

11868 LLM Systems Transformer

Lei Li



Carnegie Mellon University
Language Technologies Institute

Recap

- Design of a Deep Learning Framework
 - Tensorflow, a computation graph defined as dataflow
 - Auto differentiation
 - Scheduling of jobs

Today's Topic

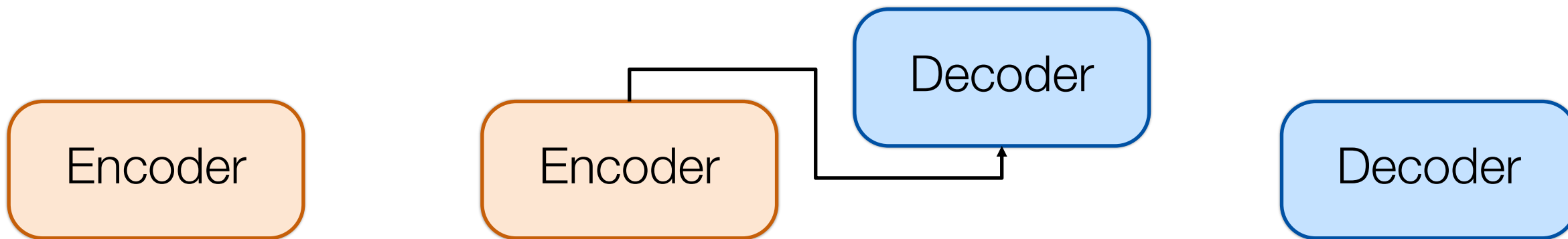
- ➡ • Transformer model
- How to implement Transformer

Type of Language Models

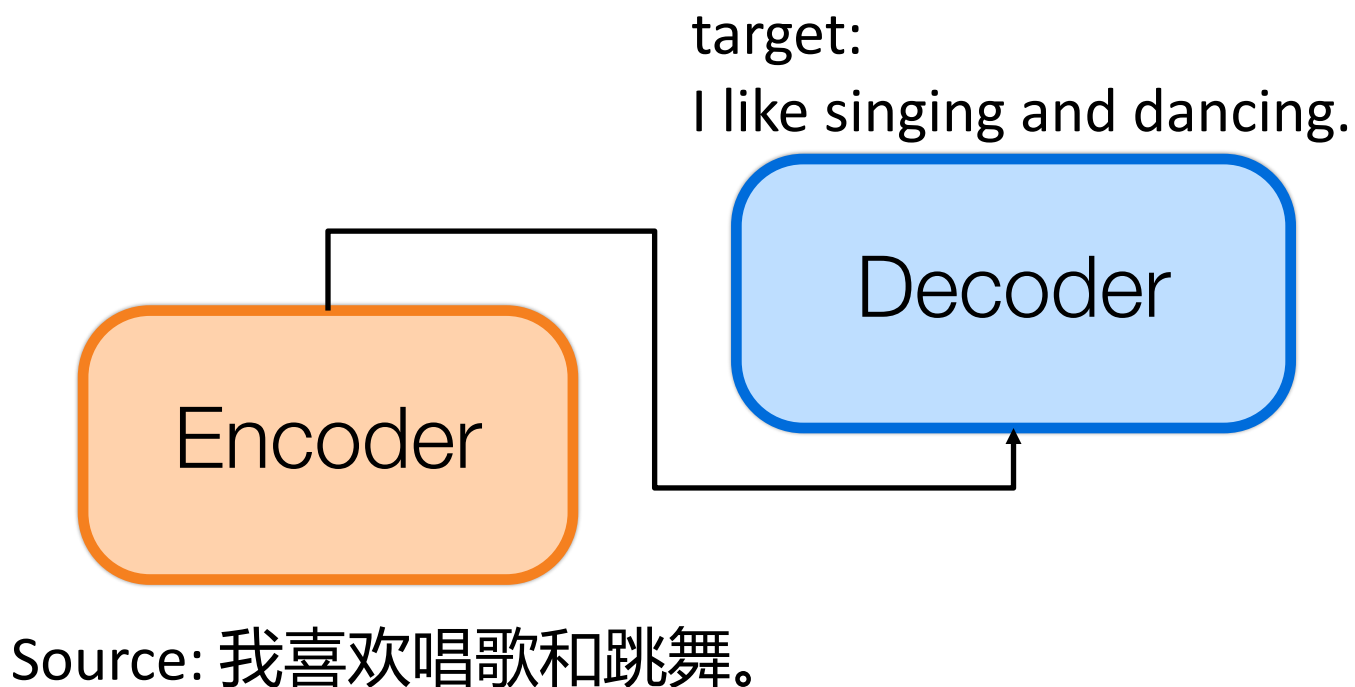
Encoder-only
Masked LM
Non-autoregressive

