《机器学习基础》

Bert

自然语言处理（NLP）

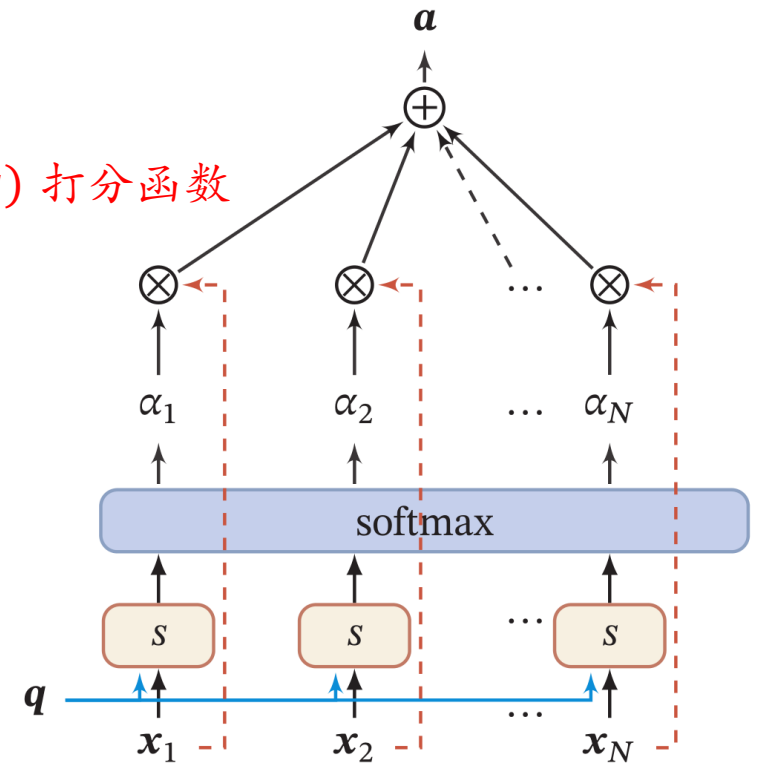
- Textual entailment(文本蕴涵)
- Question answering(问题回答)
- Semantic similarity assessment(语义相似度评估)
- Document classification(文本分类)
- Machine translation(机器翻译)

注意力模型

- 硬性注意力机制（hard attention mechanism）
- 软性注意力机制（soft attention mechanism）
- 注意力机制可以分为两步
  - 计算注意力分布α，

$$\begin{aligned}\alpha_n &= p(z=n|\boldsymbol{X},\boldsymbol{q}) \\ &= \text{softmax}\left(s(\boldsymbol{x}_n,\boldsymbol{q})\right) \\ &= \frac{\exp\left(s(\boldsymbol{x}_n,\boldsymbol{q})\right)}{\sum_{j=1}^{N}\exp\left(s(\boldsymbol{x}_j,\boldsymbol{q})\right)}\end{aligned}$$

$s(x_n, q)$ 打分函数

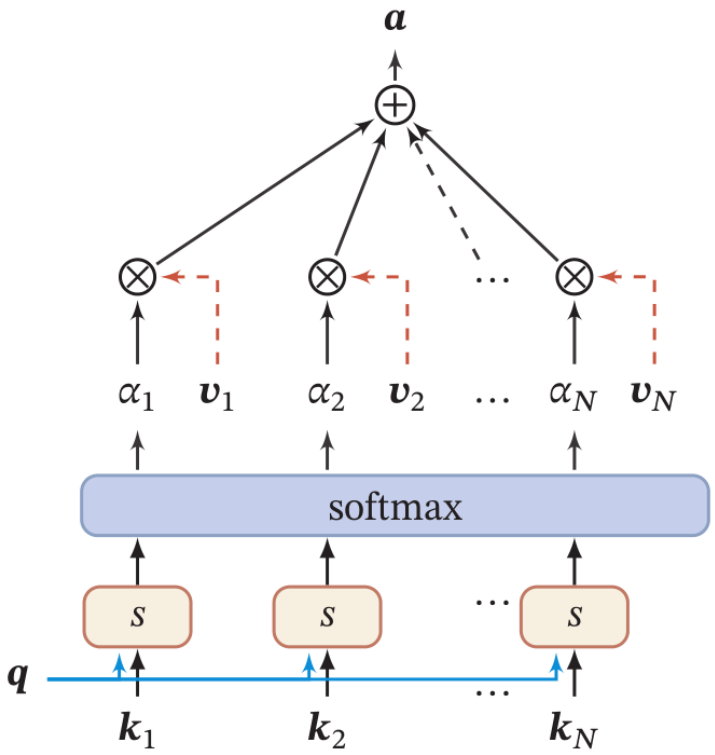


  - 根据α来计算输入信息的加权平均。

$$\begin{aligned}\text{att}(\boldsymbol{X},\boldsymbol{q}) &= \sum_{n=1}^{N}\alpha_n\boldsymbol{x}_n, \\ &= \mathbb{E}_{z\sim p(z|\boldsymbol{X},\boldsymbol{q})}[\boldsymbol{x}_z]\end{aligned}$$

