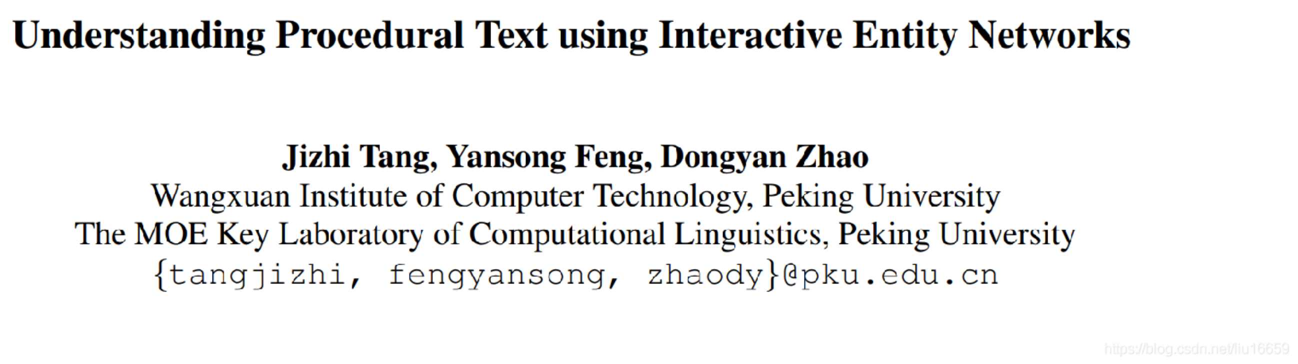
论文阅读《Understanding Procedural Text using Interactive Entity Networks》

原创 LawsonAbs 2021-03-15 10:32:54 34 收藏 编辑 版权

分类专栏： 论文阅读 文章标签： nlp 机器学习 自然语言处理

总结

- 原文出处：CSDN_LawsonAbs
- LawsonAbs的认知与思考，**望各位读者审慎阅读。**
- 论文类型：**改进模型类**，无亮点。
- 建议：如果读者尚未读到此文，建议换文阅读。
- 论文地址



Understanding Procedural Text using Interactive Entity Networks

Jizhi Tang, Yansong Feng, Dongyan Zhao
Wangxuan Institute of Computer Technology, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{tangjizhi, fengyansong, zhaody}@pku.edu.cn

本文主要按照如下三个方面来叙述：

1 Target：Understanding procedural text

2 Method：By using Interactive Entity Network

3 Contributions:

- introduce an novel module——IEN(by using attention)
- conduct intensive experiments

4 Experiments

~

---

1 Task

1.1 What’s procedural text?

e.g., scientific articles, instruction books, or recipes sentence 1:

> The water breaks into oxygen, hydrogen, and electrons.

sentence 2:

> Blood travels to the lungs.
> Carbon dioxide is removed from the blood.
> Oxygen is added to your blood.

1.2 Entity state tracking is the key task for procedural text comprehension.

本文的任务：Focus on scientific process understanding task(Dalvi et al., 2018), in which the tracking targets are action and location of entities.

1.3 Example

The state tracking task is to predict the states of each entity e_i after reading each sentence s_t, where an entity's state is a value of a property p_j.

> CO2 enters leaf.

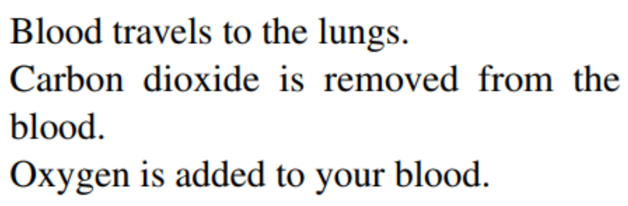
> System:
> the existence(p_j) of CO2 (e_i) is true
> the location(p_j) of CO2 is leaf

1.4 Difficulties

- dynamic nature
- the involvement of multiple entities => [`The water breaks into oxygen, hydrogen, and electrons.`]
- the complexity of tracking targets

1.5 Challenge

how to properly capture the relationship between entity interactions and their state changes.