Encoder-decoder

Decoder-only
Autoregressive



Encoder-Decoder Paradigm



$$p_{\theta}(y|x) = \prod_i \underline{p(y_i|x, y_{1:i-1})}$$

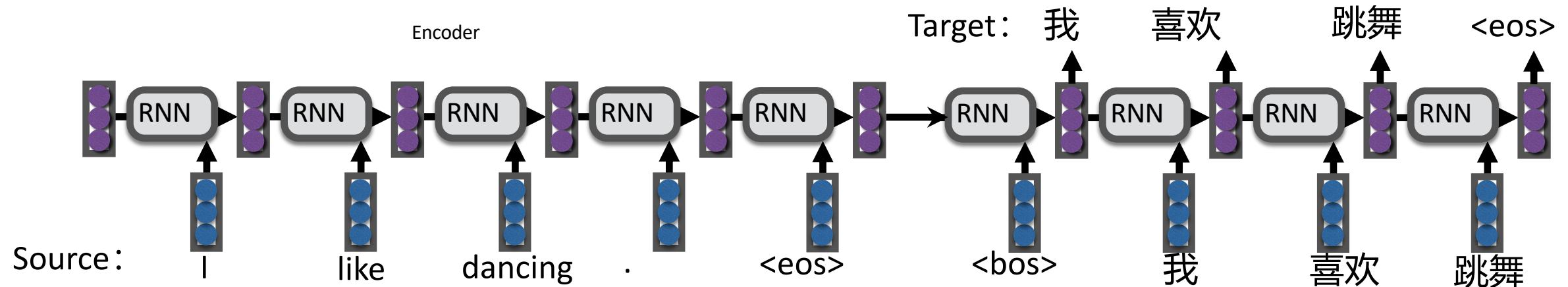
conditional prob. modeled by
neural networks

Sequence to Sequence Learning

- Conditional text generation: directly learning a function mapping from source sequence to target sequence

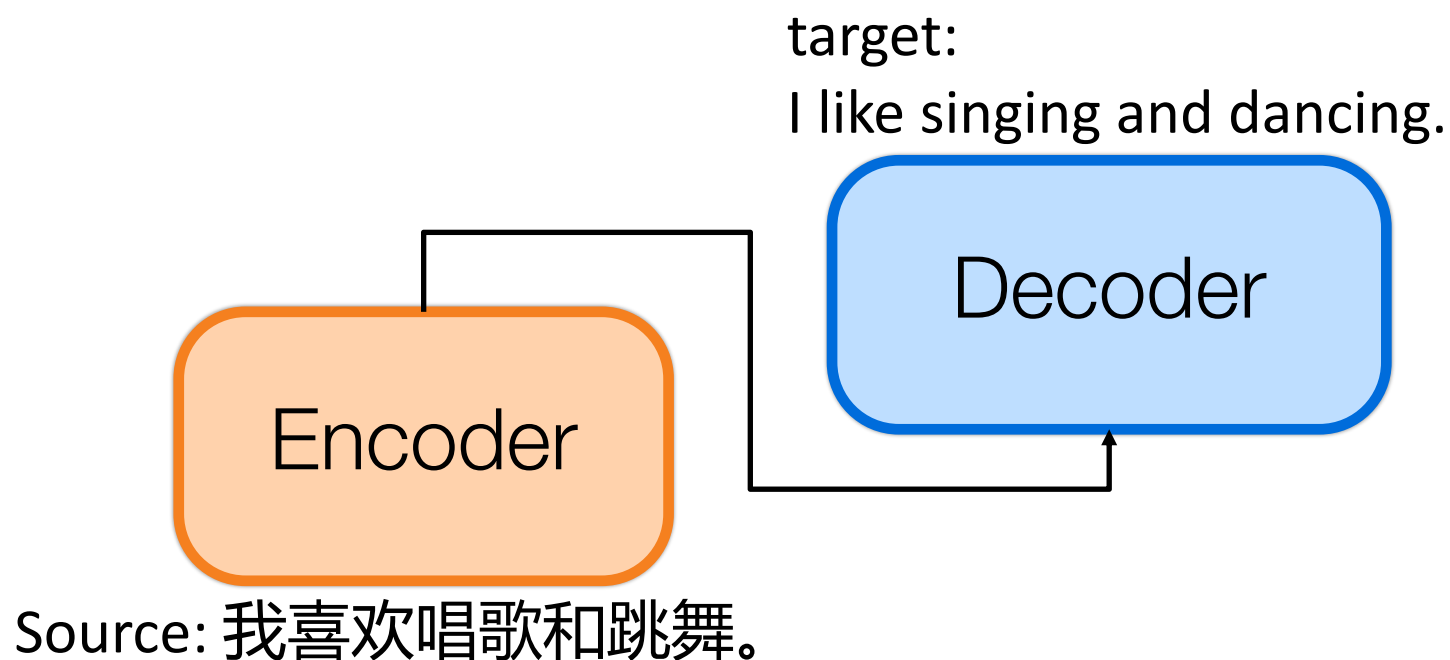
$$p_{\theta}(y|x) = \prod_t p(y_t | x, y_{1:t-1}; \theta)$$

