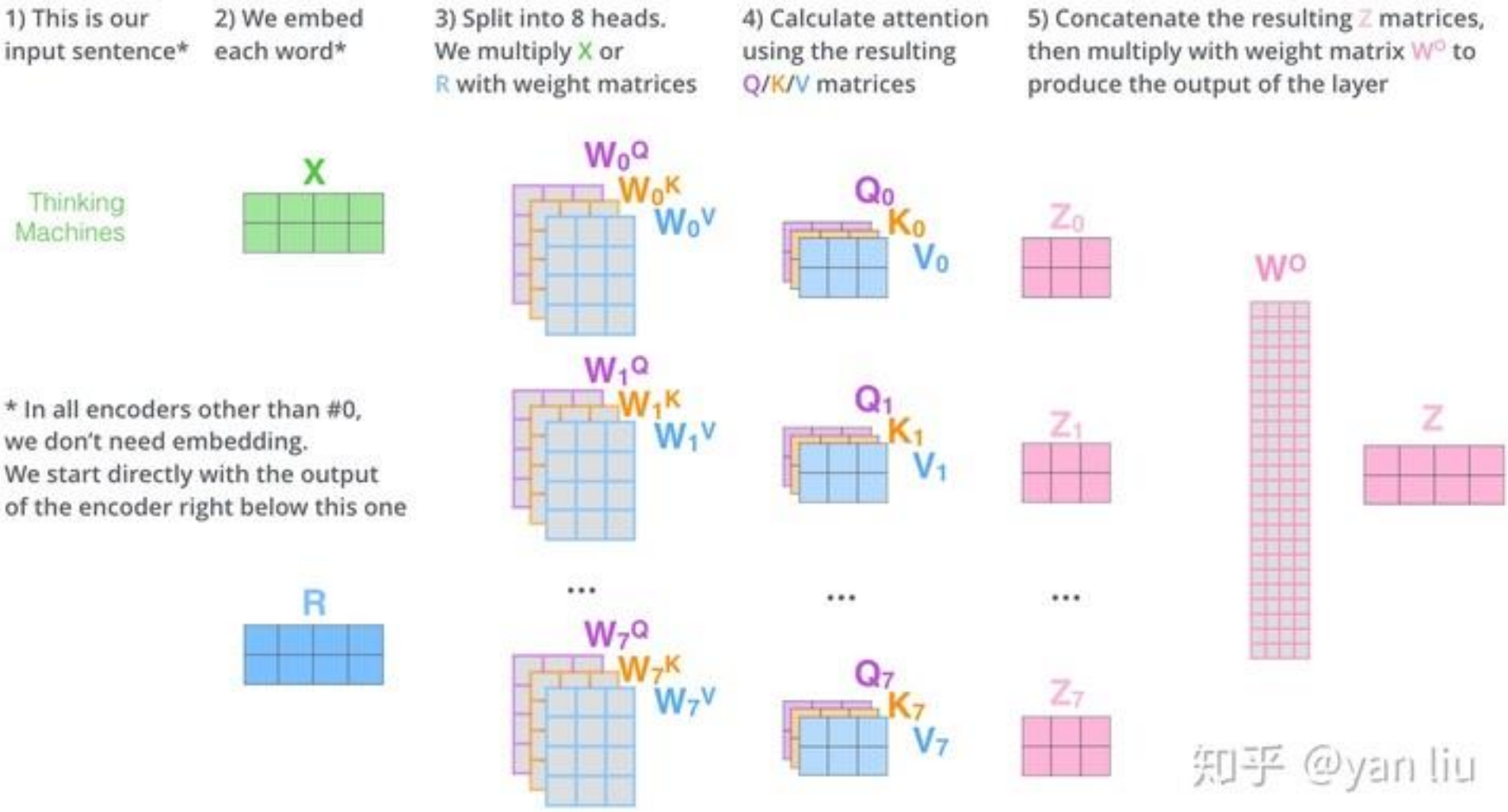
- It will become another boring parameter tuning problem

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

Multi-head

- Multi-head is to ensure the diversity for each representation vector

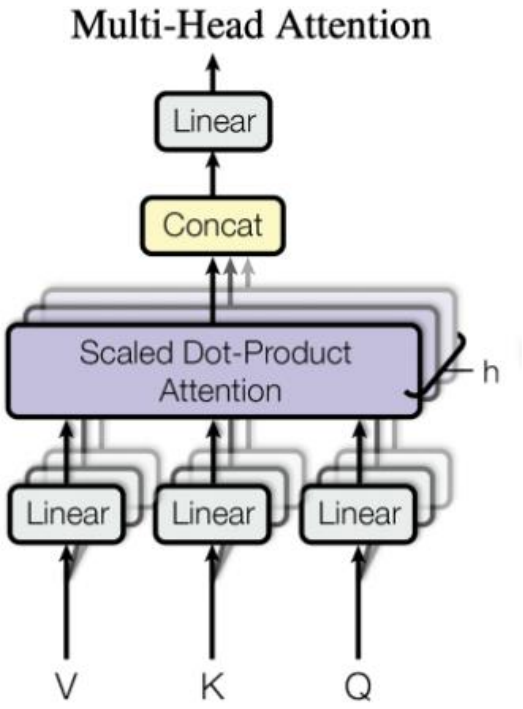
Multi-head


1) This is our input sentence*
2) We embed each word*
3) Split into 8 heads. We multiply X or R with weight matrices
4) Calculate attention using the resulting Q/K/V matrices
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer
Thinking Machines
X
R
* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one
W0Q W0K W0V
W1Q W1K W1V
W7Q W7K W7V
Q0 K0 V0
Q1 K1 V1
Q7 K7 V7
Z0
Z1
Z7
WO
Z
知乎 @yan liu

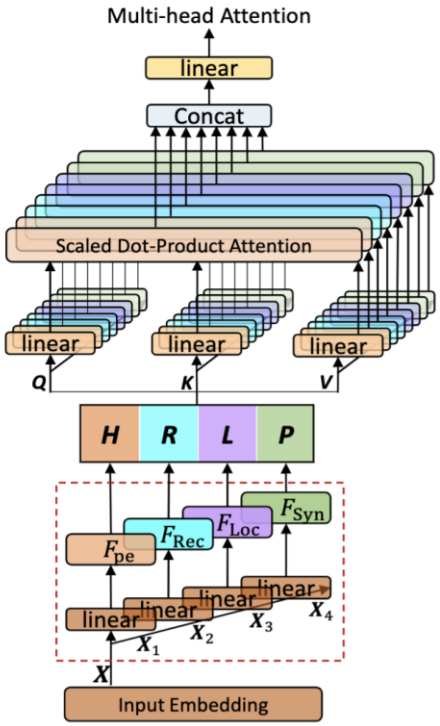
How to Enlarge the Diversity?

