

预训练模型

from word2vec to bert

One-Hot

为了得到索引为 i 的任意词的one-hot向量表示，我们创建了一个全为0的长度为 N 的向量，并将位置 i 的元素设置为1。这样，每个词都被表示为一个长度为 N 的向量，可以直接由神经网络使用。

由于任意两个不同词的one-hot向量之间的余弦相似度为0，所以one-hot向量不能编码词之间的相似性。

$$\frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|} \in [-1, 1].$$

$$W = \begin{bmatrix} 1 & \dots & \dots & \dots & 0 \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0 & \dots & 1 & \dots & 0 \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0 & \dots & \dots & \dots & 1 \end{bmatrix}$$

Window-based Co-occurrence Matrix

建立起窗口大小为1的共生矩阵，矩阵中数字代表两个单词出现在同一个窗口的频率，通过建立共生矩阵就可以很好的计算两个单词的相似性了。

Corpus: I like deep learning. I like NLP. I enjoy flying.

Dictionary: ['I', 'like', 'enjoy', 'deep', 'learning', 'NLP', 'flying', '.']

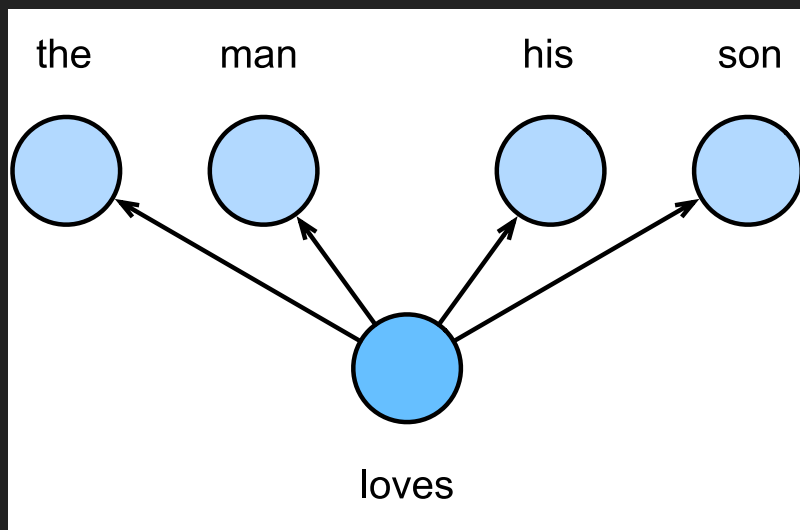
counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Self-supervised word2vec

word2vec工具是为了解决上述问题而提出的。它将每个词映射到一个固定长度的向量，这些向量能更好地表达不同词之间的相似性和类比关系。word2vec工具包含两个模型，即跳元模型（**skip-gram**）和连续词袋（**CBOW**）。其后有各种优化版本的词向量出现，比较广泛应用的有**glove**和**fasttext**。

The Skip-Gram Model

$P(\text{"the", "man", "his", "son"} \mid \text{"loves"})$.

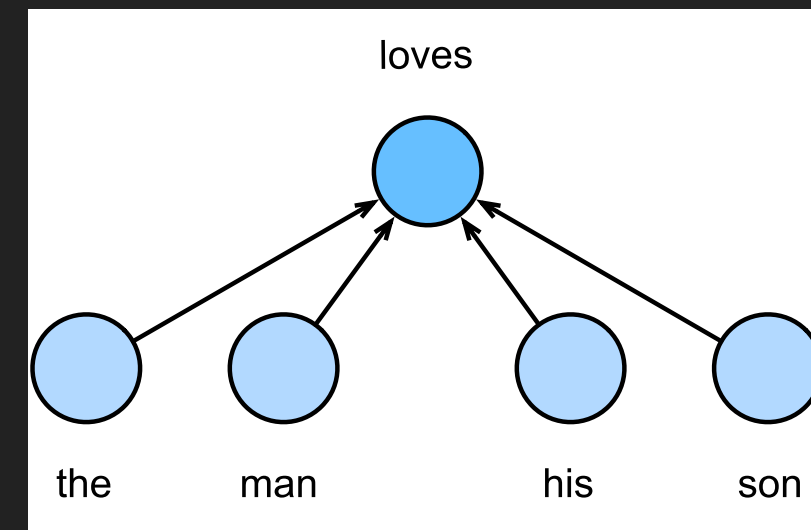


$$P(w_o \mid w_c) = \frac{\exp(\mathbf{u}_o^\top \mathbf{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \mathbf{v}_c)}, v_i \in \mathbb{R}^d \& u_i \in \mathbb{R}^d$$

$$\arg \max \prod_{t=1}^T \prod_{-m \leq j \leq m, j \neq 0} P(w^{(t+j)} \mid w^{(t)})$$

The CBOW Model

$P(\text{"loves"} \mid \text{"the", "man", "his", "son"})$.



$$P(w_c \mid w_{o_1}, \dots, w_{o_{2m}}) = \frac{\exp(\frac{1}{2m} \mathbf{u}_c^\top (\mathbf{v}_{o_1} + \dots + \mathbf{v}_{o_{2m}}))}{\sum_{i \in \mathcal{V}} \exp(\frac{1}{2m} \mathbf{u}_i^\top (\mathbf{v}_{o_1} + \dots + \mathbf{v}_{o_{2m}}))}, v_i \in \mathbb{R}^d \& u_i \in \mathbb{R}^d$$

$$\arg \max \prod_{t=1}^T P(w^{(t)} \mid w^{(t-m)}, \dots, w^{(t-1)}, w^{(t+1)}, \dots, w^{(t+m)})$$

Self-supervised word2vec

The Skip-Gram Model

$$\text{loss} = - \sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log P(w^{(t+j)} \mid w^{(t)})$$

$$\log P(w_o \mid w_c) = \mathbf{u}_o^\top \mathbf{v}_c - \log \left(\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \mathbf{v}_c) \right)$$

$$\begin{aligned} \frac{\partial \log P(w_o \mid w_c)}{\partial \mathbf{v}_c} &= \mathbf{u}_o - \frac{\sum_{j \in \mathcal{V}} \exp(\mathbf{u}_j^\top \mathbf{v}_c) \mathbf{u}_j}{\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \mathbf{v}_c)} \\ &= \mathbf{u}_o - \sum_{j \in \mathcal{V}} \left(\frac{\exp(\mathbf{u}_j^\top \mathbf{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \mathbf{v}_c)} \right) \mathbf{u}_j \\ &= \mathbf{u}_o - \sum_{j \in \mathcal{V}} P(w_j \mid w_c) \mathbf{u}_j \end{aligned}$$

对词典中索引为 i 的词进行训练后，得到 v_i （作为中心词）和 u_i （作为上下文词）两个词向量。在自然语言处理应用中，跳元模型的中心词向量通常用作词表示。

The CBOW Model

$$\text{loss} = - \sum_{t=1}^T \log P(w^{(t)} \mid w^{(t-m)}, \dots, w^{(t-1)}, w^{(t+1)}, \dots, w^{(t+m)})$$

$$\log P(w_c \mid \mathcal{W}_o) = \mathbf{u}_c^\top \bar{\mathbf{v}}_o - \log \left(\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \bar{\mathbf{v}}_o) \right)$$

$$\begin{aligned} \frac{\partial \log P(w_c \mid \mathcal{W}_o)}{\partial \mathbf{v}_{o_i}} &= \frac{1}{2m} \left(\mathbf{u}_c - \sum_{j \in \mathcal{V}} \frac{\exp(\mathbf{u}_j^\top \bar{\mathbf{v}}_o) \mathbf{u}_j}{\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \bar{\mathbf{v}}_o)} \right) \\ &= \frac{1}{2m} \left(\mathbf{u}_c - \sum_{j \in \mathcal{V}} P(w_j \mid \mathcal{W}_o) \mathbf{u}_j \right) \end{aligned}$$

其他词向量的梯度可以以相同的方式获得。与跳元模型不同，连续词袋模型通常使用上下文词向量作为词表示。

Approximate Training

Negative sampling

负采样修改了原目标函数。给定中心词 w_c 的上下文窗口，任意上下文词 w_o 来自该上下文窗口的被认为是由下式建模概率的事件：

$$P(D = 1 \mid w_c, w_o) = \sigma(\mathbf{u}_o^\top \mathbf{v}_c)$$

用 S 表示上下文词 w_o 来自中心词 w_c 的上下文窗口的事件。对于这个涉及 w_o 的事件，从预定义分布 $P(w)$ 中采样 K 个不是来自这个上下文窗口噪声词。用 N_k 表示噪声词 w_k ($k = 1, \dots, K$)不是来自 w_c 的上下文窗口的事件。

$$\prod_{t=1}^T \prod_{-m \leq j \leq m, j \neq 0} P(w^{(t+j)} \mid w^{(t)})$$

$$P(w^{(t+j)} \mid w^{(t)}) = P(D = 1 \mid w^{(t)}, w^{(t+j)}) \prod_{k=1, w_k \sim P(w)}^K P(D = 0 \mid w^{(t)}, w_k)$$

