LLM Architecture

Transformer and variants

Lei Li

Language Technologies Institute | Carnegie Mellon University School of Computer Science

Recap

- Learning parameters of an NN needs gradient calculation
- Computation Graph
  - to perform computation: topological traversal along the DAG
- Auto Differentiation
  - building backward computation graph for gradient calculation
- Put together: Deep Learning Framework
  1. Define program i.e., symbolic computation graph w/ placeholders/variable/operation nodes
  2. Executes (optimized) computation graph on a set of available devices

Today's Topic

- Encoder-decoder Architecture

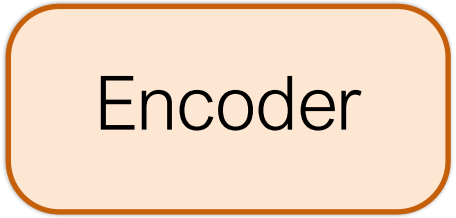
- Transformer Model
  - Embedding
  - Multihead Attention, Decoder Self-Attention
  - FFN and SwiGLU
  - Layernorm
  - RoPE

- Training Techniques and Performance of Transformer

- Pretrained LLMs

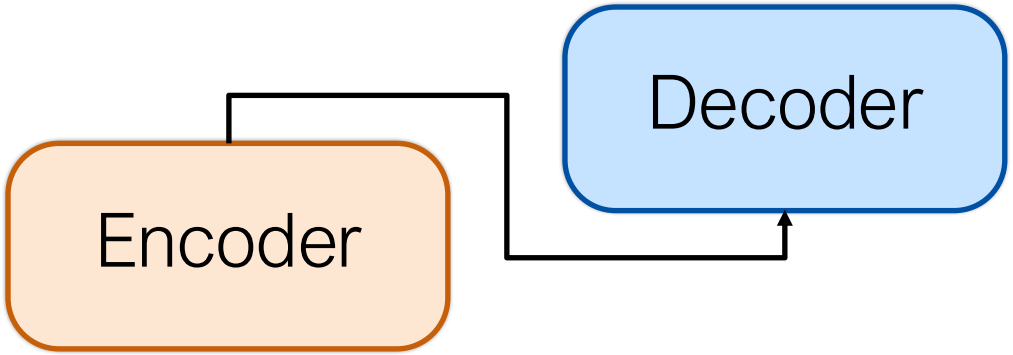
Type of Language Models

Encoder-only
Masked LM
Non-autoregressive



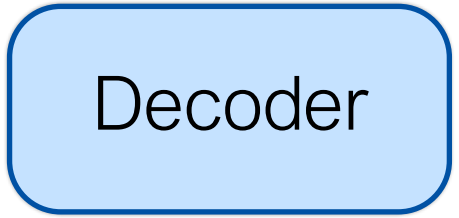
e.g. BERT
RoBERTa
ESM (for protein)

Encoder-decoder



e.g. T5, T5Gemma

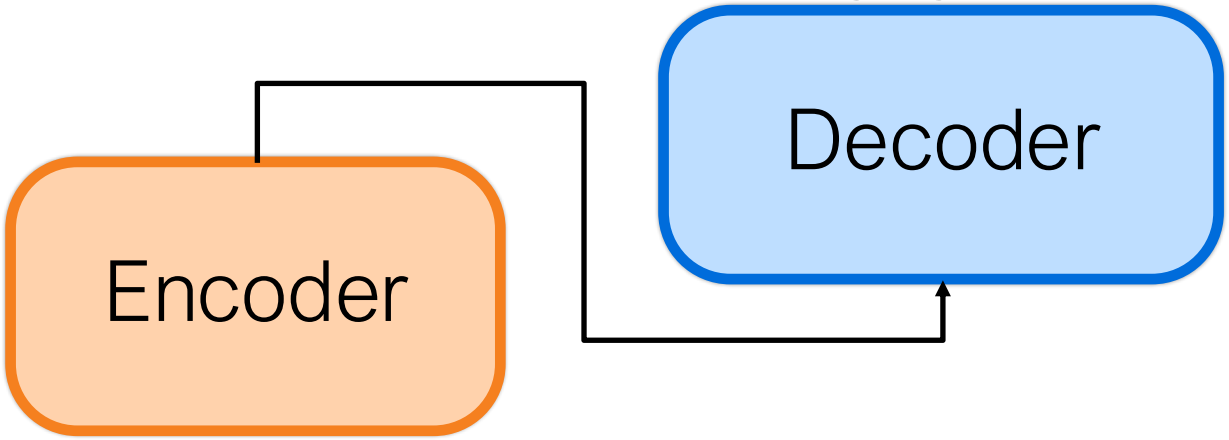
Decoder-only
Autoregressive



e.g. GPT
LLaMA
QWen
ProGen (for protein)

Encoder-Decoder Paradigm

target:
I like singing and dancing.



Source: 我喜欢唱歌和跳舞。

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

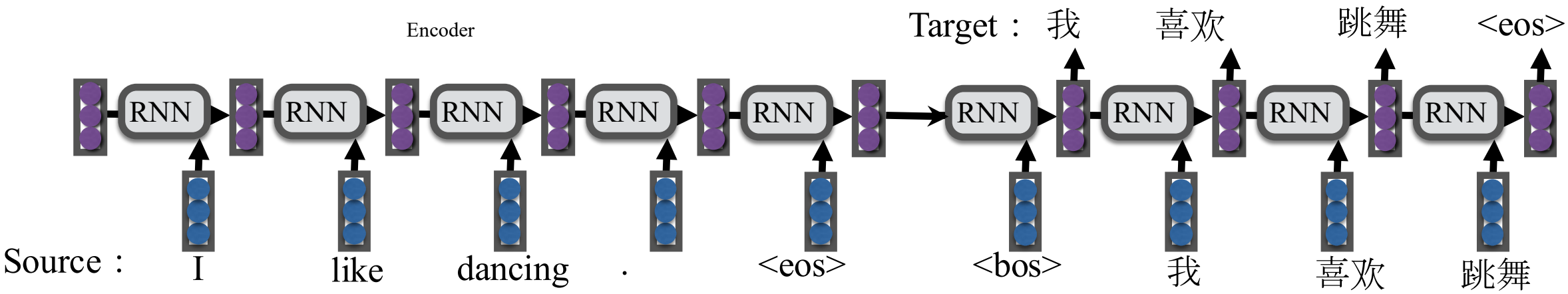
conditional prob. modeled by neural networks (Transformer)