- Multi-head is to ensure the diversity for each representation vector

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



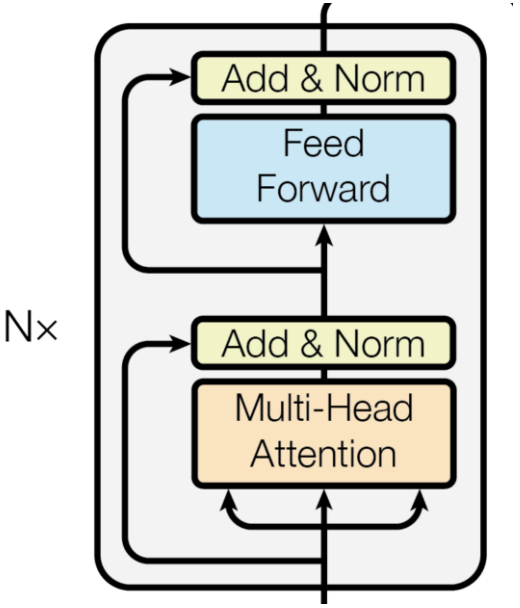
[Multi-head Structure]

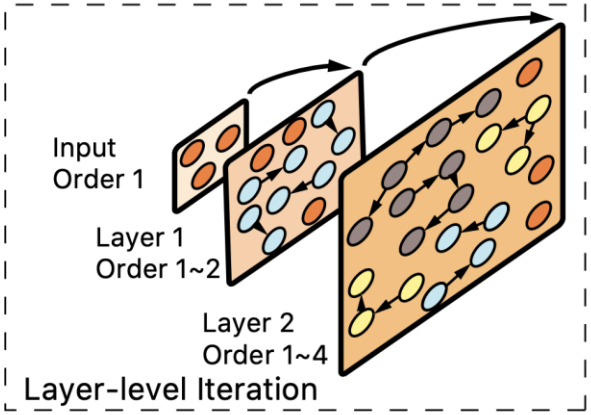


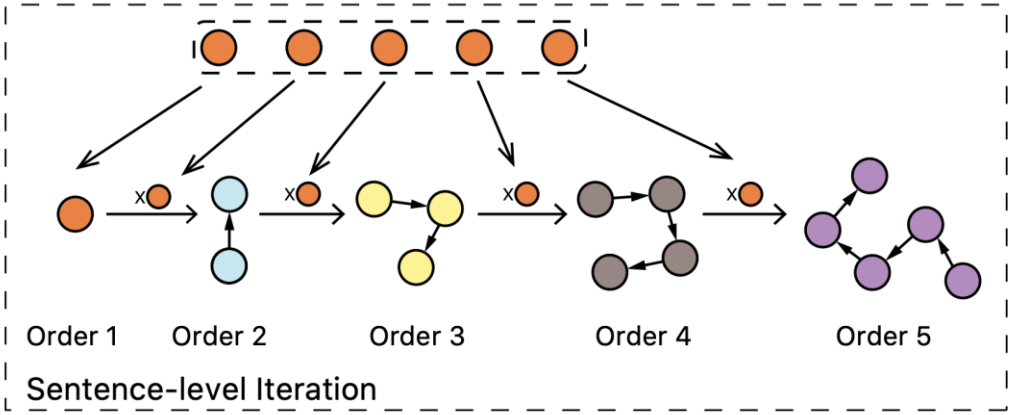
[Diverse Multi-head, Chen and Wang, TASLP-2020]

High-order Relationship

- To achieve high-order information, stacked (multi-head + feed forward) iteration network is added.

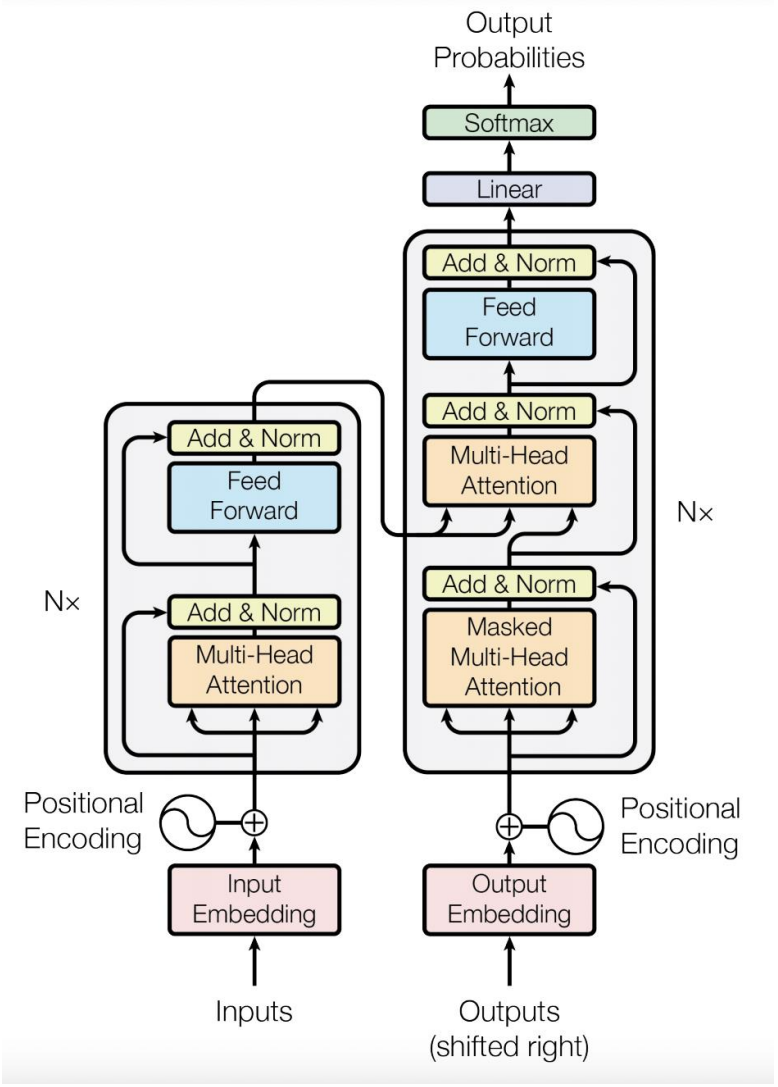






Menu

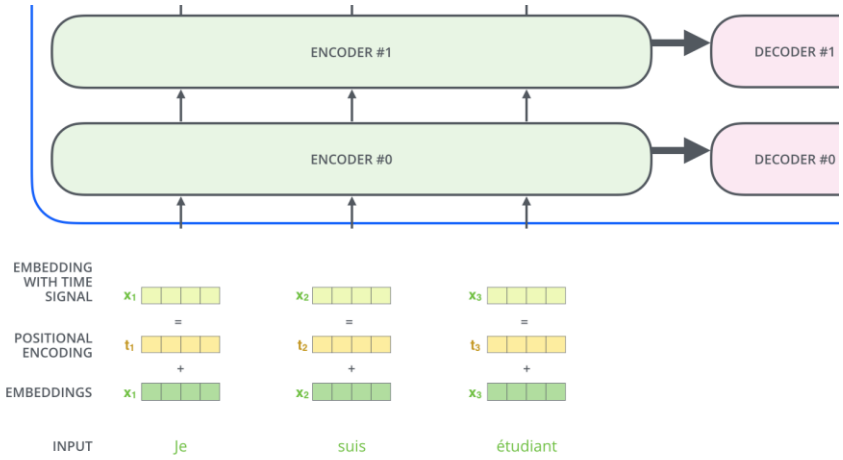
- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