Approximate Training

Negative sampling

$$\begin{aligned}l &= -\log P(w^{(t+j)} \mid w^{(t)}) = -\log P(D = 1 \mid w^{(t)}, w^{(t+j)}) - \sum_{k=1, w_k \sim P(w)}^K \log P(D = 0 \mid w^{(t)}, w_k) \\&= -\log \sigma \left(\mathbf{u}_{i_{t+j}}^\top \mathbf{v}_{i_t} \right) - \sum_{k=1, w_k \sim P(w)}^K \log \left(1 - \sigma \left(\mathbf{u}_{h_k}^\top \mathbf{v}_{i_t} \right) \right) \\&= -\log \sigma \left(\mathbf{u}_{i_{t+j}}^\top \mathbf{v}_{i_t} \right) - \sum_{k=1, w_k \sim P(w)}^K \log \sigma \left(-\mathbf{u}_{h_k}^\top \mathbf{v}_{i_t} \right).\end{aligned}$$

$$\frac{\partial l}{\partial v_{i_t}} = -\sigma \left(-\mathbf{u}_{i_{t+j}}^\top \mathbf{v}_{i_t} \right) \mathbf{u}_{i_{t+j}} + \sum_{k=1, w_k \sim P(w)}^K \sigma \left(\mathbf{u}_{h_k}^\top \mathbf{v}_{i_t} \right) \mathbf{u}_{h_k}$$

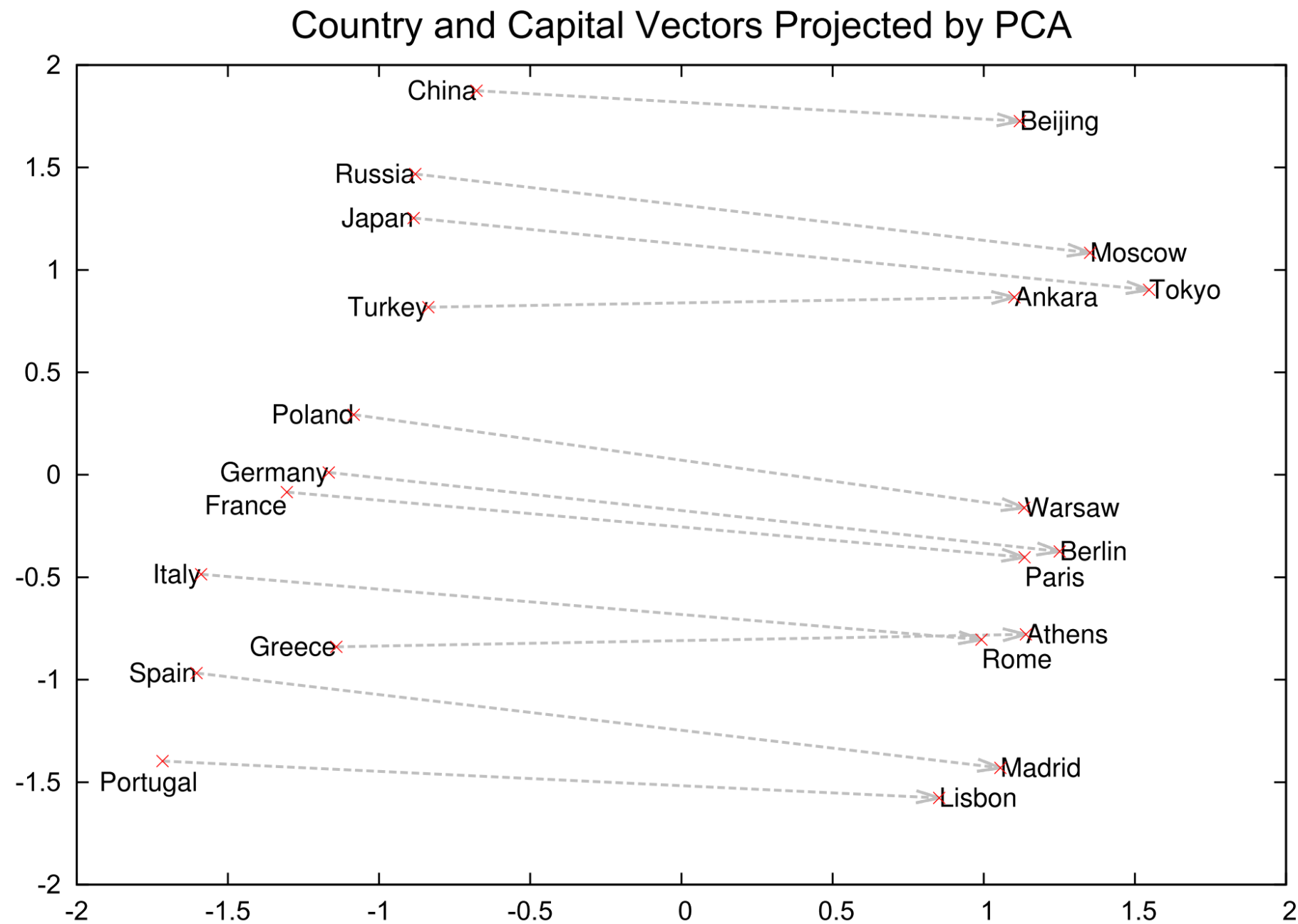
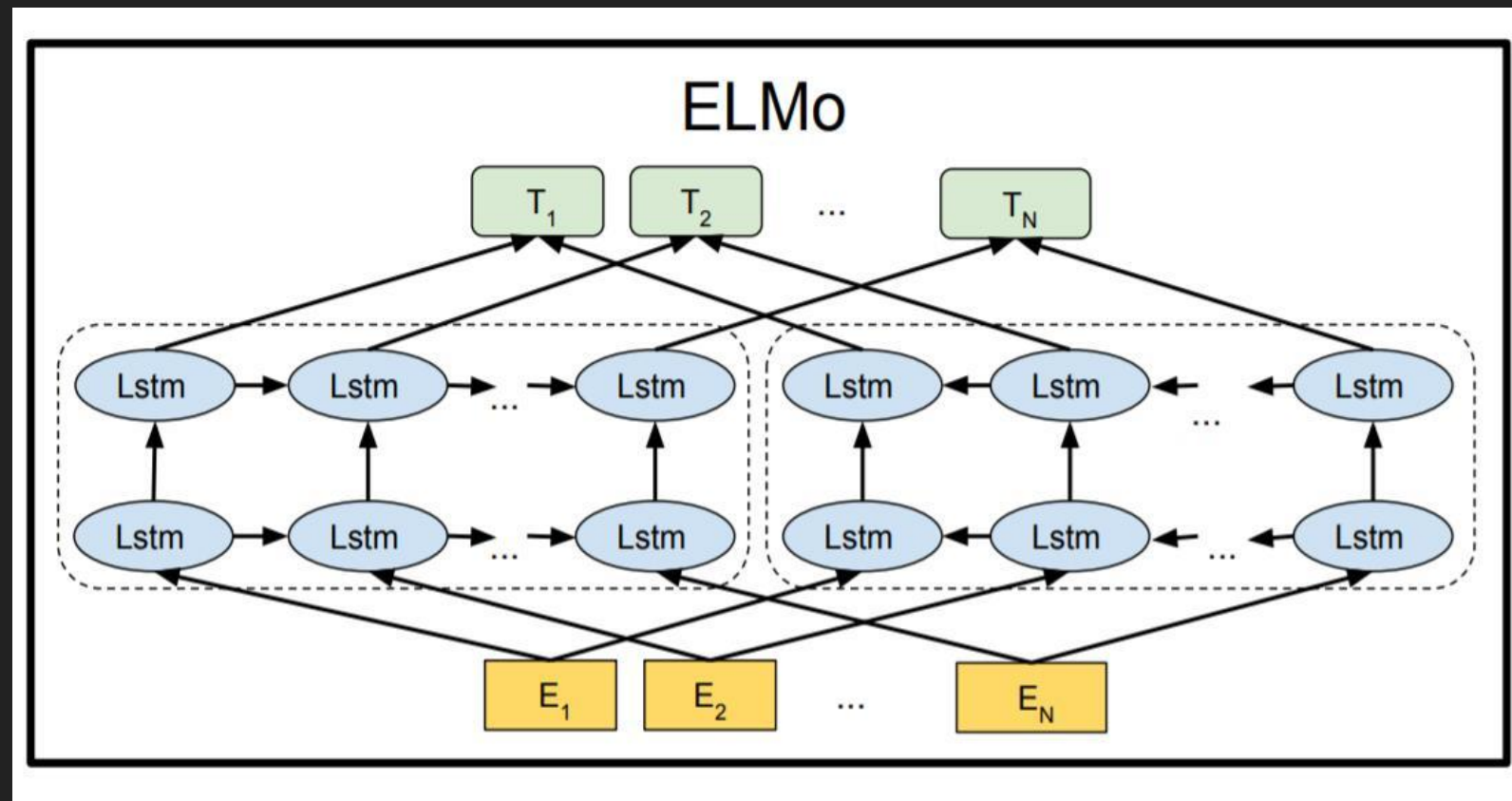


Figure 2: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. The figure illustrates ability of the model to automatically organize concepts and learn implicitly the relationships between them, as during the training we did not provide any supervised information about what a capital city means.