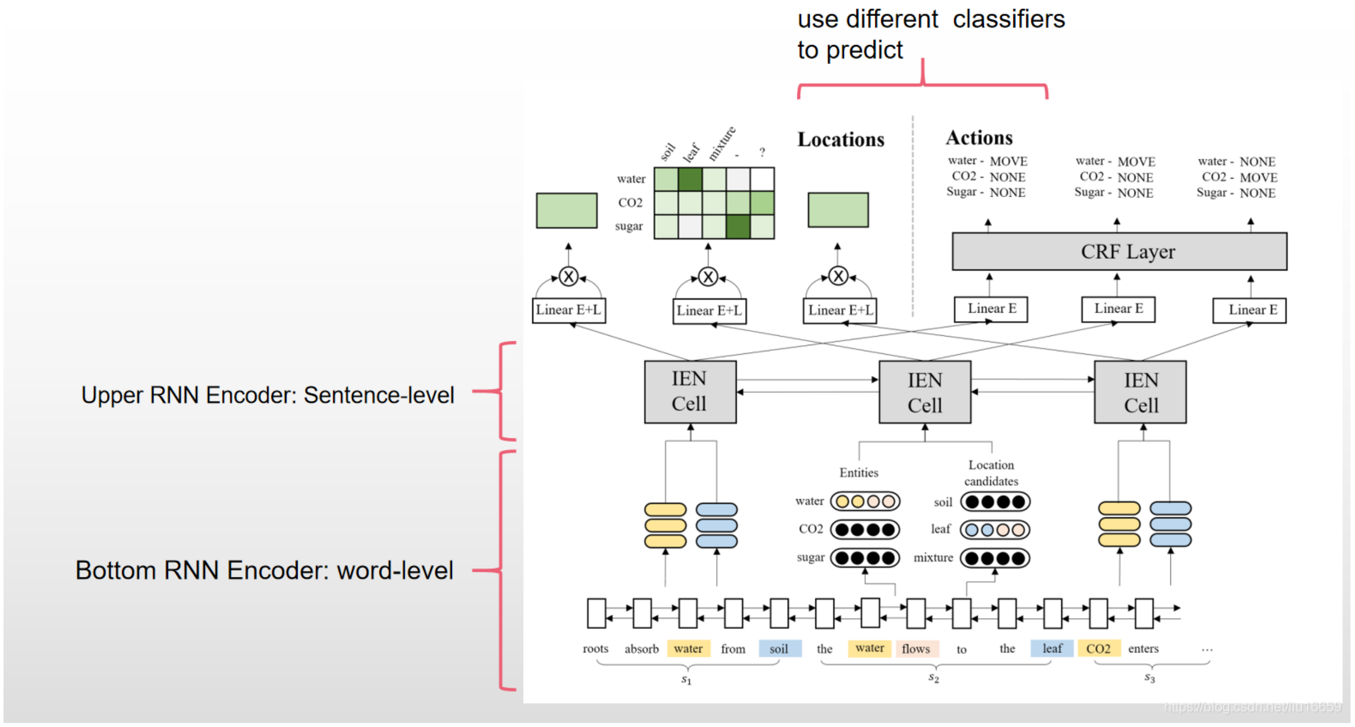

Blood travels to the lungs.
Carbon dioxide is removed from the blood.
Oxygen is added to your blood.

2 IEN: Interactive Entity Network

本文的贡献就在于提出了这个IEN模型，~~因为它提高了任务的效果，所以它有用~~。

2.1 Characteristics

- two-layer RNN model



2.1.1 Bottom RNN Encoder : Word-Level Encoding

$$\vec{\mathbf{w}}_i = [emb(w_i); v_i]$$

- 等式左边的w_i 表示的是embedding vector，程式右边的w_i 是单词i
- 实验中针对不同的 `embedding function` ，如（ `fastText`、`ELMo` ），进行对比
- v_i 是一个标量，用于表示w_i 这个词是否是动词
- 将得到的w_i 送入到BiLSTM中得到最后的一个表示 $u_i = BiLSTM([w_i])$

2.1.2 Upper RNN Encoder : Sentence-Level Encoding and IEN cell

> To track the state changes, we extract sentence features from word-level encodings by running another RNN at the sentence level.

$$\boldsymbol{x}_t^{e_i} = \begin{cases} [\boldsymbol{u}_t^{e_i}; \boldsymbol{u}_t^{v}], & \text{if} \quad e_i \in s_t \\ \mathbf{0}, & \text{otherwise} \end{cases} \tag{1}$$