- Previous encoder/decoder: LSTM or GRU

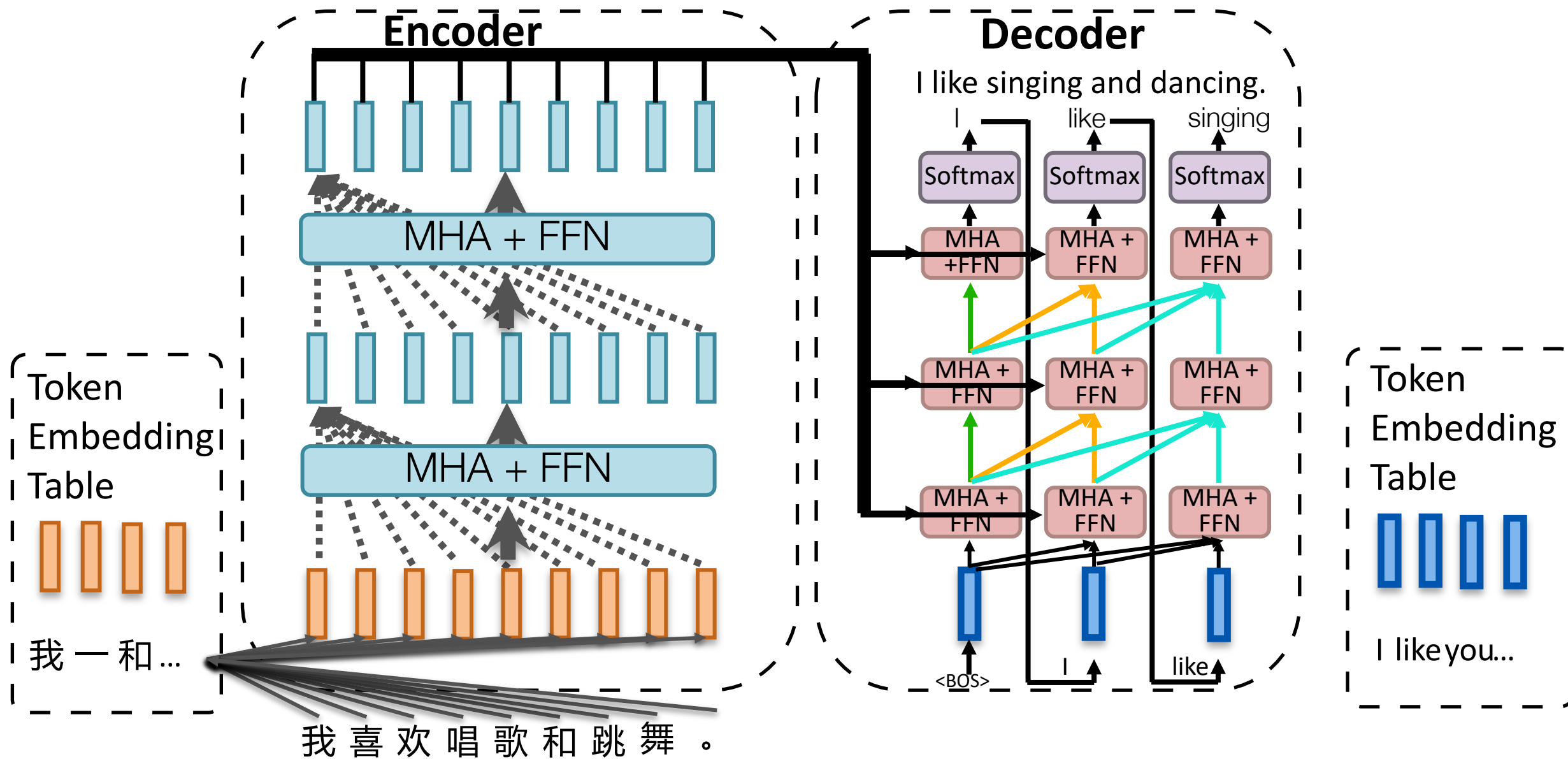


Motivation for a new Architecture

- Full context and parallel: use Attention in both encoder and decoder
- no recurrent ==> concurrent encoding



Transformer



Multi-head Attention

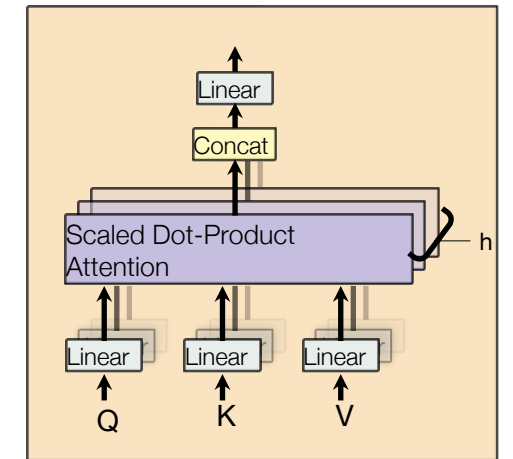
- Instead of one vector for each token
- break into multiple heads