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

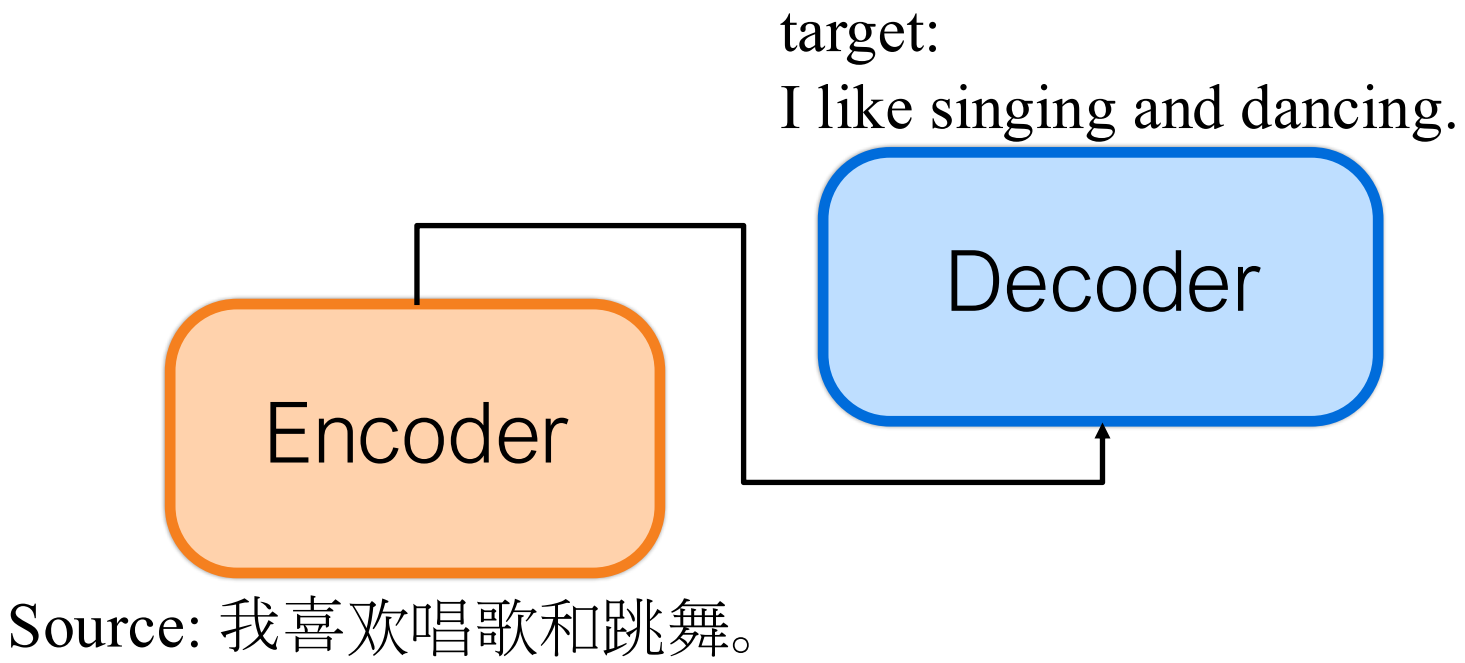


Sutskever et al. Sequence to Sequence Learning with Neural Networks. 2014

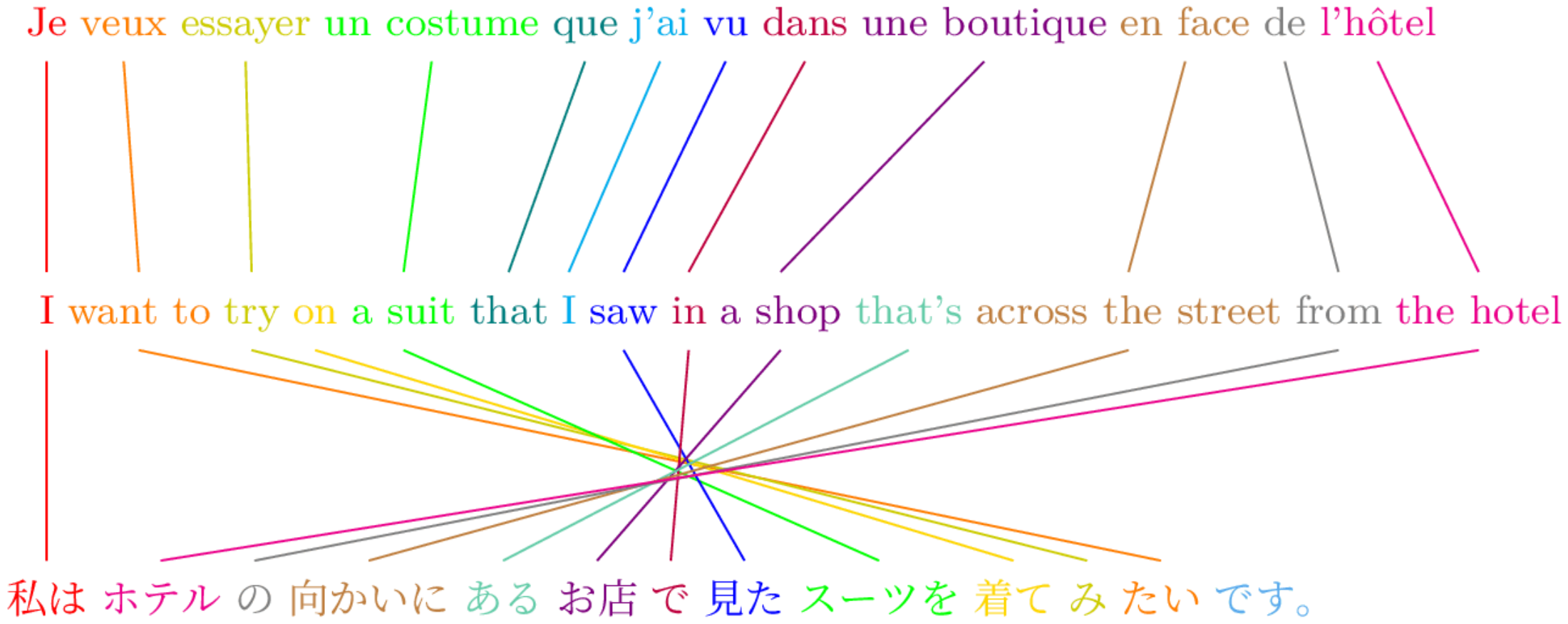
Motivation for a new Architecture

- Full context and parallel: use Attention in both encoder and decoder

- no recurrent ==> concurrent encoding



Motivation of Attention: translation need different alignment of words



Today's Topic

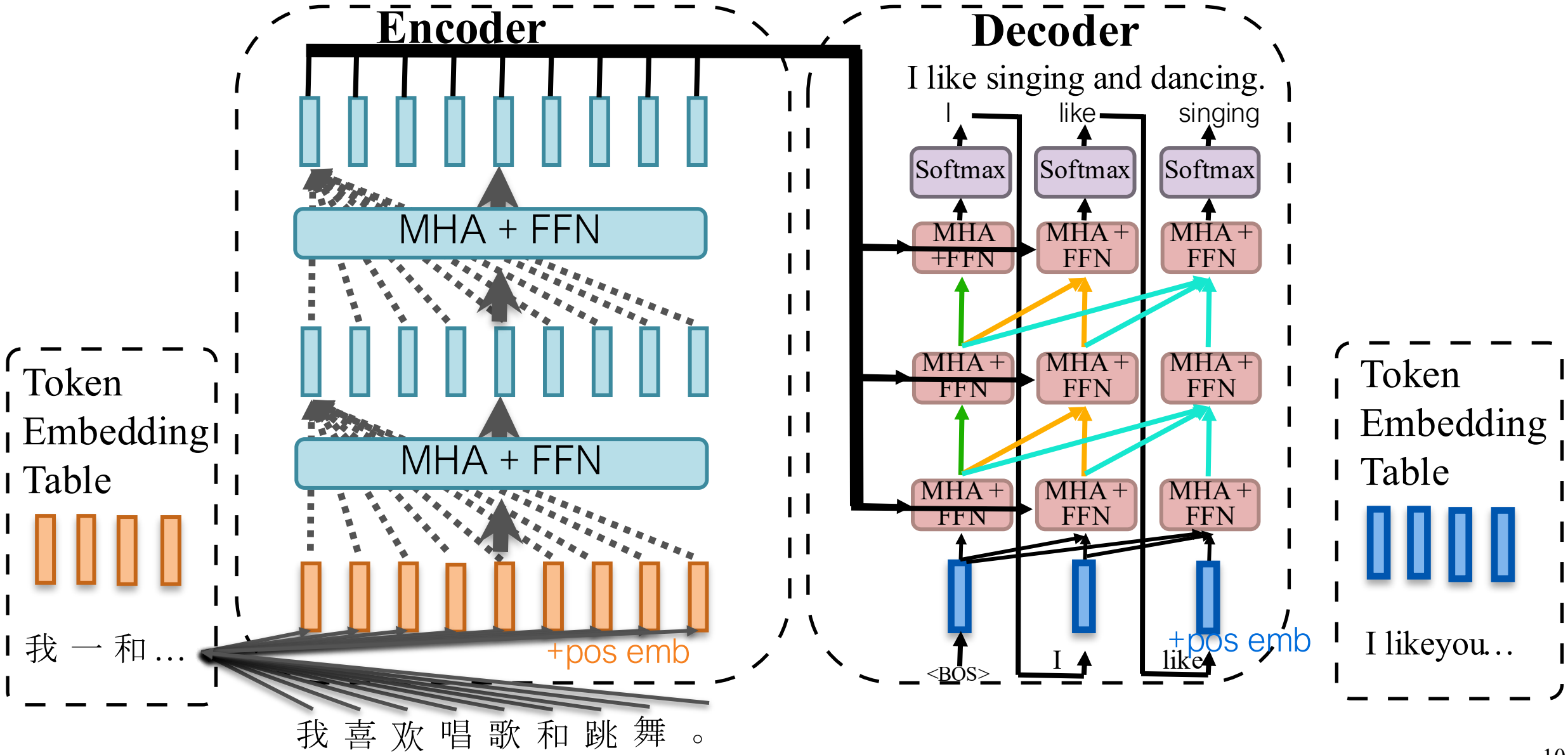
- Encoder-decoder Architecture

- Transformer Model
  - Embedding
  - Multihead Attention, Decoder Self-Attention
  - FFN and SwiGLU
  - Layernorm
  - RoPE

- Training Techniques and Performance of Transformer

- Pretrained LLMs

Transformer



Vaswani et al. Attention is All You Need. 2017

Embedding

- Token Embedding: (tokenization next lec.)
  - token index → vector
  - Shared (tied) input and output embedding from lookup table

- Positional Embedding:
  - to distinguish words in different position, Map position labels to embedding, dimension is same as Tok Emb, for t-th pos, i-th dim