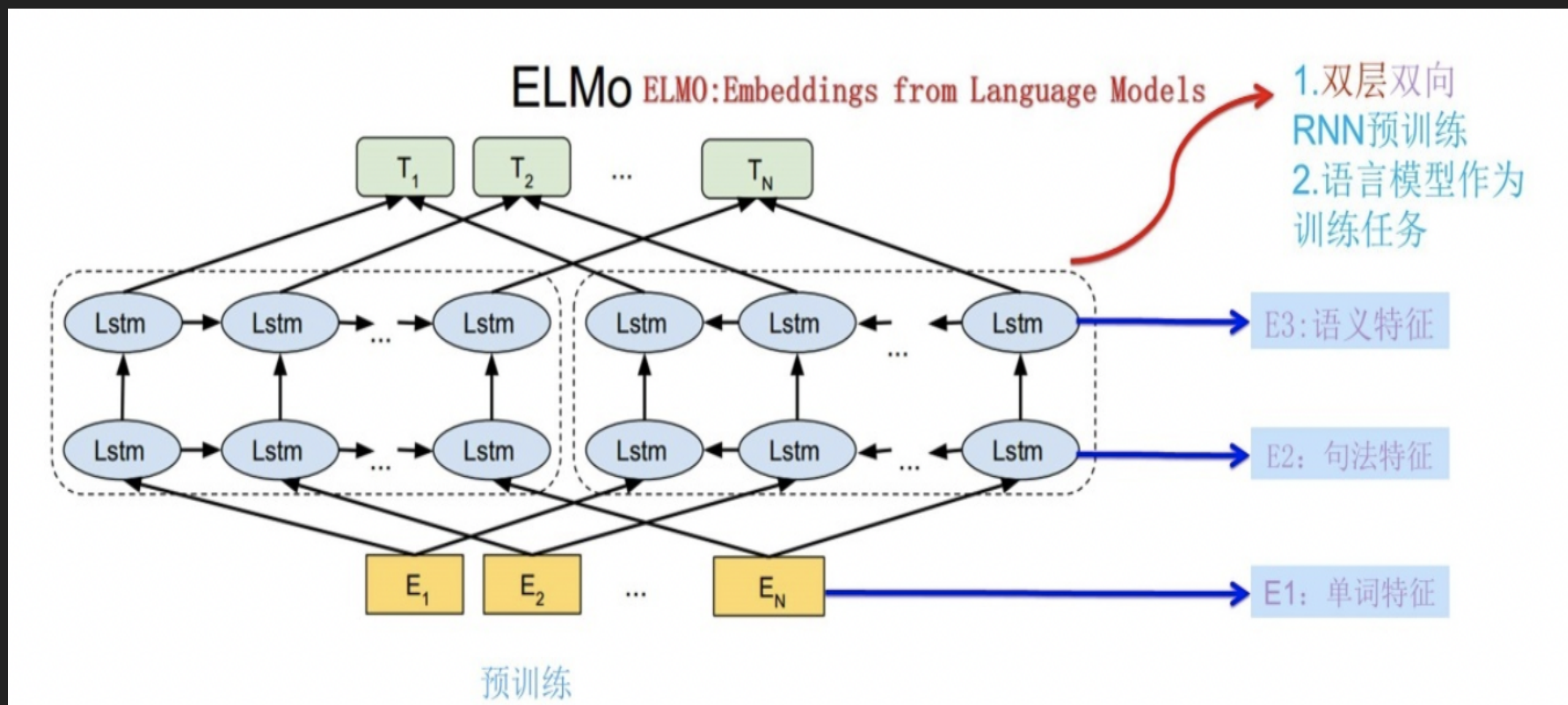
ELMo

word2vec和GloVe都将相同的预训练向量分配给同一个词，而不考虑词的上下文（如果有的话）。单身人的来由：原来是喜欢一个人；现在是喜欢一个人。“一个人”一词有完全不同的含义；因此，同一个词可以根据上下文被赋予不同的表示。



$$\sum_{k=1}^N (\log p(t_k | t_1, \dots, t_{k-1}; \Theta_x, \vec{\Theta}_{LSTM}, \Theta_s) + \log p(t_k | t_{k+1}, \dots, t_N; \overleftarrow{\Theta}_{LSTM}, \Theta_s))$$

ELMo



Attention Mechanisms

Using the nonvolitional cue

直播

换一换

查看更多 >



◎ 837 赛事综合

直播中

乔氏杯中式台球国际大师赛

UP 台球圈有料



◎ 4122 放松电台

直播中

点关注送亲亲哦

UP 保护小C



◎ 9358 放松电台

直播中

【顶级设备】限时70上...

UP 奶宝儿的超酥甜



◎ 11.6万 日常

直播中

你也可以学会的瑜伽

UP 鱼朵瑜伽



◎ 4815 视频唱见

直播中

【唱见】全B站最正经的...

UP 林二药呀



◎ 71 虚拟主播

直播中

进来看看 元气满满 唱歌狐

UP 星海玲奈



◎ 4807 吃鸡行动

直播中

鸡

UP 小霖QL



◎ 9150 职业技能

直播中

一起学习python吧

UP 程序员璐璐

Attention Mechanisms

Using the volitional cue

直播

换一换

查看更多 >



◎ 837 赛事综合

直播中

乔氏杯中式台球国际大师赛

UP 台球圈有料



◎ 4122 放松电台

直播中

点关注送亲亲哦

UP 保护小C



◎ 9358 放松电台

直播中

【顶级设备】限时70上...

UP 奶宝儿的超酥甜



◎ 11.6万 日常

直播中

你也可以学会的瑜伽

UP 鱼朵瑜伽



◎ 4815 视频唱见

直播中

【唱见】全B站最正经的...