- each head perform attention

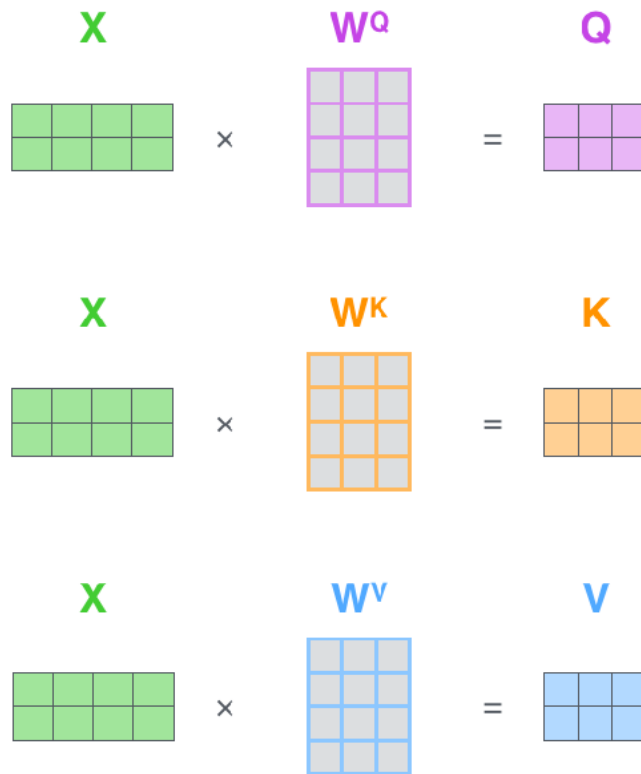
$$\text{Head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$\text{MultiHead}(Q, K, V)$$