$$PE_{t,2i} = \sin\left(\frac{t}{1000^{2i/d}}\right)$$

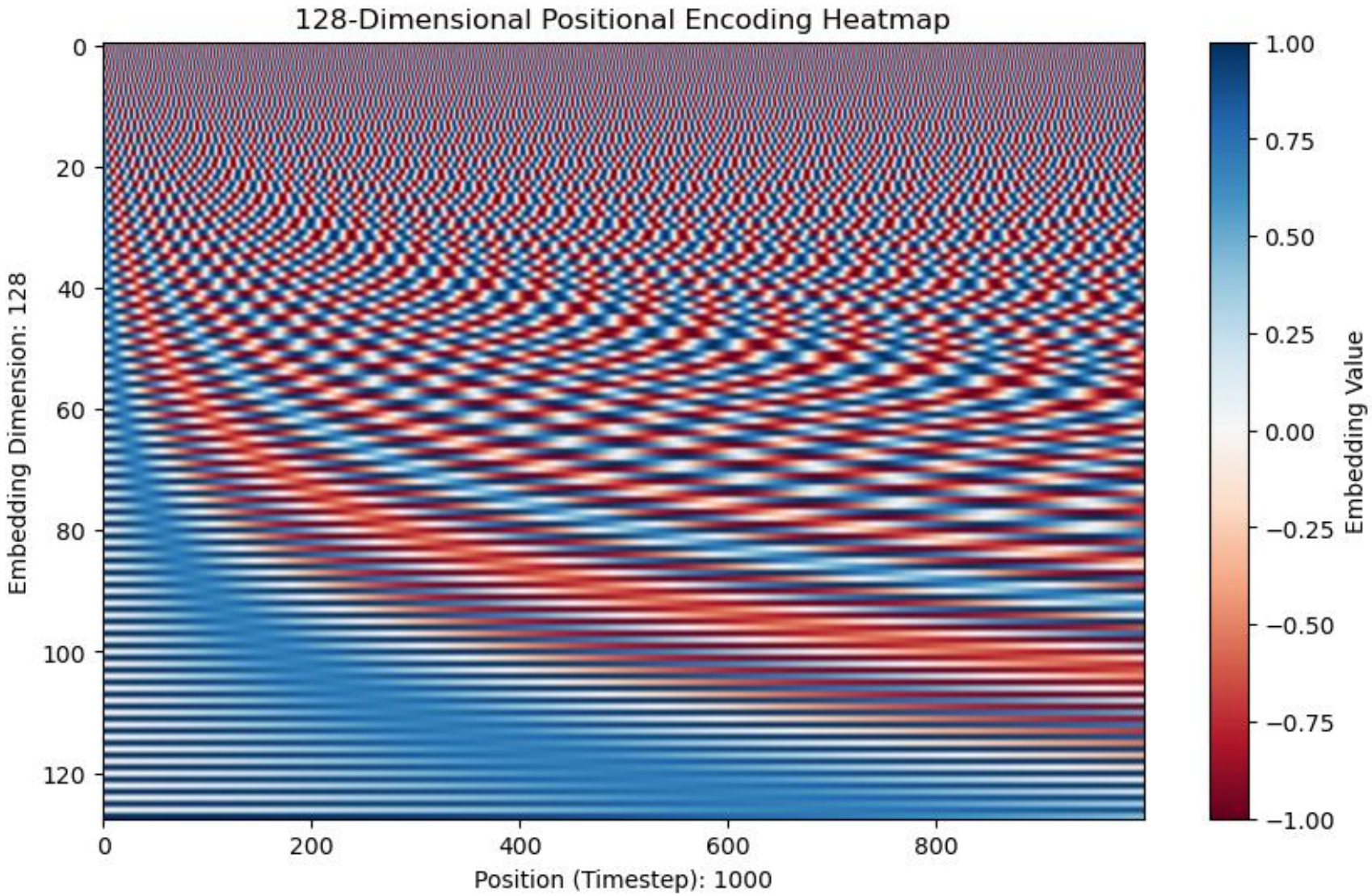
$$PE_{t,2i+1} = \cos\left(\frac{t}{1000^{2i/d}}\right)$$

Sinusoidal Positional Embedding

for t-th pos, i-th dim

$$PE_{t,2i} = \sin\left(\frac{t}{1000^{2i/d}}\right)$$

$$PE_{t,2i+1} = \cos\left(\frac{t}{1000^{2i/d}}\right)$$



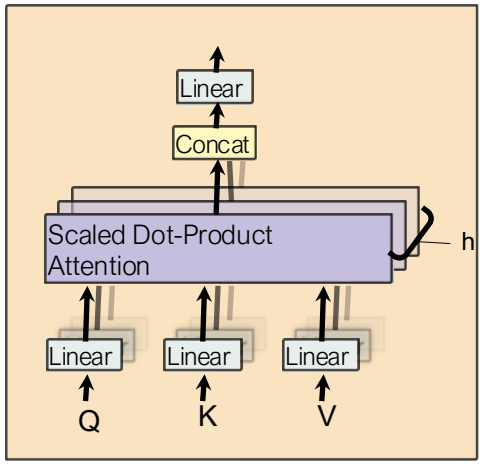
Multi-head Attention

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

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

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

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

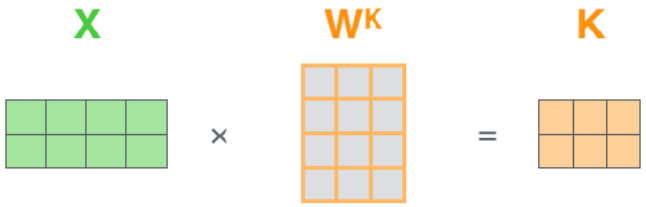


Multi-head Attention

X are input embeddings from previous layer (num of tok * dim)

Query

Key



Value

sent len x sent len

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

$$= Z$$

len x dim

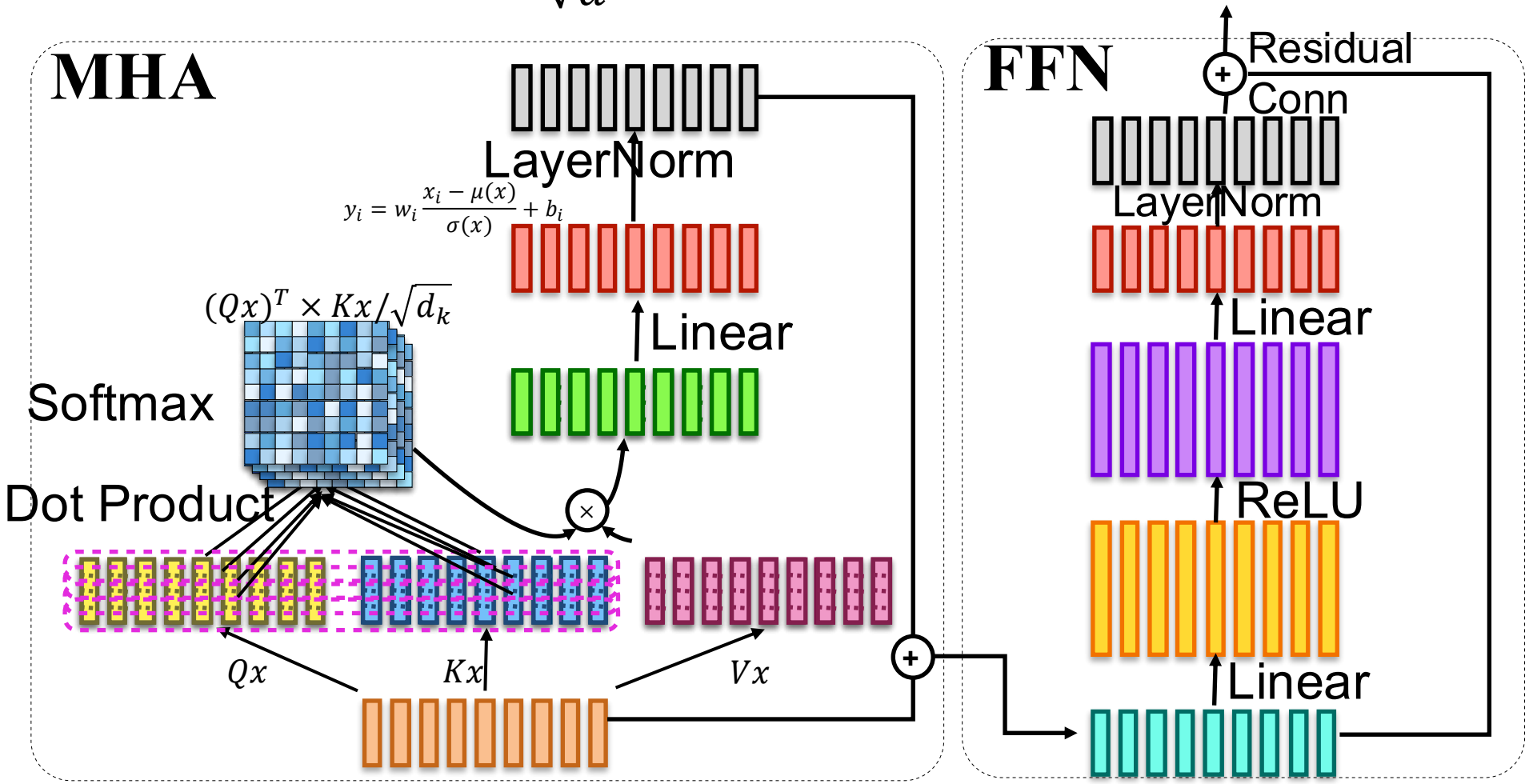
Q: why divided by sqrt(d)?

Alammar, The Illustrated Transformer

Multihead Attention and FFN

$$\text{Attention}(Q,K,V,x) = \text{Softmax}(\frac{(Qx)^T Kx}{\sqrt{d}}) \cdot (Vx)^T$$