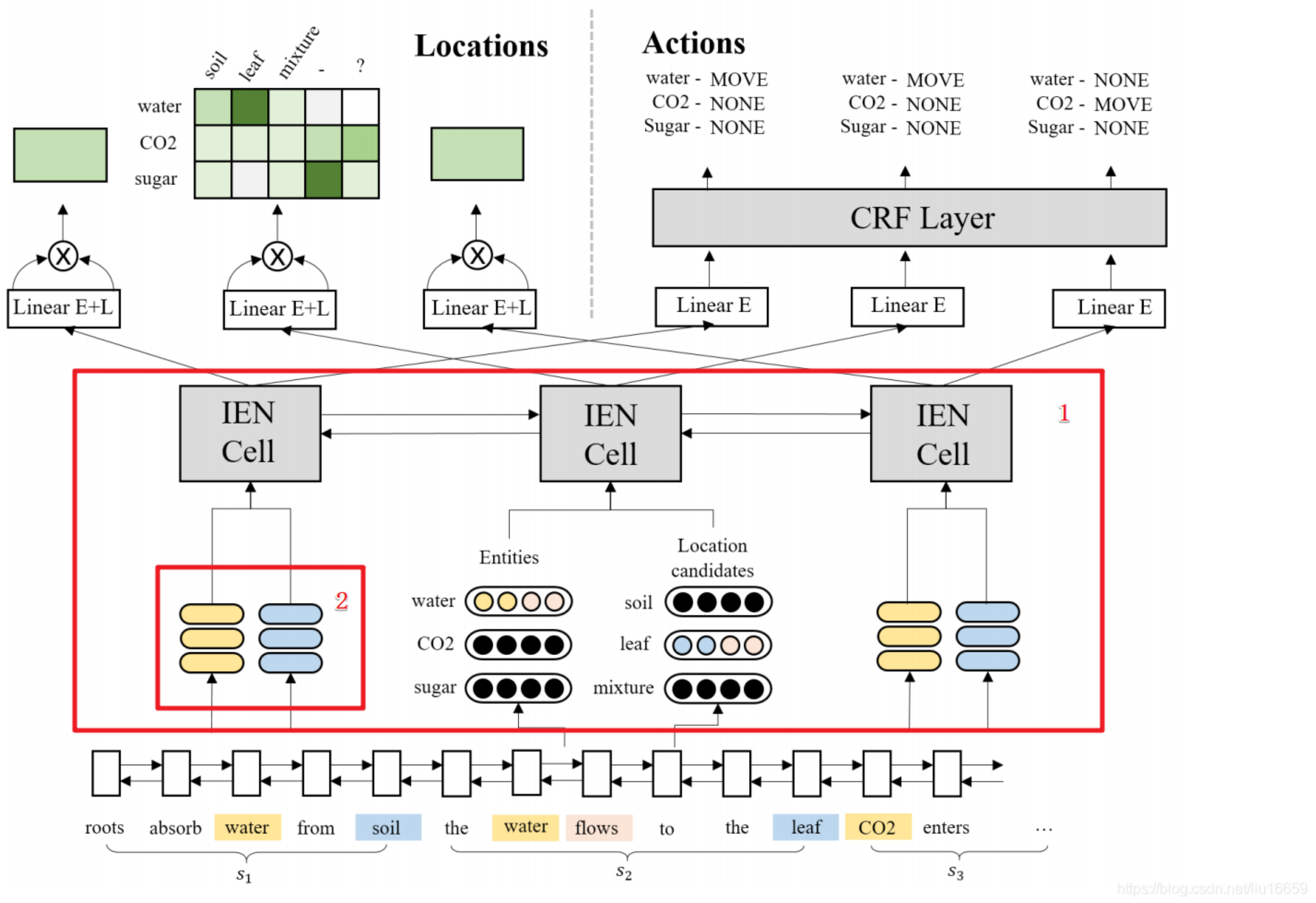
$$\boldsymbol{x}_t^{l_j} = \begin{cases} [\boldsymbol{u}_t^{l_j}; \boldsymbol{u}_t^{v}], & \text{if} \quad l_j \in s_t \\ \mathbf{0}, & \text{otherwise} \end{cases} \tag{2}$$

$x_t^{e_i}$： 表示的就是在句子 t 中的 e_i 实体；
$x_t^{l_j}$： 表示的就是在句子 t 中的 l_j 位置；
u_t^v：表示的是句子t中的谓语动词。