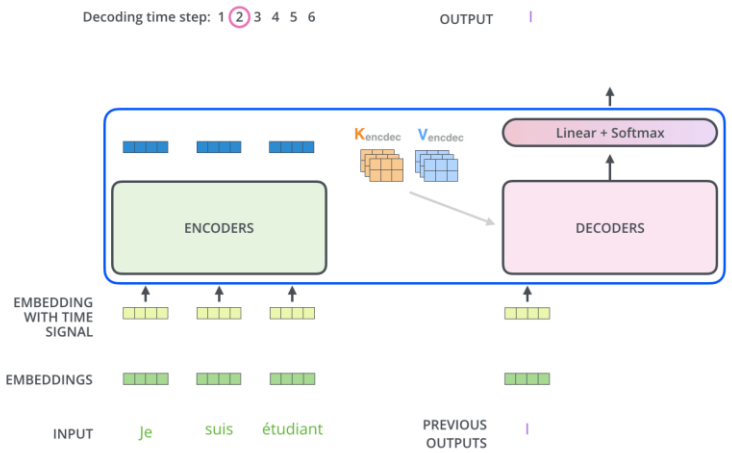
Position Encoding (PE)

- Languages have different order
- MT needs to know the alignment between positions
- Specific for MT

$$\mathbf{pe}_{(j,2i)} = \sin(j/10000^{2i/d_{model}}),$$
$$\mathbf{pe}_{(j,2i+1)} = \cos(j/10000^{2i/d_{model}}),$$



Encoder



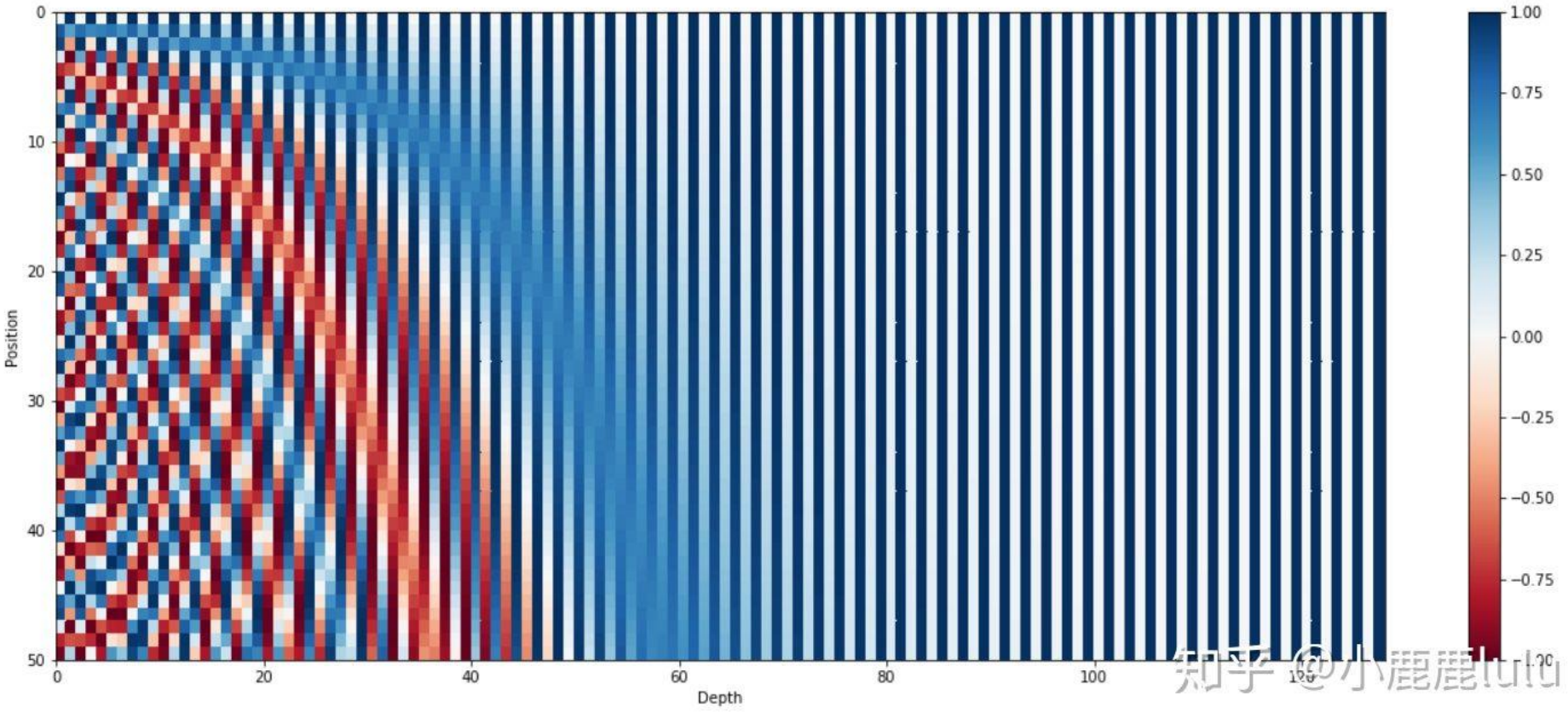
Decoder

Why Using Sin and Cos

- This problem is not well-studied.

$$\mathbf{pe}_{(j,2i)} = \sin(j/10000^{2i/d_{model}}),$$

$$\mathbf{pe}_{(j,2i+1)} = \cos(j/10000^{2i/d_{model}}),$$



Achieving Reordering

- No **hard** attention is learned.

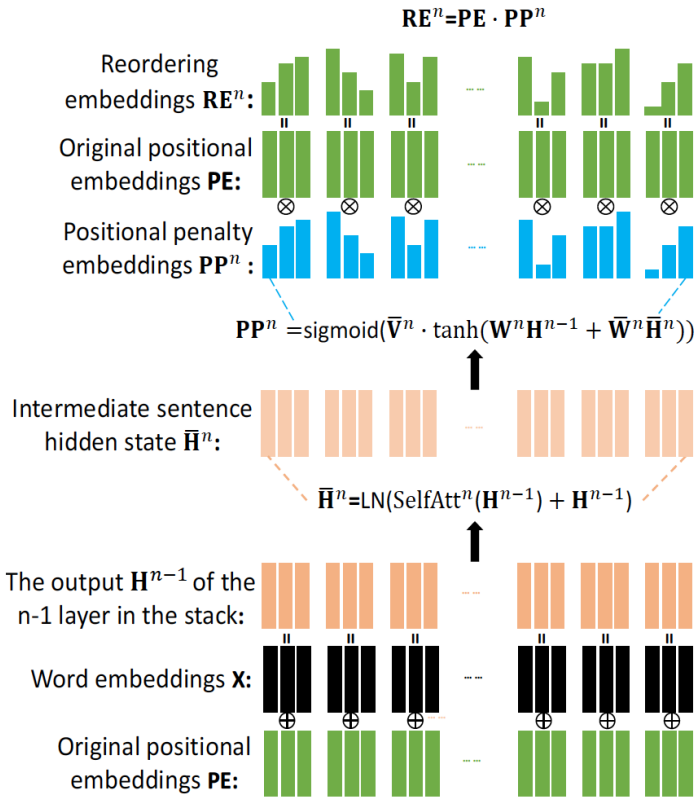


Figure 1: Learning reordering embeddings for the n-th layer in the stack.