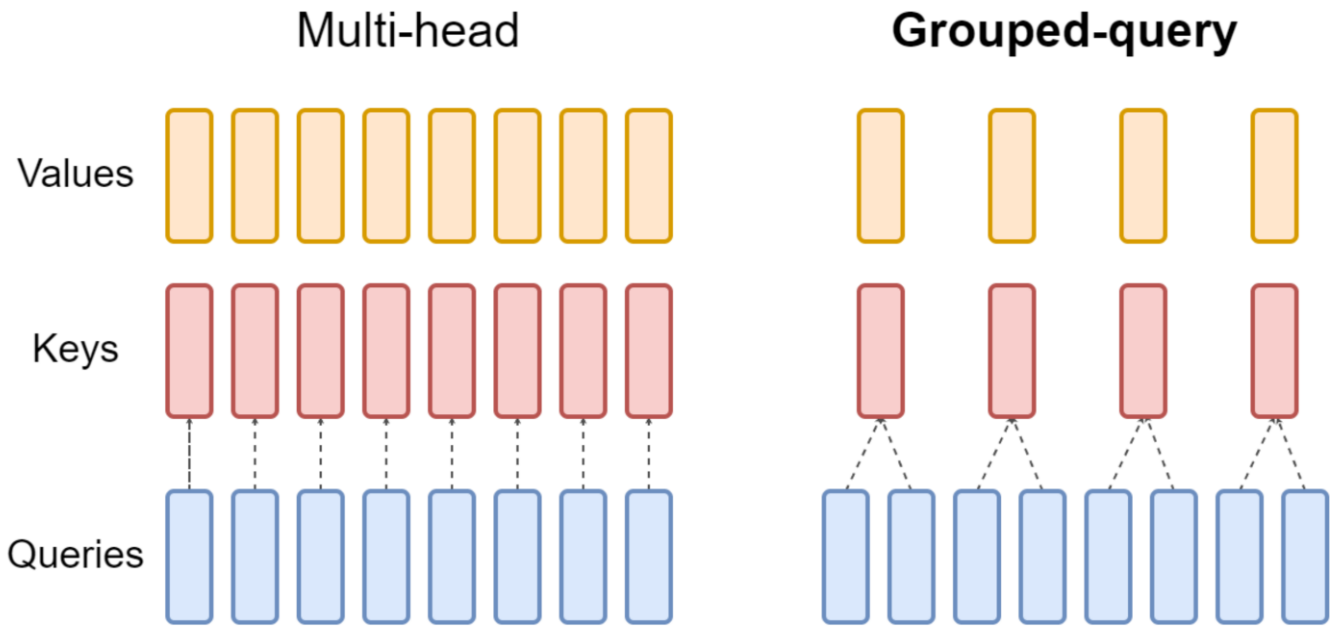
$$\text{FFN}(x) = max(0, x \cdot W_1 + b_1) \cdot W_2 + b_2$$



Attention Variant: Grouped-Query Attention (GQA)

$$\text{Head}_i = \text{Attention}(XW_h^Q, XW_{g(h)}^K, XW_{g(h)}^V)$$

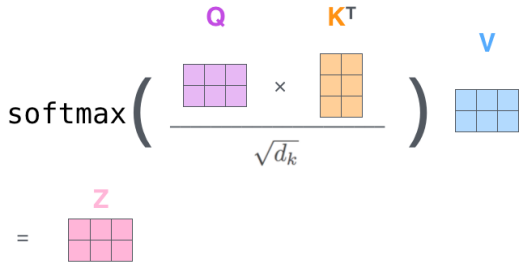
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{Head}_1, \text{Head}_2, \ldots, \text{Head}_h)W^o$$



GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, Sumit Sanghai. EMNLP 2023.

Attention Variant: Multi-head Latent Attention (MLA)

- Keys and values occupy large GPU memory (re-visit later in PagedAttention and KV cache lecture)
  - $2 \cdot len \cdot n_h \cdot d_h$



- instead of caching d_h vector (for each tk, head), reduce to a latent space with rank r

$d \times r$ $\quad$ $r \times (n \cdot d_h)$

$$Q = X \cdot W_Q^D \cdot W_Q^U$$
$$K = X \cdot W_{KV}^D \cdot W_K^U$$
$$V = X \cdot W_{KV}^D \cdot W_V^U$$

Key and Value share down-projection

- First proposed in Deepseek-V2, used in all modern LLMs

Multi-head Latent Attention (MLA)

$d \times r$ $\qquad$ $r \times (n \cdot d_h)$

$$Q = X \cdot W_Q^D \cdot W_Q^U$$
$$K = X \cdot W_{KV}^D \cdot W_K^U$$
$$V = X \cdot W_{KV}^D \cdot W_V^U$$

Key and Value share down-projection, and across all heads

$$MLA(x)_h$$

$$= \text{Softmax}\left(\frac{X \cdot W_Q^D \cdot W_{Q,h}^U \cdot \left(X \cdot W_{KV}^D \cdot W_{K,h}^U\right)^T}{\sqrt{d_h}}\right) \cdot X \cdot W_{KV}^D \cdot W_{V,h}^U$$

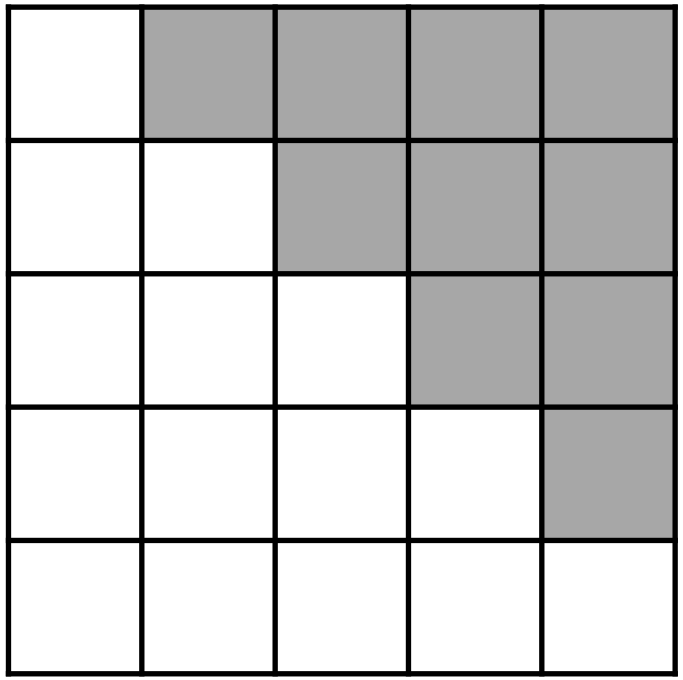
$$= \text{Softmax}\left(\frac{C_Q \cdot W_{QK,h} \cdot C_{KV}^T}{\sqrt{d_h}}\right) \cdot C_{KV} \cdot W_{V,h}^U$$

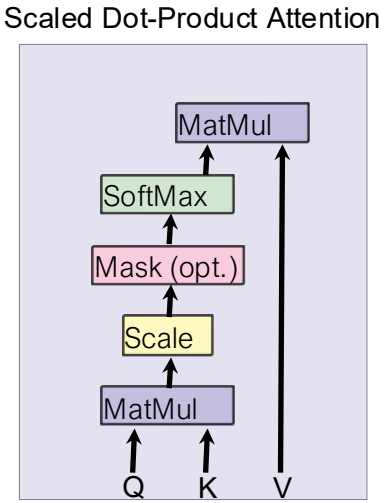
Precomputed/cached

$$W_{QK,h} = W_{Q,h}^U \cdot \left(W_{K,h}^U\right)^T$$
$$C_Q = X \cdot W_Q^D$$
$$C_{KV} = X \cdot W_{KV}^D$$

Decoder Self-Attention Mask

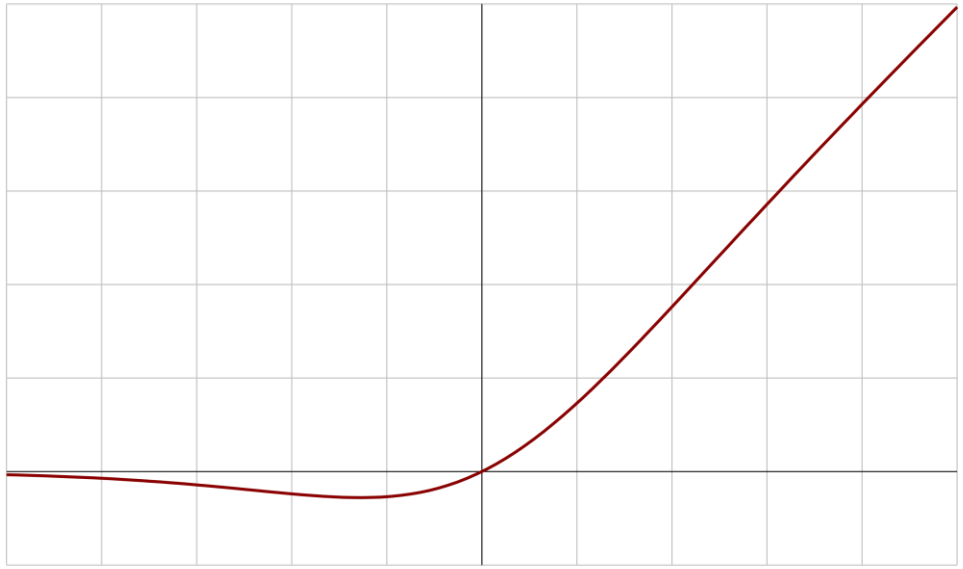
- Encoder uses bidirectional attention
- Decoder uses unidirectional attention
  - Maskout right side before softmax (-inf)





FFN with SwiGLU

Swish activation



$$swish(x) = x\sigma(\beta x)$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

FFN with ReLU

$$\mathrm{FFN}(x) = max(0, x \cdot W_1 + b_1) \cdot W_2 + b_2$$

dim=4d

FFN with SwiGLU

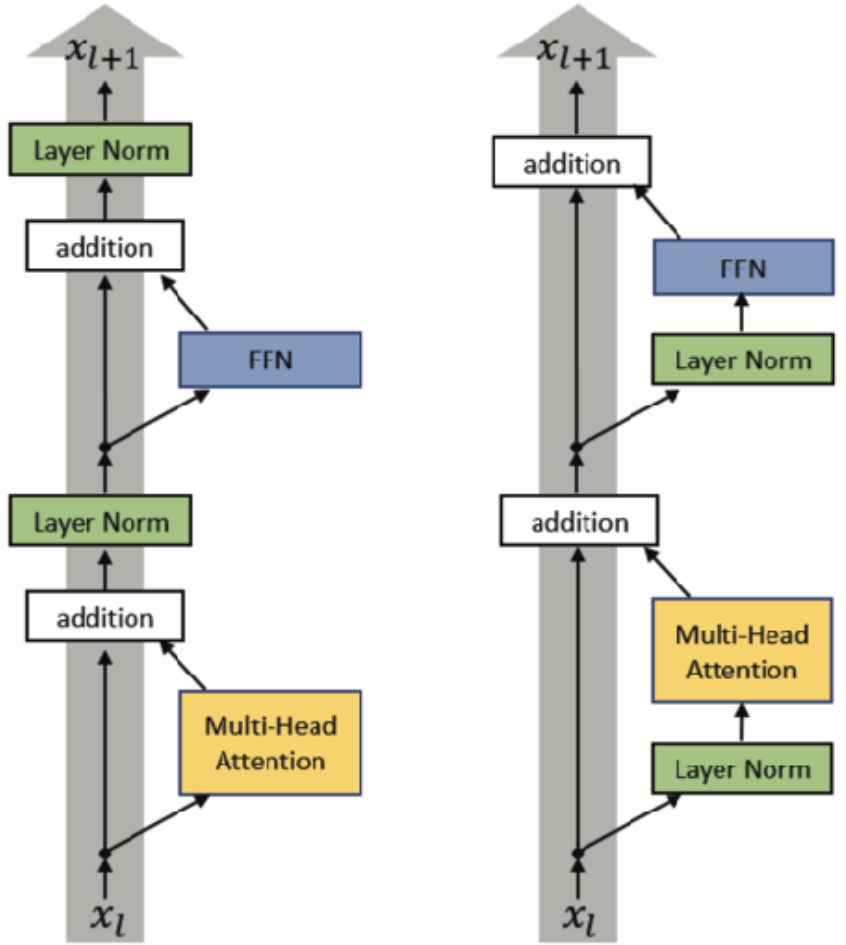
$$\mathrm{FFN}_{SwiGLU}(x) = \left(Swish(x \cdot W_1 + b_1) \odot (x \cdot W_2 + b_2)\right) \cdot W_3 + b_2$$

dim=2/3 * 4d

used in LLaMA and all later LLMs

Residual Connection and Layer Normalization

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



Relative Positional Embedding

- “I have a dream”
- “Today I have a dream”
- What are Attention weights for “I” and “dream”?
- Goal: make sure the similarity is only dependent on the distance between two embeddings, independent of their absolute positions

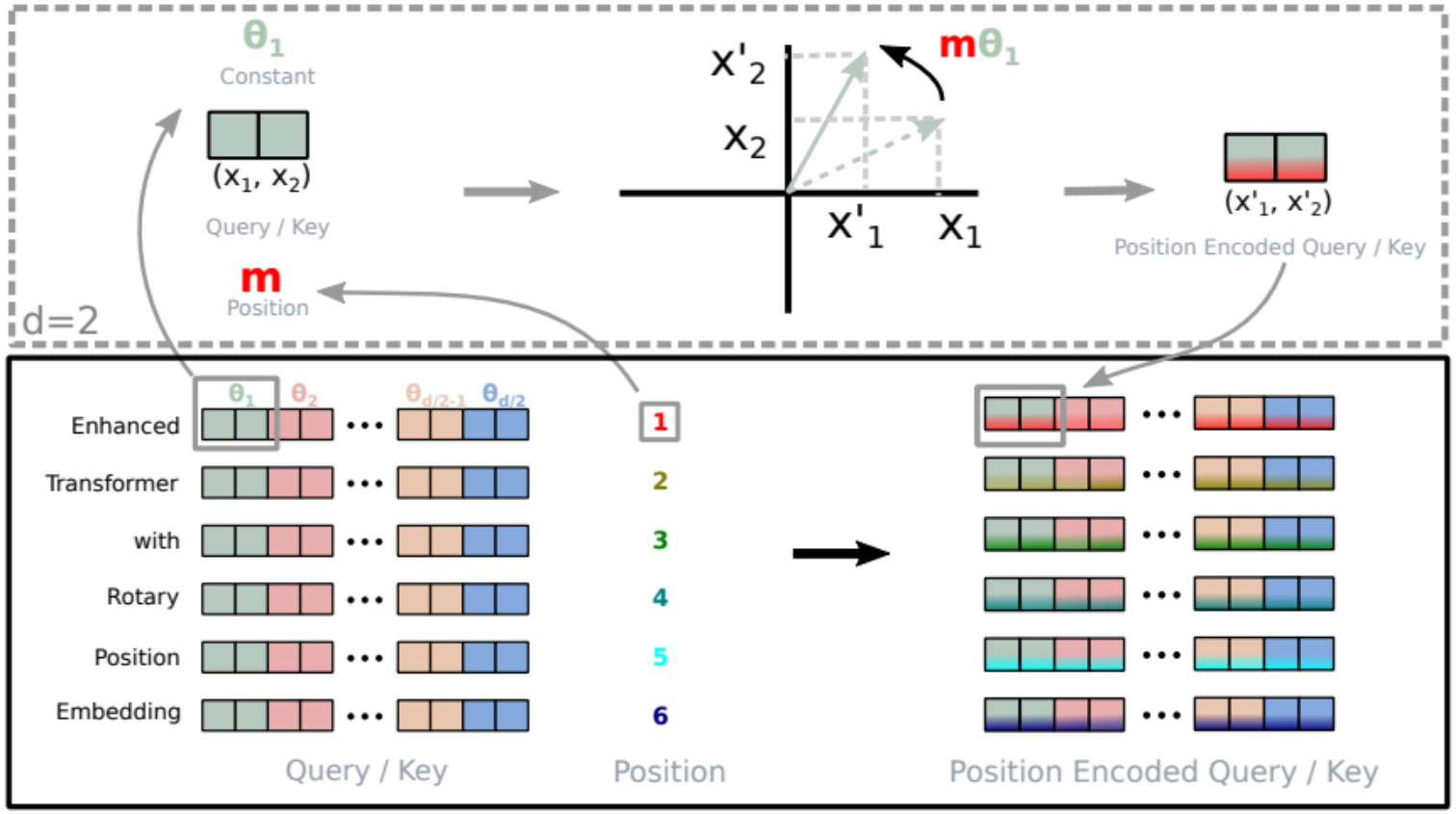
$$Att(x_i, x_j) = Att(x_{i+\Delta}, x_{j+\Delta})$$

Rotary Embedding (RoPE)

make the attention weight depending (only) on position distance

$$f(x_m, m) = \begin{pmatrix} \cos(m\theta) & -\sin(m\theta) \\ \sin(m\theta) & \cos(m\theta) \end{pmatrix} \begin{pmatrix} x_{m,1} \\ x_{m,2} \end{pmatrix}$$

$$f(x_m, m) \cdot f(x_n, n) = x_m^T R_{n-m} x_n$$



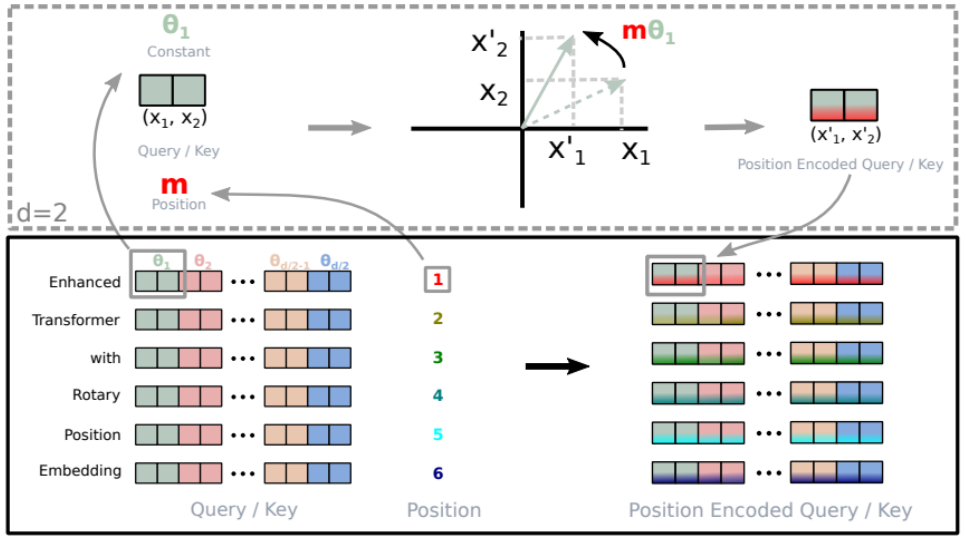
Su et al. RoFormer: Enhanced Transformer with Rotary Position Embedding 2021.