补充：

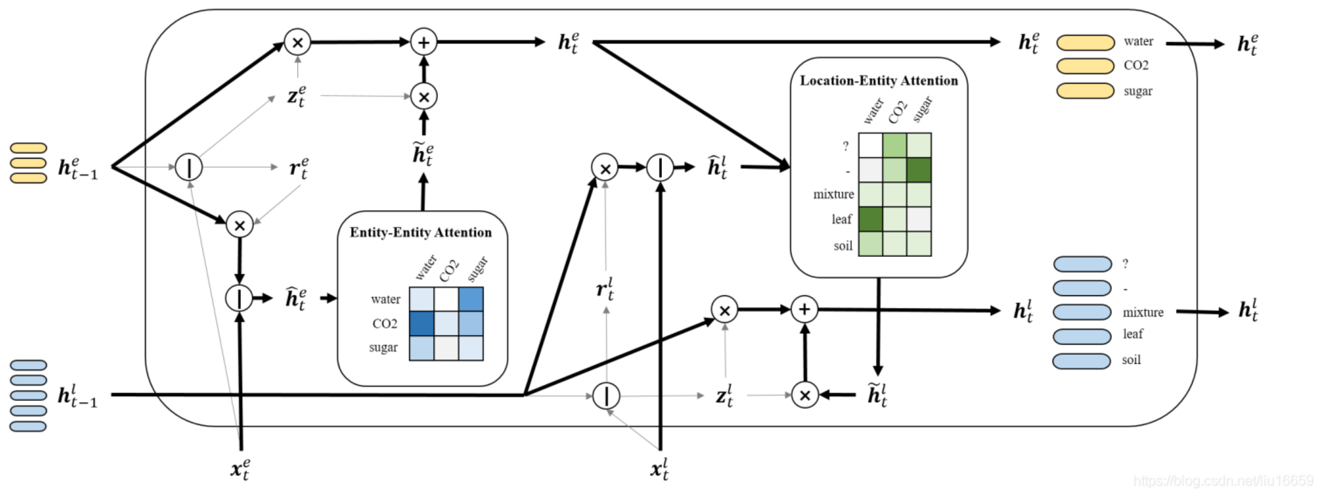
- 如果entity 或者 location 是由多个词构成的，那么就执行一个mean pooling 操作。

接着分别将所有的实体、位置表示拼接在一起得到 $x_t^e \in R^{n*d}$， $x_t^l \in R^{m*d}$。 x_t^e 和 x_t^l 就是第 t 个IEN cell 的输入。
下图中红框1就是IEN cell 部分，而其中的红框2就是IEN cell 的输入。



接着来看一下IEN cell 是如何设计的?

2.2 Structure of an IEN cell



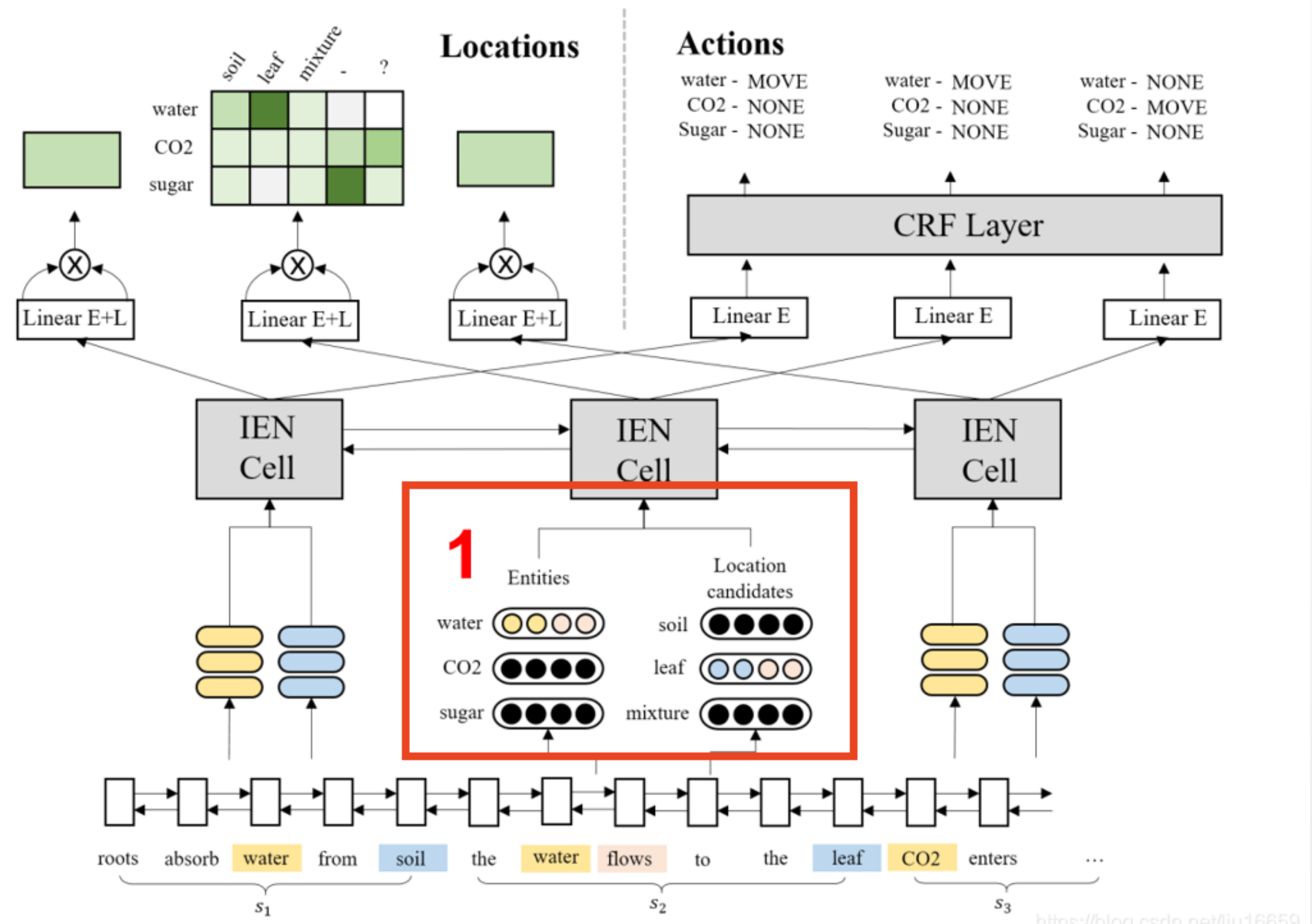
在每个 IEN cell中，都有 n 个实体 slots、 m个location slots，与上面的x_t^e 和 x_t^l 相对应。