UP 林二药呀



◎ 71 虚拟主播

直播中

进来看 元气满满 唱歌狐 兼搞笑艺人

UP 星海玲奈



◎ 4807 吃鸡行动

直播中

鸡

UP 小霖QL



◎ 9150 职业技能

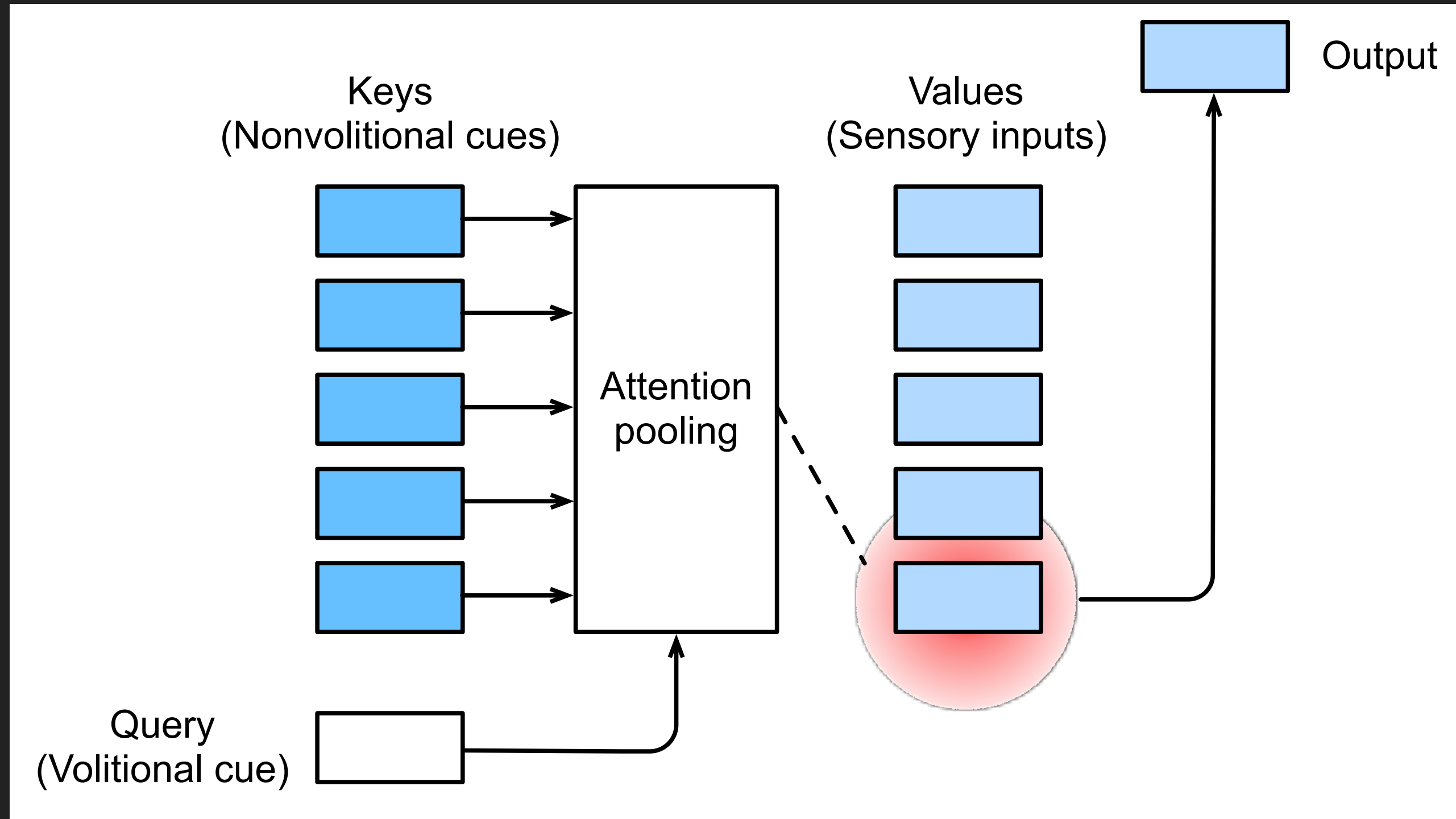
直播中

一起学习python吧

UP 程序员璐璐

Attention Mechanisms

Using the volitional cue

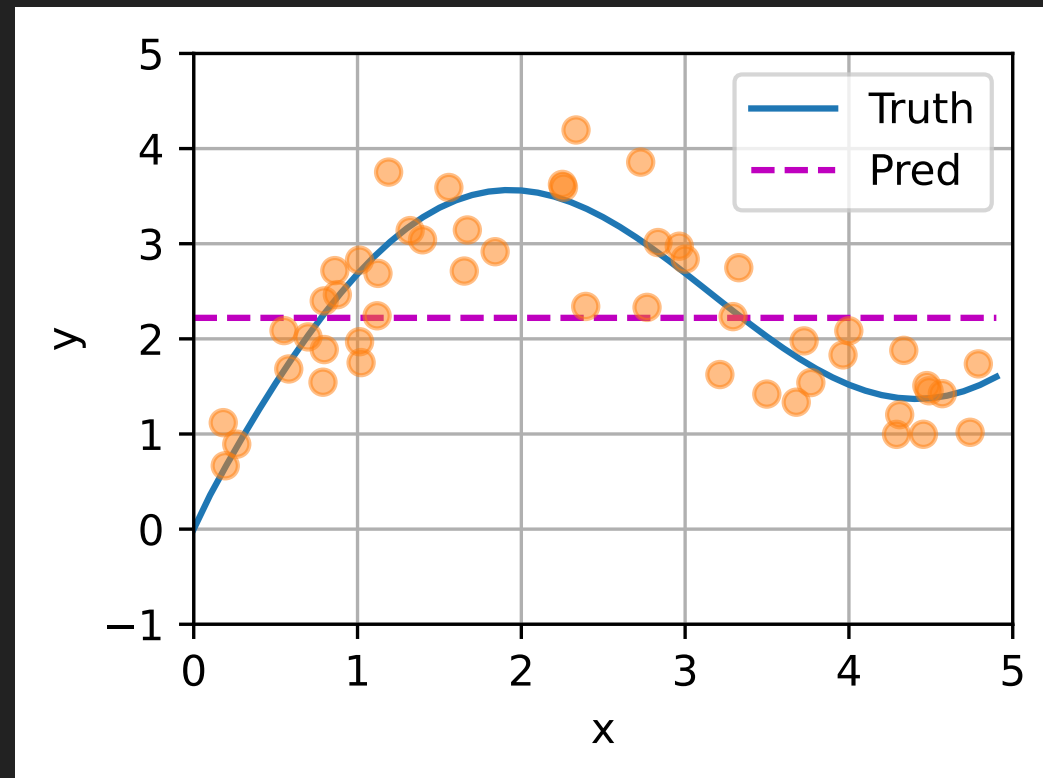


Attention Mechanisms

Attention Pooling **Nadaraya-Watson** Kernel Regression $y_i = 2 \sin(x_i) + x_i^{0.8} + \epsilon$,