$$= \text{Concat}(\text{Head}_1, \text{Head}_2, \dots, \text{Head}_h)W^O$$



Multi-head Attention



sent len x sent len

$$\text{softmax} \left(\frac{Q \times K^T}{\sqrt{d_k}} \right) \times V$$

=

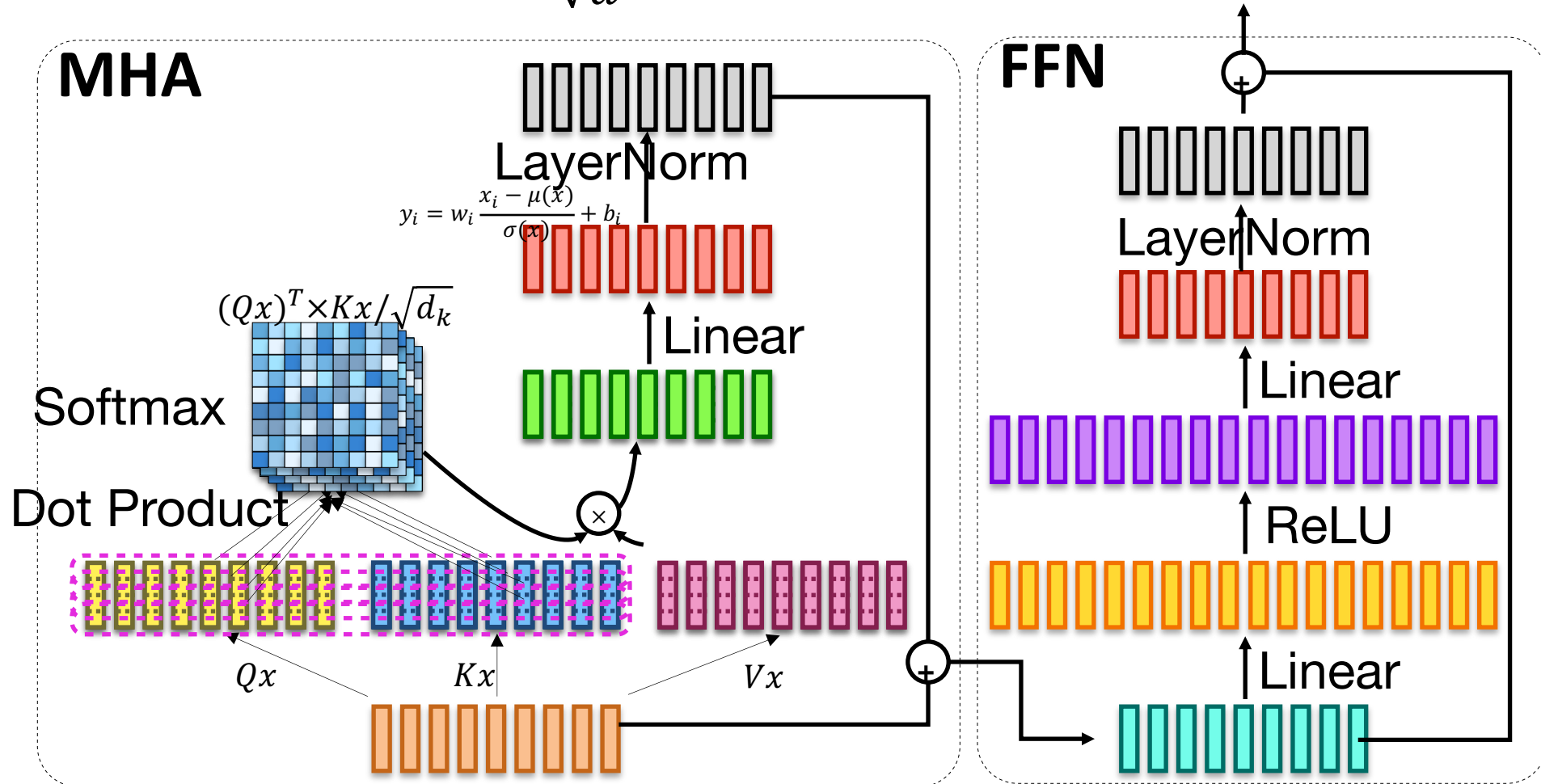
Z

The result Z is a 2x3 pink grid.

sent len x dim

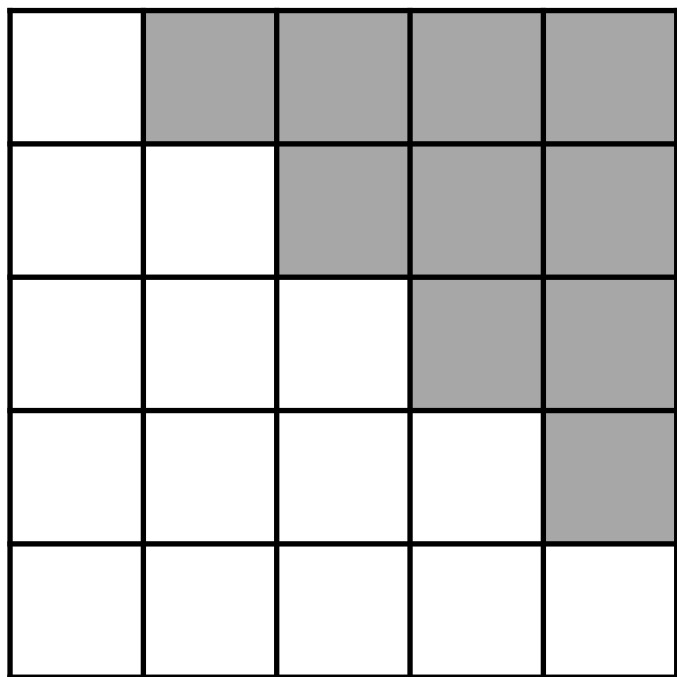
Multihead Attention and FFN

$$\text{Attention}(Q, K, V, x) = \text{Softmax}\left(\frac{(Qx)^T Kx}{\sqrt{d}}\right) \cdot (Vx)^T \quad \text{FFN}(x) = \max(0, x \cdot W_1 + b_1) \cdot W_2 + b_2$$

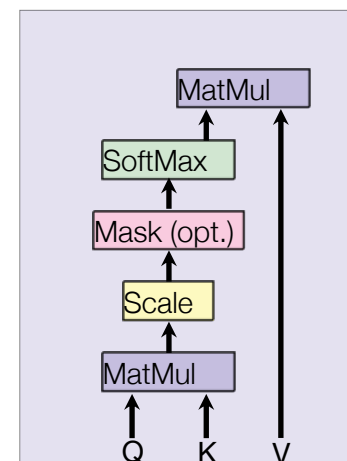


Self-Attention for Decoder

- Maskout right side before softmax (-inf)