注意力机制的变体

- 键值对注意力（key-value pair attention）

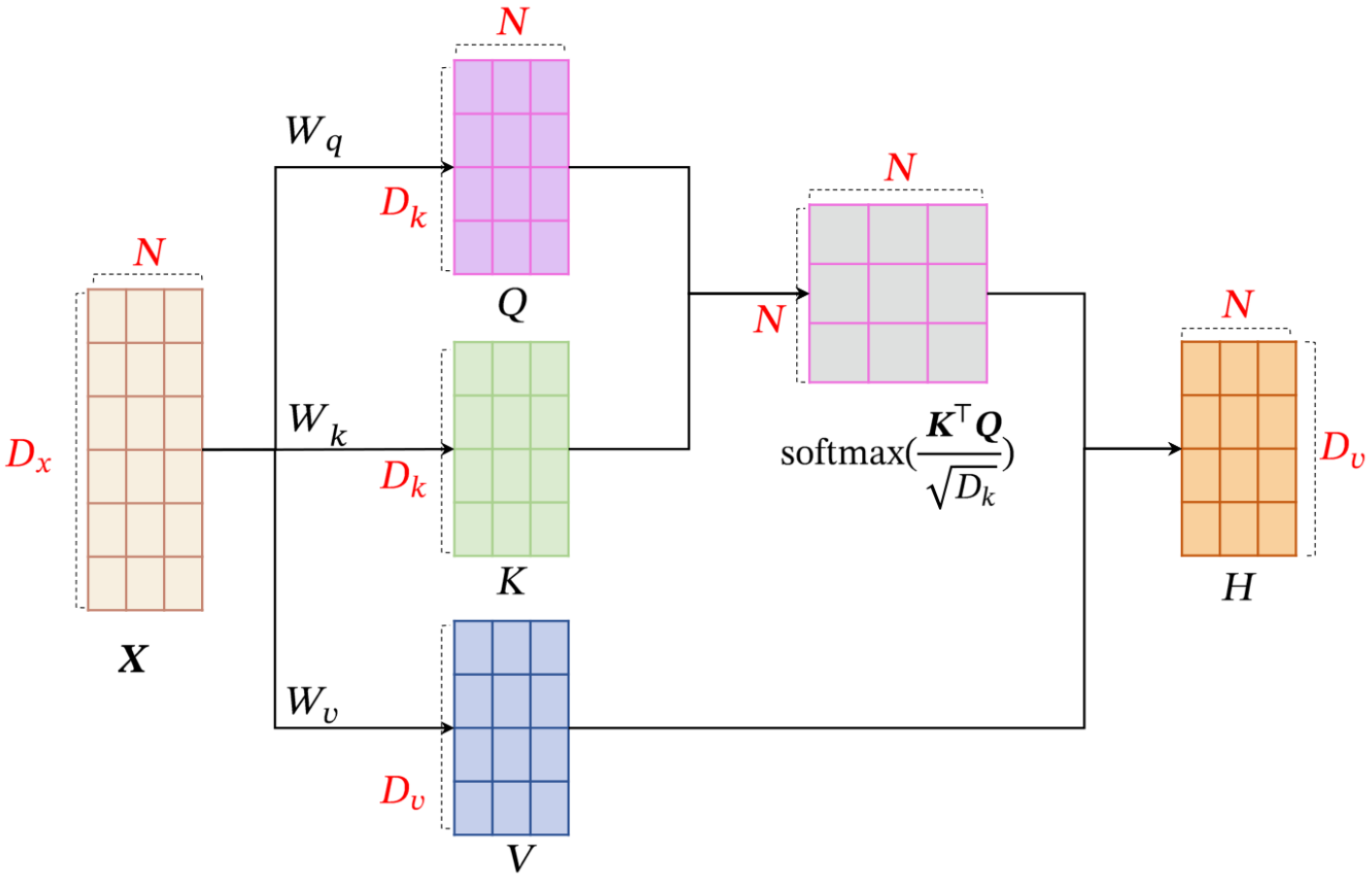


用$(K,V) = [(\boldsymbol{k}_1, \boldsymbol{v}_1), \cdots, (\boldsymbol{k}_N, \boldsymbol{v}_N)]$表示N个输入信息

$$\text{att}\Big((\boldsymbol{K},\boldsymbol{V}),\boldsymbol{q}\Big) = \sum_{n=1}^{N} \alpha_n \boldsymbol{v}_n,$$

$$= \sum_{n=1}^{N} \frac{\exp\Big(s(\boldsymbol{k}_n,\boldsymbol{q})\Big)}{\sum_j \exp\Big(s(\boldsymbol{k}_j,\boldsymbol{q})\Big)} \boldsymbol{v}_n$$

QKV模式（Query-Key-Value）



自注意力模型

- 输入序列为$X = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \in R^{D_x \times N}$
- 首先生成三个向量序列

$$\boldsymbol{Q} = \boldsymbol{W}_q \boldsymbol{X} \in \mathbb{R}^{D_k \times N},$$

$$\boldsymbol{K} = \boldsymbol{W}_k \boldsymbol{X} \in \mathbb{R}^{D_k \times N},$$

$$\boldsymbol{V} = \boldsymbol{W}_v \boldsymbol{X} \in \mathbb{R}^{D_v \times N},$$

- 计算$\boldsymbol{h}_n$

$$\boldsymbol{h}_n = \text{att}\Big((\boldsymbol{K}, \boldsymbol{V}), \boldsymbol{q}_n\Big)$$

- 如果使用缩放点积来作为注意力打分函数，输出向量序列可以简写为

$$\boldsymbol{H} = \boldsymbol{V}\,\text{softmax}\left(\frac{\boldsymbol{K}^{\top}\boldsymbol{Q}}{\sqrt{D_k}}\right),$$

多头（multi-head）注意力

- 多头注意力（multi-head attention）
  - 利用多个查询 $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_M]$，来并行地从输入信息中选取多组信息．每个注意力关注输入信息的不同部分．

$$\text{att}\Big((\boldsymbol{K}, \boldsymbol{V}), \boldsymbol{Q}\Big) = \text{att}\Big((\boldsymbol{K}, \boldsymbol{V}), \boldsymbol{q}_1\Big) \oplus \cdots \oplus \text{att}\Big((\boldsymbol{K}, \boldsymbol{V}), \boldsymbol{q}_M\Big)$$

多头（multi-head）自注意力

- 多头自注意力（multi-head self-attention）
  - 结合了自注意力、多头注意力
  - 加入了位置编码，扩展了模型专注于不同位置的能力