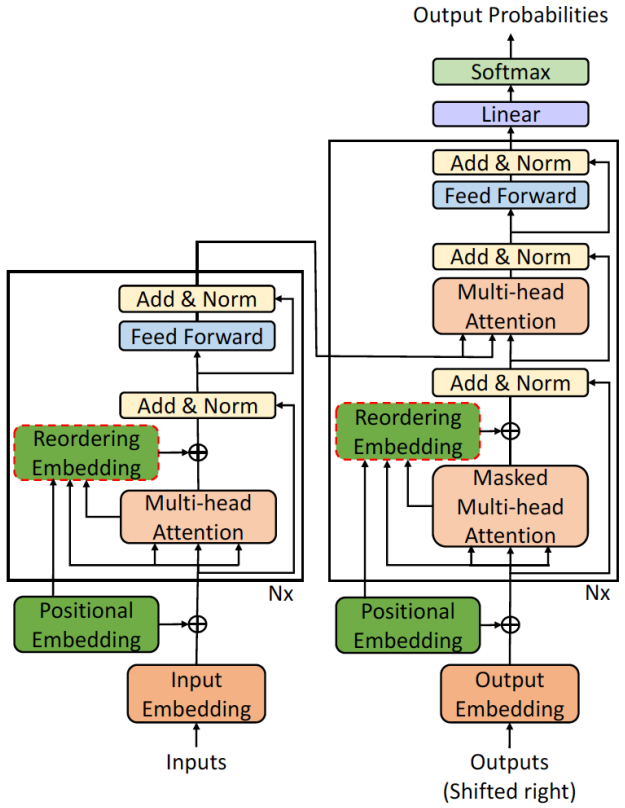
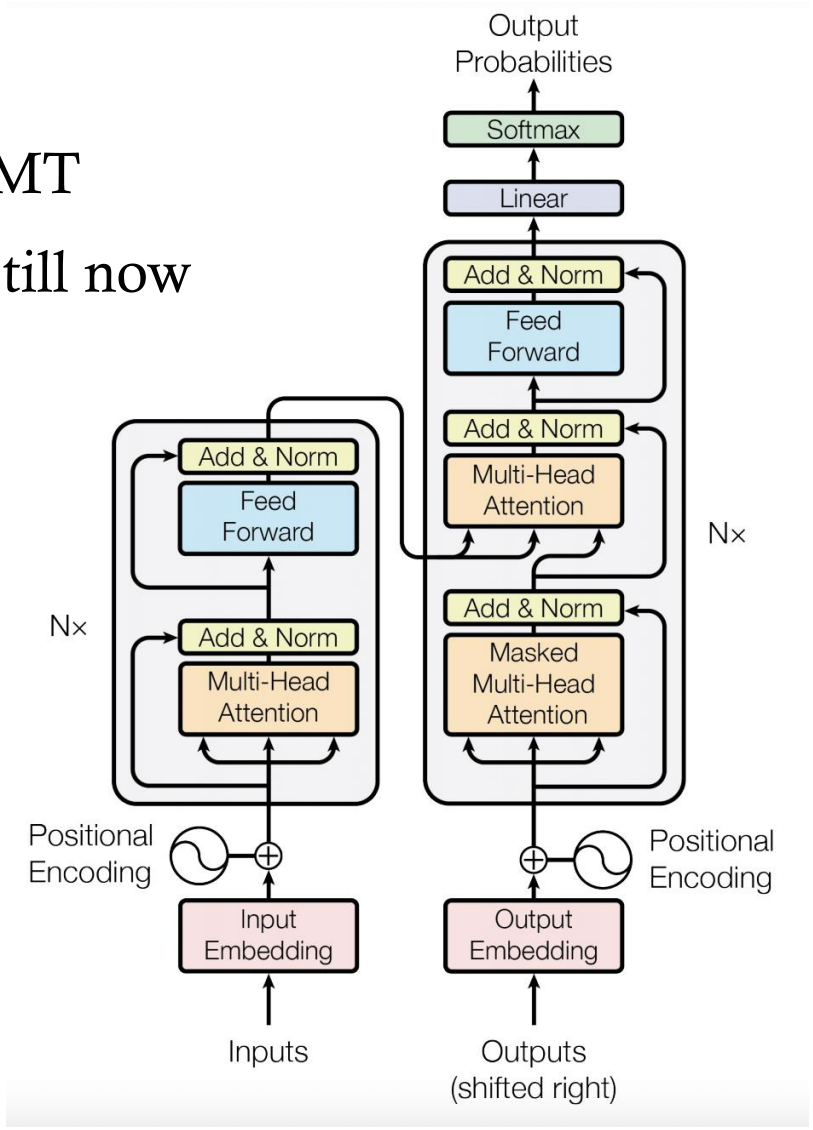


Figure 2: The architecture of Transformer with reordering embeddings.

[NMT with Reordering, Chen and Wang, ACL-2019]

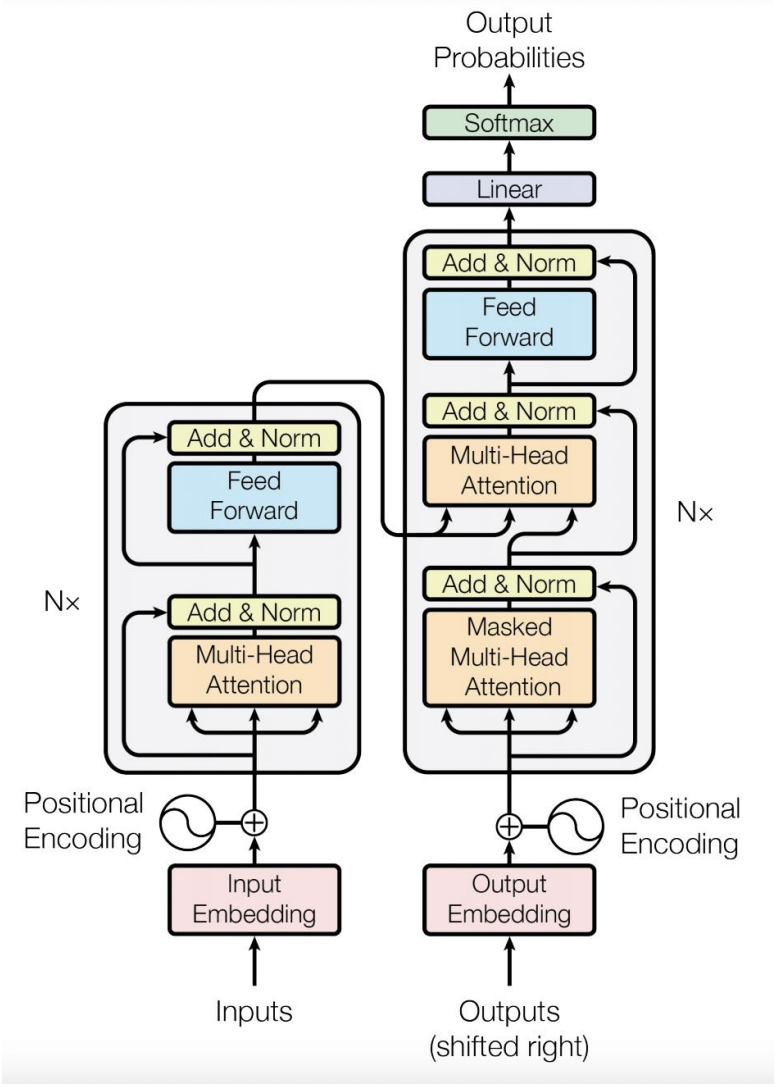
Entire Transformer Strucure

- "Attention is all you need" in 2017.
- Just 2 years after the RNN-based NMT
- No further significant improvement till now