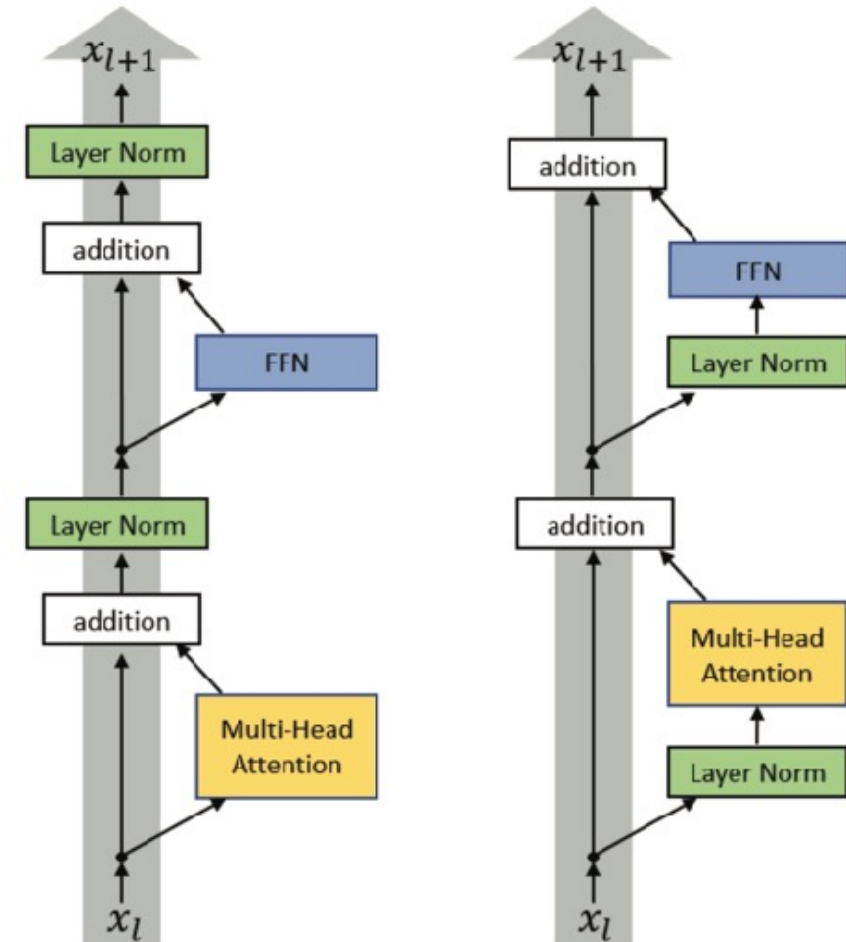


Scaled Dot-Product Attention



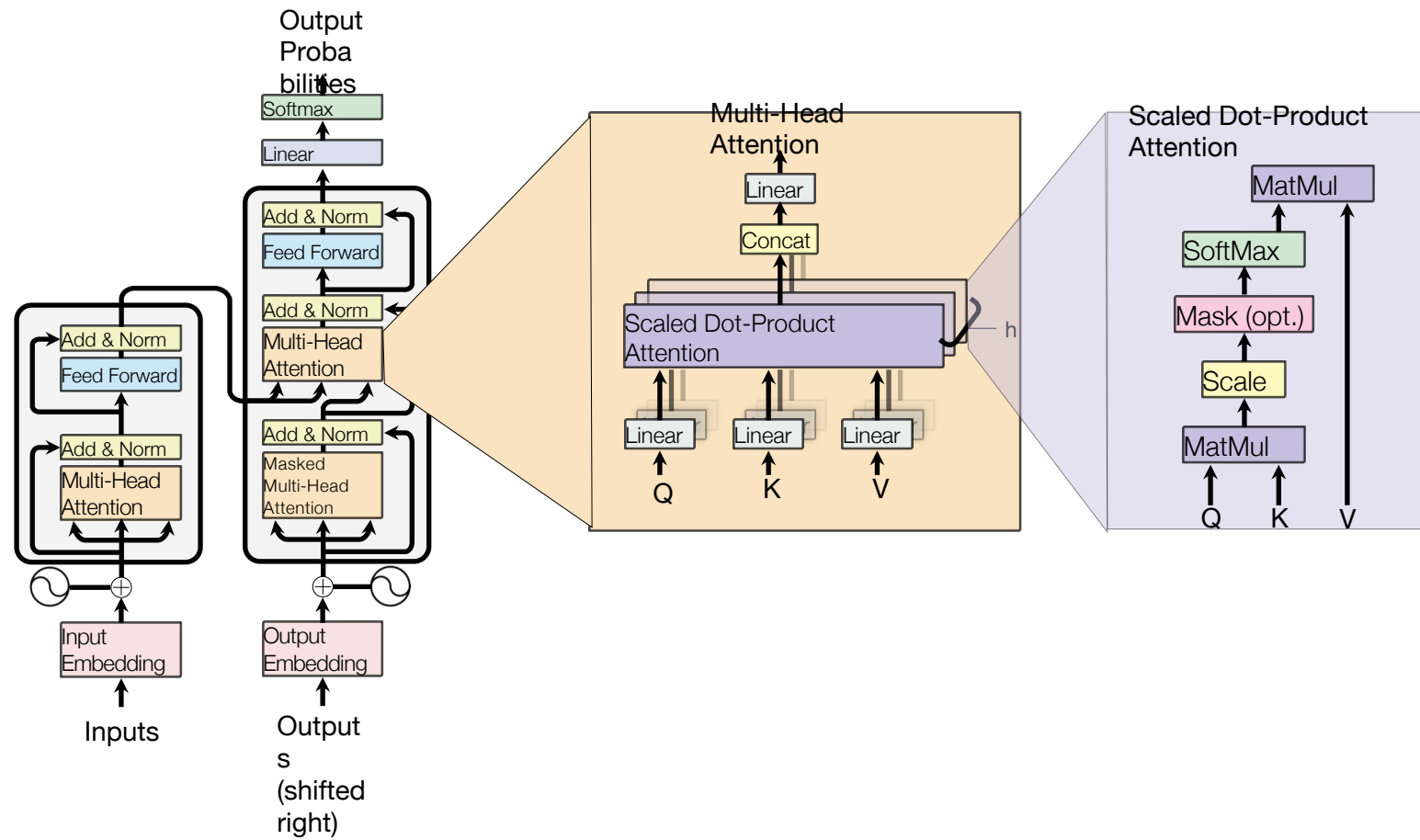
Residual Connection and Layer Normalization