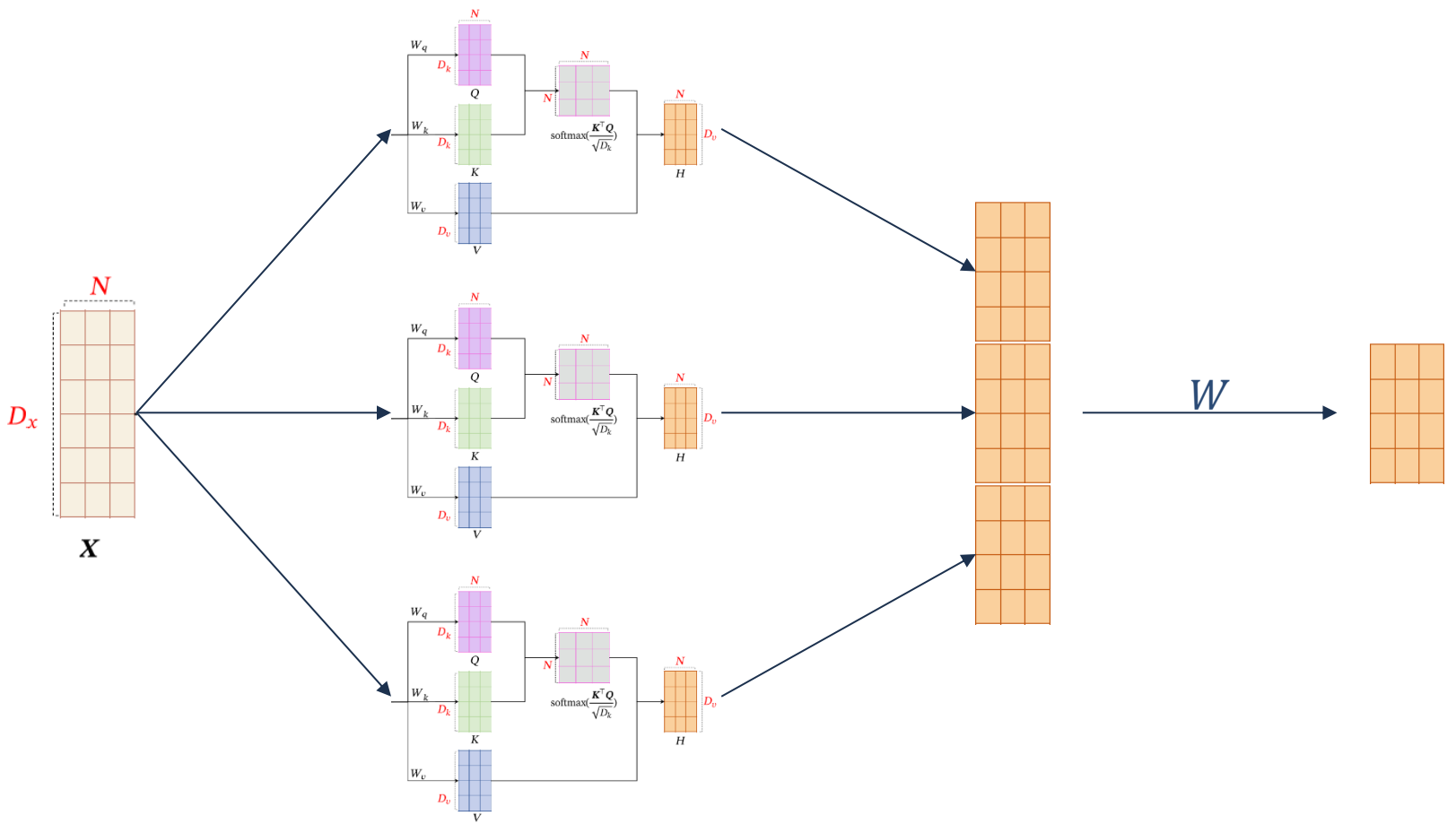
$$\text{MultiHead}(\boldsymbol{H}) = \boldsymbol{W}_o[\text{head}_1; \cdots; \text{head}_M],$$

$$\text{head}_m = \text{self-att}(\boldsymbol{Q}_m, \boldsymbol{K}_m, \boldsymbol{V}_m),$$

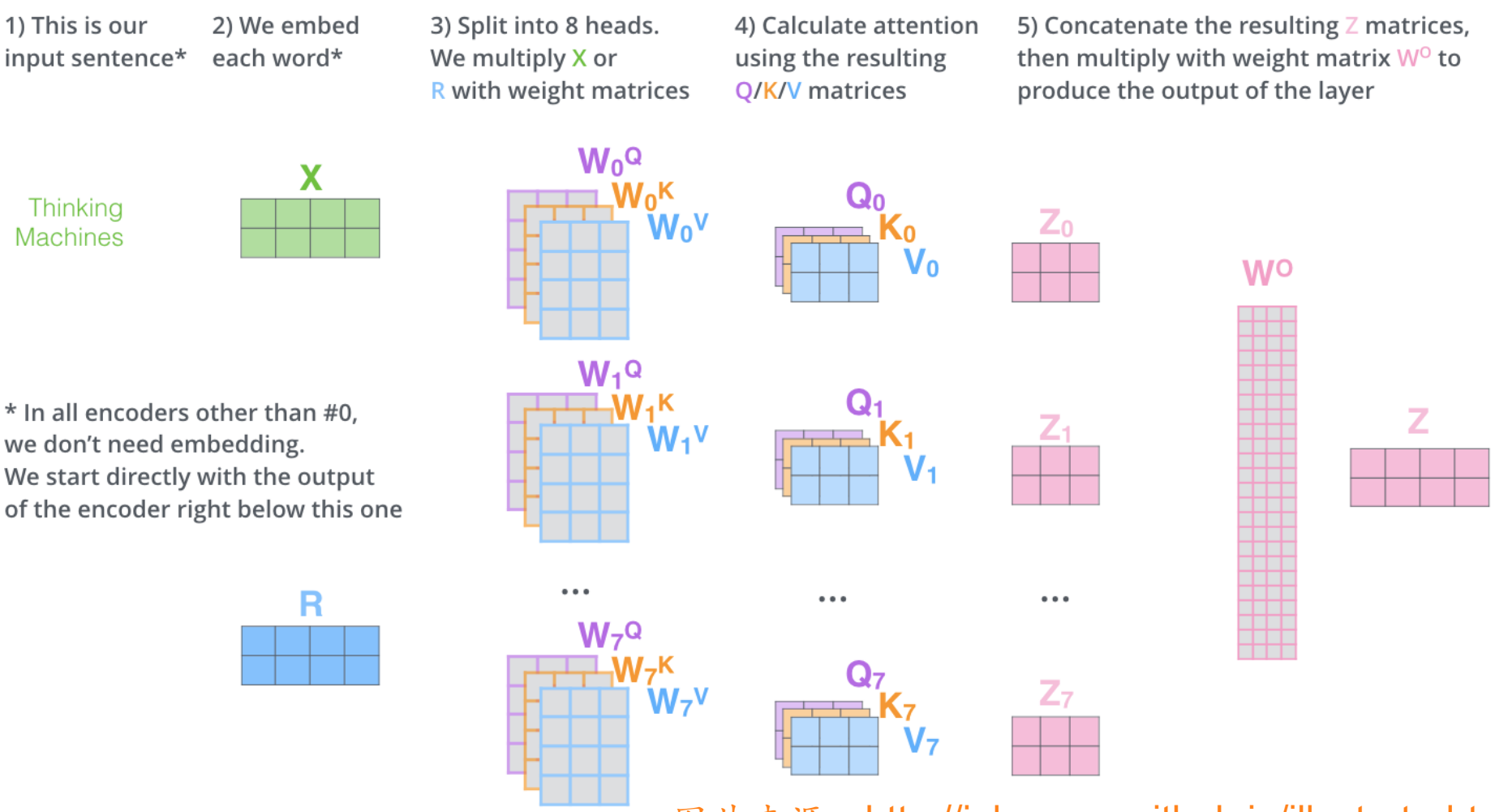
$$\forall m \in \{1, \cdots, M\}, \quad \boldsymbol{Q}_m = \boldsymbol{W}_q^m \boldsymbol{H}, \boldsymbol{K} = \boldsymbol{W}_k^m \boldsymbol{H}, \boldsymbol{V} = \boldsymbol{W}_v^m \boldsymbol{H},$$

其中 $\boldsymbol{W}_o \in \mathbb{R}^{D_h \times M d_v}$ 为输出投影矩阵，$\boldsymbol{W}_q^m \in \mathbb{R}^{D_k \times D_h}$，$\boldsymbol{W}_k^m \in \mathbb{R}^{D_k \times D_h}$，$\boldsymbol{W}_v^m \in \mathbb{R}^{D_v \times D_h}$ 为投影矩阵，$m \in \{1, \cdots, M\}$.

多头（multi-head）自注意力模型

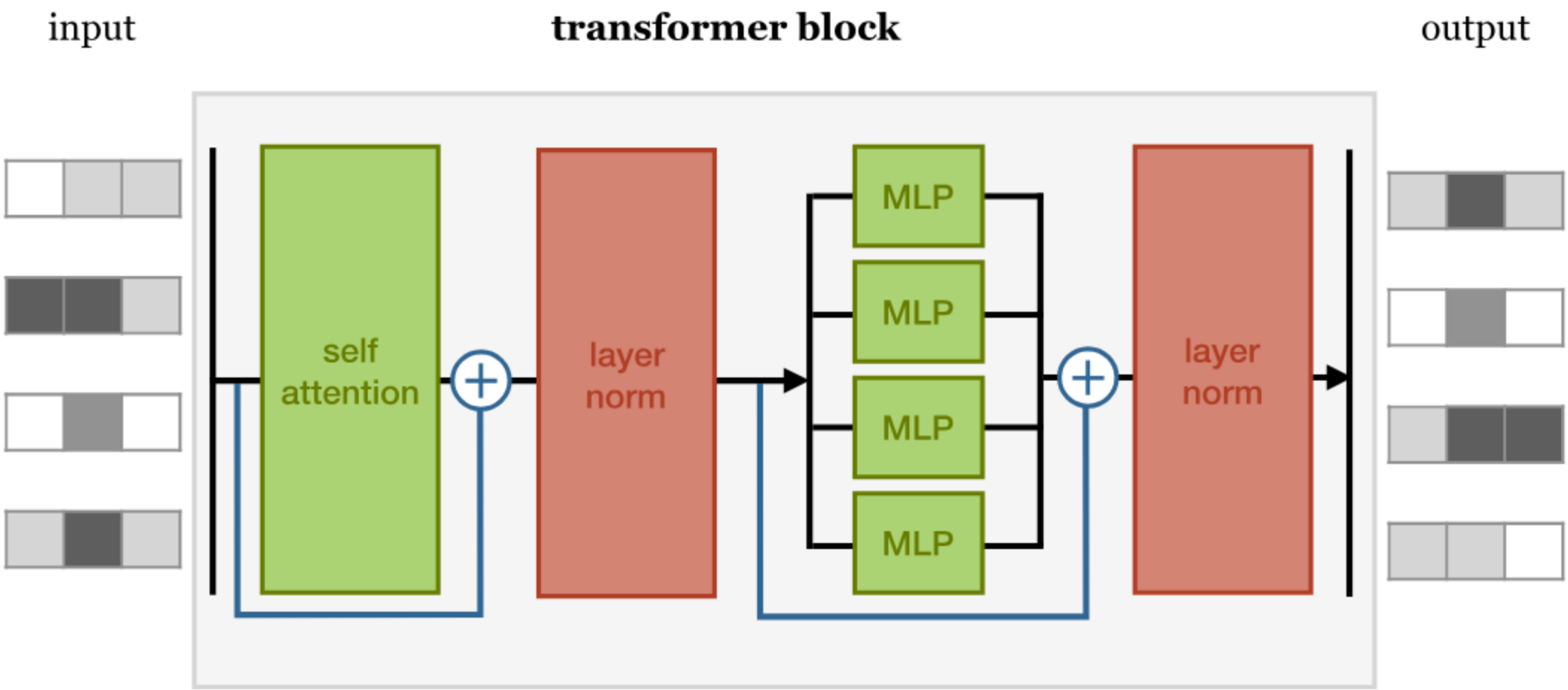


多头（multi-head）自注意力模型



图片来源：http://jalammar.github.io/illustrated-transformer/

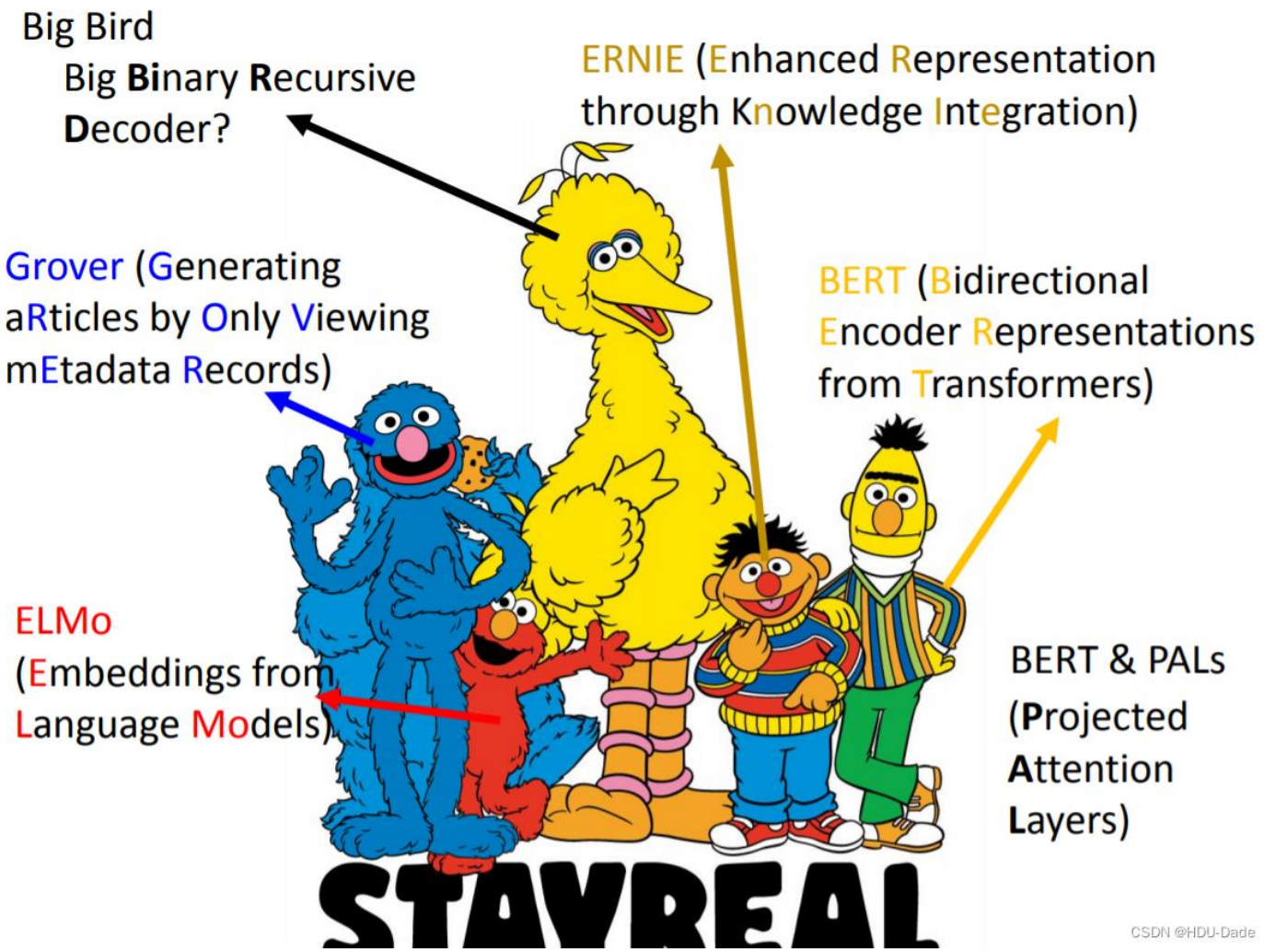
Transformer块



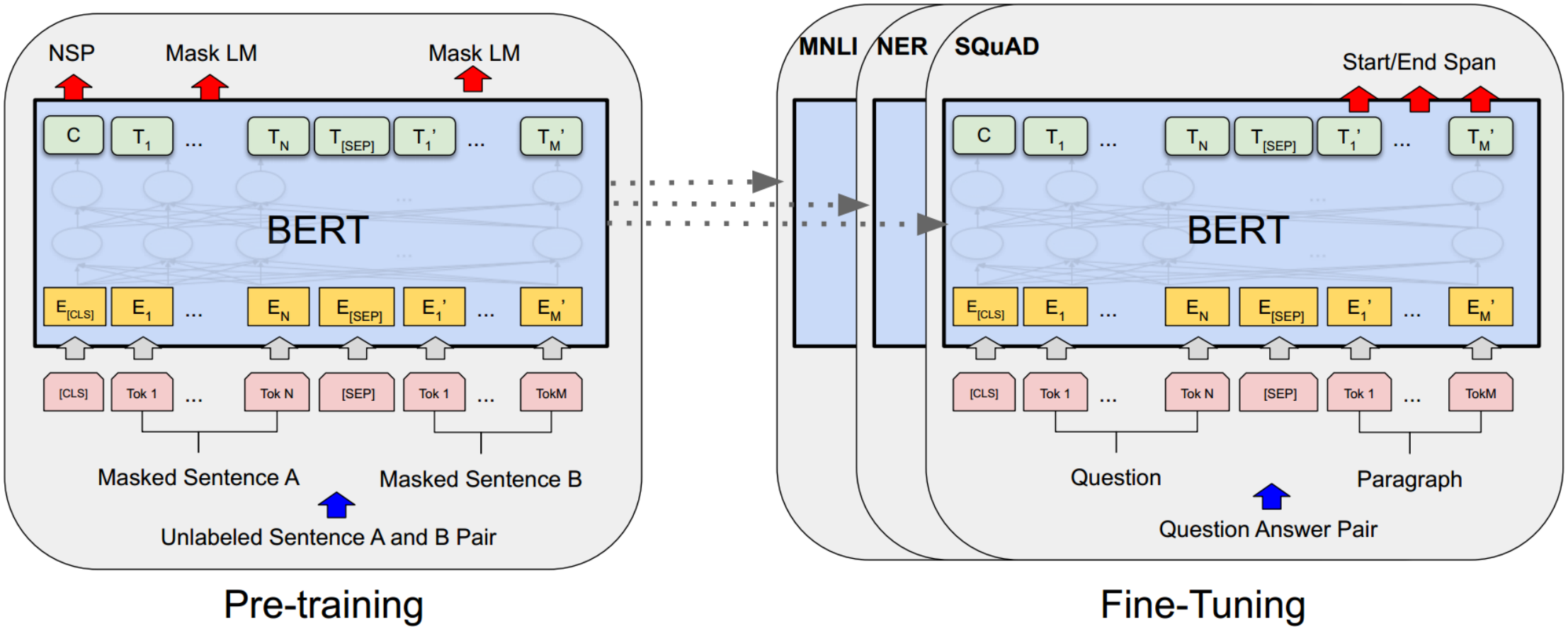
- Transformer

- Bert

芝麻街系列NLP模型


Big Bird
Big Binary Recursive Decoder?
ERNIE (Enhanced Representation through Knowledge Integration)
Grover (Generating aRticles by Only Viewing mEtadata Records)
BERT (Bidirectional Encoder Representations from Transformers)
ELMo (Embeddings from Language Models)
BERT & PALs (Projected Attention Layers)
STAYREAL
CSDN @HDU-Dade

Bert


NSP
Mask LM
Mask LM
C
T1
...
TN
T[SEP]
T1'
...
TM'
BERT
E[CLS]
E1
...
EN
E[SEP]
E1'
...
EM'
[CLS]
Tok 1
...
Tok N
[SEP]
Tok 1
...
TokM
Masked Sentence A
Masked Sentence B
Unlabeled Sentence A and B Pair
Pre-training
MNLI
NER
SQuAD
Start/End Span
Question
Paragraph
Question Answer Pair
Fine-Tuning

预训练（Pre-training）

- 一个已经在大规模数据上训练过的模型可以提供一个好的参数初始值
  - 好的初始值会使得网络收敛到一个泛化能力高的局部最优解.