Rotary Embedding (RoPE)

$$f(x_m, m)$$

$$= \begin{pmatrix} \cos(m\theta_1) & -\sin(m\theta_1) & & \\ \sin(m\theta_1) & \cos(m\theta_1) & & \\ & & \cos(m\theta_2) & -\sin(m\theta_2) \\ & & \sin(m\theta_2) & \cos(m\theta_2) \end{pmatrix} \begin{pmatrix} x_{m,1} \\ x_{m,2} \\ x_{m,3} \\ x_{m,4} \end{pmatrix}$$

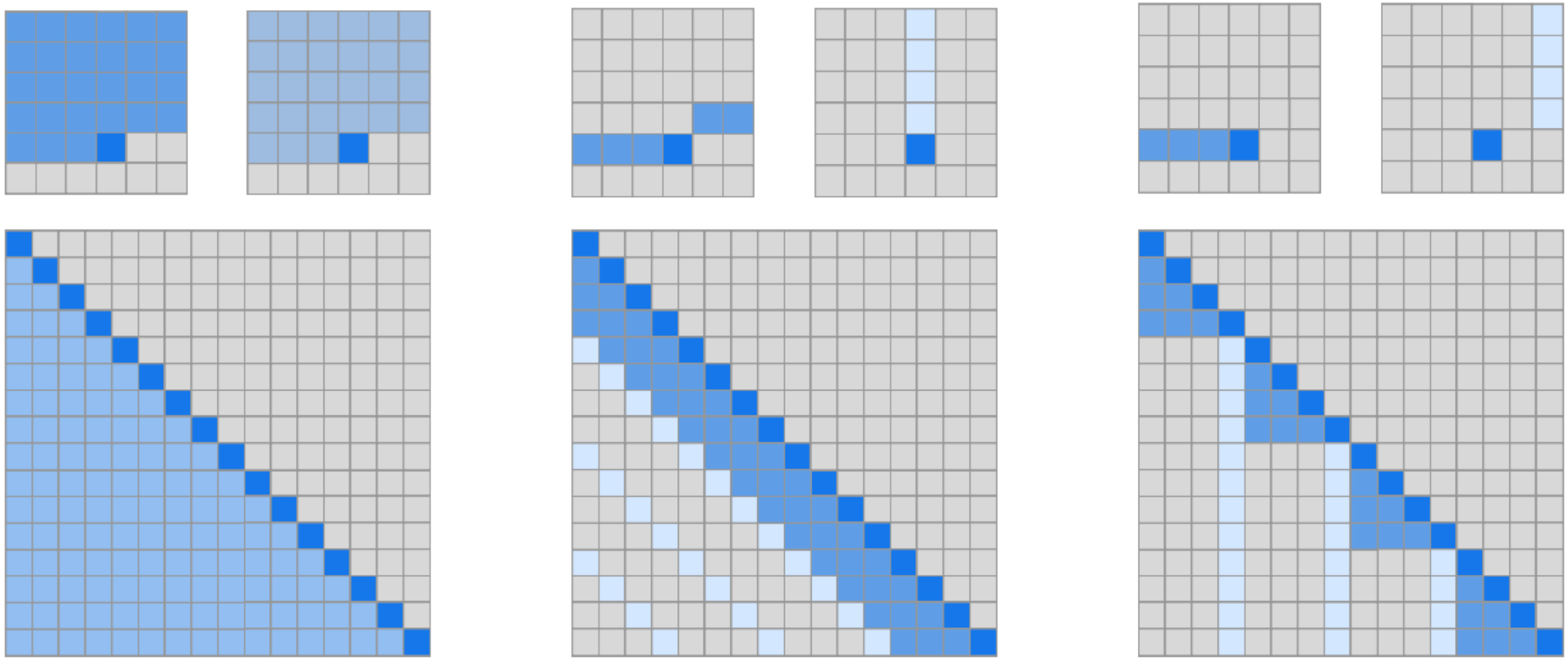
$$f(x_m, m)^T \cdot f(x_n, n) = x_m^T R_{n-m} x_n$$



Su et al. RoFormer: Enhanced Transformer with Rotary Position Embedding 2021.

Sparse Attention

- Alternating dense and locally banded **sparse attention** patterns (GPT3)



(a) Transformer (b) Sparse Transformer (strided) c) Sparse Transformer (fixed)

Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

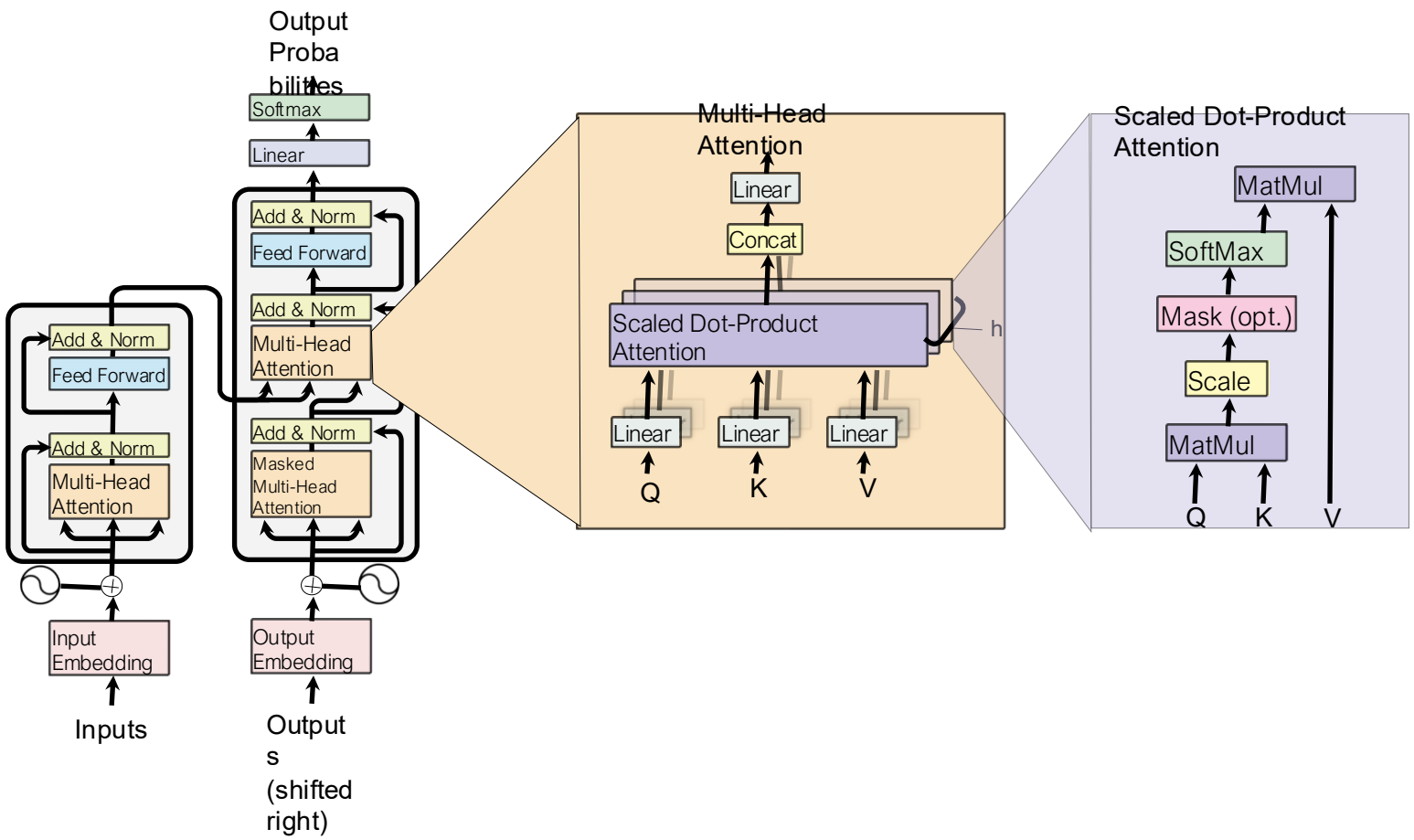
Today's Topic

- Encoder-decoder Architecture
- Transformer Model
  - Embedding
  - Multihead Attention, Decoder Self-Attention
  - FFN and SwiGLU
  - Layernorm
  - RoPE

- Training Techniques and Performance of Transformer
- Pretrained LLMs

Transformer in Original Paper

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



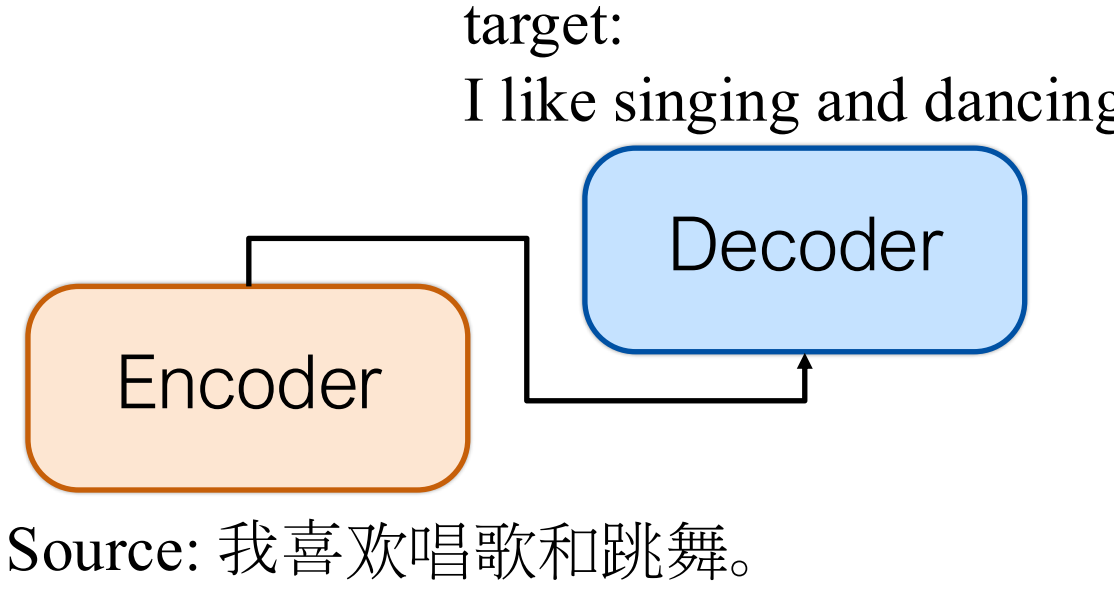
Training Transformer

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

- Training loss: Cross-Entropy

$$l = -\sum_n \sum_t \log f_\theta(x_n, y_{n,1}, \cdots, y_{n,t-1})$$

- 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 ~ 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
  - $\eta = \frac{1}{\sqrt{d}} min(\frac{1}{\sqrt{t}}, \frac{t}{\sqrt{t_0^3}})$