- Residual Connection
- Make it zero mean and unit variance within layer
- Post-norm
- Pre-norm



Transformer in Original Paper

- C layers of encoder (=6)
- D layers of decoder (=6)
- Token Embedding: 512 (base), 1024 (large)
- FFN dim=2048

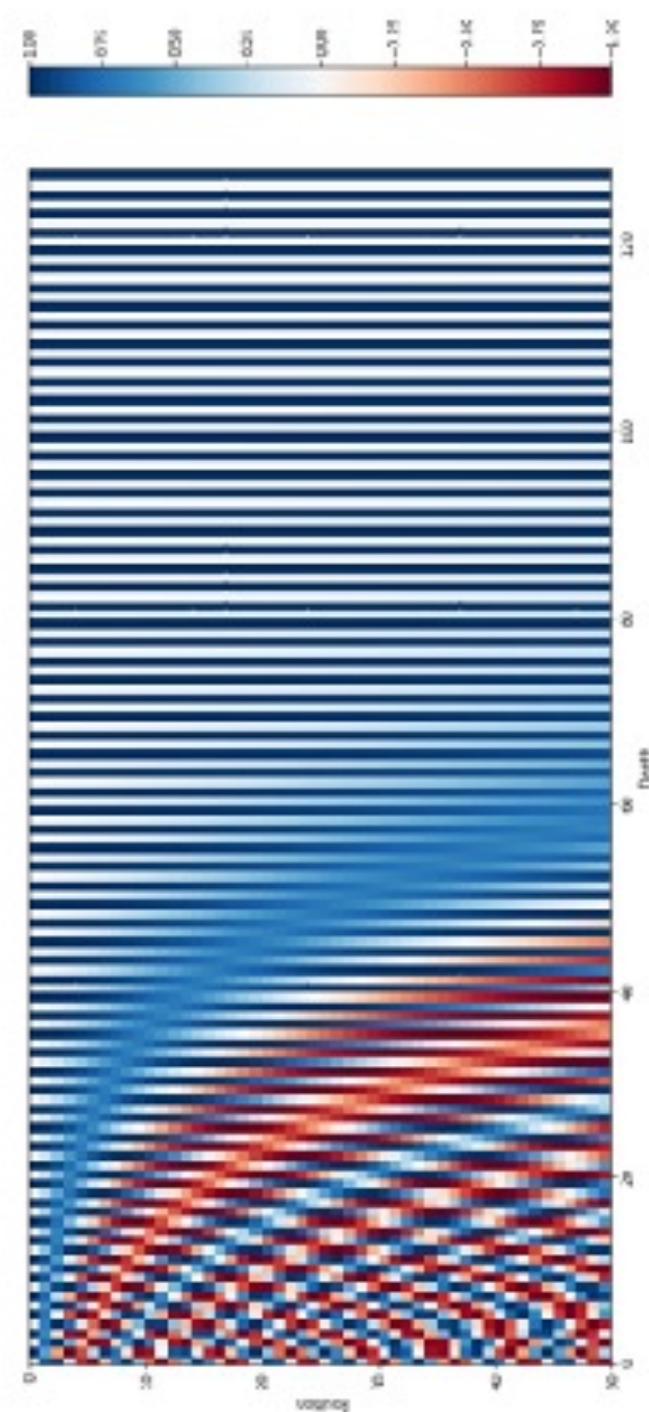


Embedding

- Token Embedding:
 - Shared (tied) input and output embedding
- Positional Embedding:
 - to distinguish words in different position, Map position labels to embedding, dimension is same as Tok Emb