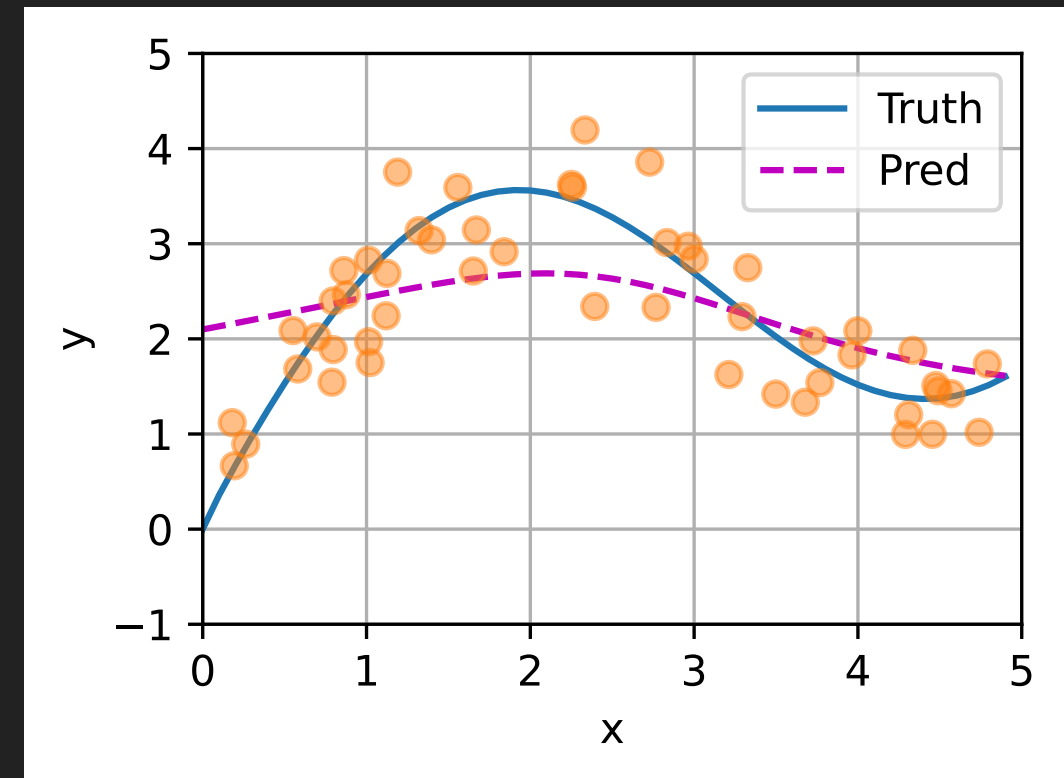
average pooling

$$f(x) = \frac{1}{n} \sum_{i=1}^n y_i$$

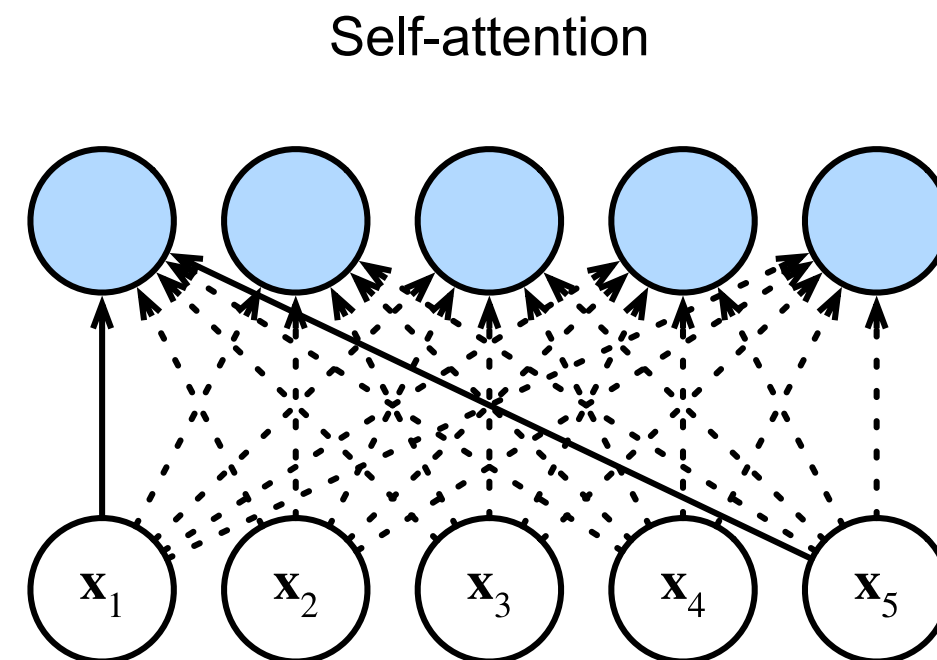
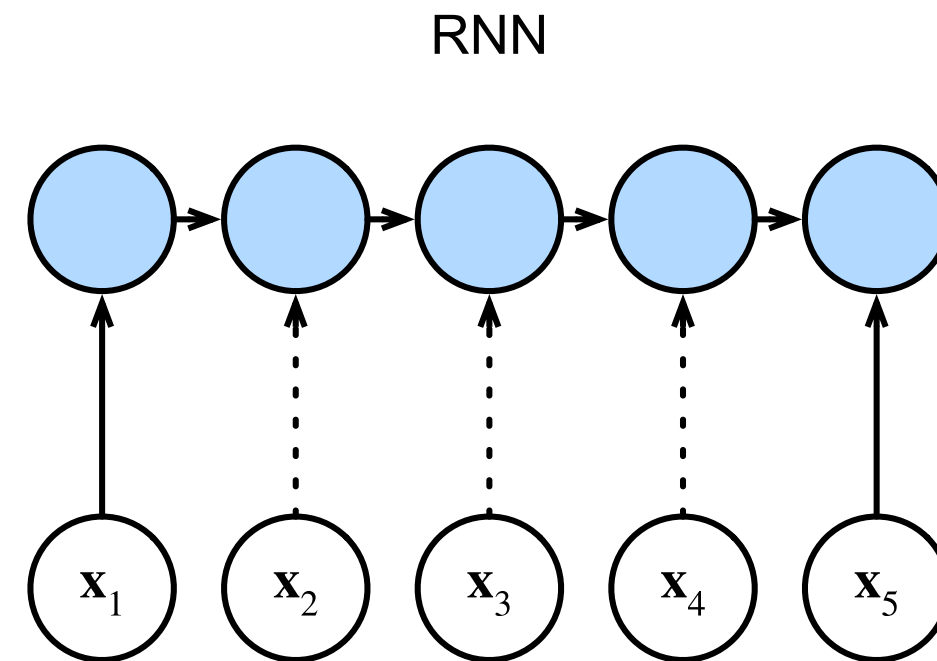
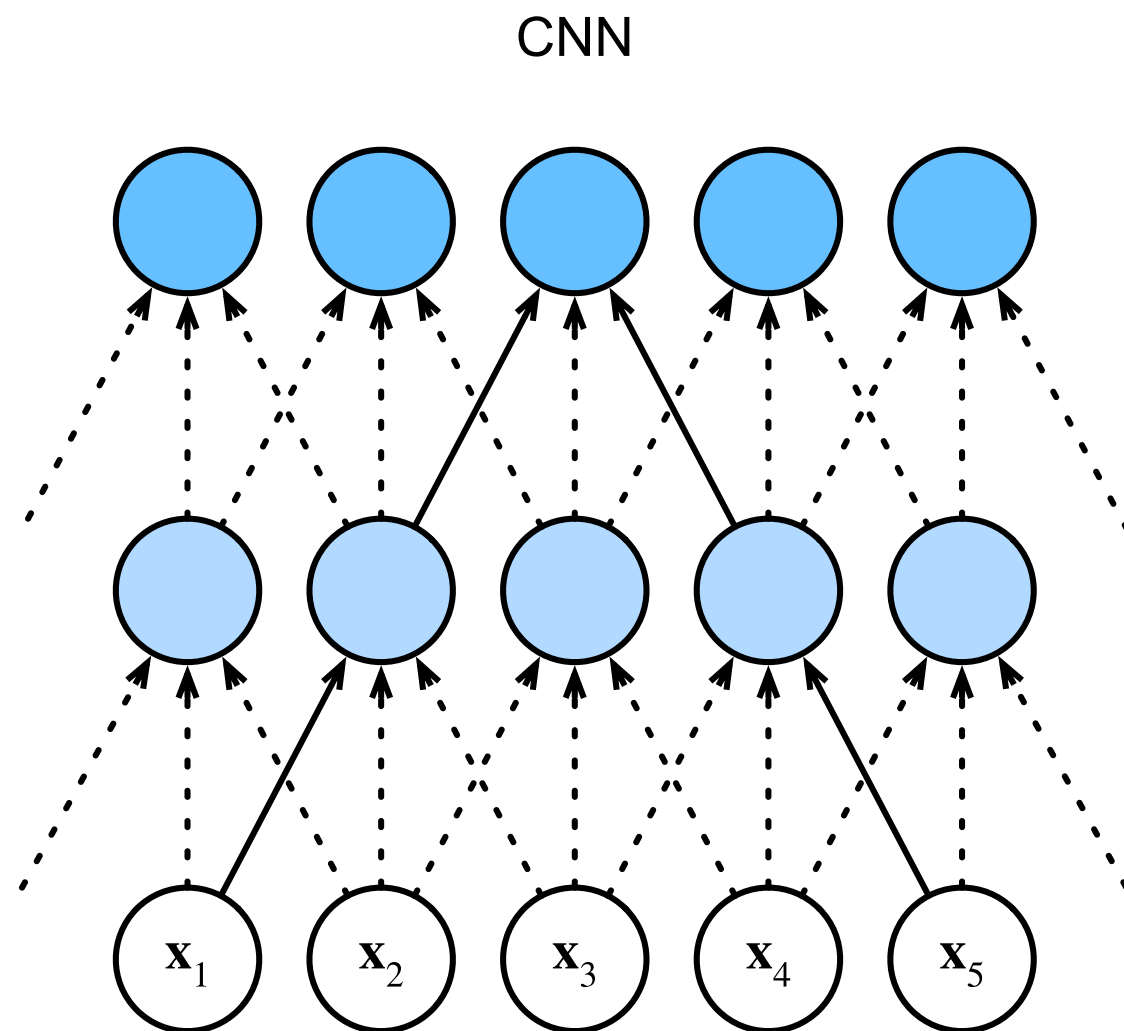


attention pooling

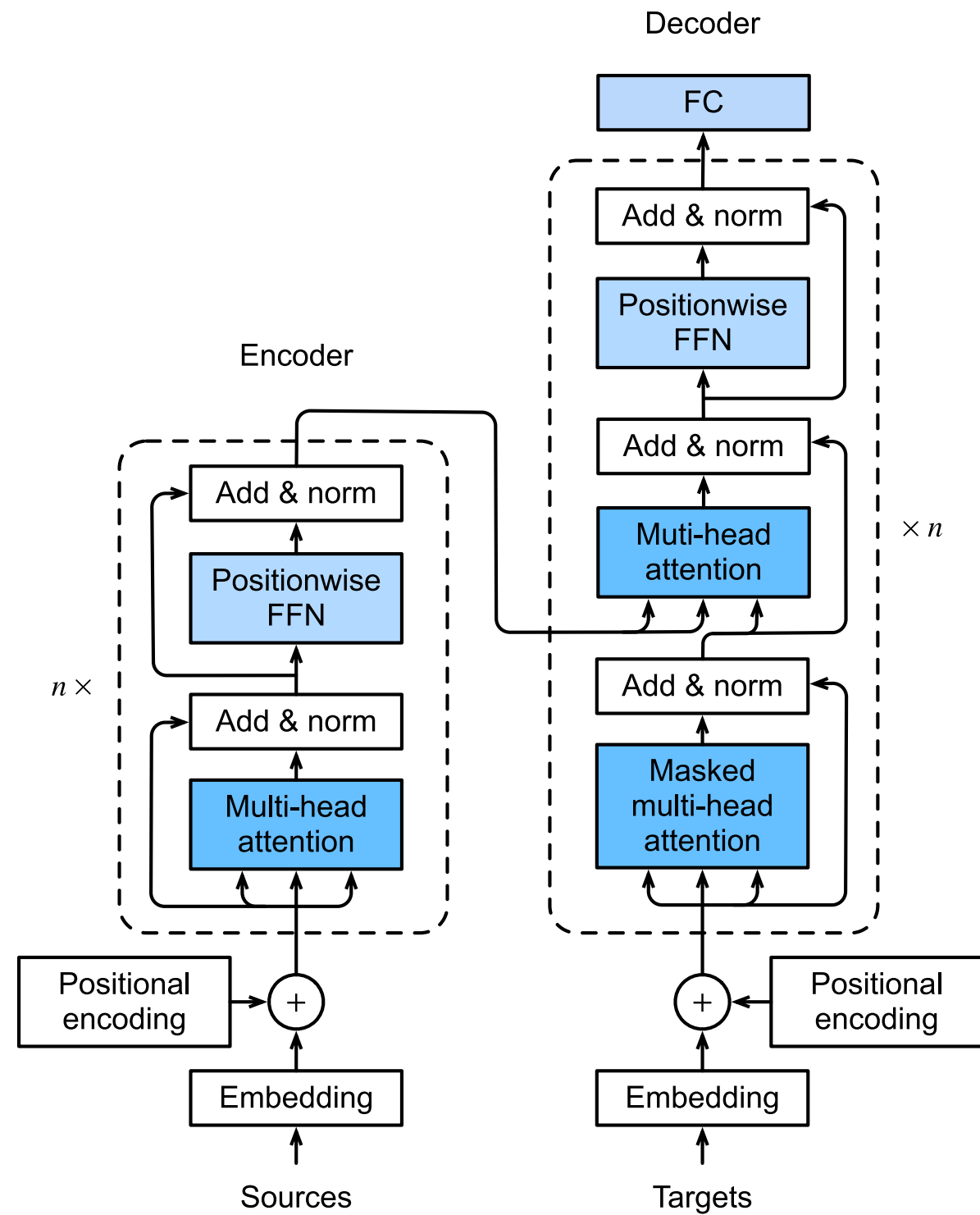
$$f(x) = \sum_{i=1}^n \frac{K(x - x_i)}{\sum_{j=1}^n K(x - x_j)} y_i,$$



