

Cache-based Recurrent Transformer Network

概览

对于时间步 t ，输入片段（向量）为 $X_t \in R^{L \times d}$ ，Memory 为 $\mathcal{M} = \{K_i, V_i\}_{i=1}^N$ ，网络的更新步骤大致分为

查询

查询应该符合如下原则：

- 对于片段中的每个词，应有一个查询结果。
- 对于每个词的查询，这个词之后的词应该每 **mask** 住。
- 对于片段的第一个词，后面的词都是看不见的，查询时信息太少，应该拼接上一个片段 X_{t-1} 参与查询
- 从大小为 N 的缓存中查询中 k 个片段

$$\{\alpha_i, Z_i\}_k^{(j)} = \text{Query}(X_{t-1}, \text{Masked}_j(X_t), \mathcal{M}) \quad (1)$$

其中 Z_i 为需要进行回忆（以 Transformer Memory 的方式拼接进当前片段）的 k 个片段， α_i 为其对应的权重， j 表示计算第 j 个词对应的查询结果， $j = 1, 2, \dots, L$
 \mathcal{M} 是 Memory，存储 N 个 Key-Value 对，Value 即为片段，对应的 Key 为该片段的一个意义向量。

更新 hidden state

使用 Transformer 网络更新 m 层隐状态

$$h_{t,j}^{1:m} = \text{Transformer}(\alpha_{1:k}^{(j)}, Z_{1:k}^{(j)}, X_t) \quad (2)$$

更新 Memory

$$\mathcal{M} = \text{renew}(h_t^{1:m}, \mathcal{M}) \quad (3)$$

1 查询

采取键值对匹配的方法， $Q_t^{(j)}$ 是对应的查询向量，计算方法列下， $Keys$ 是缓存中的键向量。

$$\{\alpha_i, Z_i\}_k^{(j)} = \text{topk}(\text{softmax}(Q_t^{(j)} \cdot Keys^T))$$

以下列出查询向量 $Q_t^{(j)}$ 的计算方法：

- **1.1 fixed_length** 定长方法: $Q_t^{(j)}$ 包含包括第 j 个词在内的第 j 个词之前的 L 个词（事实上已经取到 X_{t-1} 中，此处仍用 X_t 表示）

- **1.1.1 fixed_length_1** 直接使用底层的连续 L 个词，使用 **encoder** 进行计算

$$Q_t^{(j)} = (X_{t,j-L+1}, \dots, X_{t,j-1}, X_{t,j})$$

$$Q_t^{(j)} = \text{summary}(\text{Transformer}(Q_t^{(j)}))$$

- **1.1.2 fixed_length_2** 将 j 个词作为当前片段，上一片段中的后 $L-j$ 个词的表示作为 **memory**

$$Q_t^{(j)} = \text{Transformer}(X_{t-1}[j+1:L], X_t[1:j])$$

$$Q_t^{(j)} = \text{summary}(Q_t^{(j)})$$

其中 **summary** 函数与之后更新 **Memory** 时使用的相同，**Transformer** 共享模型参数

- **1.2 倍长方法**：将 X_{t-1} 的 L 个词拼接入当前输入，使用 $2L$ 个词进行计算

$$Q_t^{(j)} = \text{concat}(X_{t-1}, \text{Masked}_j(X_t))$$

$$Q_t^{(j)} = \text{Transformer}(Q_t^{(j)})$$

注意到此时 $Q_t^{(j)}$ 为 $2L$ 个词的表示，需要将其变成 L 个词的表示，才能使用 **summary** 函数

- **1.2.1 last_l** 截取后 L 个

$$Q_t^{(j)} = \text{summary}(Q_t^{(j)}[L+1:2L])$$

- **1.2.2 middel_l** 截取中间 L 个

$$Q_t^{(j)} = \text{summary}(Q_t^{(j)}[j+1:j+L])$$

- **1.2.3 linear** 线性变换法

$$Q_t^{(j)} = \text{summary}(\text{Linear}(Q_t^{(j)}))$$

其中 **summary** 函数与之后更新 **Memory** 时使用的相同，**Transformer** 共享模型参数

2 更新 hidden state

- 2.1 standard: 采用 Transformer-XL 的方法

$$h_{t,j}^{1:m} = \text{Transformer}(\alpha_{1:k}^{(j)}, Z_{1:k}^{(j)}, X_t)$$

以下省去当前时间步 t 。对于 $n=1, \dots, m$

$$\mathbf{m}_j^{n-1} = \text{concat}(\{Z_i^{(j)}\}_{i=1}^{n-1})$$

$$\tilde{\mathbf{m}}_j^{n-1} = \text{concat}(\{\alpha_i^{(j)} Z_i^{(j)}\}_{i=1}^{n-1})$$

$$\tilde{\mathbf{h}}_j^{n-1} = [\mathbf{m}_j^{n-1} \circ \mathbf{h}_t^{n-1}]$$

$$\hat{\mathbf{h}}_j^{n-1} = [\tilde{\mathbf{m}}_j^{n-1} \circ \mathbf{h}_t^{n-1}]$$

$$\mathbf{q}_j^n, \mathbf{k}_j^n, \mathbf{v}_j^n = \mathbf{h}_j^{n-1} \mathbf{W}_q^{n\top}, \tilde{\mathbf{h}}_j^{n-1} \mathbf{W}_{k,E}^{n\top}, \hat{\mathbf{h}}_j^{n-1} \mathbf{W}_v^{n\top}$$

$$\mathbf{A}_{j,i}^n = \mathbf{q}_j^{n\top} \mathbf{k}_{i,j}^n + \mathbf{q}_j^{n\top} \mathbf{W}_{k,R}^n \mathbf{R}_{i-j} + u^\top \mathbf{k}_{i,j} + v^\top \mathbf{W}_{k,R}^n \mathbf{R}_{i-j}$$

$$\mathbf{a}_j^n = \text{Masked-Softmax}(\mathbf{A}_j^n) \mathbf{v}_j^n$$

$$\mathbf{o}^n = \text{LayerNorm}(\text{Linear}(\mathbf{a}^n) + \mathbf{h}^{n-1})$$

$$\mathbf{h}^n = \text{Positionwise-Feed-Forward}(\mathbf{o}^n)$$

3 更新 Memory

Memory 的大小为 N 个 Key-Value 对, 每个 Value 为一个片段 X_i 的所有层的表示。将上一步生成的 $h_t^{1:m}$ 存入 Memory:

- 将第一个 Key-Value 对删除, 其余的向前递补, 最后一个 Key-Value 对为空, 将 $h_t^{1:m}$ 填入 Memory 中的最后一个空位

对于填入的 Key-Value 对 h_t , 需要更新其 Key, 用最顶层表示更新 Key:

$$\text{Key} = \text{summary}(h_t^n)$$

- 3.1 直接使用原向量 $\text{summary}(h_t^n) = \text{Identical}(\text{flat}(h_t^n))$
- 3.2 线性变换 $\text{summary}(h_t^n) = \text{Linear}(\text{flat}(h_t^n))$