- 预训练初始化（Pretrained Initialization）：
  - 在深度学习模型训练之前，使用预训练模型的权重作为初始权重。
  - **更快的收敛、 更好的性能、 减少过拟合**

微调（Fine-Tuning）

- 对一个已经在相关任务上预训练好的模型（预训练模型）进行额外的训练，以适应一个新的、更具体的任务（目标任务）。
- 通过利用预训练模型在大量数据上学习到的通用特征，来提高模型在新任务上的性能。

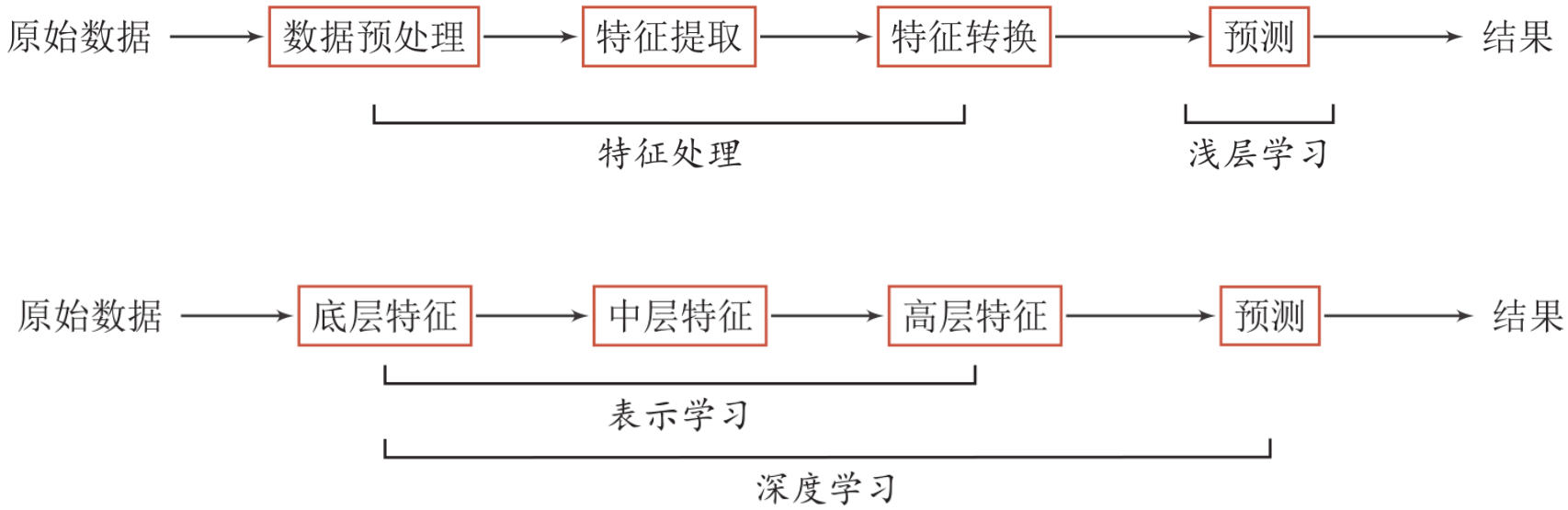
迁移学习（Transfer Learning）

- 迁移学习：**预训练模型-> 微调**

- 使用预先训练的模型为新任务提取单词/句子功能
- 需要构建一个新模型来捕获新任务所需的信息
- 通常不更新预先训练的模型的权重

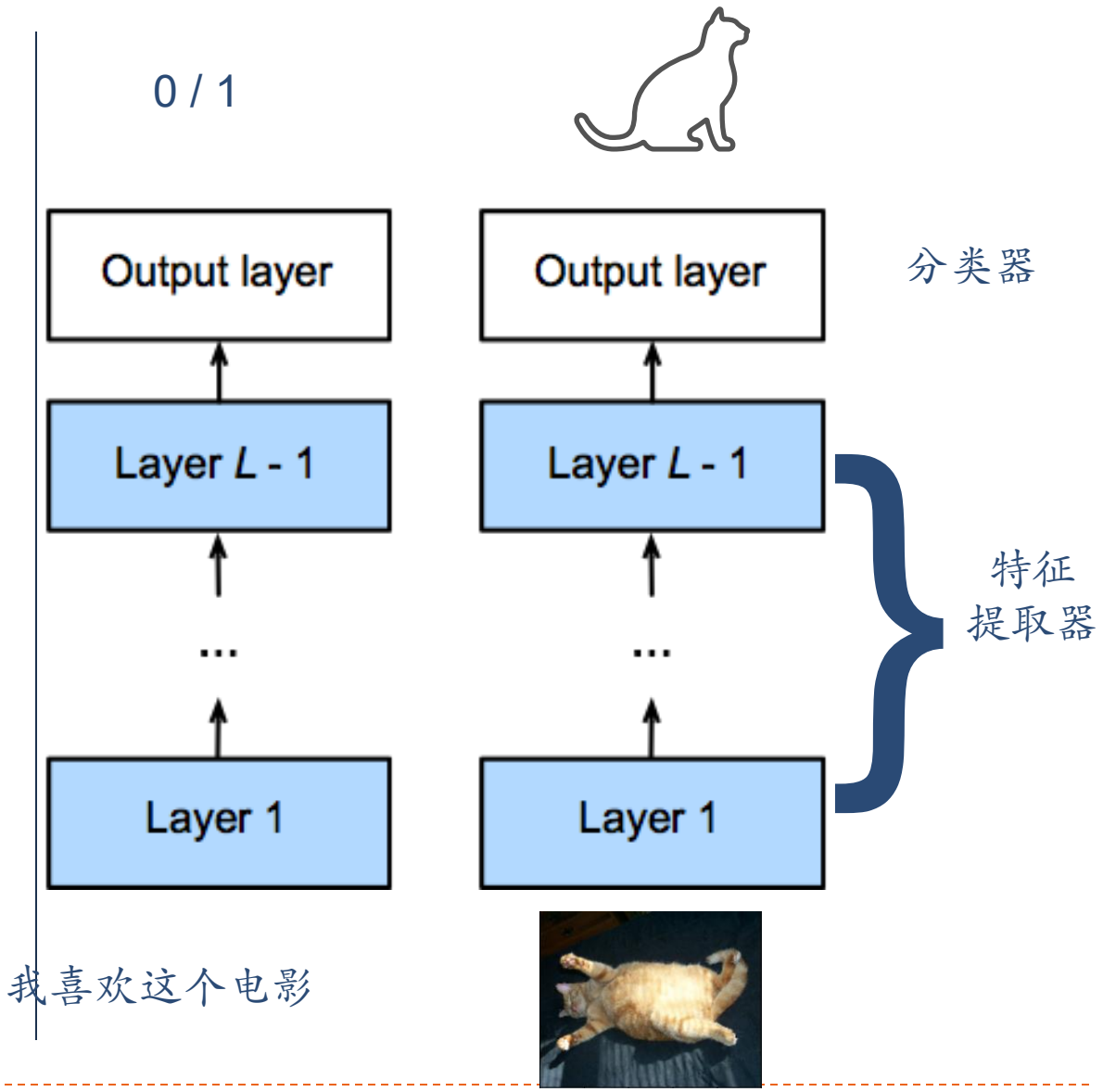
深度学习

- 通过构建具有一定“深度”的模型，可以让模型来自动学习好的特征表示（从底层特征，到中层特征，再到高层特征），从而最终提升预测或识别的准确性。



BERT 的动机

- 基于微调的 NLP 方法
- 预训练模型捕获足够多的数据信息
- 只需要为新任务添加简单的输出层

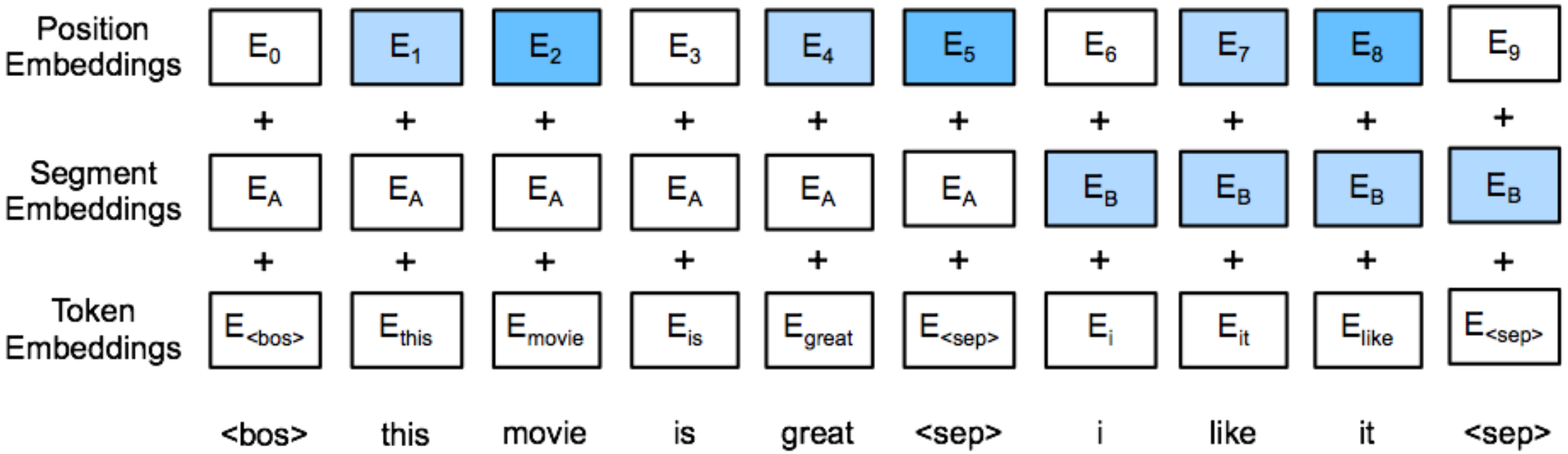


BERT 架构

- 一个（巨大的）Transformer模型编码器（没有解码器）