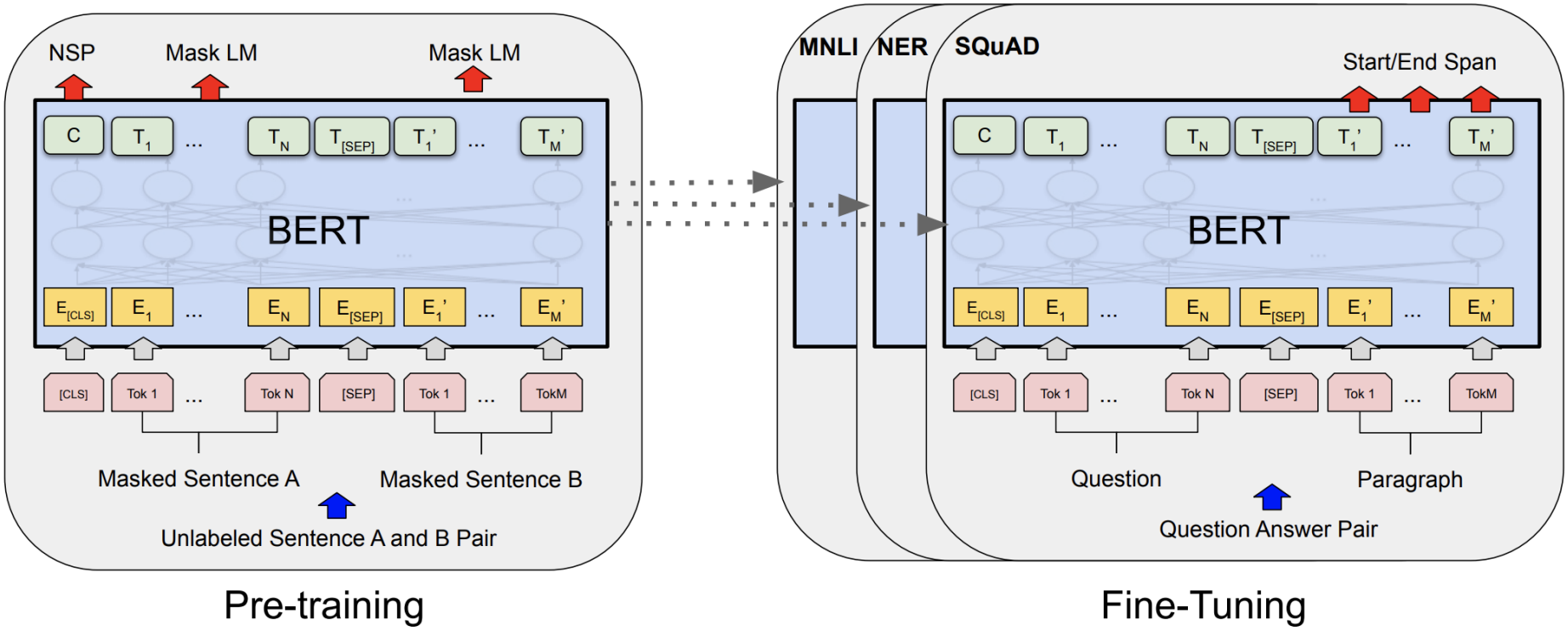
Menu

- Machine Translation
  - History
  - RNN based MT
- Break (18:45~18:55)
- Transformer
  - Self-attention
  - Multi-head and Stack Iteration
  - Position Embedding
- Applications
  - Language Modeling
  - CV



Applications

- Pretrained Language Model
  - The language model is to represent a language and then apply the model to general task.

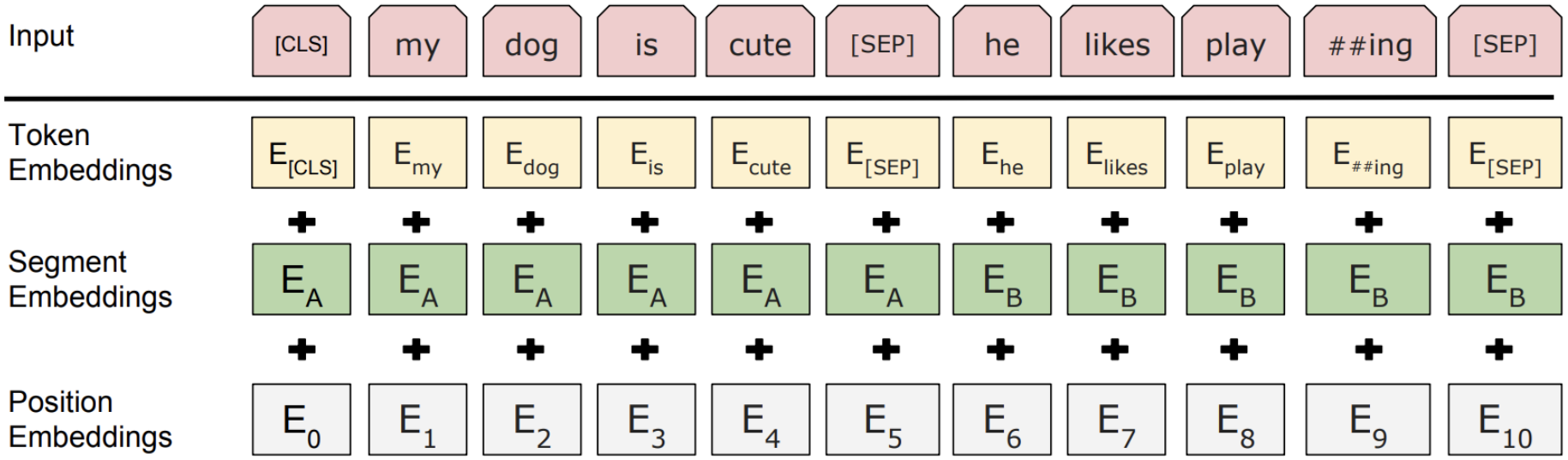


BERT

- Masking Mechanism

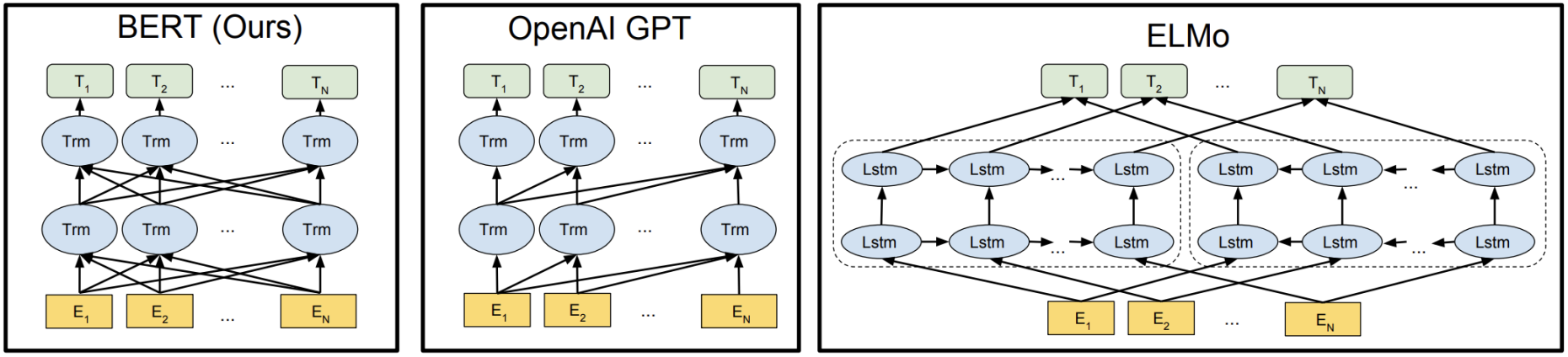
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com