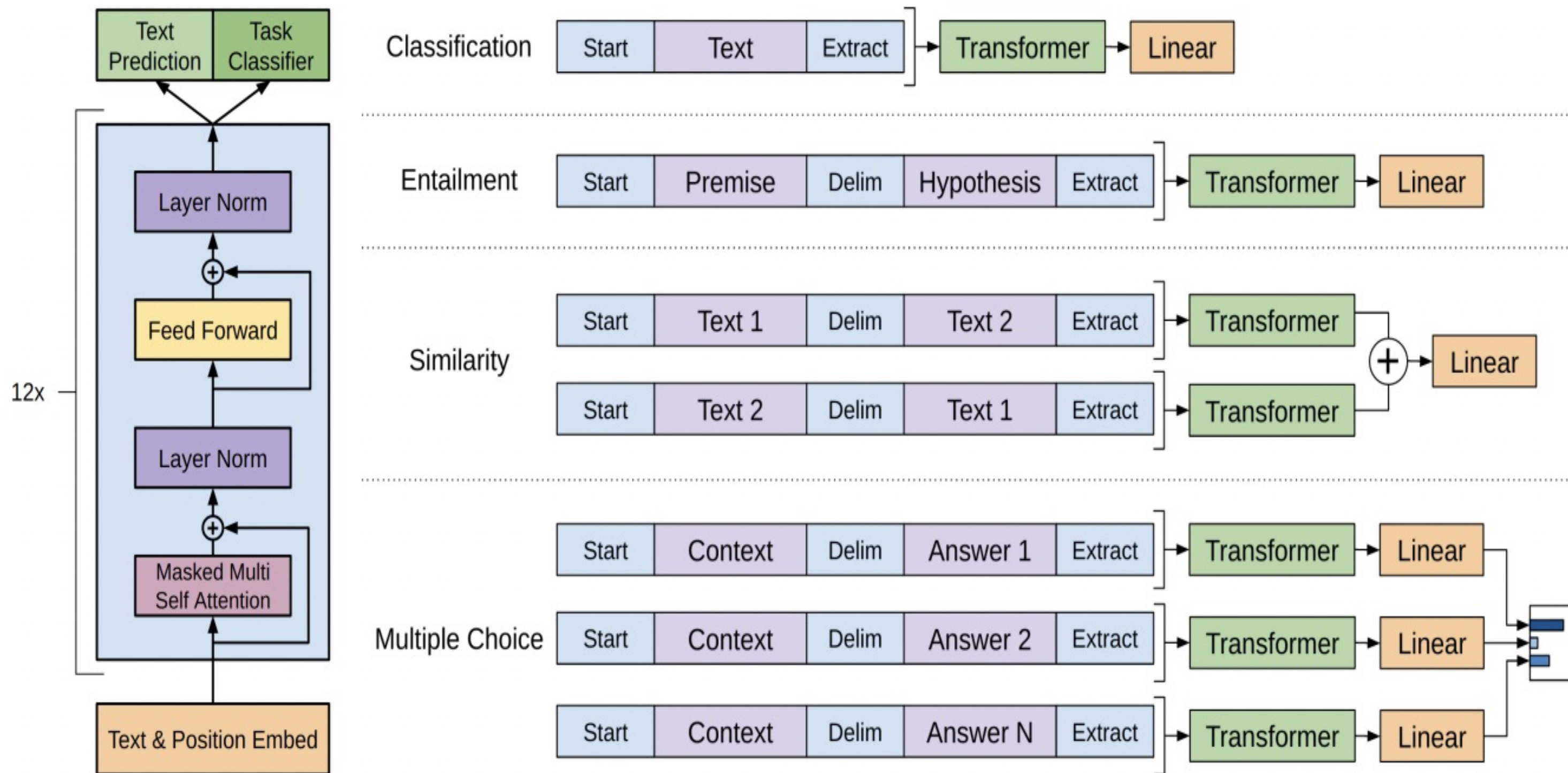
Attention Mechanisms



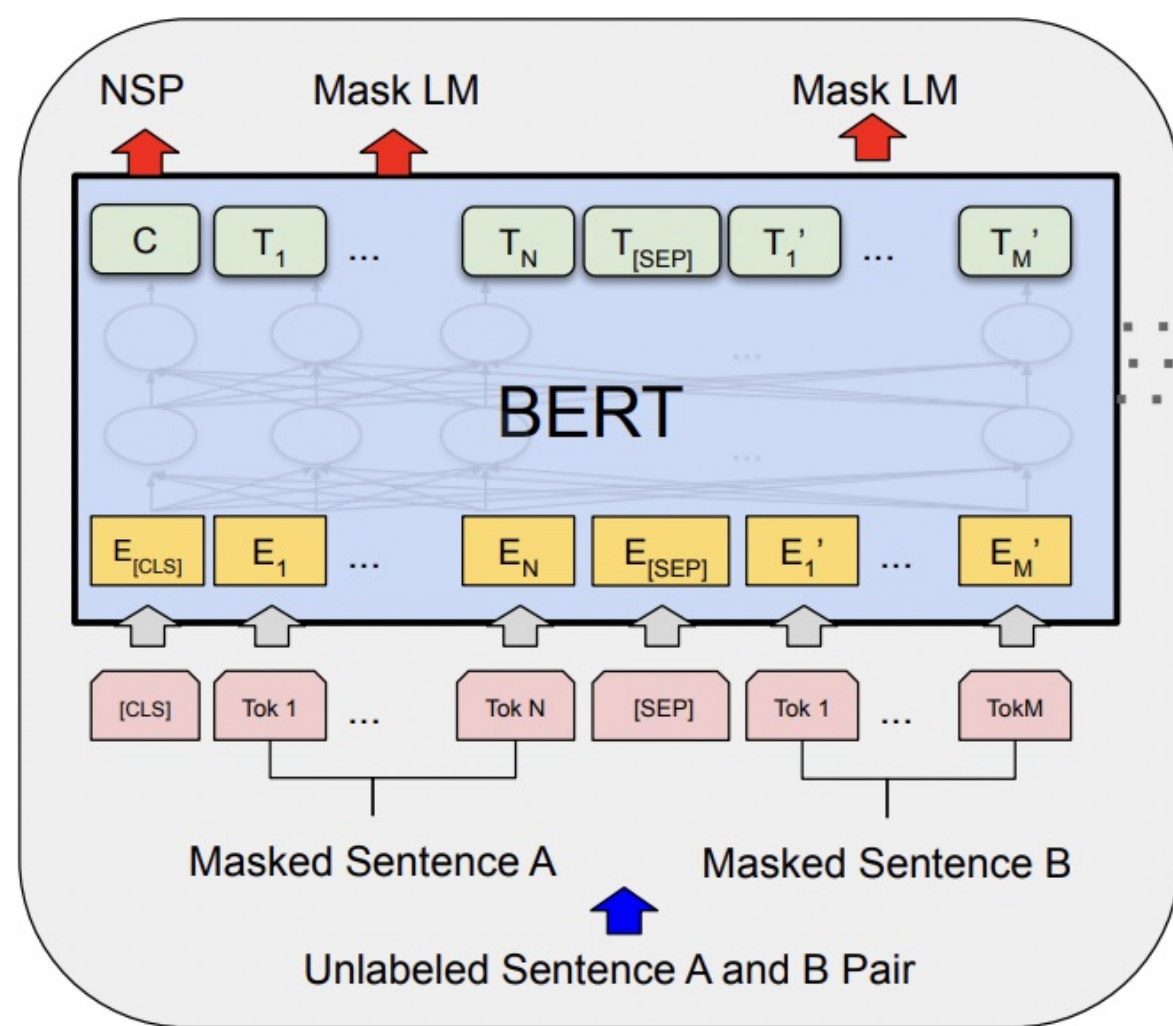
Transformer



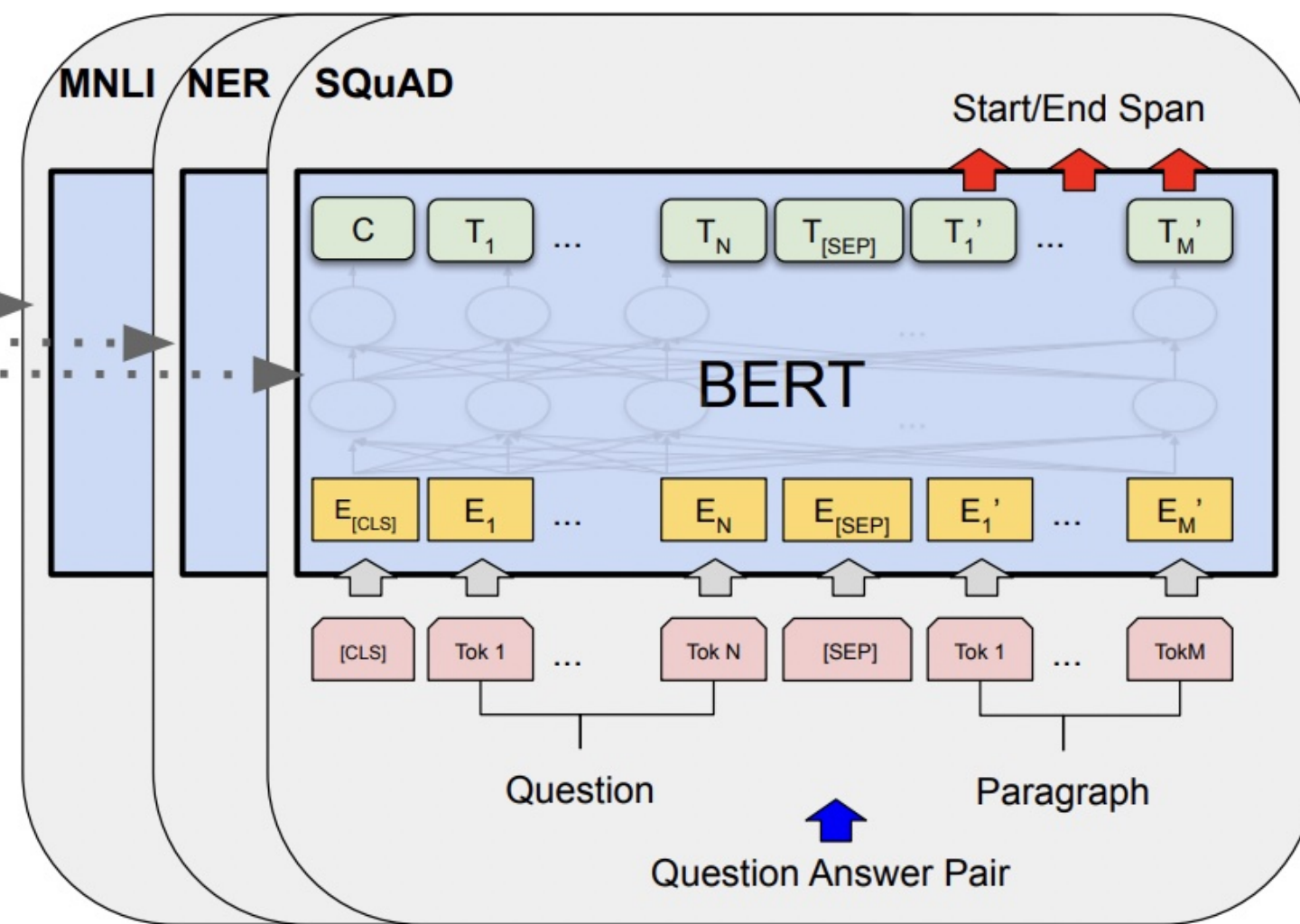
GPT



BERT

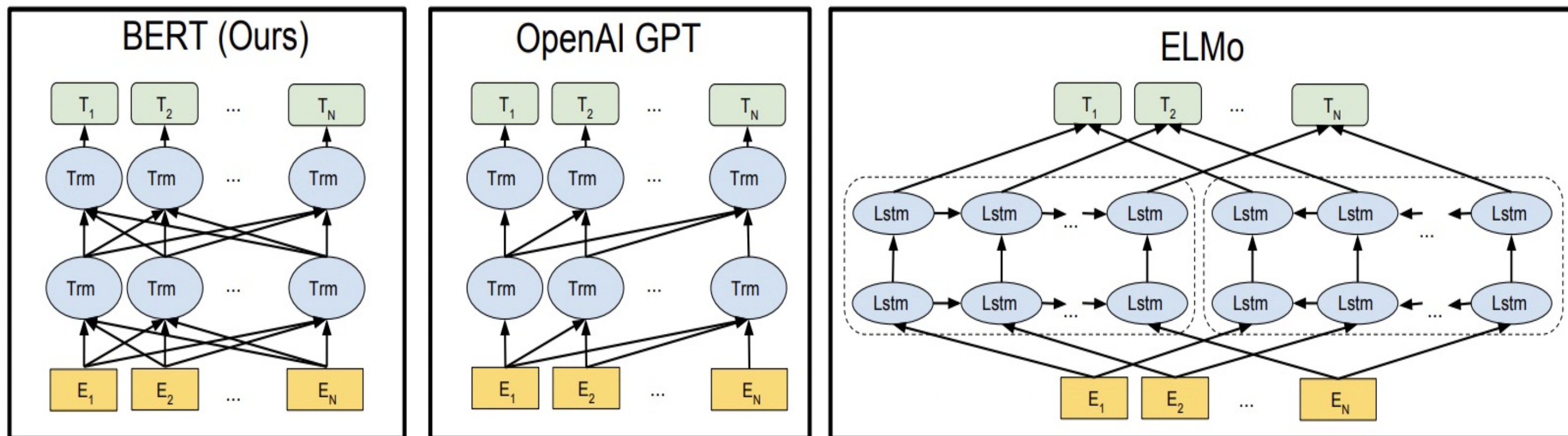


Pre-training