- 两种模型大小：
  - Base版：＃blocks = 12，隐藏层大小= 768，＃heads = 12，＃参数 = 110M
  - Large版：＃blocks = 24，隐藏层大小= 1024，＃heads = 16，＃参数 = 340M
- 使用超过30亿单词的大型语料库（书籍和维基百科）训练

输入

- 每个样本都是一对句子
- 添加其他细分嵌入



预训练任务1：掩码语言模型

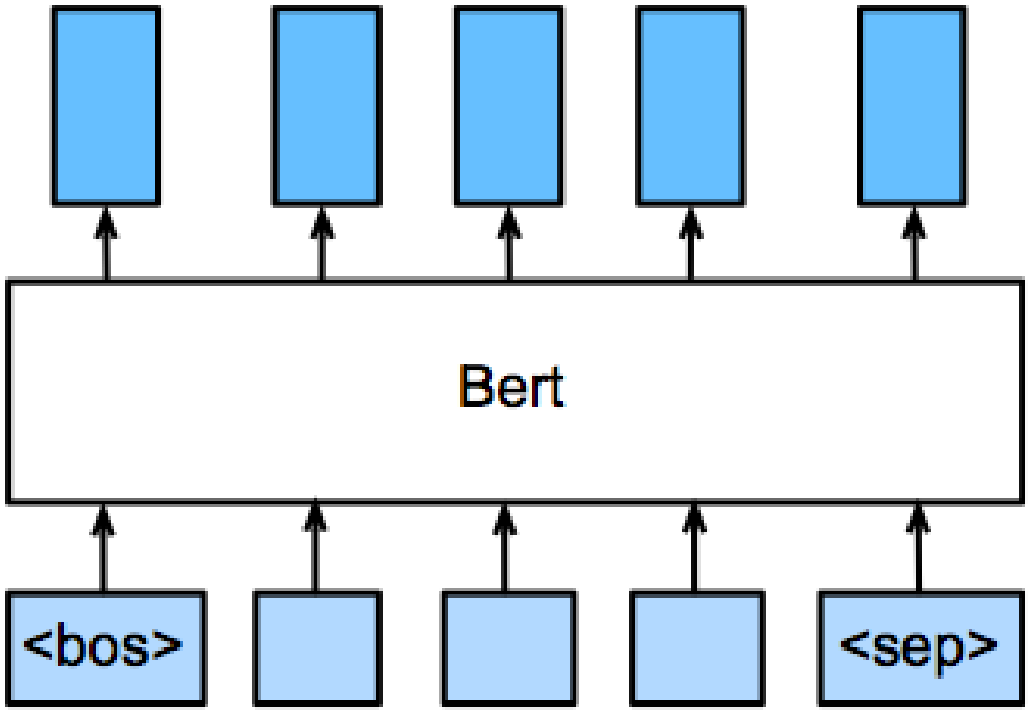
- 在每个句子中随机掩盖（例如15%）标记，预测这些掩码标记（<mask>）
  - Transformer模型是双向的，它打破了标准语言模型的单向限制
- 微调任务中没有掩码标记（<mask>）
  - 80% 的时间，用 <mask> 替换选定的标记
  - 10% 的时间，用随机挑选的picked tokens替换
  - 10% 的时间，保留原始标记

预训练任务2：下一句话预测

- 50% 的时间，选择一个连续的句子对(Positive)
  - <bos> 这部电影很棒 <sep> 我喜欢它 <sep>
- 50% 的时间，选择一个随机的句子对(Negative)
  - <bos> 这部电影很棒 <sep> hello world <sep>
- 将Transformer模型的输出 <bos> 输入到稠密层以预测它是否是顺序对（sequential pair）