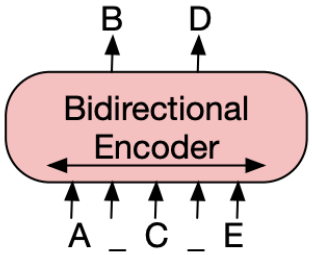
Other Language Model

- Only the encoder is pre-trained.

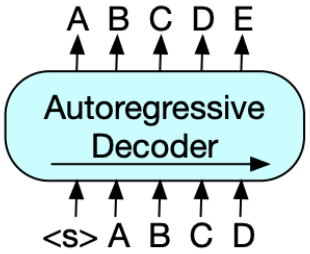


End-to-End Language Model

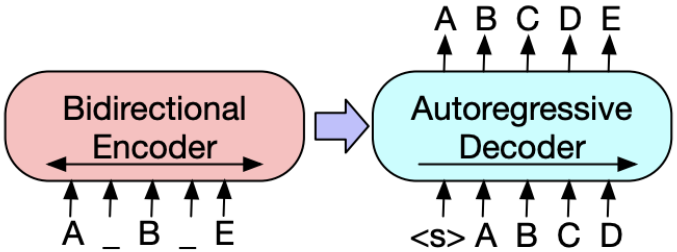
- *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*



(a) BERT: Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.

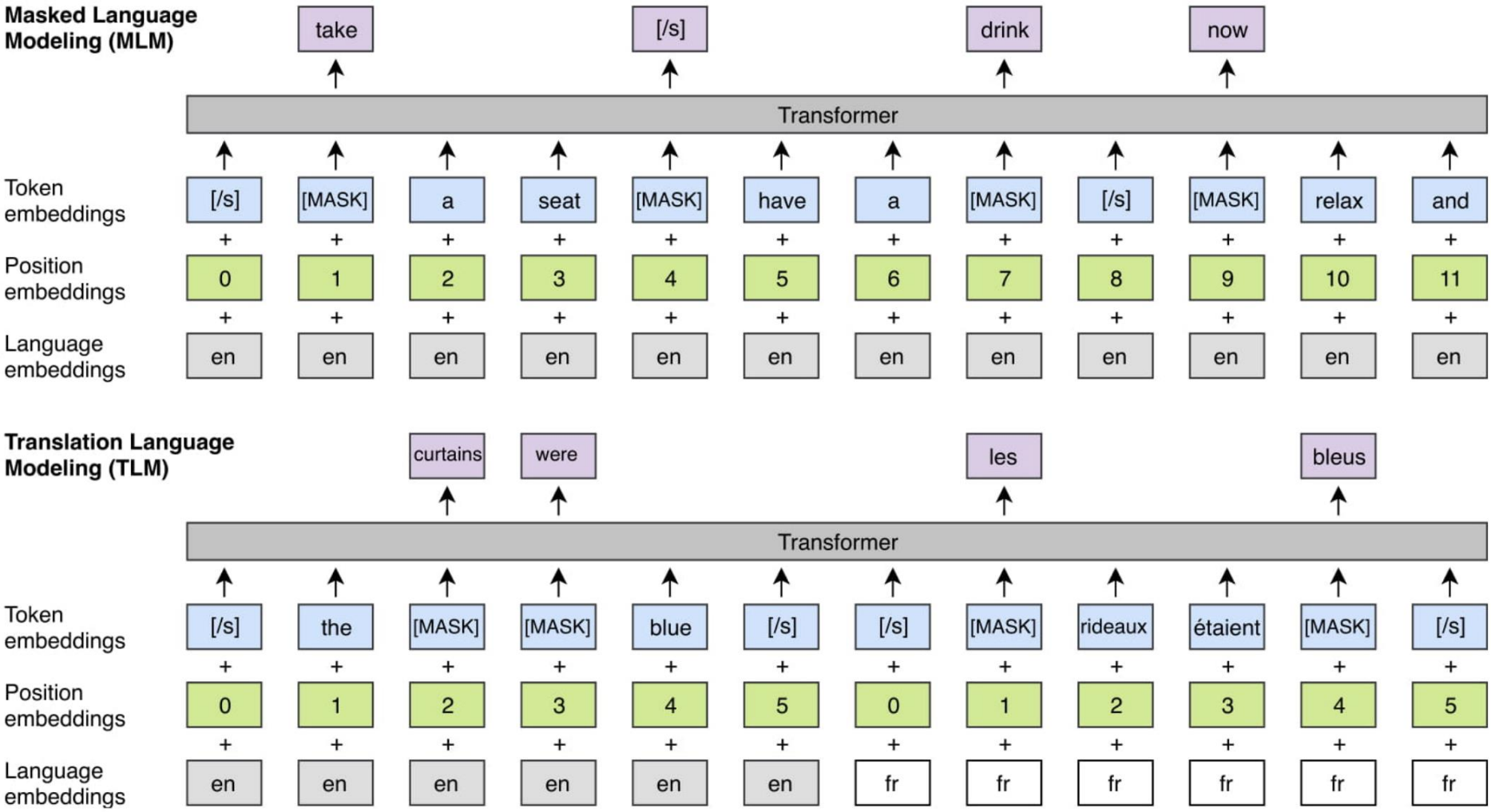


(b) GPT: Tokens are predicted auto-regressively, meaning GPT can be used for generation. However words can only condition on leftward context, so it cannot learn bidirectional interactions.



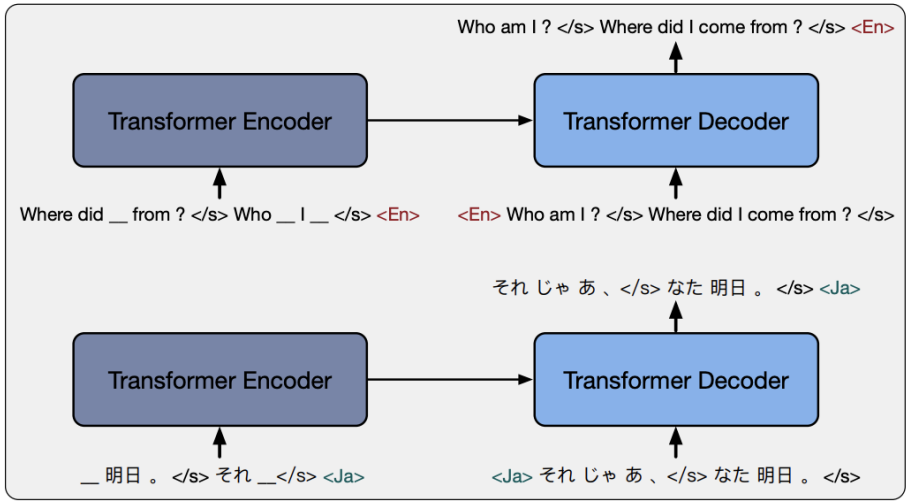
(c) BART: Inputs to the encoder need not be aligned with decoder outputs, allowing arbitary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder, and we use representations from the final hidden state of the decoder.