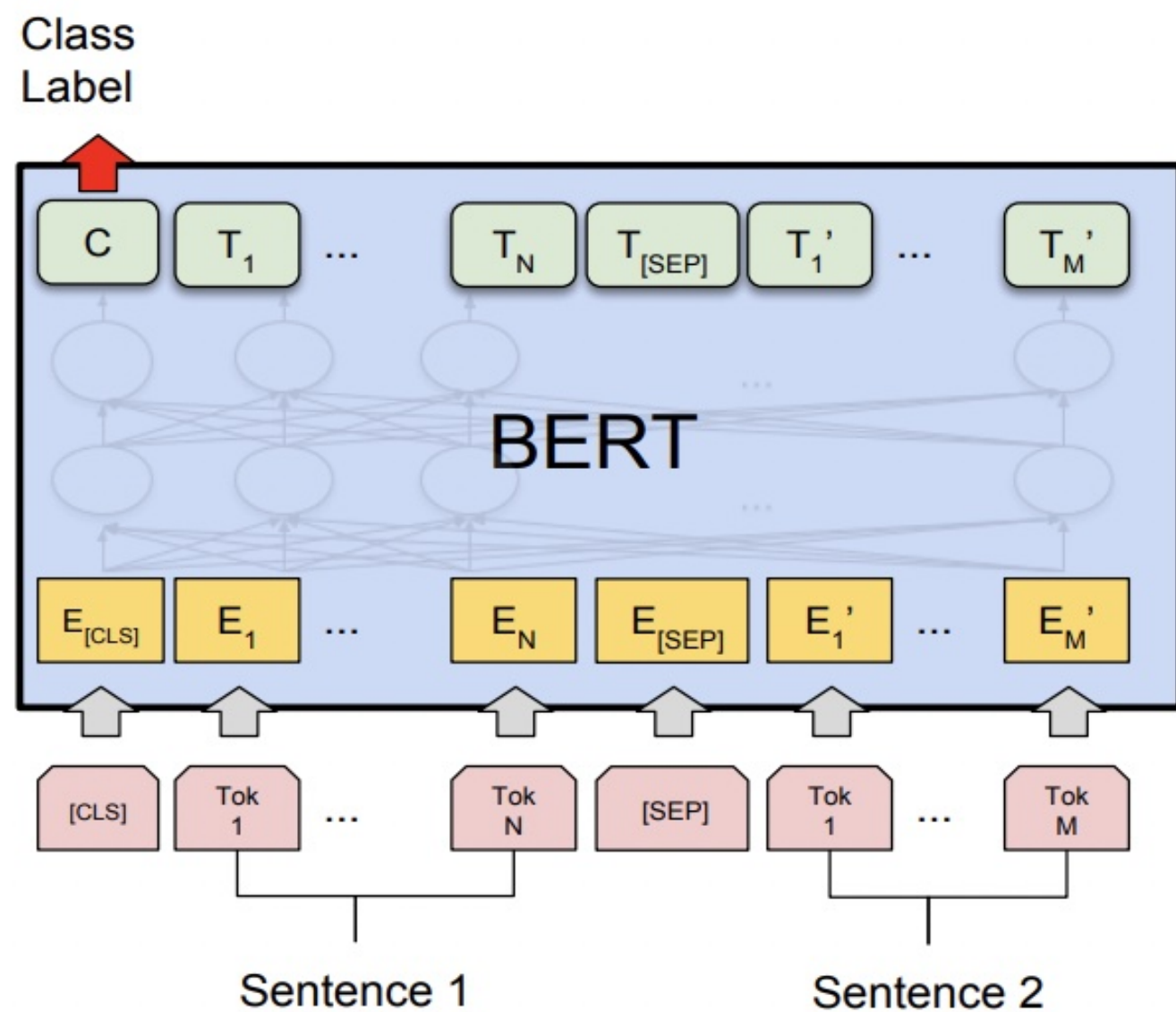
Fine-Tuning

BERT

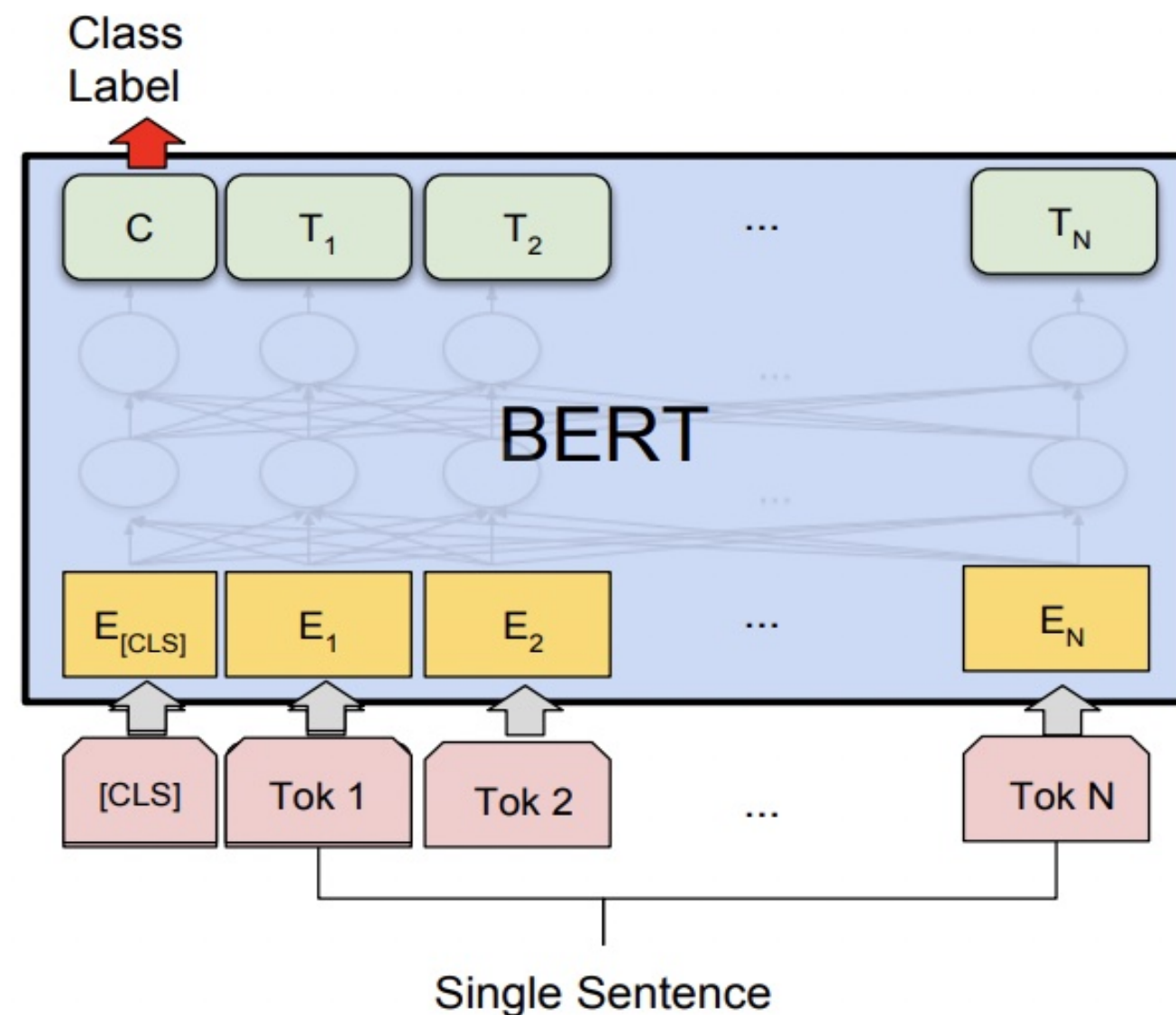


BERT是transformer的encoder部分，GPT是decoder拿掉multi-head以后的部分，把ELMo的LSTM换成transformer就是BERT。回过头，BERT可以看做是deep CBOW。

BERT

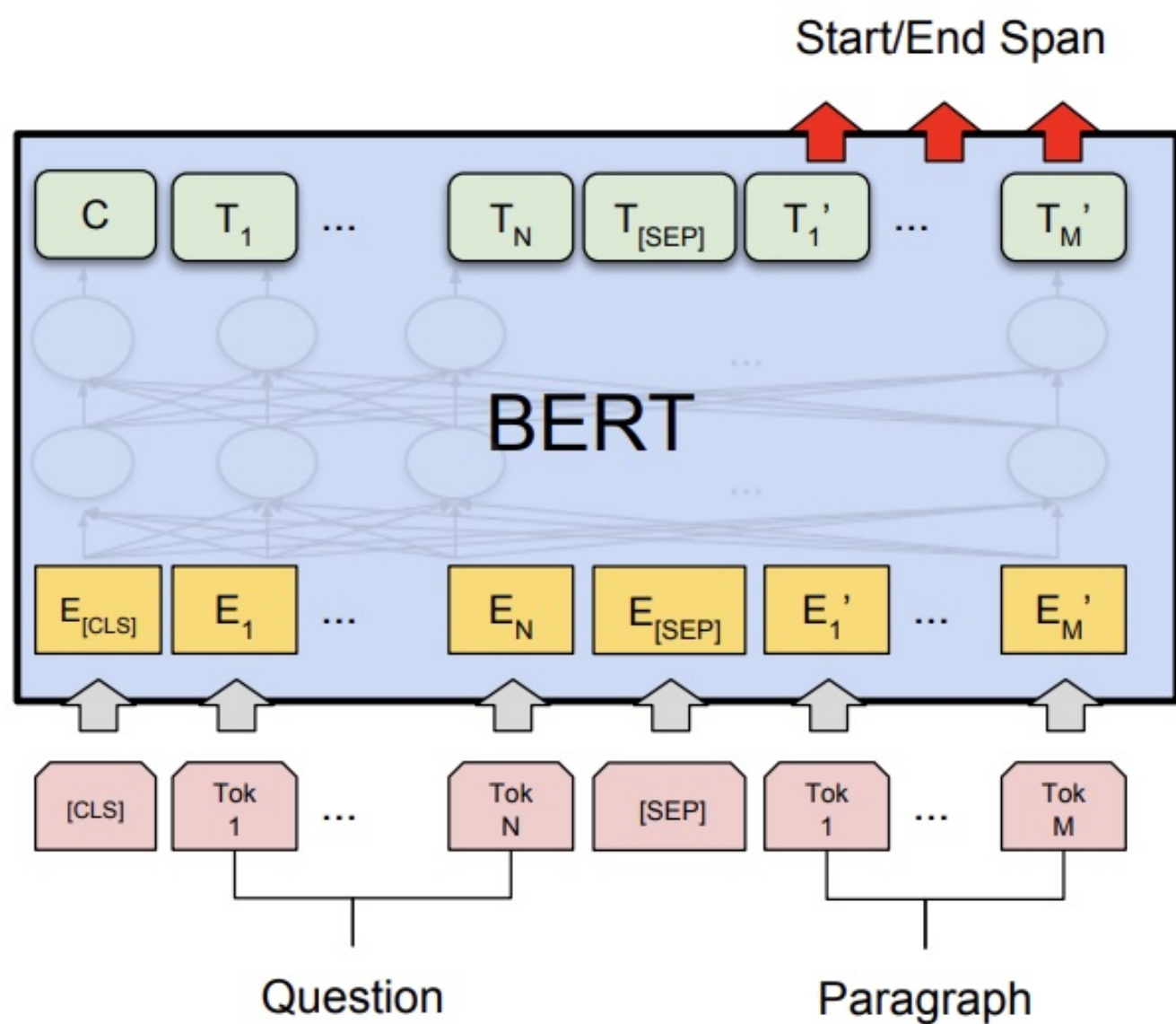