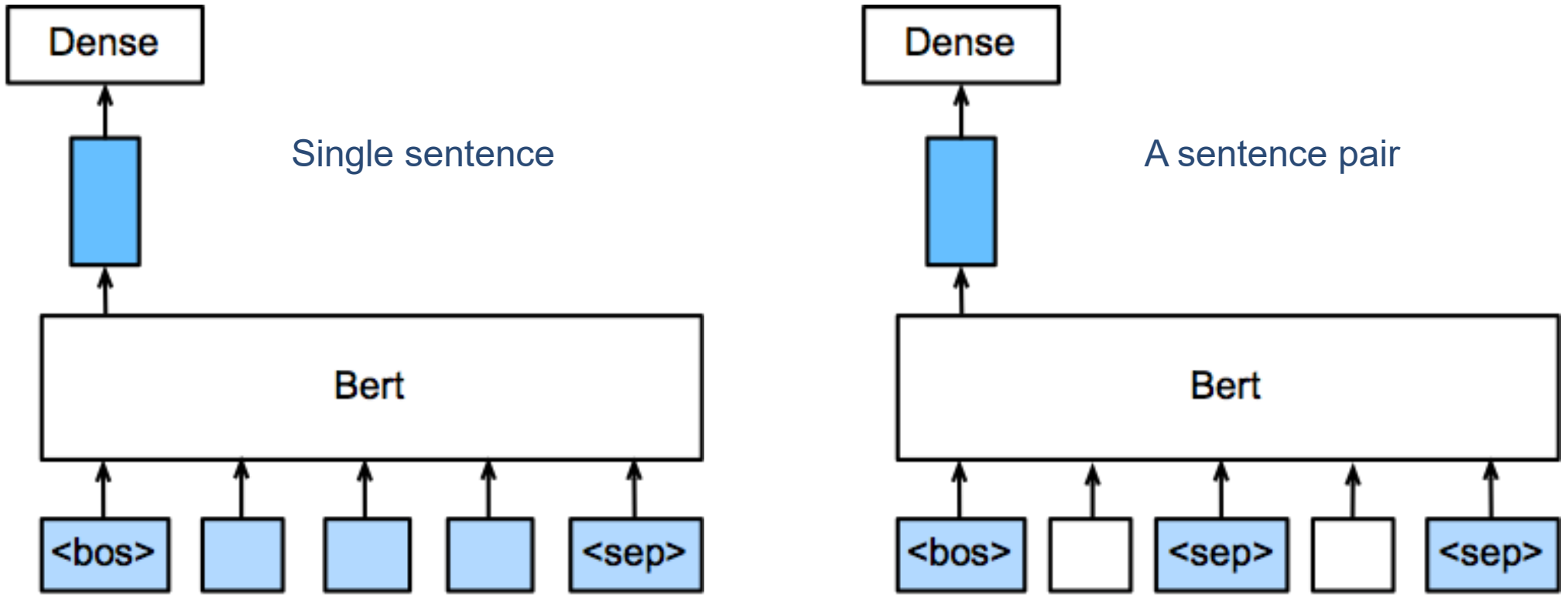
用 Bert 微调

- Bert 为捕获上下文信息的每个标记返回一个特征向量
- 不同的微调任务使用不同的向量集



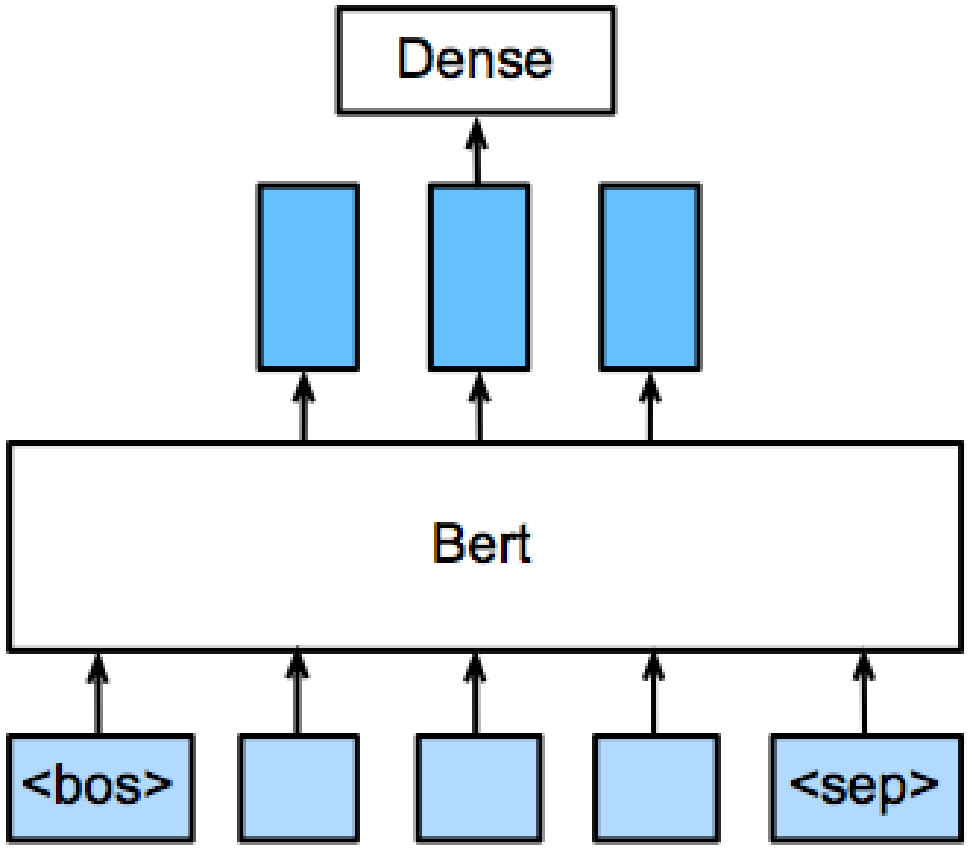
语句分类

- 将 <bos> 标记向量输入稠密输出层



命名实体识别

- 确定标记是否是命名实体，例如人员，组织和位置等等
- 将每个非特殊标记向量馈送到稠密输出层



自动问答

- 给定问题和描述文本，找到答案，这是描述中的文本段
- 给定 p_i，描述中的第 i 个标记，学习 $\mathbf{s}$ 中的 p_i，第 i 个标记是这段开始的概率：

$$p_1, \ldots, p_T = \text{softmax}(\langle \mathbf{s}, \mathbf{v}_1 \rangle, \ldots, \langle \mathbf{s}, \mathbf{v}_T \rangle)$$

- 同样可以学习第 i 个标记是这段结局的概率