Cross-Lingual Pretraining

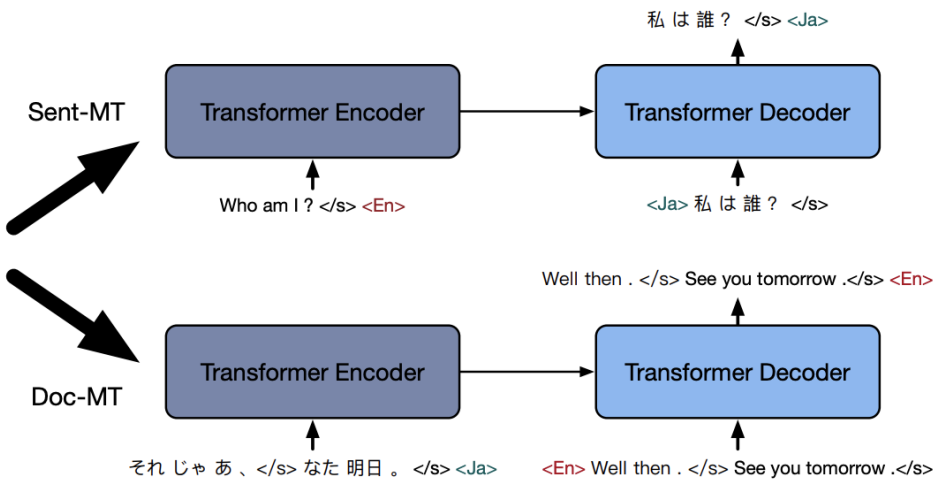


XLM: Unsupervised Cross-lingual Representation Learning at Scale

Cross-Lingual Pretraining



Multilingual Denoising Pre-Training (mBART)



Fine-tuning on Machine Translation

MBART: Multilingual Denoising Pre-training for Neural Machine Translation

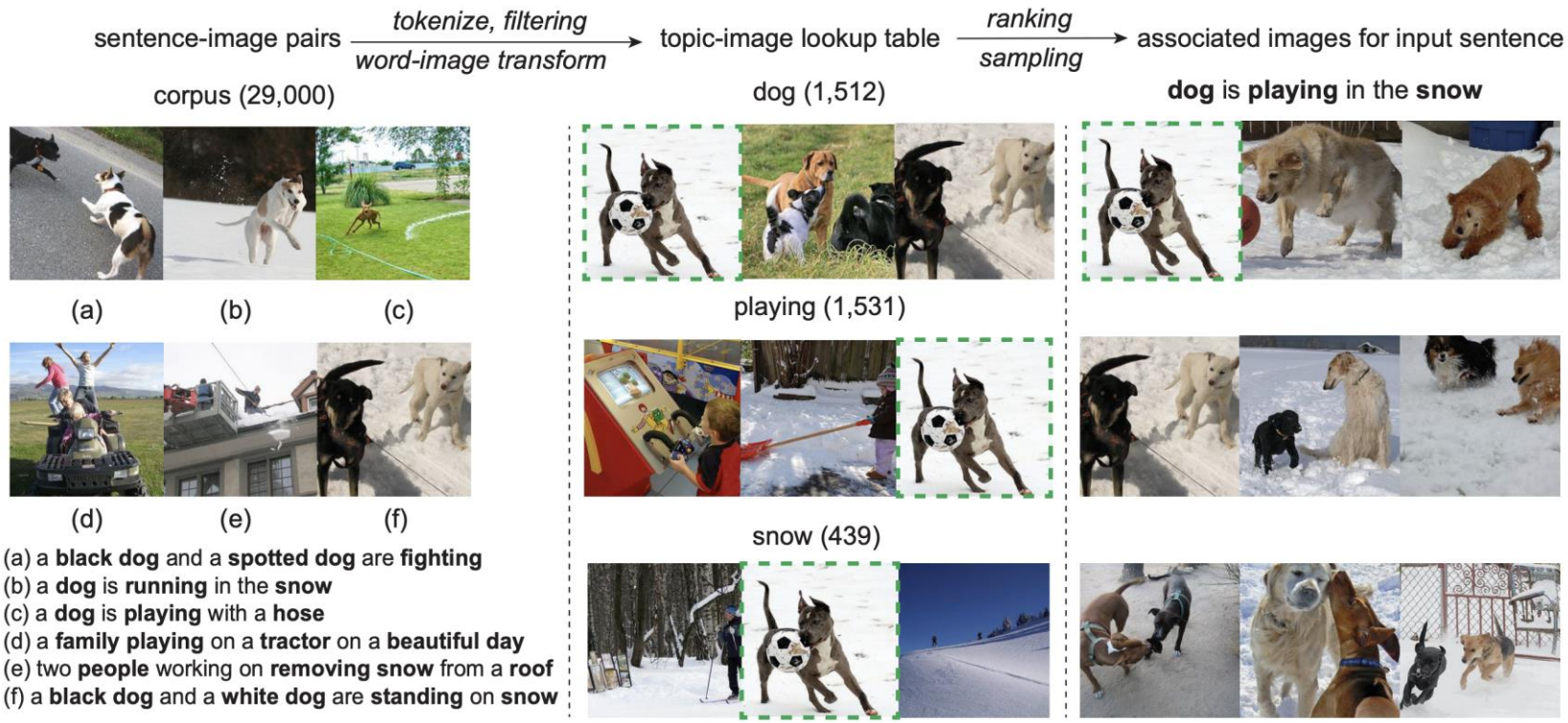
Multi-model in NLP

- How to interact between different signals
  - Pre-train by signal-A -> Fine-tune by signal-B
  - Joint Learn from signal-A and signal-B
  - How human brain deal with Coca-Cola

- No large-scale annotated data.
- How to measure the performance, especially whether the improvement is from the multi-modelity
- …

Image-Text Transformer

- We use image information to improve the MT performance
- No large-scale image-sentence annotated data is necessary



[Zhang and **Wang*** et al., 2020]

