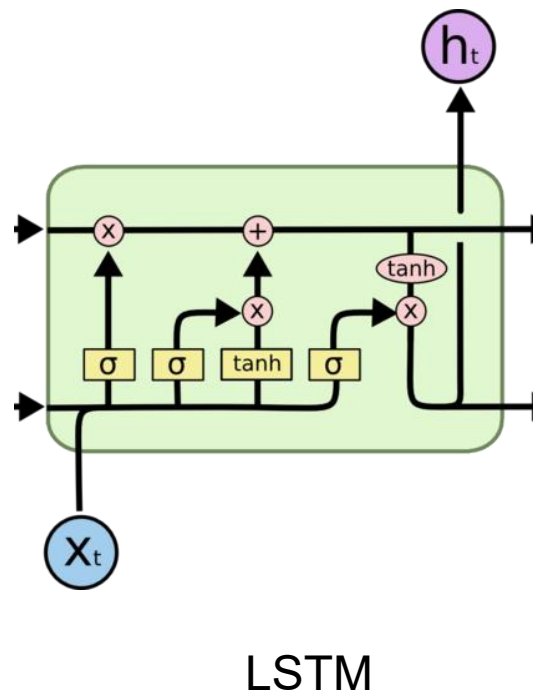
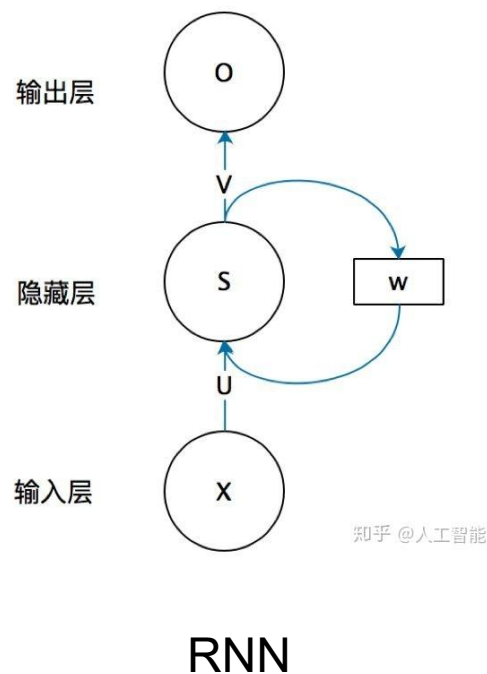


Transformer

Reporter: Bowen Xu

Background

Traditional translation model

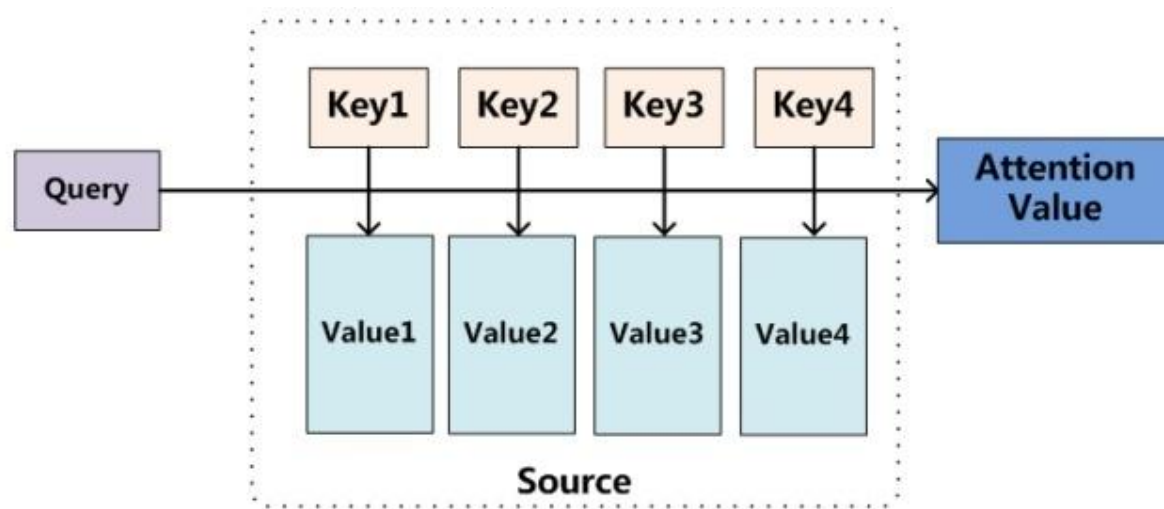


Drawbacks:

- Sequential nature precludes **parallelization**
- **Information loss** in long term

Introduction

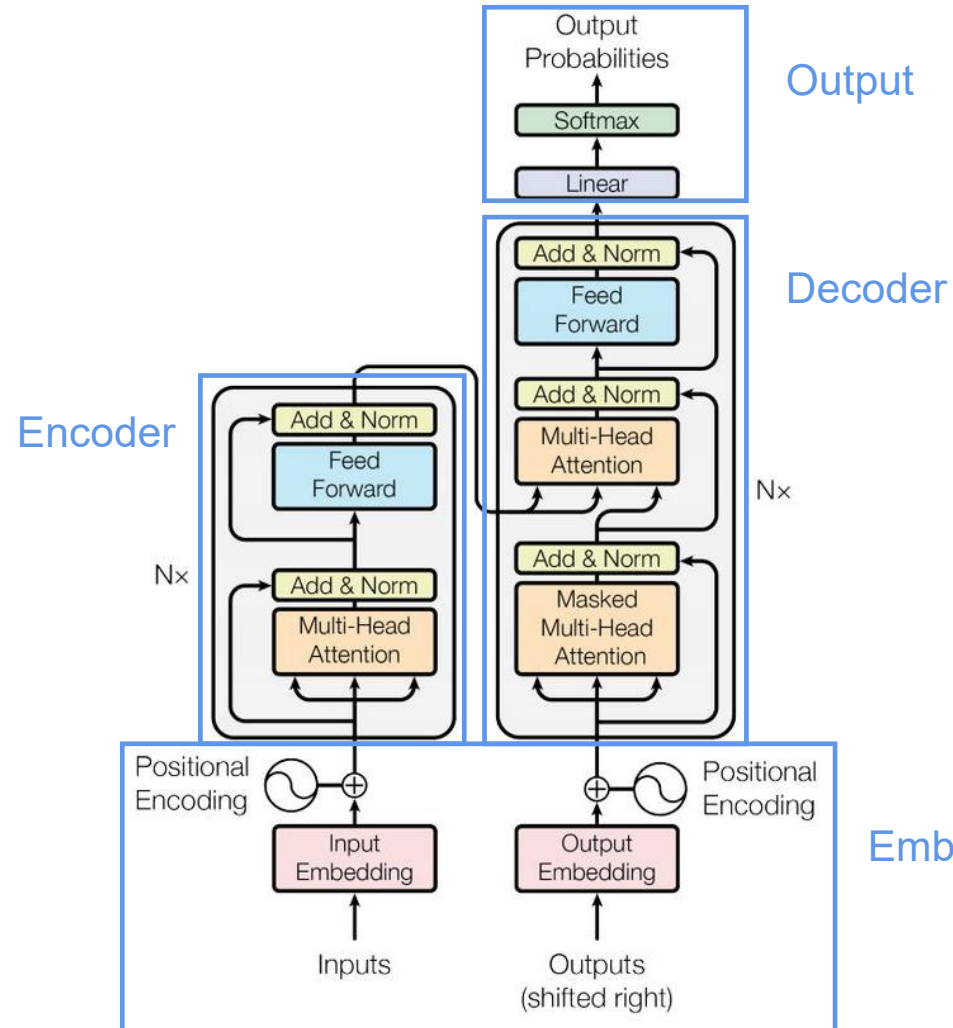
Based entirely on Attention Mechanism



Advantages:

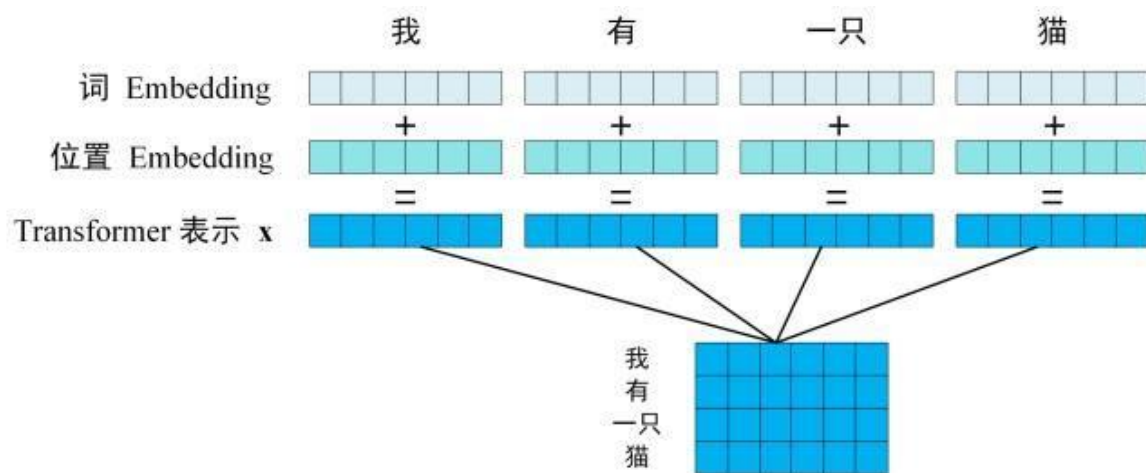
- Allowing for **parallelization**
- Allowing modeling of dependencies without regard to distance

Model Architecture



- Embedding input
- **Encoder**
- **Decoder**
- Output

Embedding Input



Why apply Position Embedding:

- Self-attention cannot get positional information

- Word Embedding: word2vec, GloVe, one-hot, etc.
- Positional Embedding:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

$$PE(pos, 2i + 1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

- d_{model} : dimension of input
- pos: position of words

Positional Embedding

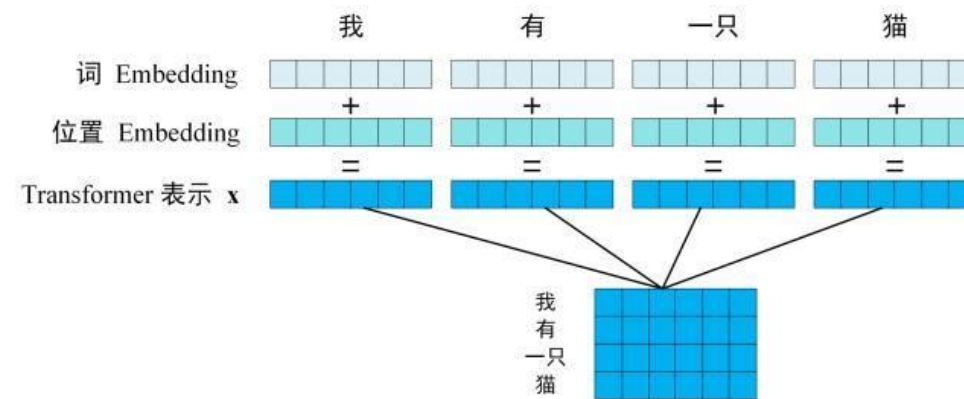
Why positional embedding:

- “Tom chases Jerry” \neq “Jerry chases Tom”
- Relative positional information:

$$\begin{cases} \sin(\alpha + \beta) = \sin\alpha\cos\beta + \sin\beta\cos\alpha \\ \cos(\alpha + \beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta \end{cases}$$

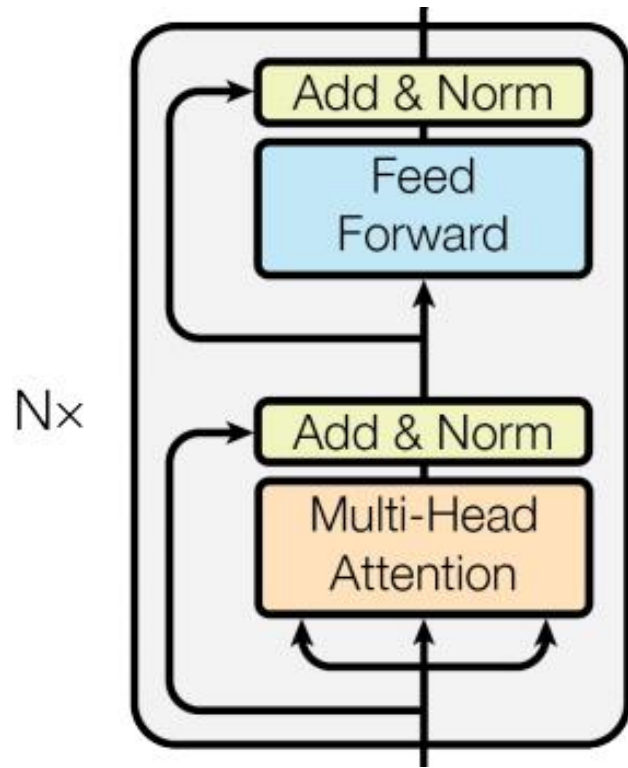
$$\begin{cases} PE(pos + k, 2i) = PE(pos, 2i)PE(k, 2i + 1) + PE(pos, 2i + 1)PE(k, 2i) \\ PE(pos + k, 2i + 1) = PE(pos, 2i + 1)PE(k, 2i + 1) - PE(pos, 2i)PE(k, 2i) \end{cases}$$

- Add word and positional embedding rather than concat
 - Reduce the vector dimension to facilitate training



Encoder

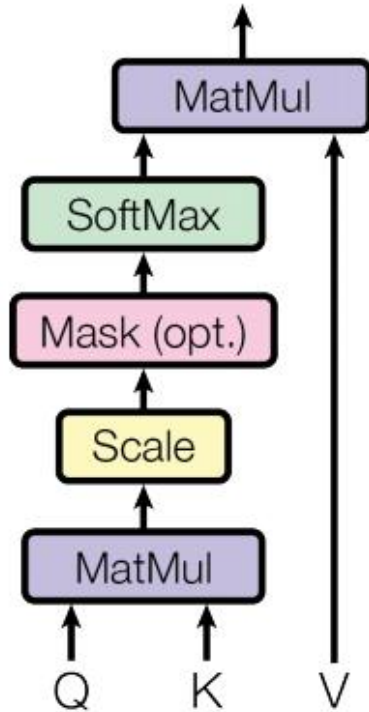
Encoder has $N=6$ identical layers



Structure of one layer:

- **Multi-Head Attention**
- Feed Forward
- Add & Norm

Self-Attention Mechanism



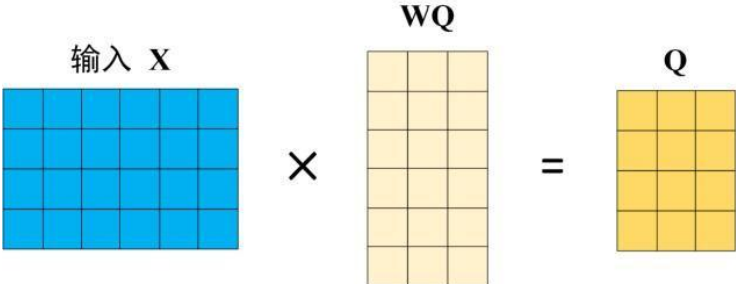
Self-Attention(without mask):

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

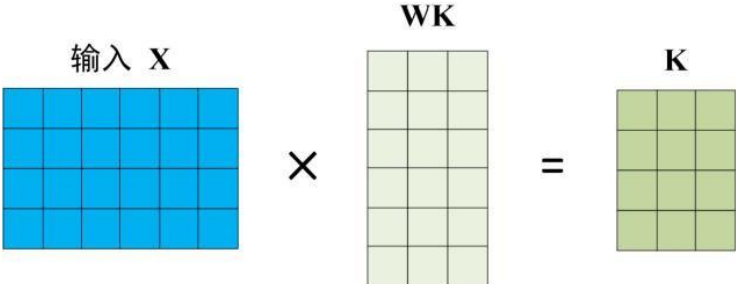
- Q: query (to match others)
- K: key (to be matched)
- V: value (information to be extracted)
- d_k : Number of columns (dimension) of Q and K

Self-Attention Input

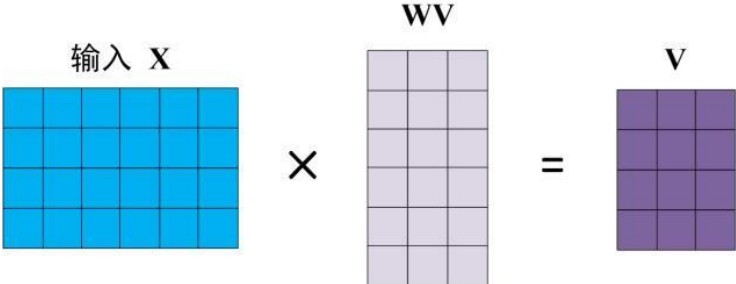
输入 X


$$\text{输入 X} \times W^Q = Q$$

输入 X


$$\text{输入 X} \times W^K = K$$

输入 X

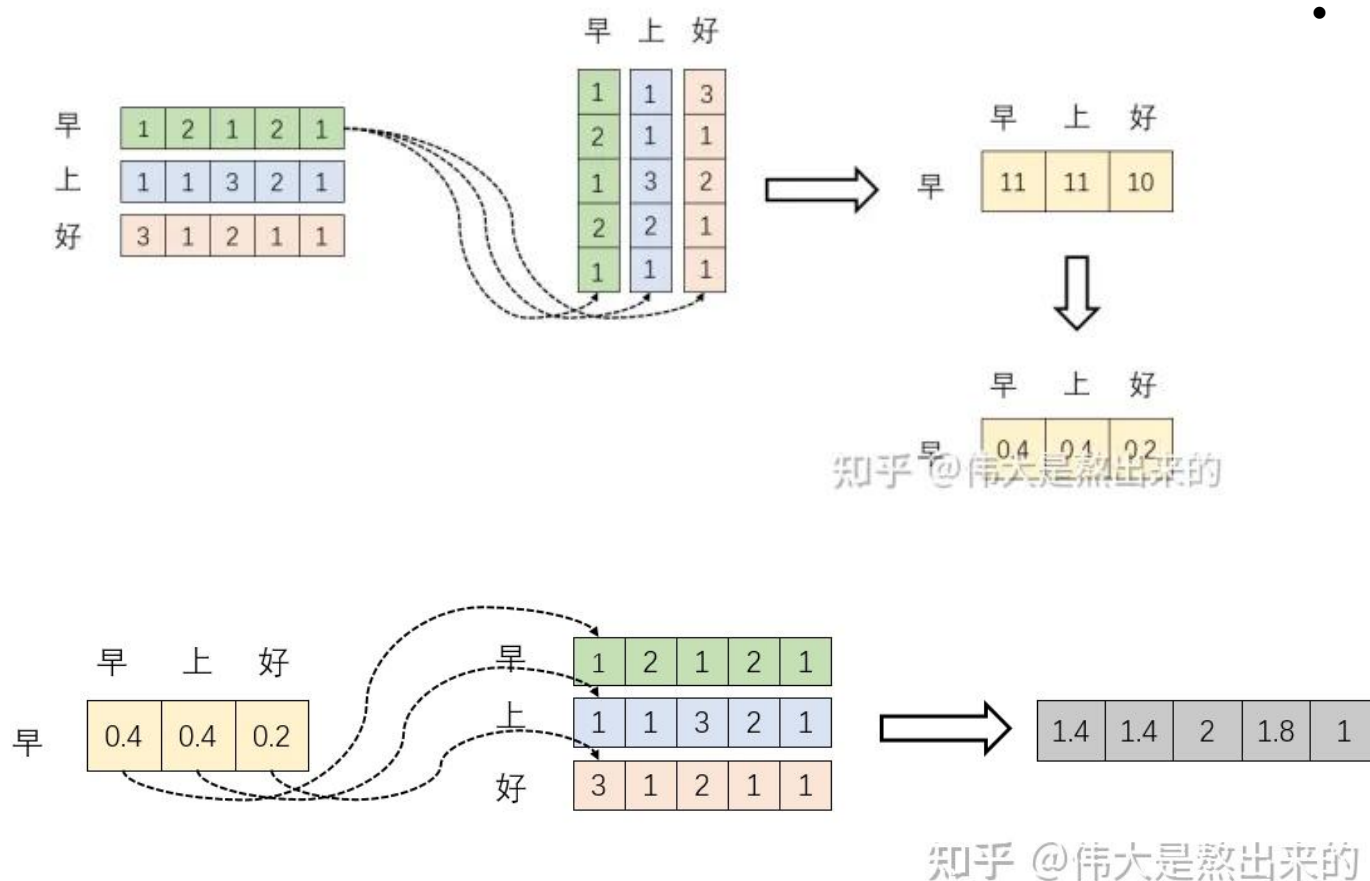

$$\text{输入 X} \times W^V = V$$

- X: Input of each layer of encoder
- W^Q, W^K, W^V : Linear transformation matrix

Self-Attention Mechanism

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

- Q, K, V: Linear transformation of input X.



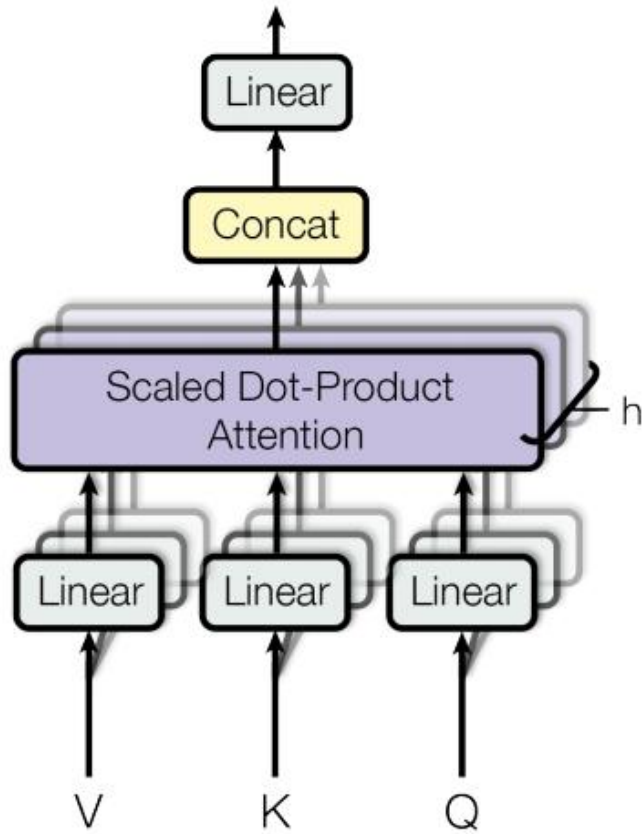
$$QK^T$$

$$softmax(QK^T)$$

$$softmax(QK^T)V$$

$\sqrt{d_k}$: Normlization

Multi-Head Attention

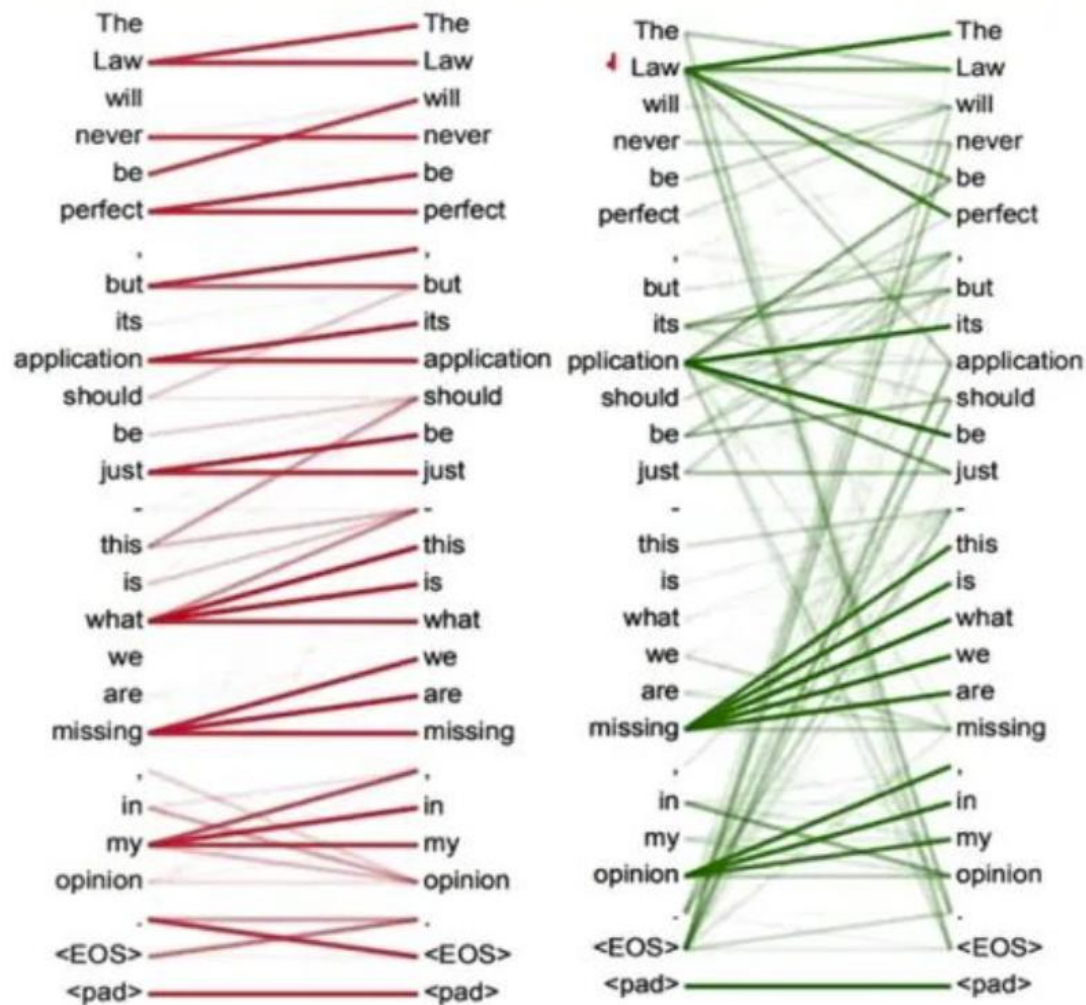


$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) W^O$$

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

Multi-Head attention ensures the **parallelism**

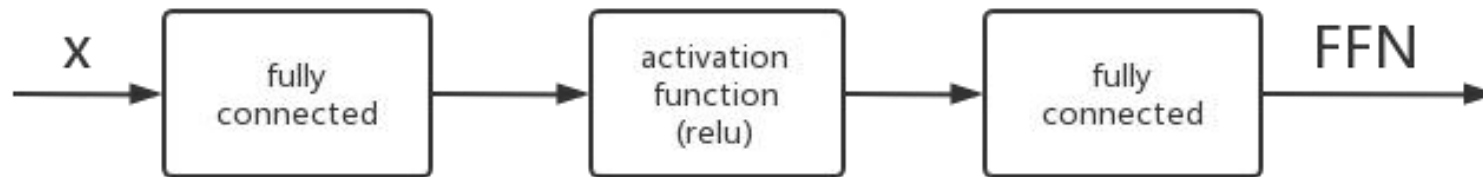
Multi-Head Attention



Results of multi-head attention:

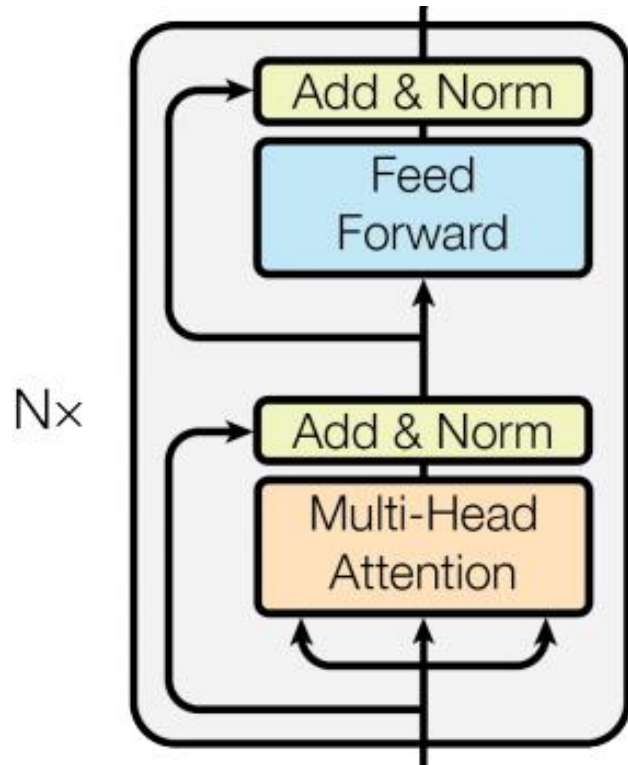
- Different attention results
- The obtained feature vectors have different expressions

Feed Forward



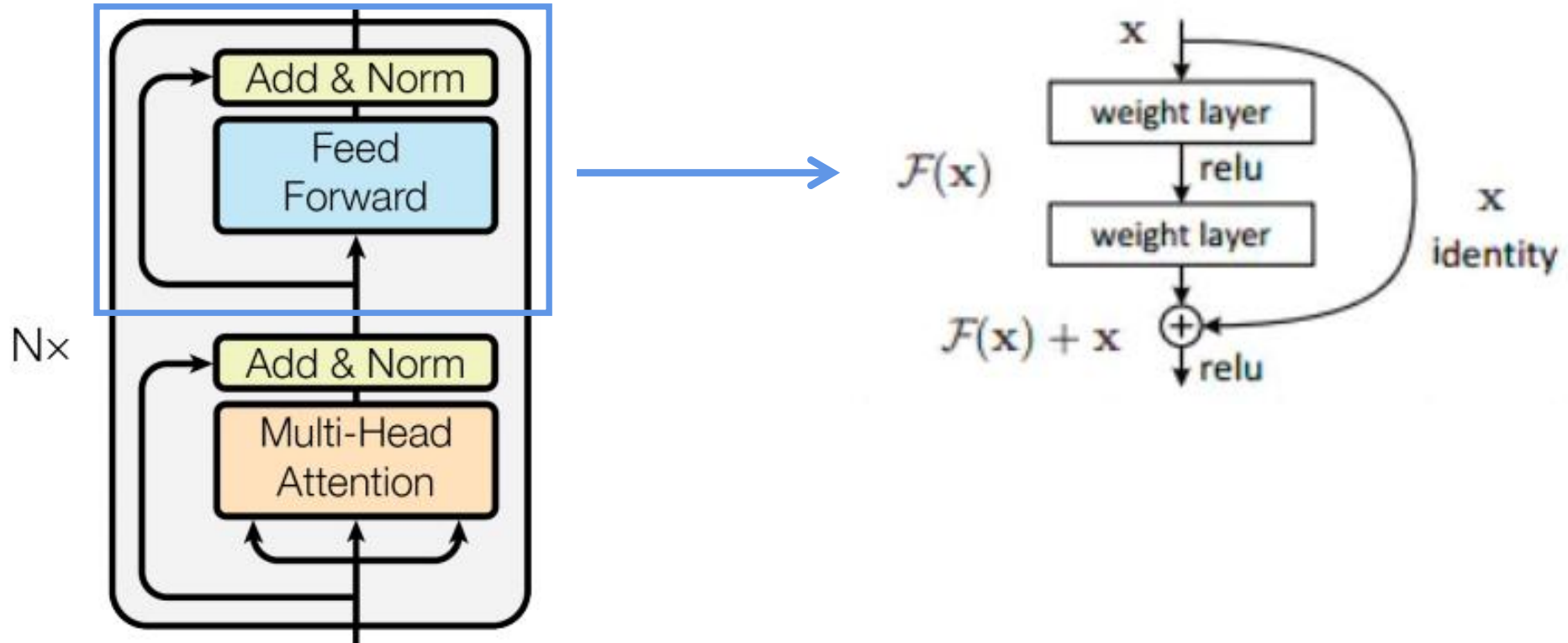
$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Add & Norm

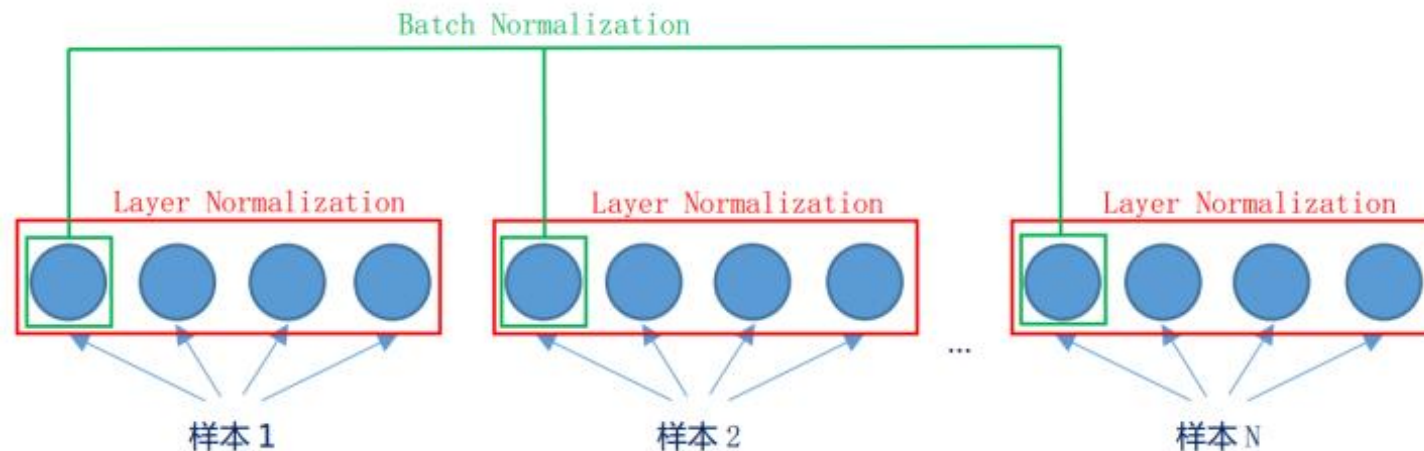


- Add: Residual connection
- Norm: Layer normalization

Residual Connection



Layer Normalization



- **Layer norm:** Normalization **between different dimensions** in the same sample
- Batch norm: Normalization between different samples in the same dimension

Layer Normalization

Why Layer normalization works:

- Forward normalization brings **distribution stability**
- **Stabilize** (μ, σ) **gradient** of back propagation
- Remove some parameters(bias and gain) to **avoid over-fitting**



Why Layer norm outperform Batch norm:

- In NLP tasks, the mean and variance always oscillate.
- The distribution of the same position of different sentences is different.

Encoder Conclusion

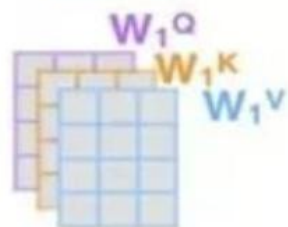
1) 这是我们的输入句子*

Thinking
Machines

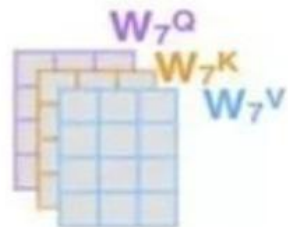
2) 编码每一个单词



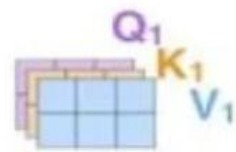
3) 将其分为8个头。将矩阵X或R乘以各个权重矩阵



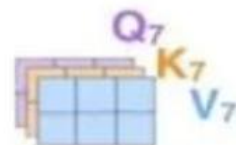
...



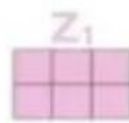
4) 通过输出的查询/键/值 (Q/K/V) 矩阵计算注意力



...



5) 将所有注意力头拼接起来，乘以权重矩阵 W^O



...

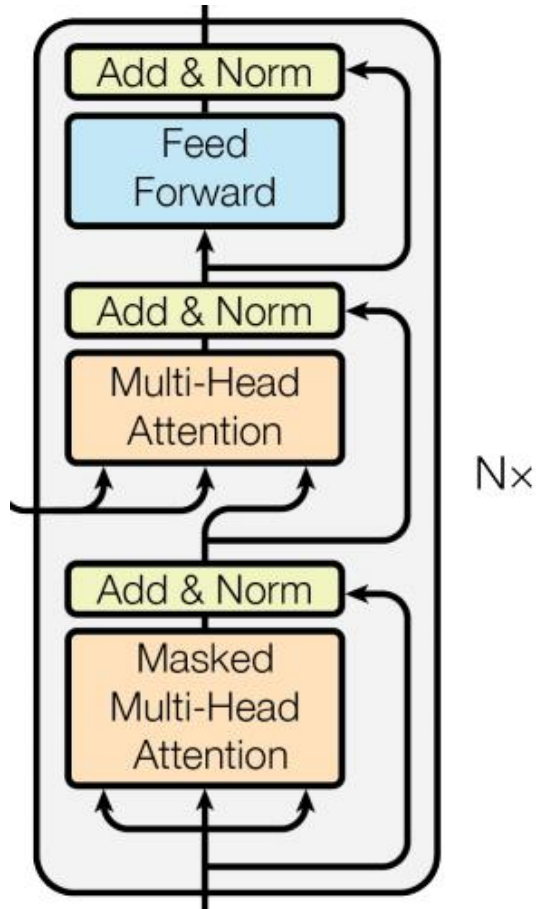


*注：除了第0个编码器，其他编码器都不需要进行词嵌入。它可以直接讲前面一层编码器的输出作为输入（矩阵R）。



Decoder

Decoder has $N=6$ identical layers



Structure of one layer:

- Masked Multi-Head Attention
- Multi-Head Attention
- Feed Forward
- Add & Norm

Mask Operation

0						
1						
2						
3						
4						

输入矩阵 X



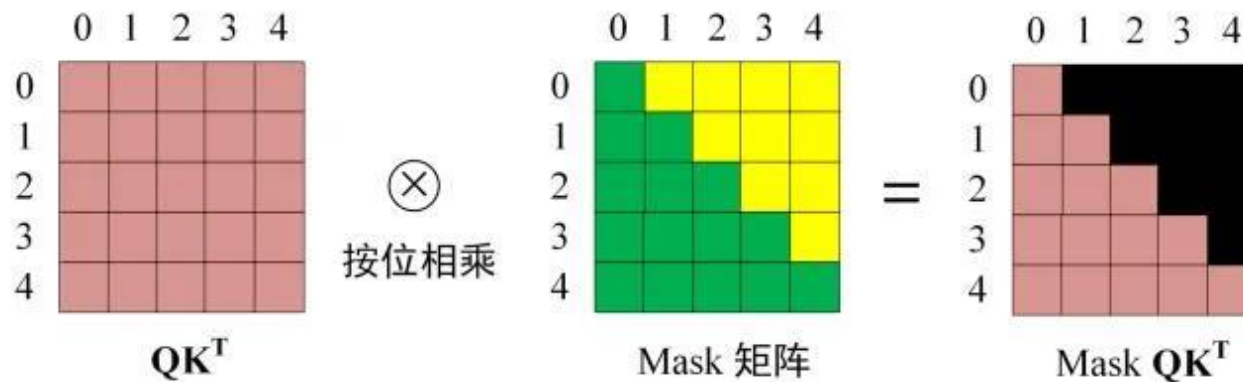
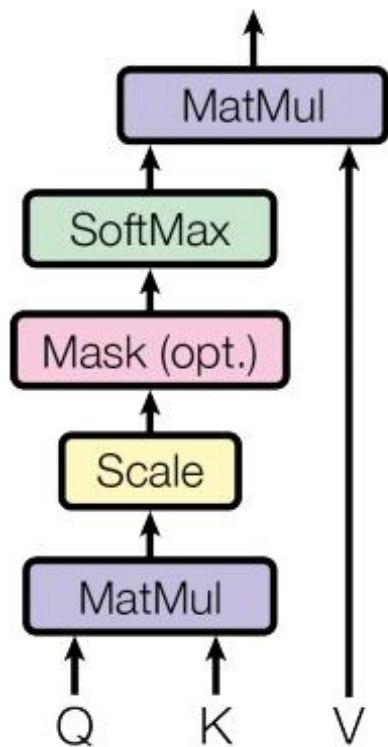
	0	1	2	3	4
0	不遮挡	遮挡	遮挡	遮挡	遮挡
1	不遮挡	不遮挡	遮挡	遮挡	遮挡
2	不遮挡	不遮挡	不遮挡	遮挡	遮挡
3	不遮挡	不遮挡	不遮挡	不遮挡	遮挡
4	不遮挡	不遮挡	不遮挡	不遮挡	不遮挡

Mask 矩阵

Sequence Mask:

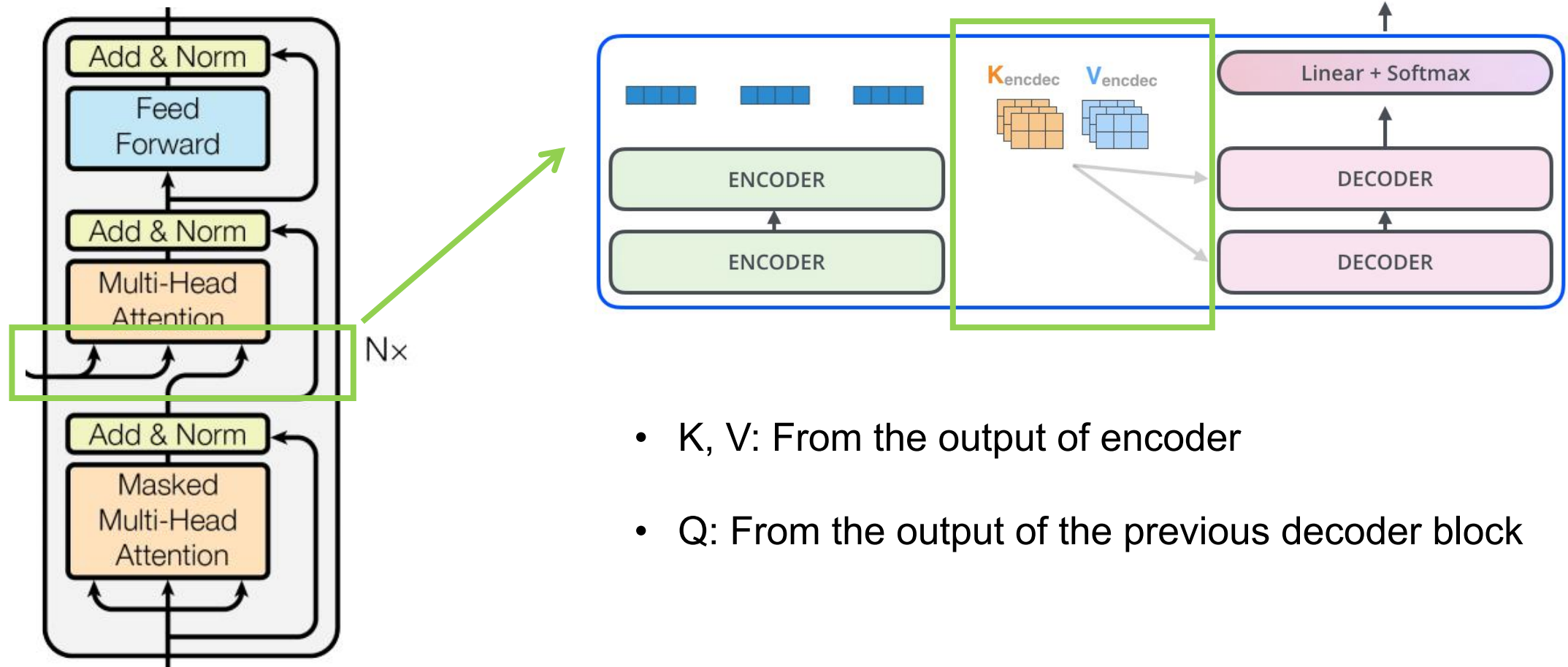
- Why: The decoder cannot see the future information.
- How: Setting masked position to $-\infty$.

Masked Multi-Head Attention

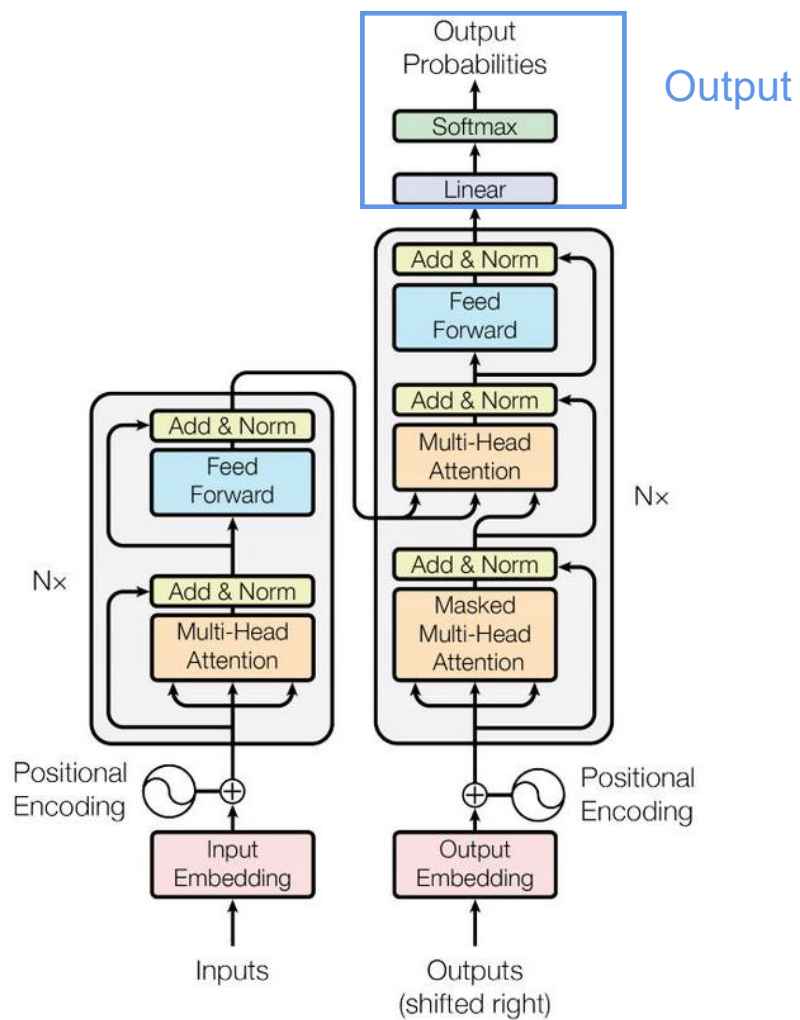


- Mask matrix is a lower triangular matrix (non-0 elements are all 1).

Multi-Head Attention In Decoder



Output



Structure of output block:

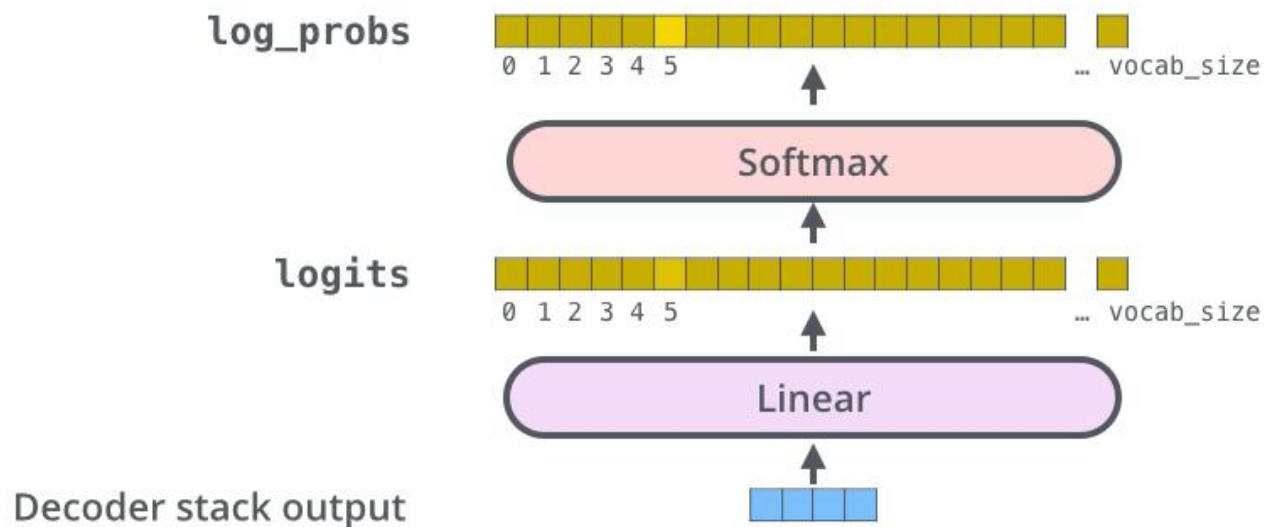
- Linear layer
- Softmax

Output

- How to predict a word:

Which word in our vocabulary
is associated with this index?

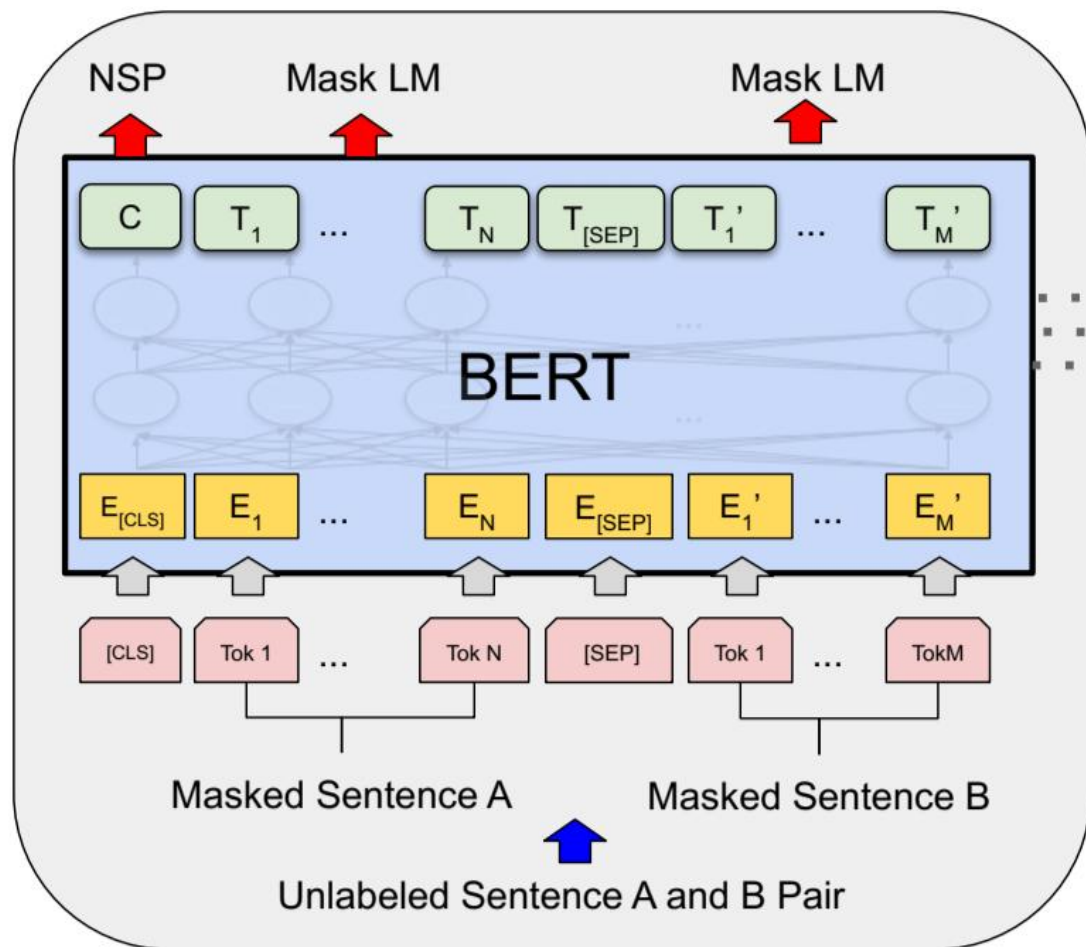
Get the index of the cell
with the highest value
(**argmax**)



Comparison With Common Neural Networks

- FNN: Consists of an input layer, one or more hidden layers, and an output layer. Not well suited for **sequential data** like text.
- CNN: Primarily used for **image and signal processing** tasks. Good at identifying patterns in images and not well suited for **sequential data** like text.
- RNN: Well suited for handling sequential data like text. But less effective for **longer sequences** than Transformer.
- LSTM: Well suited for handling sequential data like text, speech and time-series data. Able to handle longer sequences than vanilla RNNs. But less effective for **longer sequences** than Transformer.

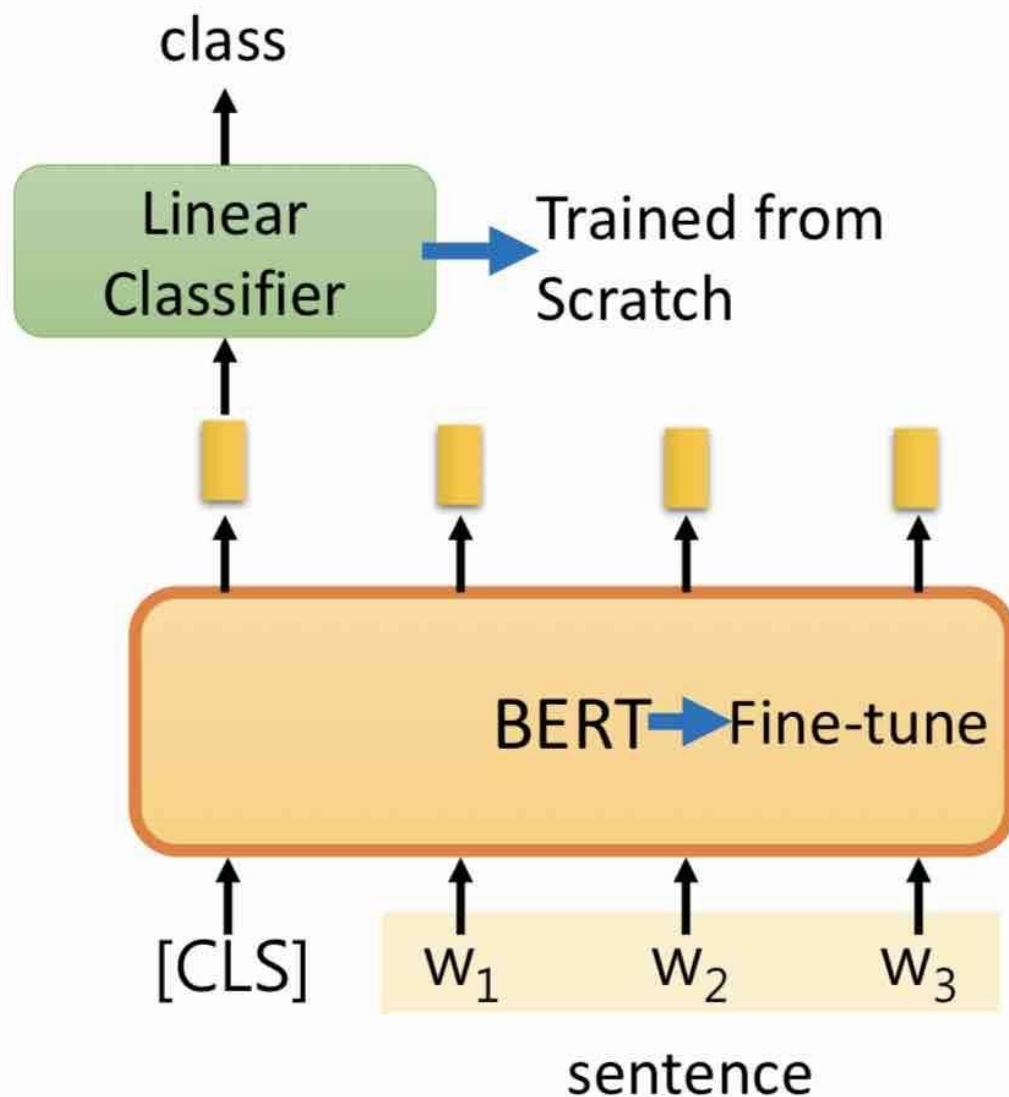
Applications: BERT



BERT (Bidirectional Encoder Representation from Transformer)

- A language representation model
- Main applications:
 - Sentence Classification tasks
 - Extracted-based Question Answering
 - Natural Language Inference
 -

Sentence Classification Tasks



Input: single sentence,
output: class

Example:
Sentiment analysis (our
HW),
Document Classification

Extracted-based Question Answering

阅读理解题，输入是文章和问题，
输出是理解的答案位置。

文章: $D = \{d_1, d_2, \dots, d_N\}$

问题: $Q = \{q_1, q_2, \dots, q_N\}$



结果: $A = \{q_s, \dots, q_e\}$

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?

gravity

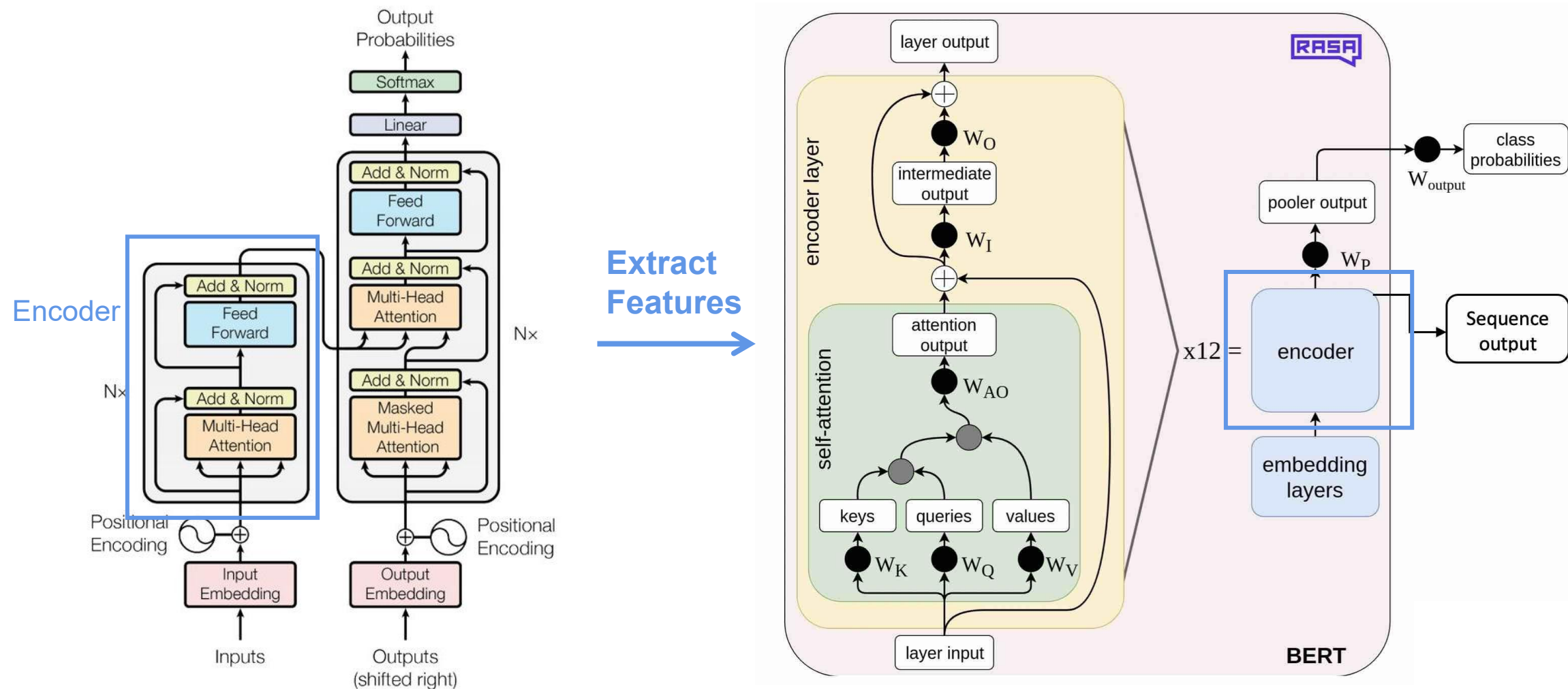
What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?

graupel

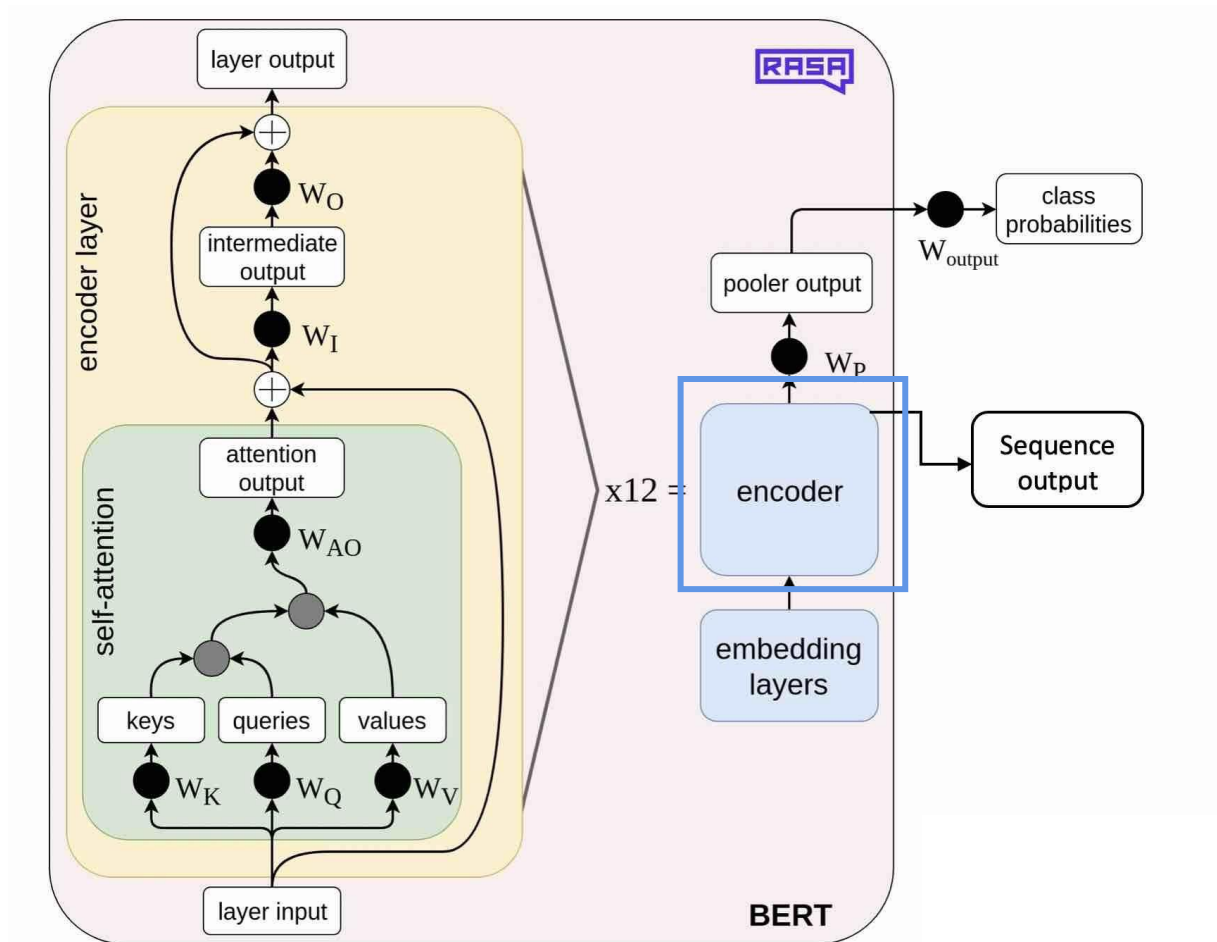
Where do water droplets collide with ice crystals to form precipitation?

within a cloud

Transformer In BERT



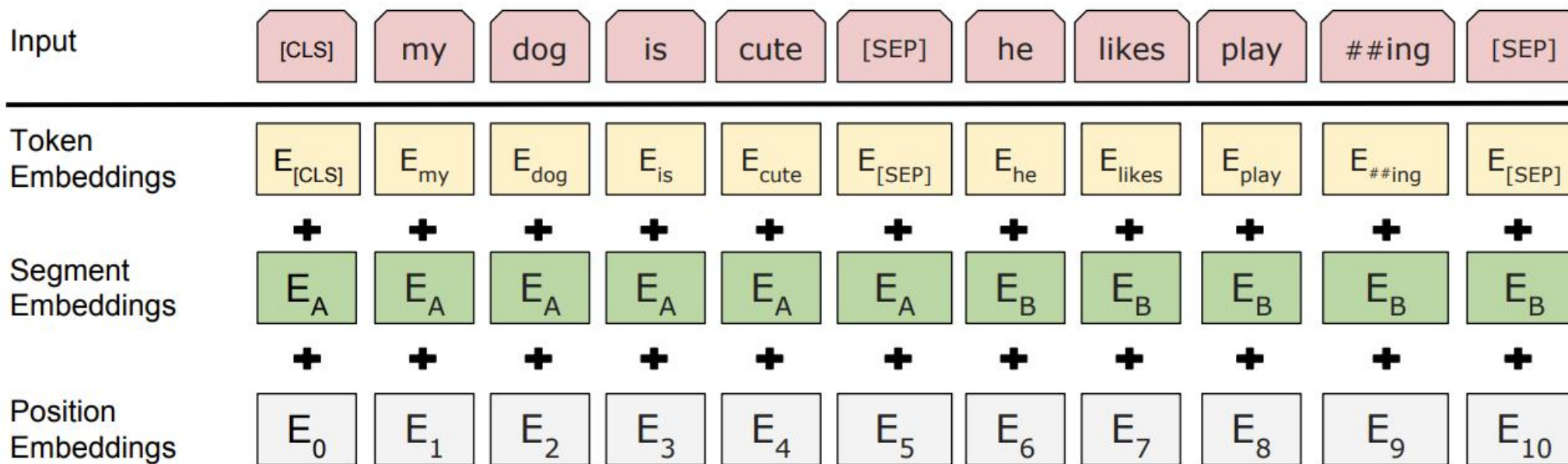
Transformer In BERT



The role of Encoder in BERT:

- Extract the features of texts
- Enable BERT training does not require labeled texts.

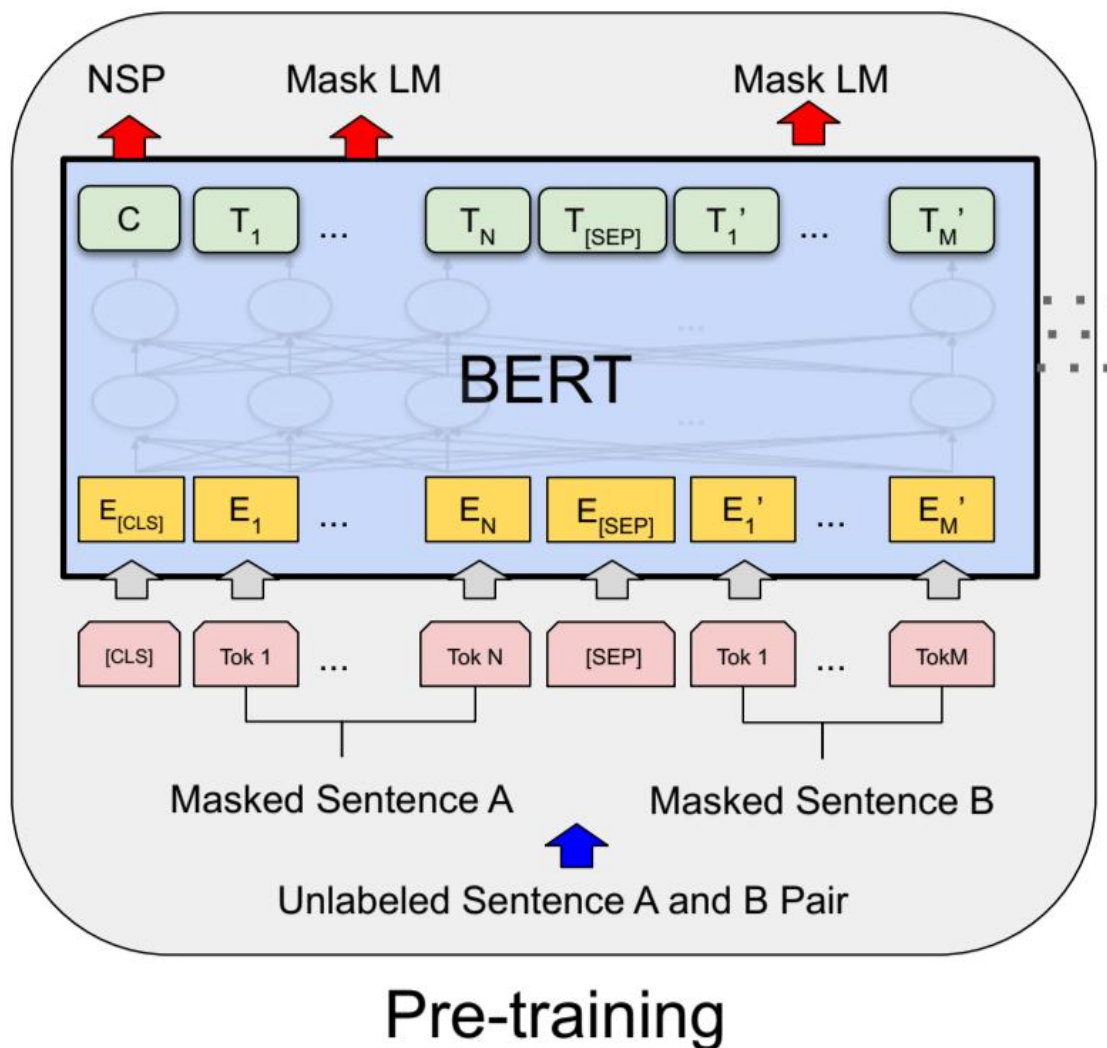
Model Input



- Token Embeddings: Convert each word into a fixed dimension vector.
- Segment Embeddings: Distinguish two sentences in a sentence pair.
- Position Embeddings: Mark the position of each word in the sentence.

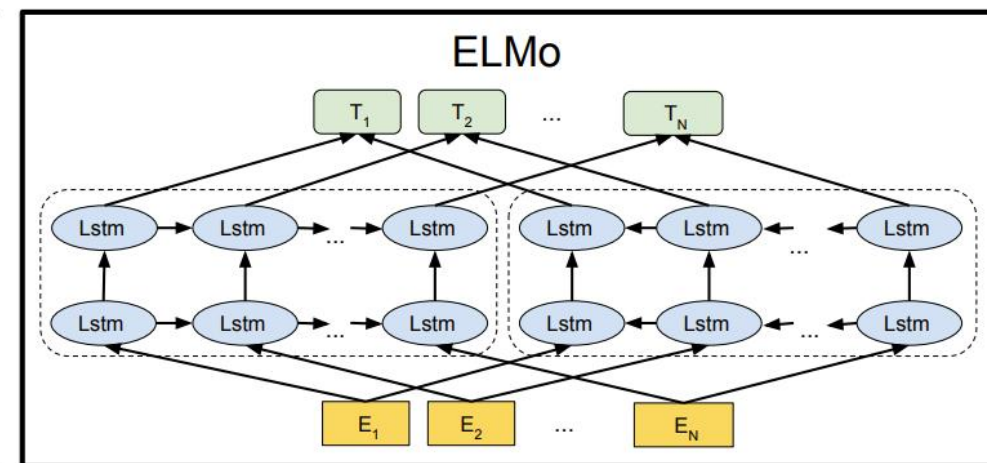
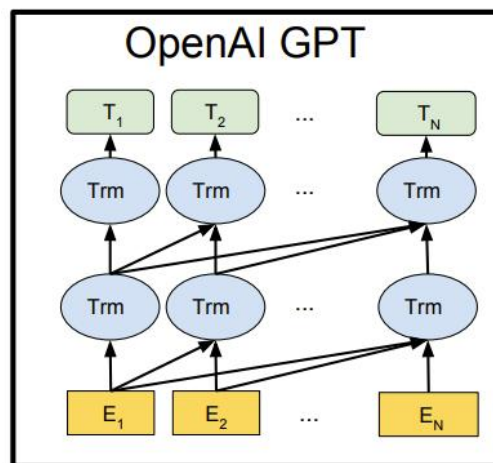
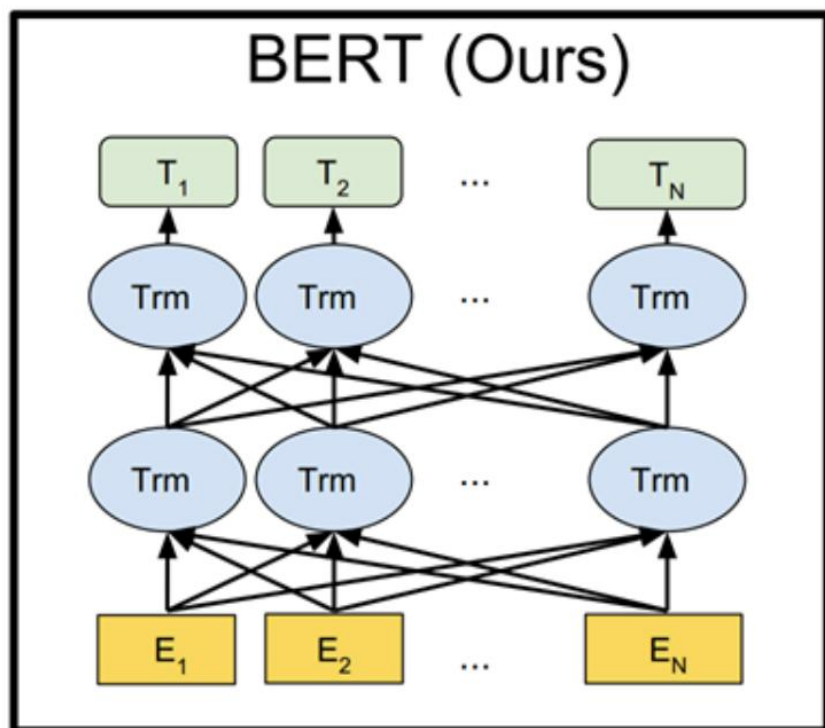
In Transformer

Pre-Training Tasks



- Masked Language Model (MLM):
 - Predict the masked English words.
- Next Sentence Prediction (NSP):
 - Determine whether sentence B is the context of sentence A.

Pre-Training Model Architecture



Pre-training model architecture in BERT:

- Jointly conditioned on both left and right context in all layers.
- Obtain the bidirectional information of the sequence.

Conclusion

Advantages:

- Avoid sequence model and can realize **parallelization**.
- Solves the problem of **long distance dependence**.
- Self-attention can produce more **interpretable models**.

Disadvantages:

- Loss of **position information** (Compare with RNN)
- Lost the ability to capture **local features** (Compare with CNN)

Transformer is widely used in Deep Learning fields (NLP, CV, etc.).

*Thank
you*