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)

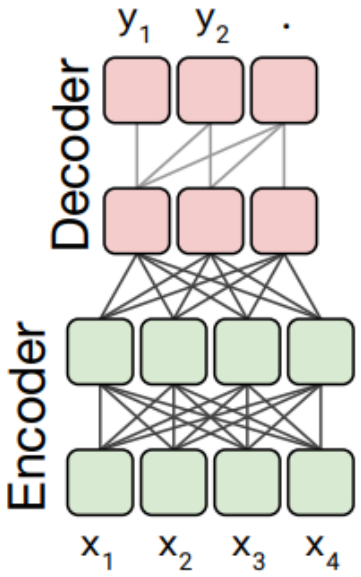
- decoding length: within source length + 50
  - more on decoding in next lecture

Today's Topic

- Encoder-decoder Architecture
- Transformer Model
  - Embedding
  - Multihead Attention, Decoder Self-Attention
  - FFN and SwiGLU
  - Layernorm
  - RoPE
- Training Techniques and Performance of Transformer
- Pretrained LLMs

T5

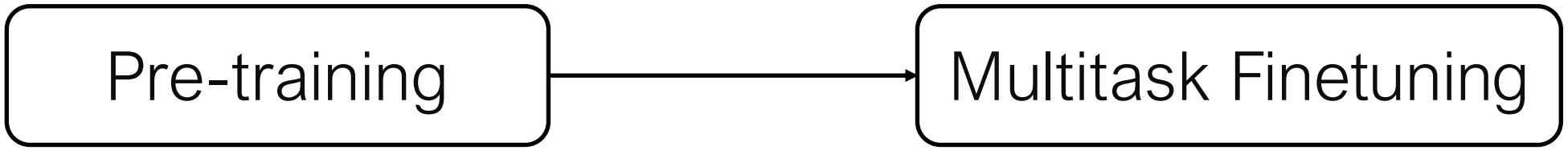
- Model Architecture
  - Standard encoder-decoder Transformer
  - Decoding: beam search
    - Beam width=4, length penalty=0.6



- Model Size
  - T5-base: 220 million parameters
    - 12 blocks, d_{ff} = 3072, d_{kv} = 64, 12-headed attention, d_{model} = 768
  - T5-3B
    - 24 blocks, d_{model} = 1024 , d_{kv} = 128, d_{ff} = 16384, 32-headed attention
  - T5-11B
    - 24 blocks, d_{model} = 1024 , d_{kv} = 128, d_{ff} = 65536, 128-headed attention

Raffel, Colin, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." *The Journal of Machine Learning Research* 21.1 (2020): 5485-5551.

Training

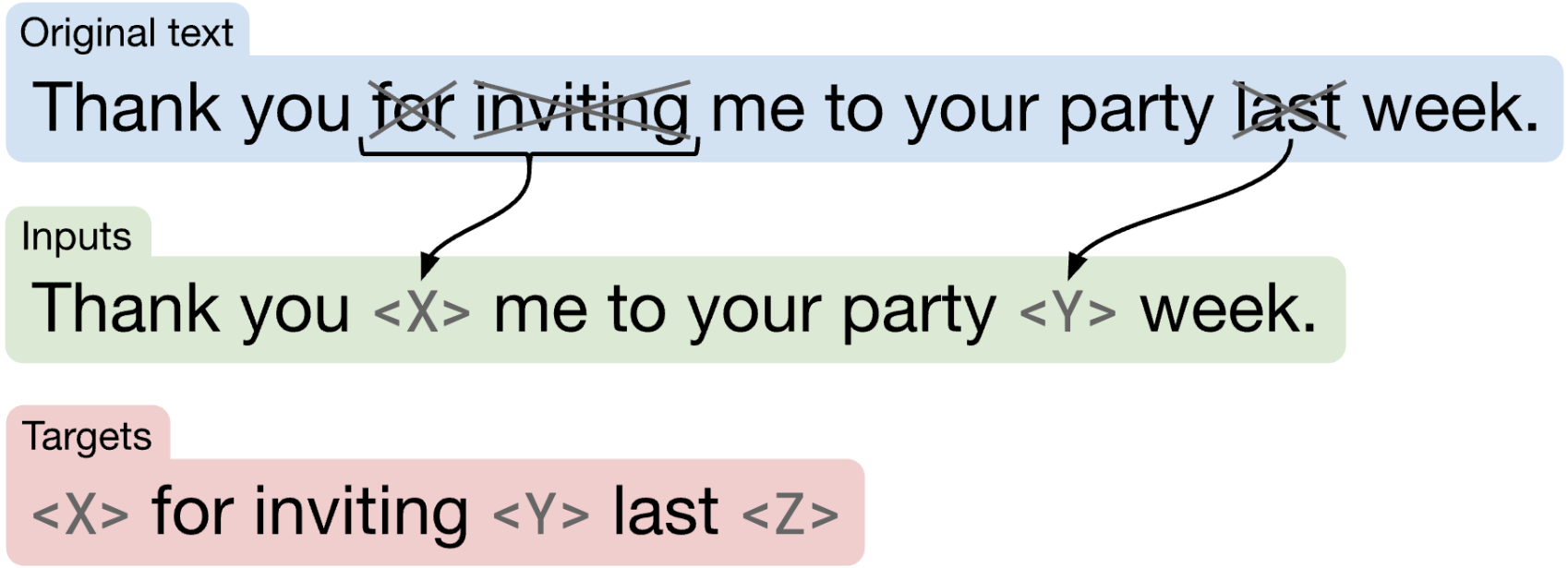


- Pretraining: C4: filtered English corpus from Common Crawl
  - 750GB, still used often today
- Supervised fine-tuning
  - Language understanding/Text classification: GLUE/SuperGLUE
  - Summarization: CNN/Daily mail corpus
  - Question answering: SQuAD
  - Translation: WMT English to German, French, Romanian

T5 Pre-training: Recover randomly corrupted spans

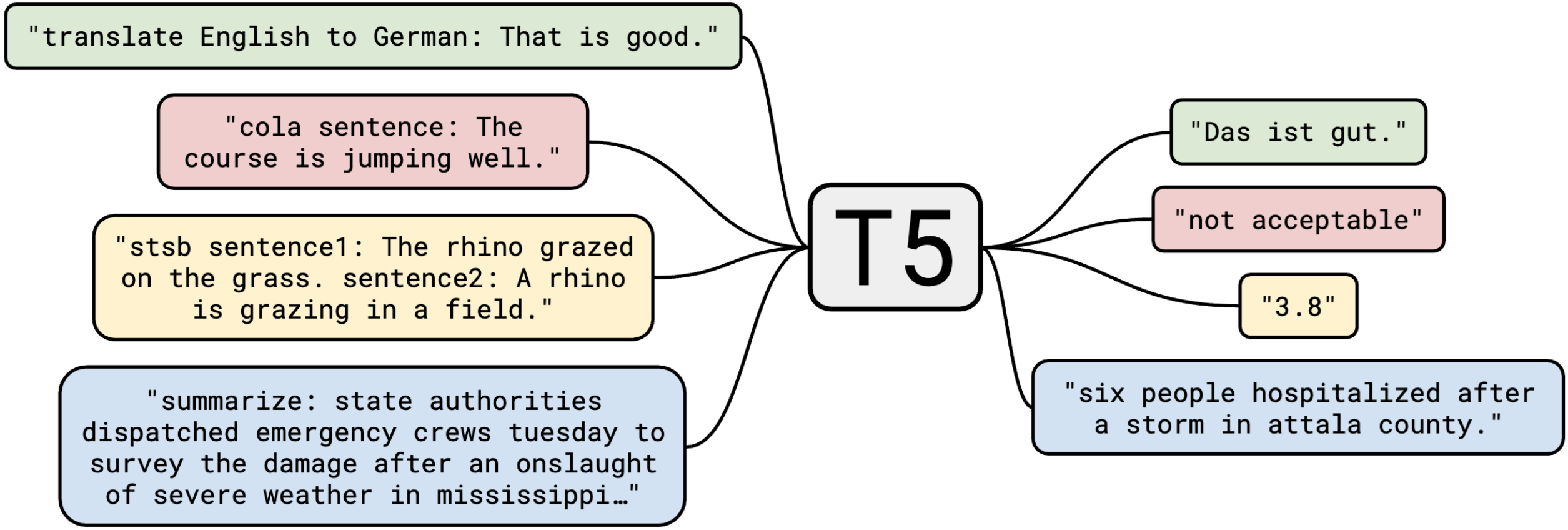
Standard next token cross entropy loss, plus … Cloze-style QA

15% of text are corrupted



Pre-train for 0.5 million steps on a batch size of 128 sequences of length 512. packing multiple seqs 65k tokens per batch, result in 34B trained tokens.