- $PE_{pos,2i} = \sin(\frac{pos}{1000^{2i/d}})$

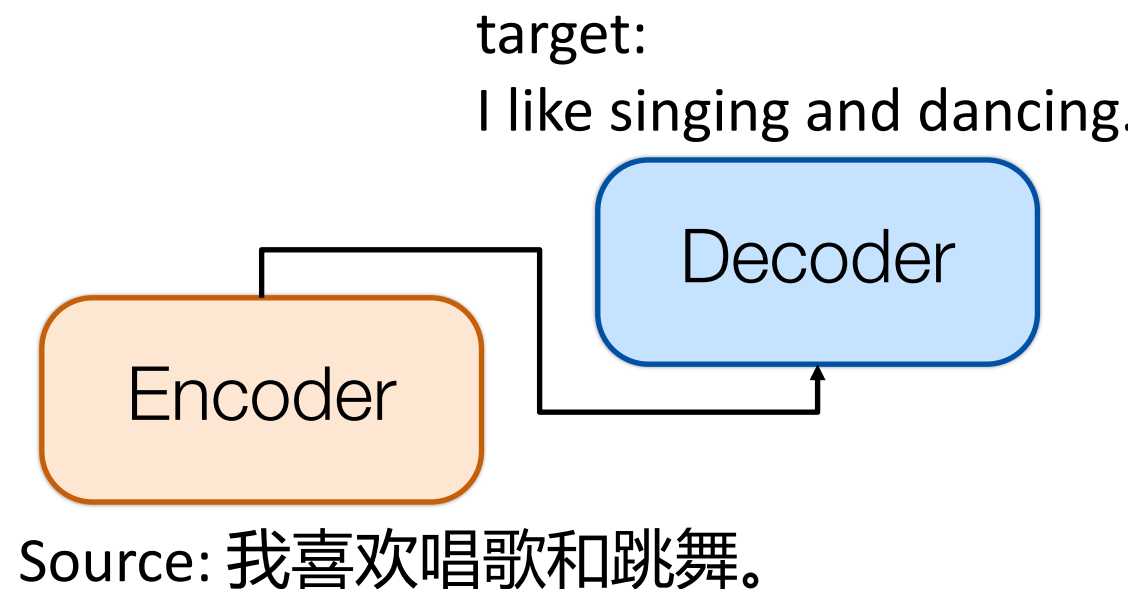
- $PE_{pos,2i+1} = \cos(\frac{pos}{1000^{2i/d}})$



Training Transformer

$$P(Y|X) = \prod P(y_t | y_{<t}, x)$$

- Training loss: Cross-Entropy
$$l = -\sum_n \sum_t \log f_\theta(x_n, y_{n,1}, \dots, y_{n,t-1})$$
- Teacher-forcing during training.
- pretend to know groundtruth for prefix



Training Transformer for MT

- Dropout
 - Applied to before residual
 - and to embedding, pos emb.
 - $p=0.1 \sim 0.3$
- Label smoothing
 - 0.1 probability assigned to non-truth
- Vocabulary:
 - En-De: 37K using BPE
 - En-Fr: 32k word-piece (similar to BPE)

Label Smoothing

- Assume $y \in R^n$ is the one-hot encoding of label

$$y_i = \begin{cases} 1 & \text{if belongs to class } i \\ 0 & \text{otherwise} \end{cases}$$

- Approximating 0/1 values with softmax is hard

- The smoothed version

$$y_i = \begin{cases} 1 - \epsilon & \text{if belongs to class } i \\ \epsilon / (n - 1) & \text{otherwise} \end{cases}$$

- Commonly use $\epsilon = 0.1$

Training

- Batch
 - group by approximate sentence length
 - still need shufflingHardware
 - one machine with 8 GPUs (in 2017 paper)
 - base model: 100k steps (12 hours)
 - large model: 300k steps (3.5 days)
- Adam Optimizer
 - increase learning rate during warmup, then decrease
 - $\eta = \frac{1}{\sqrt{d}} \min\left(\frac{1}{\sqrt{t}}, \frac{t}{\sqrt{t_0^3}}\right)$

ADAM

$$m_{t+1} = \beta_1 m_t - (1 - \beta_1) \nabla \ell(x_t)$$

$$v_{t+1} = \beta_2 v_t + (1 - \beta_2) (\nabla \ell(x_t))^2$$

$$\hat{m}_{t+1} = \frac{m_{t+1}}{1 - \beta_1^{t+1}}$$

$$\hat{v}_{t+1} = \frac{v_{t+1}}{1 - \beta_2^{t+1}}$$

$$x_{t+1} = x_t - \frac{\eta}{\sqrt{\hat{v}_{t+1}} + \epsilon} \hat{m}_{t+1}$$

Model Average

- A single model obtained by averaging the last 5 checkpoints, which were written at 10-minute interval (base)
- decoding length: within source length + 50

Summary

- Sequence-to-sequence encoder-decoder framework for conditional generation, including Machine Translation
- Key components in Transformer
 - Positional Embedding (to distinguish tokens at different pos)
 - Multihead attention
 - Residual connection
 - layer norm

Code Go-through

<https://nlp.seas.harvard.edu/annotated-transformer/>

Reading for Next Class

- Neural Machine Translation of Rare Words with Subword Units. Sennrich et al. 2016.
- SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. Kudo and Richardson. 2018