Image-Text Transformer

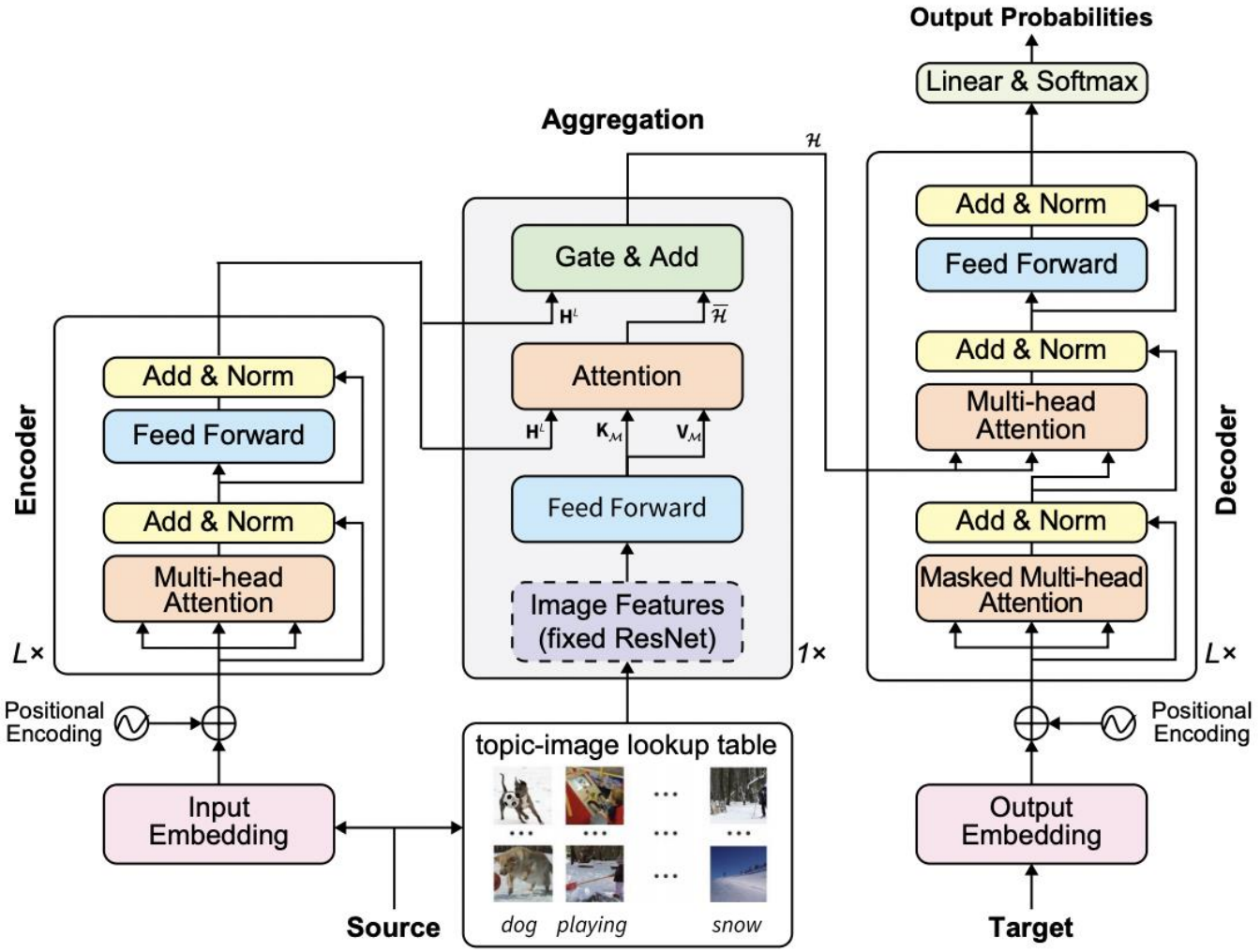
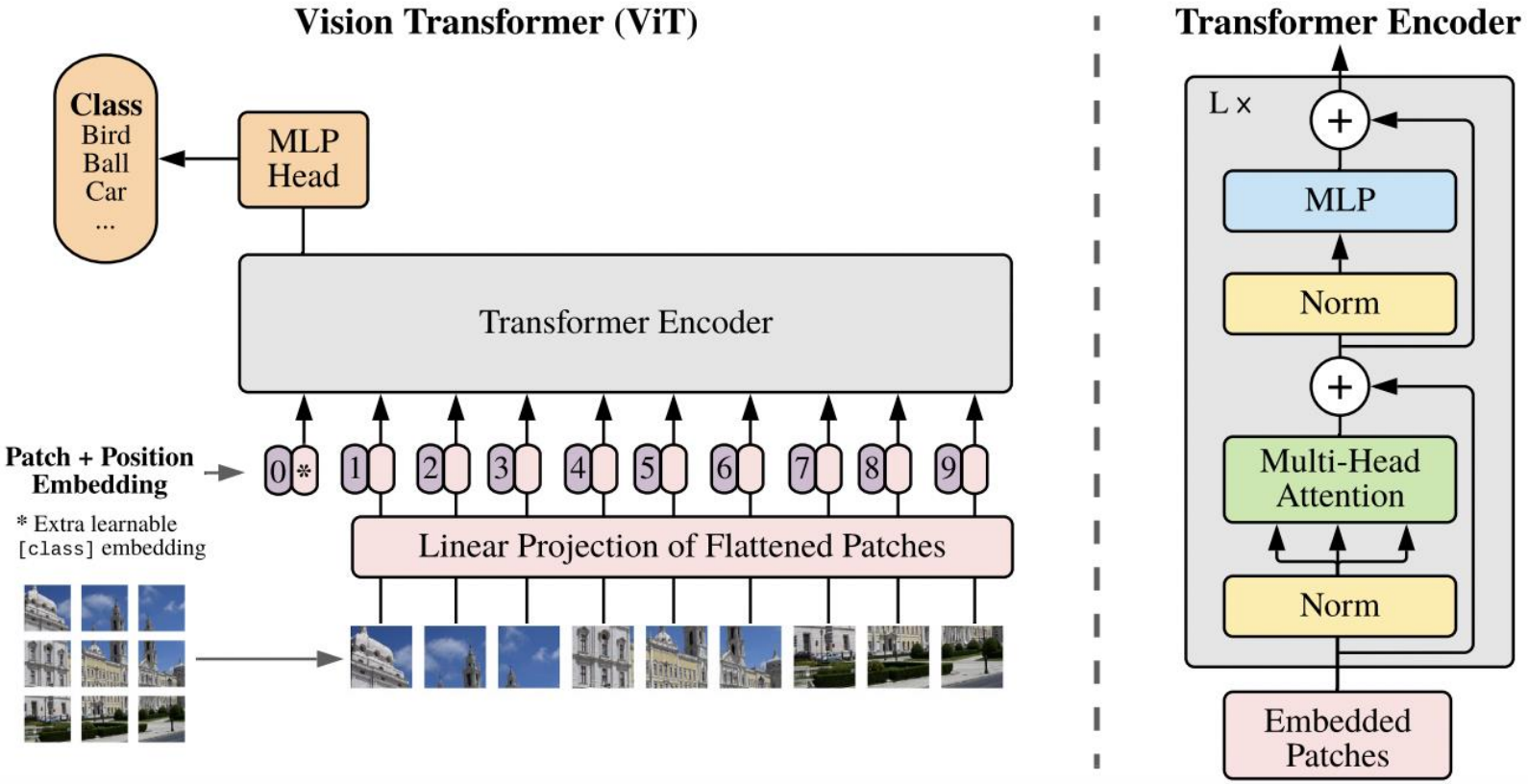


Figure 2: Overview of the framework of our proposed method.

Transformer in CV

- How model self-attention
  - Image is not a sequence.
  - Too many pixels.



Thank You!