T5 Multitask SFT with Task Instruction



Unified format to put task instructions as natural language in the input, enables transfer to new tasks. with more instruction tuning ➔ T0, Flan-T5

LLaMA

- Model Architecture:
  - Based on Transformer decoder-only, with a few improvements.
  - Pre-normalization [GPT3]
  - SwiGLU activation function [PaLM]: Swish-Gated Linear Unit
  - Rotary Embeddings [RoFormer]

Touvron, Hugo, et al. "Llama: Open and efficient foundation language models." arXiv preprint arXiv:2302.13971 (2023).

LLaMA

- Model Size

params	dimension	n heads	n layers	learning rate	batch size	n tokens
6.7B	4096	32	32	$3.0e^{-4}$	4M	1.0T
13.0B	5120	40	40	$3.0e^{-4}$	4M	1.0T
32.5B	6656	52	60	$1.5e^{-4}$	4M	1.4T
65.2B	8192	64	80	$1.5e^{-4}$	4M	1.4T

Touvron, Hugo, et al. "Llama: Open and efficient foundation language models." arXiv preprint arXiv:2302.13971 (2023).

LLaMA

- Training Strategy
  - Trained with the standard language modeling loss function: the average log probability of all tokens without label smoothing
  - Auxiliary loss to encourage the softmax normalizer to be close to 0

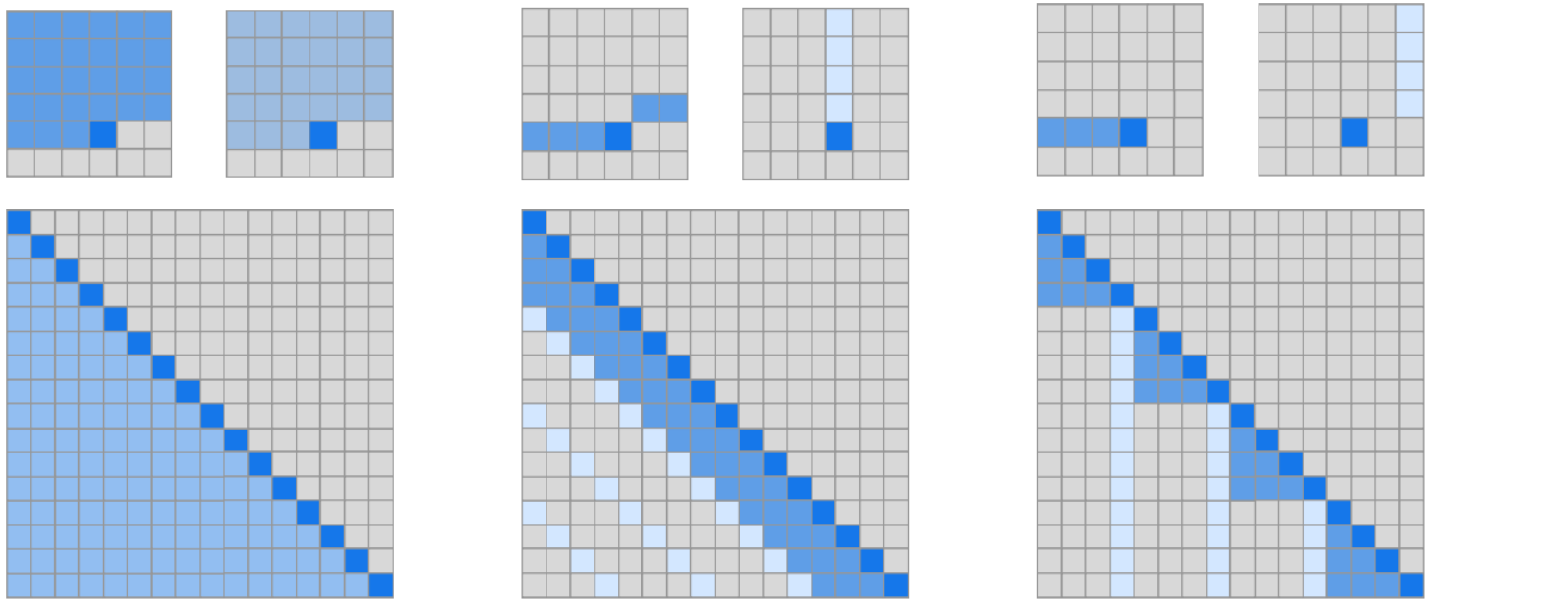
- Pre-training Details
  - Using only open-source data

Dataset	Sampling prop.	Epochs	Disk size
CommonCrawl	67.0%	1.10	3.3 TB
C4	15.0%	1.06	783 GB
Github	4.5%	0.64	328 GB
Wikipedia	4.5%	2.45	83 GB
Books	4.5%	2.23	85 GB
ArXiv	2.5%	1.06	92 GB
StackExchange	2.0%	1.03	78 GB

Touvron, Hugo, et al. "Llama: Open and efficient foundation language models." arXiv preprint arXiv:2302.13971 (2023).

GPT3

- Model Architecture
  - Based on the standard Transformer architecture
  - With modified initialization, pre-normalization, and reversible tokenization
  - Alternating dense and locally banded **sparse attention** patterns



(a) Transformer (b) Sparse Transformer (strided) c) Sparse Transformer (fixed)

Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

GPT3

- Model Size

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

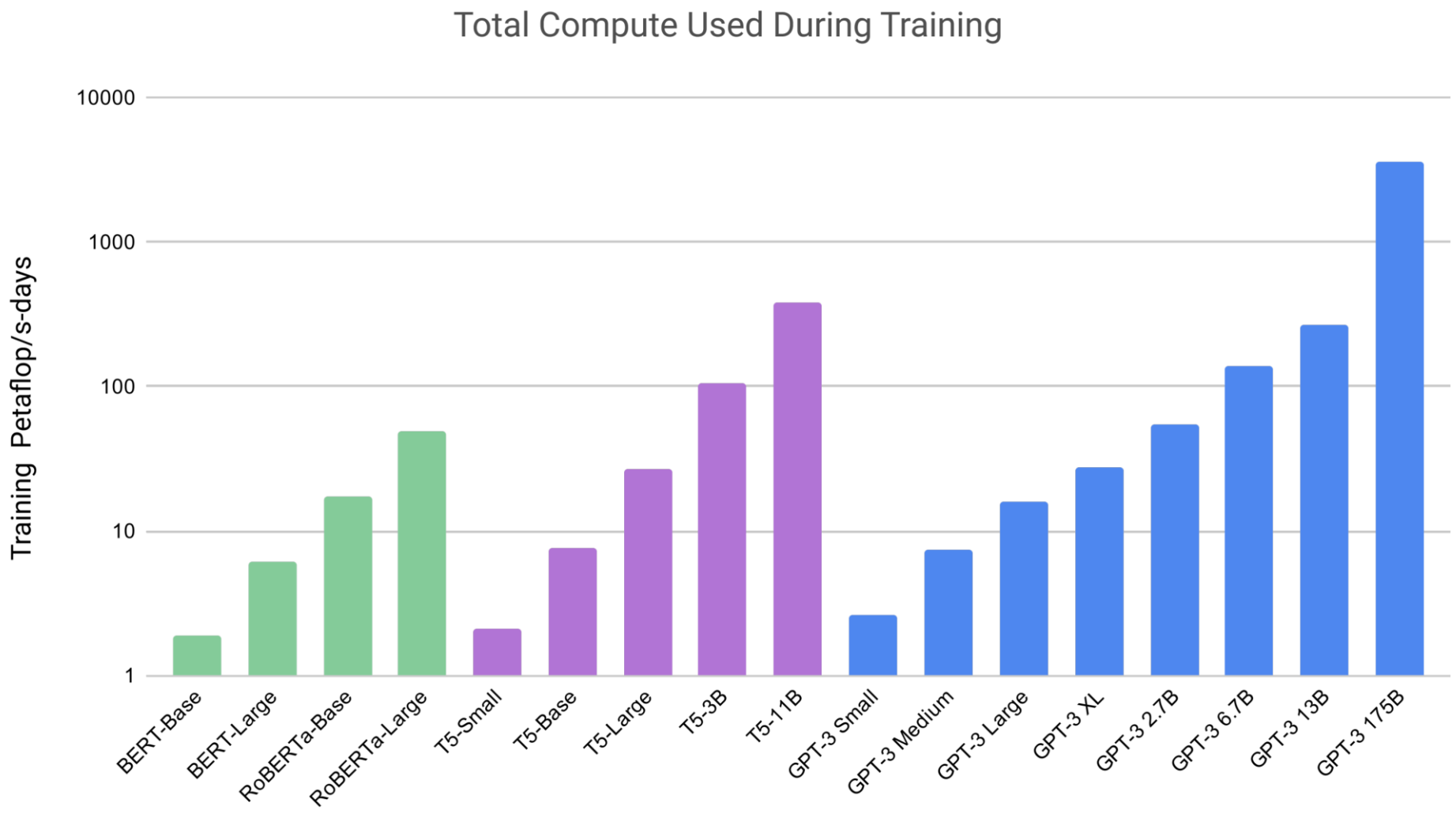
GPT3

- Training Strategy
  - Unsupervised Pre-training
- Training Details

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

Computation



Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

Summary

- Key components in Transformer and LLM Architecture
  - Positional Embedding (to distinguish tokens at different pos)
  - Multihead attention
  - Residual connection
  - layer norm
  - FFN and SwishGLU
  - RoPE

Summary

- Training Technique:
  - Dropout
  - Label smoothing
  - optimization alg
- Pretraining: Mask-labeled recovery of random spans + next token prediction
- Multitask supervised fine-tuning with instruction templates
  - T5, InsturctGPT

Code Go-through

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

Quiz

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