(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG

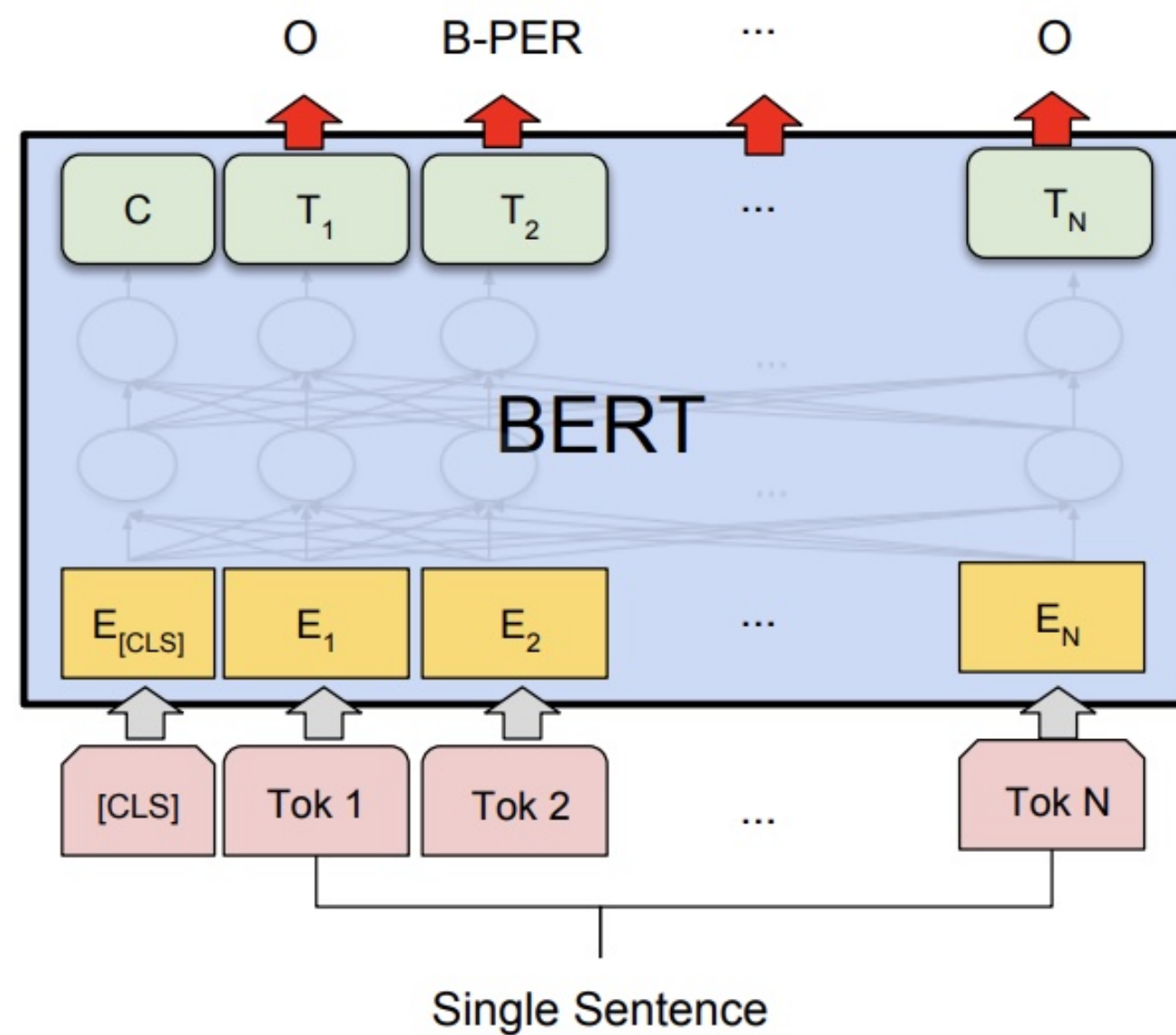


(b) Single Sentence Classification Tasks:
SST-2, CoLA

BERT



(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER