



计算机系 宋曦轩 张瀚宸

Summer24.net9.org

自然语言处理

- 如何表示一个词的含义?
- 如何获取词在句子中的向量表示?
- 如何在模型中存储知识?
- 如何基于以上原理构建语言模型?

🤗 如何表示一个词的含义?

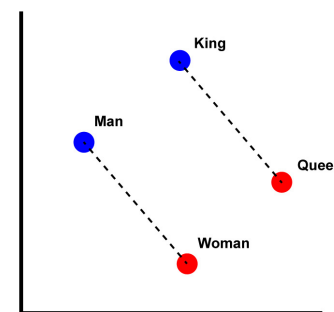
- 用一个id表示一个词?

北京	🐙	=344
南京	🐙	=345
中学		=364
大学		=365
北大		=344+365=709?

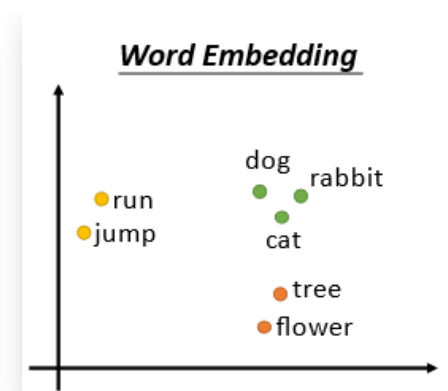
- 离散的id型词语如何输入到神经网络之中?
- 神经网络的输入对象通常是feature -> 我们需要得到词的feature

😊 Word Embedding 词嵌入

- 将一个词映射为一个 *embedding* dim 维的向量
- 每一维具有一定的含义（具体含义可能很抽象）



维度	金属	生物	👋	😊
电子羊	0.9	0.1	0.3	0.2
仿生人	0.5	0.4	0.4	0.8
拔罐王	0.2	0.8	0.9	1.0



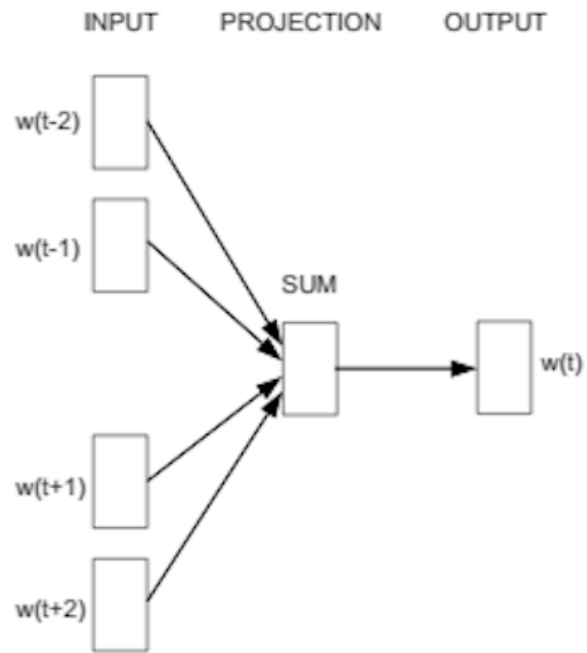
🤗 如何获取 Word Embedding ?

- 手工构造?
 - “北京在北纬40度，所以必须有一维向量的值是40” ?
- 通过在任务中拟合数据获得
 - 例如对于 Next token prediction （根据前缀预测下个词）的任务：
 1. 随机初始化所有词 v 的 Word Embedding E_v
 2. 将前缀的 Embedding 求和取平均 $E_{predict} = \frac{1}{k} \sum_{j=0}^{k-1} E_{v_j}$
 3. 下一个词是 Embedding 与 $E_{predict}$ 最接近的词
$$v_{predict} = \max_{v_i} \cos \langle E_{predict}, E_{v_i} \rangle$$
 4. 计算loss, 反向传播

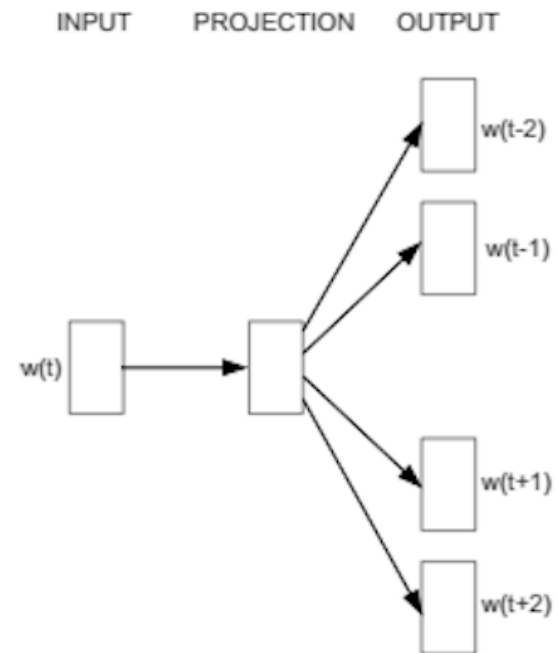
🤗 如何获取词在句子中的向量表示?

- **An Apple a Day Keeps the Doctor Away**
- 一天一部 **Iphone** 让我与博士学位失之交臂?
- 🍏 or 🍏?
- 有没有被咬一口?
- 一个简单的想法:
- 将句中所有词的向量加权求和, 表示词在句中的含义。

👉 Skip-Gram & CBOW

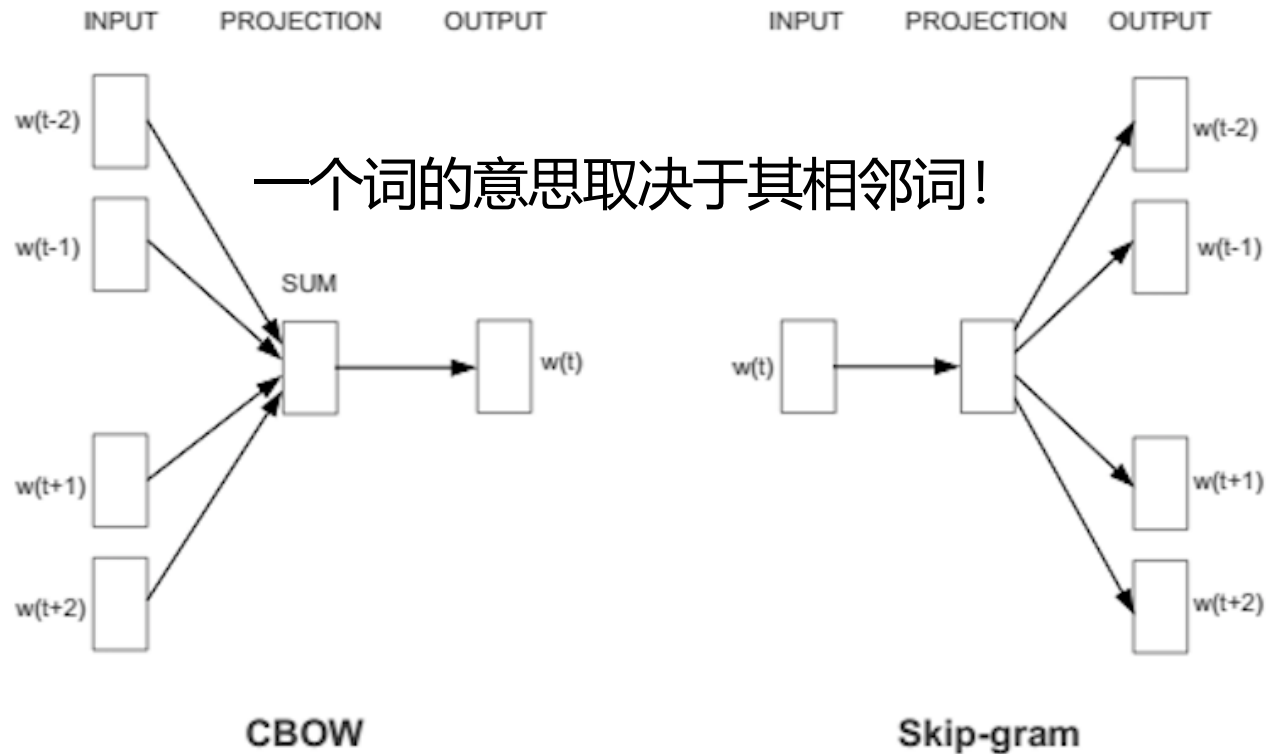


CBOW



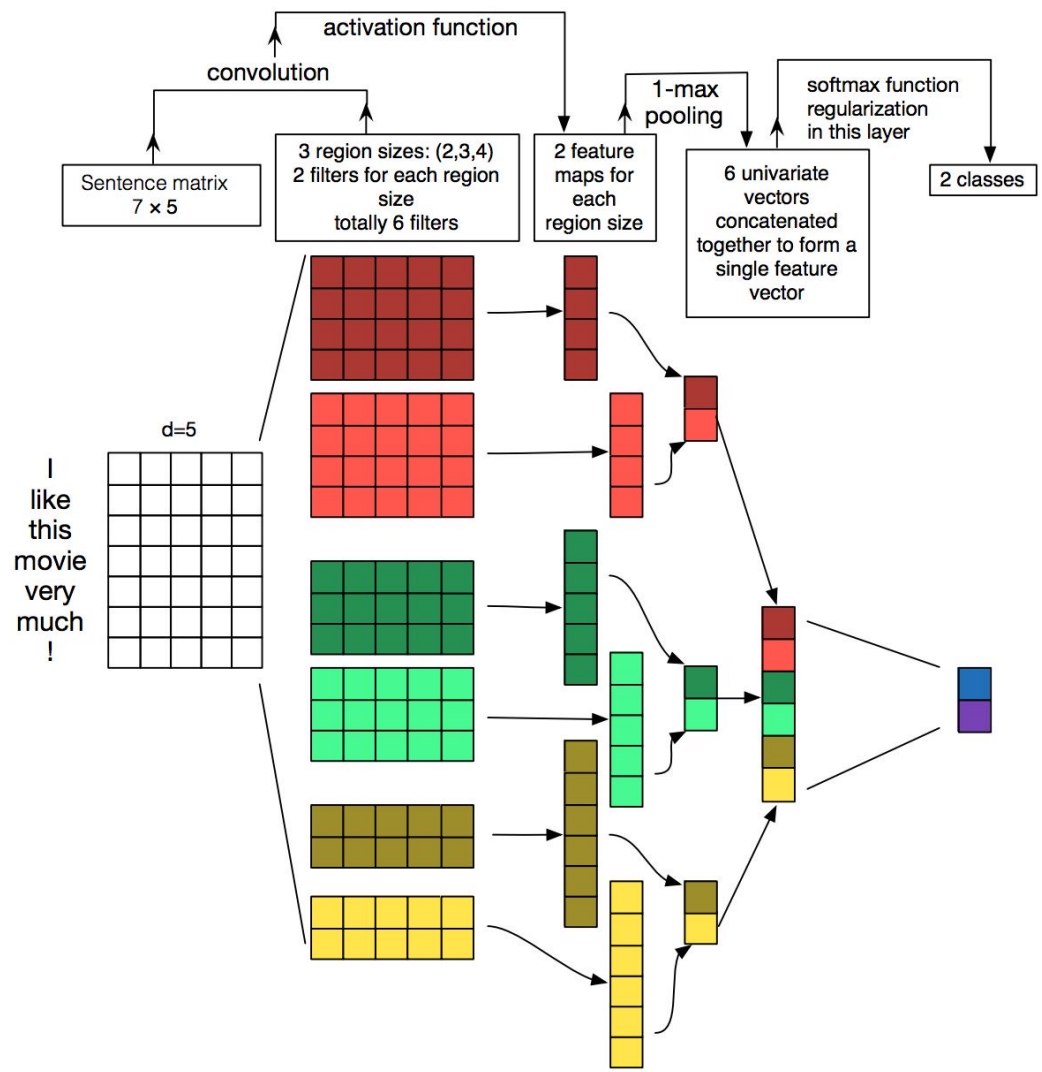
Skip-gram

😊 Skip-Gram & CBOW



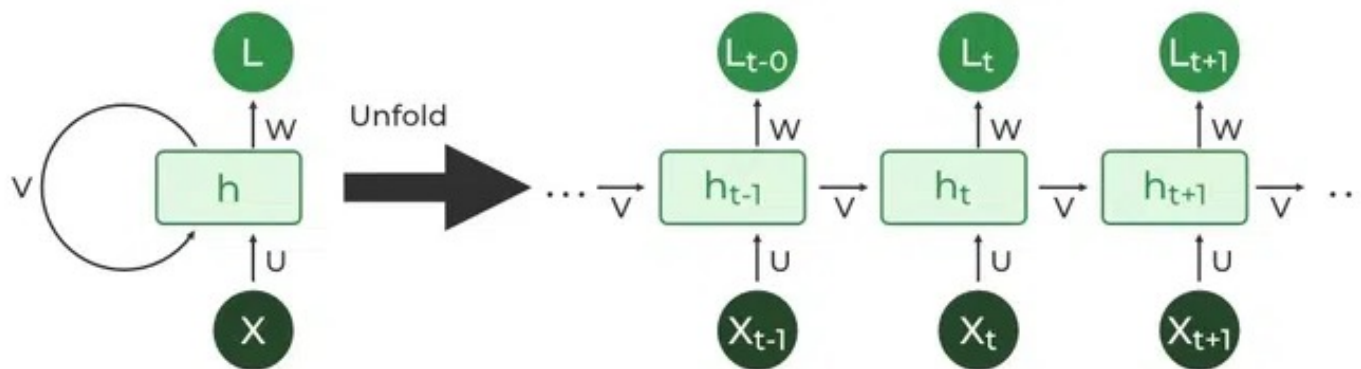
😊 TextCNN

使用卷积来聚合特征



👋 RNN

- 预测下一个词



长距离依赖？

- 小明从小就对大自然充满好奇，他总是喜欢在乡间的小道上漫步，观察周围的动植物。每当清晨的阳光穿过树梢洒在草地上，露珠闪烁着微光，他总会蹲下身子，细细地看着那晶莹透亮的小水珠，仿佛整个世界都安静了下来。
- 小时候，村子里有一位老爷爷，精通植物的知识。他经常跟在老爷爷身后学习，哪一种树是松树，哪一株草是蒲公英，他都能轻易辨认，并能说出它们的用处。一段时间后，孩子们都称他为“小博物学家”，村里人也说他将来一定有出息。
- 岁月如梭，转眼间，他已经是中学生了，尽管学业繁重，他仍旧没有放弃对自然的热爱。他报名参加了学校的生物社团，和老师、同学们一起进行野外考察，参与了许多有趣的实验项目。记得有一次，班级组织去山林采集植物标本，他兴奋地向大家展示他找到的一种罕见的蘑菇，并详细讲解了它的生态习性，不仅让同学们佩服，也让老师对他刮目相看。
- 在高中的生物竞赛中，凭借扎实的基础知识和敏锐的观察力，他取得了优异的成绩。家人也为他感到骄傲，特别是那个曾经教他认识植物的老爷爷，看到一天天长大成人，内心充满了欣慰。老爷爷鼓励他继续追寻自己的梦想，不要畏惧困难和挑战。
- 高中毕业后，他成功考入了一所著名的农业大学，选择了植物保护专业。大学生活丰富多彩，但他更专注于自己的研究领域。经常待在实验室里，熬夜查看实验数据，与导师探讨课题，甚至利用假期走访各地，调查不同地区的植物生态。他的一些研究成果还在国内外学术会议上得到高度评价。
- 随着专业知识的不断积累，他并没有止步于此，决定去国外深造，学习更多先进的知识和技术。通过不懈的努力，最终获得了一所国际知名学府的博士录取通知书。前往异国他乡的那一天，他带着家乡泥土的气息和老爷爷赠送的一本植物图鉴，内心充满憧憬和期待。
- 国外的学习生活使他更加开阔眼界，也结识了许多志同道合的朋友，尽管遇到不少学术难题，但始终秉持着对自然的热爱，坚定不移地在科学道路上前行。几年后，他以优异的成绩完成学业，带着知识和热情回到祖国，投入植保事业，为保护环境贡献力量。
- 回到故乡时，村里的小伙伴们看到他个个惊叹不已，纷纷围了上来，想听他讲述在外面的见闻。在给大家讲述旅途中的故事时，那位老爷爷微笑着遥望远方，仿佛看到了一片生机勃勃的未来。一个看似平凡的名字，却承载了不凡的梦想和希望，每当那么熟悉的呼喊声在耳边响起时，时间仿佛又回到了那个无忧无虑的童年时代。
- 他的名字叫做：_____

👉 Attention is all you need!

- *attention probs*:

	🍏	are	good	for	❤️
🍏	0.4	0.05	0.2	0.05	0.3
are	0.05	0.75	0.1	0.05	0.05
good
for
❤️

$$E_{\text{🍏 in sentence}} = 0.4E_{\text{🍏}} + 0.05 E_{\text{are}} + 0.2 E_{\text{good}} + 0.05 E_{\text{for}} + 0.3 E_{\text{❤️}}$$

👉 Attention Probs

- 如何获取 *attention probs*? 两个词越相关越大?

- 以两个词 *Embedding* 的余弦夹角表示相似度:

$$\text{attention scores}(v_i, v_j) = \cos \langle E_{v_i}, E_{v_j} \rangle$$

- 使用 *Softmax* 归一化:

$$\text{attention probs}(v_i, v_j) = \frac{e^{\text{attention scores}(v_i, v_j)}}{\sum_{k=0}^{\text{sequence length}} e^{\text{attention scores}(v_i, v_k)}}$$

$$E_{v_i \text{ in sentence}} = \sum_{j=0}^{\text{seq len}} \text{attention probs}(v_i, v_j) E_j$$

🤗如何科学获取Attention Score?

- *Embedding*的余弦夹角：与 v_i 相似的特征真的是 v_i 需要的吗? 🤔
 - 假如句子为：🍏🍏🍏🍏🍏🍏🍏🍏🍏🍏 are good.
 - 与 $E_{\text{🍏}}$ 最接近的总是 $E_{\text{🍏}}$, $E_{\text{🍏 in sentence}} = E_{\text{🍏}}$

• *Embedding* \rightarrow (*Query*, *Key*) 🤗🤗🤗

• *Query*: 想查询的特征

• *Key*: 想被查询的特征

$$\text{attention scores}(v_i, v_j) = \cos \langle Q_{v_i}, K_{v_j} \rangle$$

🤔 Attention

- Embedding $\xrightarrow{\text{Linear}}$ (*Query*, *Key*, *Value*)
 $\text{attention scores}(v_i, v_j) = \cos \langle Q_{v_i}, K_{v_j} \rangle$

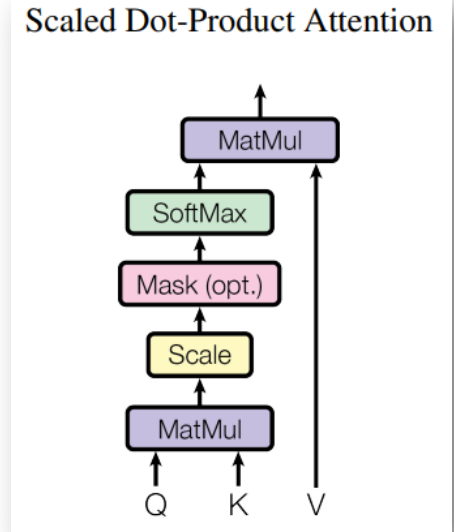
$$\text{attention probs}(v_i, v_j) = \frac{e^{\text{attention scores}(v_i, v_j)}}{\sum_{k=0}^{\text{sequence length}} e^{\text{attention scores}(v_i, v_k)}}$$

$$V_{v_i \text{ in sentence}} = \sum_{j=0}^{\text{seq len}} \text{attention probs}(v_i, v_j) V_j$$

$$E_{v_i \text{ in sentence}} \xleftarrow{\text{Linear}} V_{v_i \text{ in sentence}}$$

😊 Attention

- Input: *Embeddings* $shape = (seq\ len, embedding\ dim)$
- $Q = Q_{proj}(Embeddings)$ $shape = (seq\ len, embedding\ dim)$
- $K = K_{proj}(Embeddings)$ $shape = (seq\ len, embedding\ dim)$
- $V = V_{proj}(Embeddings)$ $shape = (seq\ len, embedding\ dim)$
- *Attention Probs* $(Q, K) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)$ $shape = (seq\ len, seq\ len)$
- *Attention* $(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$ $shape = (seq\ len, embedding\ dim)$
- *Output* $= O_{proj}(Attention(Q, K, V))$ $shape = (seq\ len, embedding\ dim)$



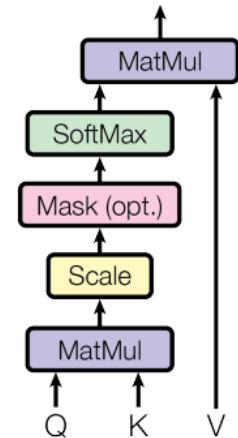
👉 Attention

- Query: 我想找到他的名字...
- Key:
 - 小明 * 他的名字 = 0.8
 - 他 * 他的名字 = 0.1
 - 小博物学家 * 他的名字 = 0.05
 - ...
- Result = $0.8 * V_{\text{小明}} + 0.1 * V_{\text{他}} + 0.05 * V_{\text{小博物学家}} + \dots$

😊 Attention

- Input: *Embeddings* $shape = (seq\ len, embedding\ dim)$
- $Q = Q_{proj}(Embeddings)$ $shape = (seq\ len, query\ dim)$
- $K = K_{proj}(Embeddings)$ $shape = (seq\ len, key\ dim), key\ dim = query\ dim$
- $V = V_{proj}(Embeddings)$ $shape = (seq\ len, value\ dim)$
- $Attention\ Probs(Q, K) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)$ $shape = (seq\ len, seq\ len)$
- $Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$ $shape = (seq\ len, value\ dim)$
- $Output = O_{proj}(Attention(Q, K, V))$ $shape = (seq\ len, embedding\ dim)$

Scaled Dot-Product Attention



👉如何加入位置信息🧭?

- Attention机制中，并没有考虑向量在sequence中的位置。
- *Position embedding*
- 为每一个位置预设一个向量 PE_{pos}

$$E_{v_i \text{ input}} = E_{v_i} + PE_{pos}$$

🤗 模型中知识存在何处?

- 知识: **Key-Value** Pair: (K, V)

- Neural Memory, 使用 x 查询 k_i :

$$p(k_i|x) \propto e^{k_i \cdot x}$$

$$MN(x) = \sum_{i=1}^{dim} p(k_i|x) \cdot v_i$$

$$K = [k_i], V = [v_i]$$

$$MN(x) = \text{softmax}(xK^T)V$$

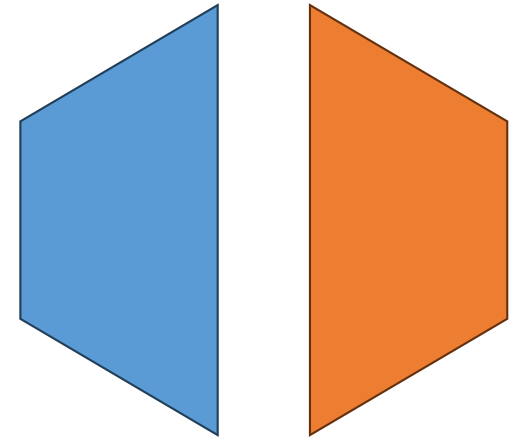
- 用两个线性层实现前馈神经网络:

$$FFN(x) = f(xK^T)V$$

😊 Feedforward Neural Network

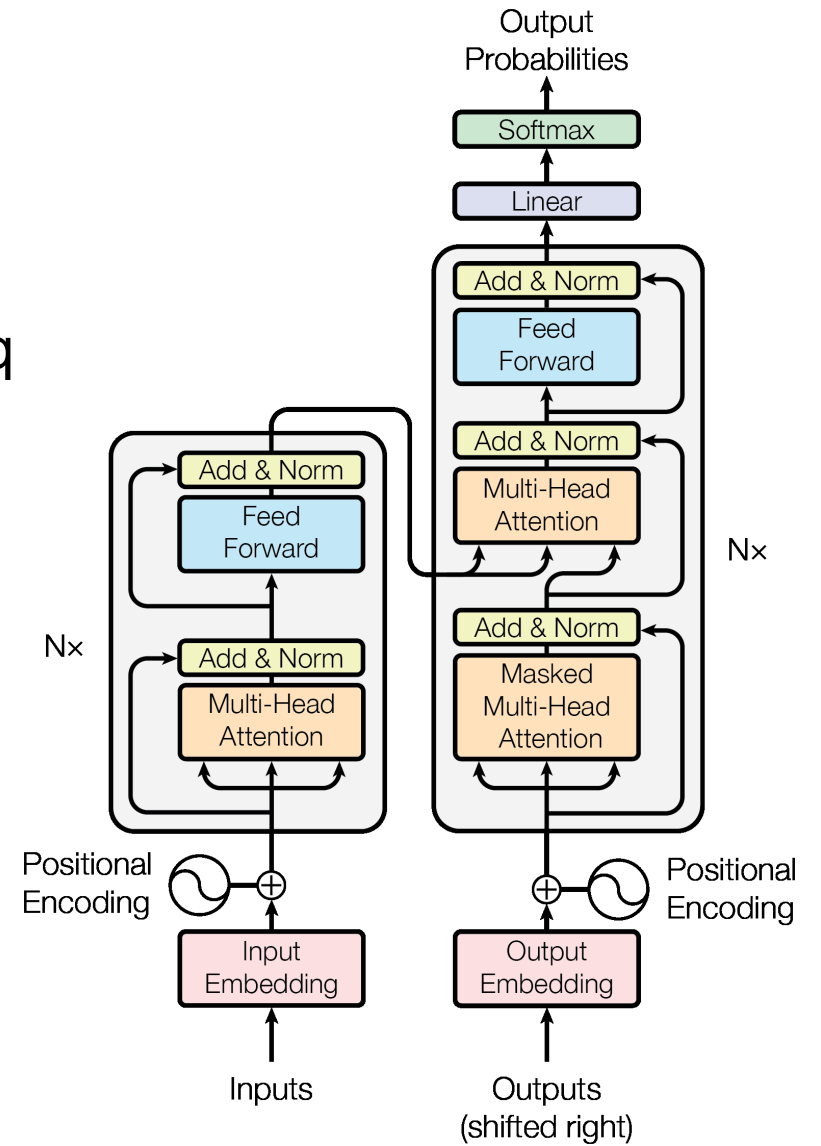
$$\text{FFN}(x) = f(xK^T)V$$

- f 一般使用 $relu$ 或其变种
- 一般inner *hidden* dim = 4 *embedding* dim
- 参数量: $8 \times \text{embeddings dim}^2$



👋 Transformer

- 从Input序列到Output序列——Seq2Seq



👉如何基于以上原理构建语言模型？

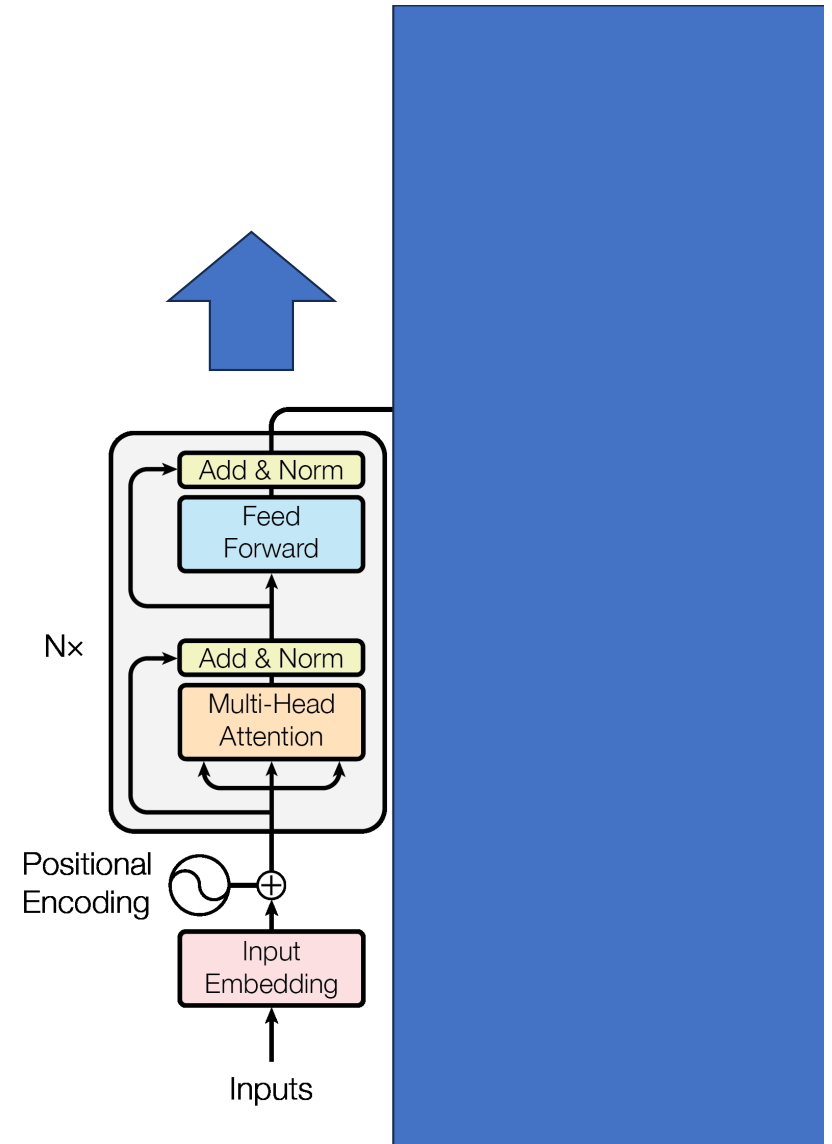
- 语言模型：

$$P(w_i | context)$$

- *context*：上下文
- w_i ：某个位置上的某个词的概率
- 自编码语言模型（如BERT）：
 - “北京在纬度[MASK]度”，求[MASK]填词的概率分布：
 $P(\cdot | \text{北京在纬度[MASK]度})$
- 自编码语言模型（如GPT）：
 - “北京在北”，求下一个字的概率分布： $P(\cdot | \text{北京在北})$

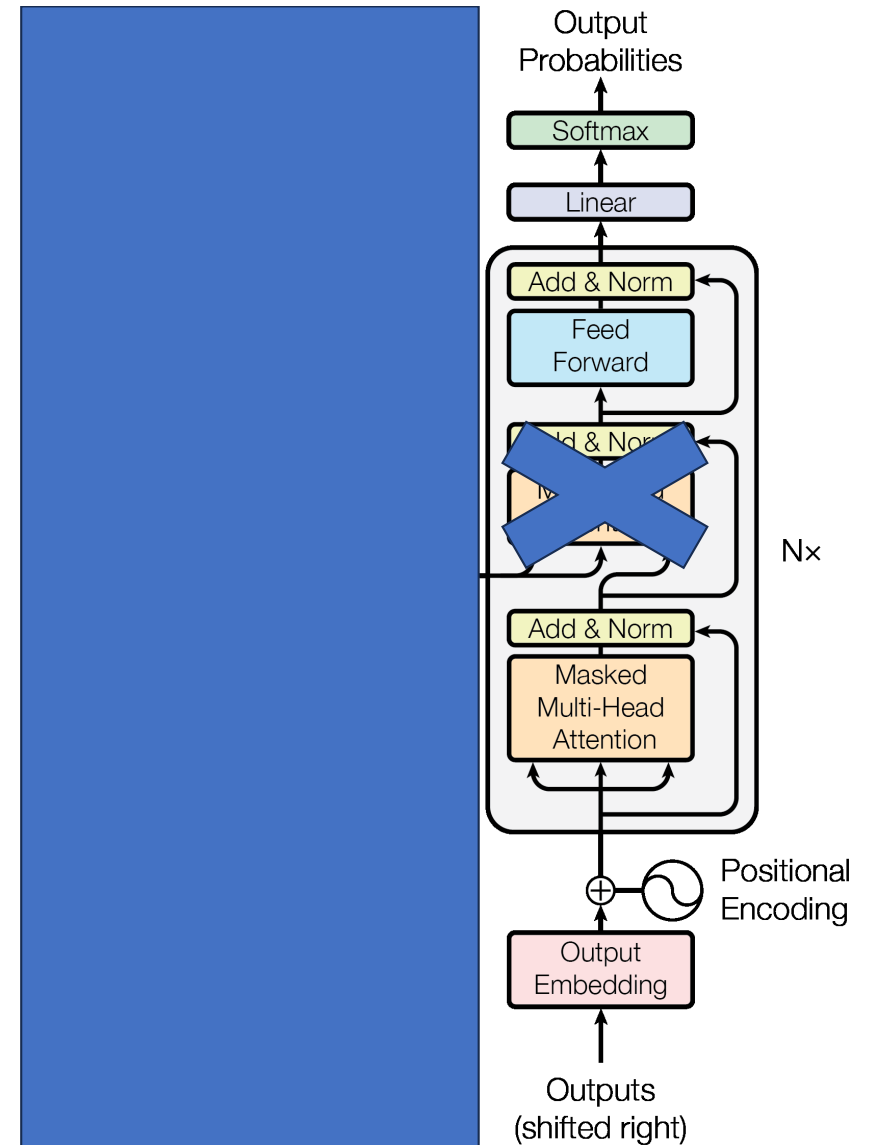
😊 Encoder Only

- 从Input序列到分类输出——
Classification
- 典型模型：BERT



😊 Decoder Only

- 从输入序列到输入序列的延长——
Generation
- 典型模型：GPT



👋 Attention Types

- Self Attention QKV来自同一序列

Q \ K	🍎	are	good	for	❤️
🍎	0.4	0.05	0.2	0.05	0.3
are	0.05	0.75	0.1	0.05	0.05
good
for
❤️



Attention Mask

👋 Attention Types

- Cross Attention QKV来自不同序列

Q \ K	苹果	对	我们	身体	好
🍎	0.4	0.05	0.2	0.05	0.3
are	0.05	0.75	0.1	0.05	0.05
good
for
❤️

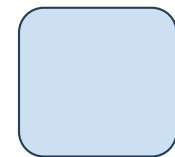


Attention Mask

👋 Attention Types

- 一种特殊的Self Attention

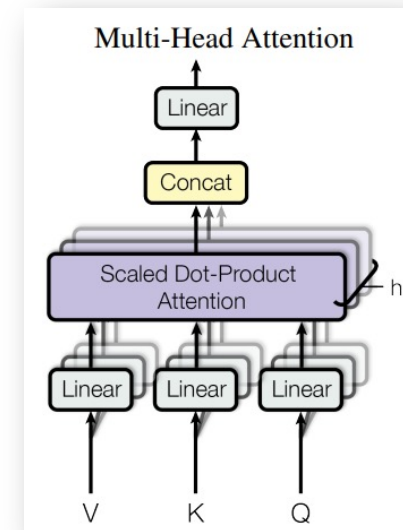
Q \ K	🍏	are	good	for	❤️
🍏	0.55	0.05	0.4	0	0
are	0.15	0.75	0.1	0	0
good	0	0
for	0	0	0
❤️	0	0	0



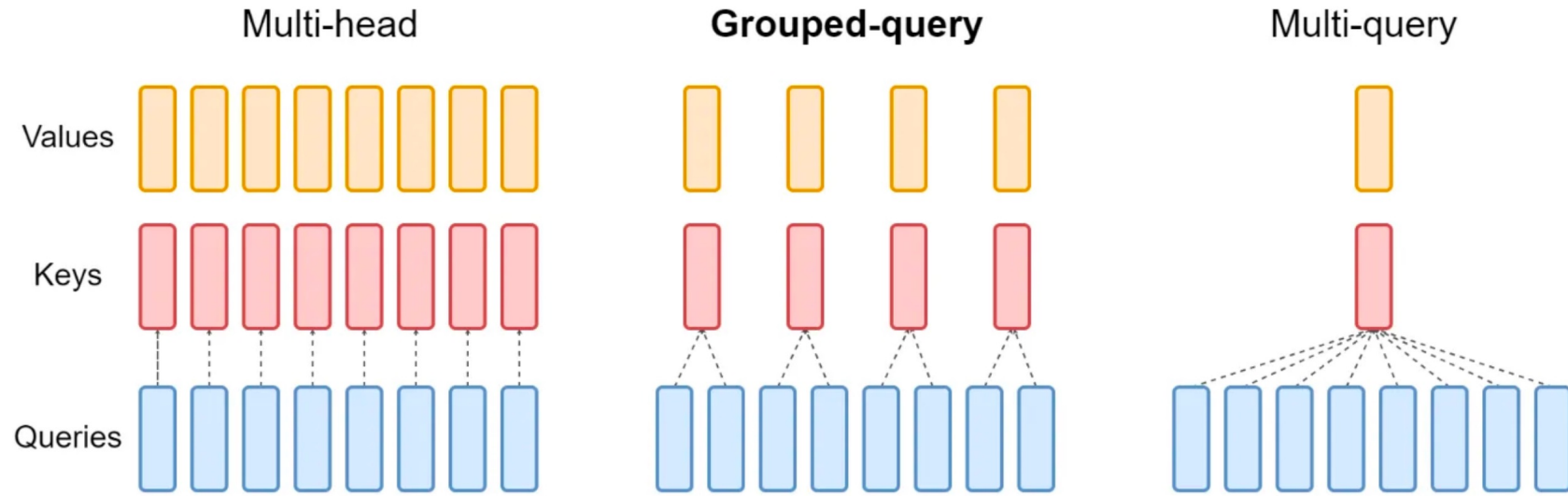
Attention Mask

👋 Multi Head Attention 😊😊😊

- $(Q_0, Q_1, Q_2, Q_3) = Q = Q_{proj}(Embeddings)$ $Q_i.shape = (seq\ len, \frac{embeddings\ dim}{num\ head})$
- $(K_0, K_1, K_2, K_3) = K = K_{proj}(Embeddings)$ $K_i.shape = (seq\ len, \frac{embeddings\ dim}{num\ head})$
- $(V_0, V_1, V_2, V_3) = V = V_{proj}(Embeddings)$ $V_i.shape = (seq\ len, \frac{embeddings\ dim}{num\ head})$
- $Output = O_{proj}(Concat(Attention(Q_i, K_i, V_i)))$
- 参数量: $4 \times embeddings\ dim^2$
- 不同Head关注不同的语义关系



👉 Group Query Attention



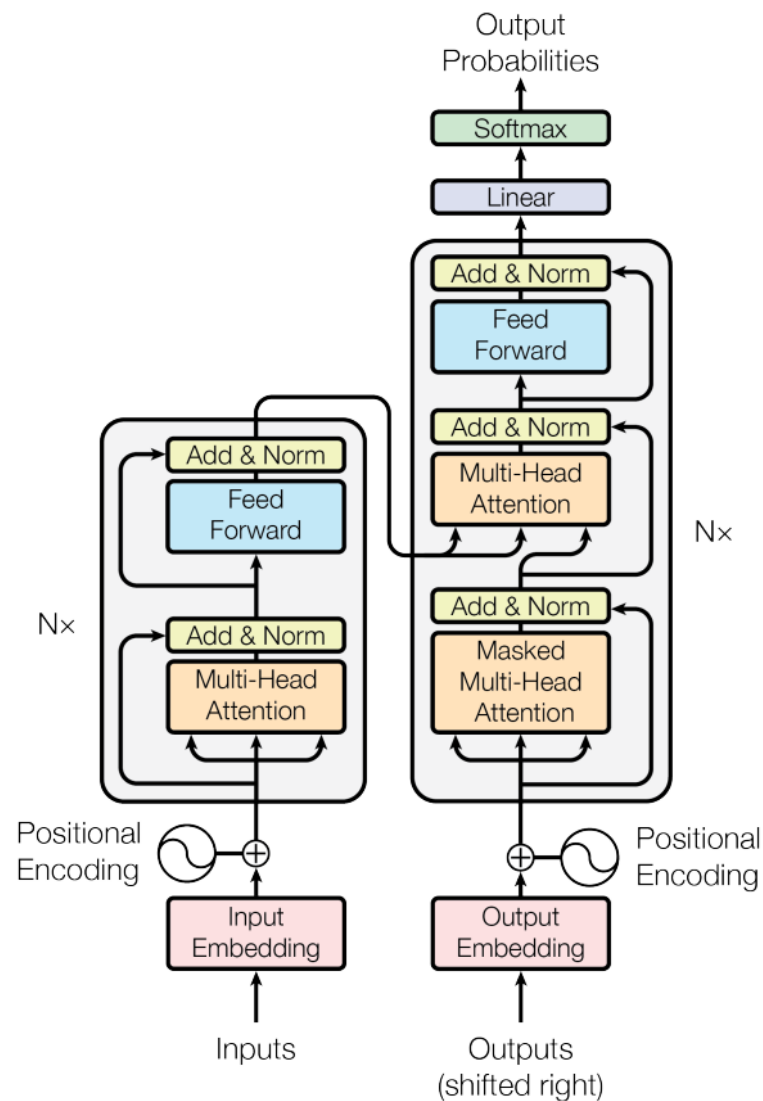
👋 Transformer

- *LM Head*: 一个线性层, 输入维度为 *embedding dim*; 输出维度为词表大小 *vocab size*。

- 输出每个位置上各词未归一化的对数概率 *logits*:

$$P_{\theta}(\cdot | context) = \text{Softmax}(\text{logits})$$

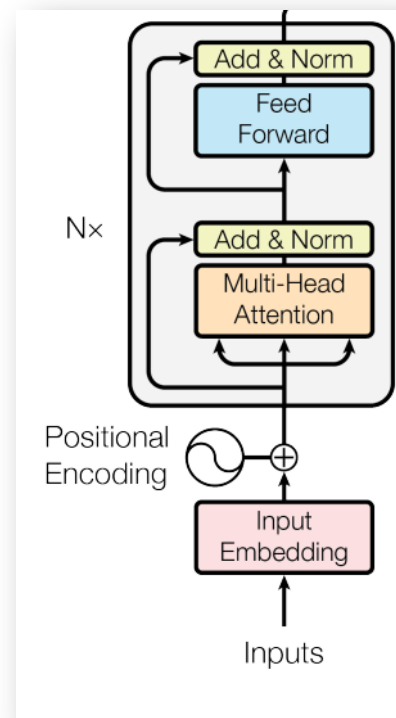
- 左侧为Encoder
- 右侧为Decoder



😊 Encoder

- BERT
- 一次前向传播即可计算出所有[MASK]的概率分布
- 文本理解任务

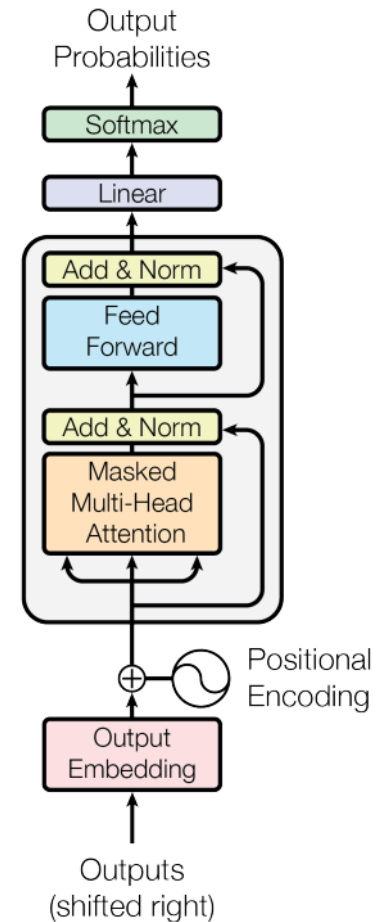
Q \ K	🍎	are	[MASK]	for	❤️
🍎	0.4	0.05	0.2	0.05	0.3
are	0.05	0.75	0.1	0.05	0.05
[MASK]
for
❤️



😊 Decoder

- GPT
- 自回归生成:
- 每次生成下一个词
- Masked Attention:
- 每个词做Attention时只能加权到之前的词
- 文本生成任务

Q \ K	🍎	are	good	for	❤️
🍎	1.0				
are	0.3	0.7			
good		
for	
❤️



语言模型训练

- 如何训练语言模型?
- 如何使用语言模型完成具体下游任务?
- 如何使用有限的硬件资源进行微调?
- 如何让语言模型更好的理解人类意图?

如何训练语言模型?

- 随机初始化权重 θ
- 反复随机初始化权重 θ ，直到权重能较好地完成任务✖
- 使用大量人工标注文本对模型进行训练😬😬😬
- 使用大量无标注文本对模型进行训练🤔😊👏

👉 预训练

- Mask filling or Next token prediction?

- Mask filling:

- 我是[MASK], [MASK]考试没有一次[MASK]。


$$\max_{\theta} (P_{\theta}(\text{大学生}|\text{context}) \cdot P_{\theta}(\text{幼儿园}|\text{context}) \cdot P_{\theta}(\text{参加}|\text{context}))$$

- Next token prediction:

- 我是大学生,

$$\begin{aligned} & \max_{\theta} \left(\prod P_{\theta}(\text{next token}|\text{prefix}) \right) \\ & = \max_{\theta} (P_{\theta}(\text{大学生}|\text{我是}) \cdot P_{\theta}(, |\text{我是大学生})) \end{aligned}$$

如何使用语言模型完成具体下游任务?

- 反复随机初始化权重 θ , 直到权重 θ 能较好地完成任务✖
- 在预训练模型的基础上继续使用大量无标注文本训练😞
- 使用少量标注数据 (x, y) 模型进行训练

🤗 如何使用语言模型完成具体下游任务?

- 例：分类任务
- 替换预训练模型中的LM Head为Classification Head
- Classification Head输入维度为 *embedding dim*
- 输出维度为类别数目 *num labels*
- ~~然后直接拿去部署~~
- 使用标注的数据 (x, y) 进行训练
 - x : 文本; y : 类别

如何使用有限的硬件资源进行微调?

- **LoRA: Low-Rank Adaptation**

Low-Rank Adaption

$$\Delta W = W_a W_b$$

$$W = W + W_a W_b$$

$$\text{size}(W_a) = n \times r$$



Fine-Tuning

$$W = W + \Delta W$$

W : New Weights

W : Raw Weights

ΔW : Update

$$\text{size}(\Delta W) = n^2$$

Too Big!

👉 如何让语言模型更好的理解人类意图?

- 指令微调
- 使用人工标注的 (prompt, response) 对模型进行微调
- 强化学习