2.2.1 IEN cell Inputs

> the representations of all entities and all location candidates in a signle sentence st, or a mask vector if the entity or location candidate is not in st .

$$x_t^{e_i} = \begin{cases} [u_t^{e_i}; u_t^v], & \text{if} \quad e_i \in s_t \\ \mathbf{0}, & \text{otherwise} \end{cases} \qquad (1)$$

$$x_t^{l_j} = \begin{cases} [u_t^{l_j}; u_t^v], & \text{if} \quad l_j \in s_t \\ \mathbf{0}, & \text{otherwise} \end{cases} \qquad (2)$$



memory slots 是什么?

- we place memory slots inside IEN cells and let them recurrently update as GRU;

- Each memory slot represents the state of a specific entity or a location candidate.

$h_t^e \in R^{n*d}$ 表示第t个IEN cell中所有实体的 memory slots； $h_t^l \in R^{n*d}$ 表示第t个IEN cell中所有location的 memory slots。更新过程如下：

entity memory slots

$$
\begin{aligned}
\boldsymbol{z}_t^e &= \sigma([\boldsymbol{h}_{t-1}^e; \boldsymbol{x}_t^e] \cdot \boldsymbol{W}_z^e) && (3)\\
\boldsymbol{r}_t^e &= \sigma([\boldsymbol{h}_{t-1}^e; \boldsymbol{x}_t^e] \cdot \boldsymbol{W}_r^e) && (4)\\
\hat{\boldsymbol{h}}_t^e &= [\boldsymbol{r}_t^e \odot \boldsymbol{h}_{t-1}^e; \boldsymbol{x}_t^e] && (5)\\
\tilde{\boldsymbol{h}}_t^e &= \tanh(\mathrm{Att}(\hat{\boldsymbol{h}}_t^e, \hat{\boldsymbol{h}}_t^e, \hat{\boldsymbol{h}}_t^e) \cdot \boldsymbol{W}^e) && (6)\\
\boldsymbol{h}_t^e &= (1 - \boldsymbol{z}_t^e) \odot \boldsymbol{h}_{t-1}^e + \boldsymbol{z}_t^e \odot \tilde{\boldsymbol{h}}_t^e && (7)
\end{aligned}
$$

这里的 Att 是一个 scaled key-value attention function. 定义公式如下：

are trainable parameters. Att is a scaled key-value attention function (Vaswani et al., 2017), defined as:

$$\mathrm{Att}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}) = \mathrm{Softmax}\left(\frac{(\boldsymbol{q}\boldsymbol{W}^q)(\boldsymbol{k}\boldsymbol{W}^k)^T}{\sqrt{d_a}}\right)(\boldsymbol{v}\boldsymbol{W}^v) \tag{8}$$

where $\boldsymbol{W}^q \in \mathbb{R}^{d \times d_a}$, $\boldsymbol{W}^k \in \mathbb{R}^{d \times d_a}$ and $\boldsymbol{W}^v \in \mathbb{R}^{d \times d_a}$ are trainable parameters.

模型的亮点就在于将 attention 引入到了GRU中，这样显式地模拟出 entity-entity 以及 entity-location 之间的交互。

location memory slots

这个更新过程同 entity memory slots 很相似，不再叙述。

2.2.3 输出

2.2.3.1 Entity Actions

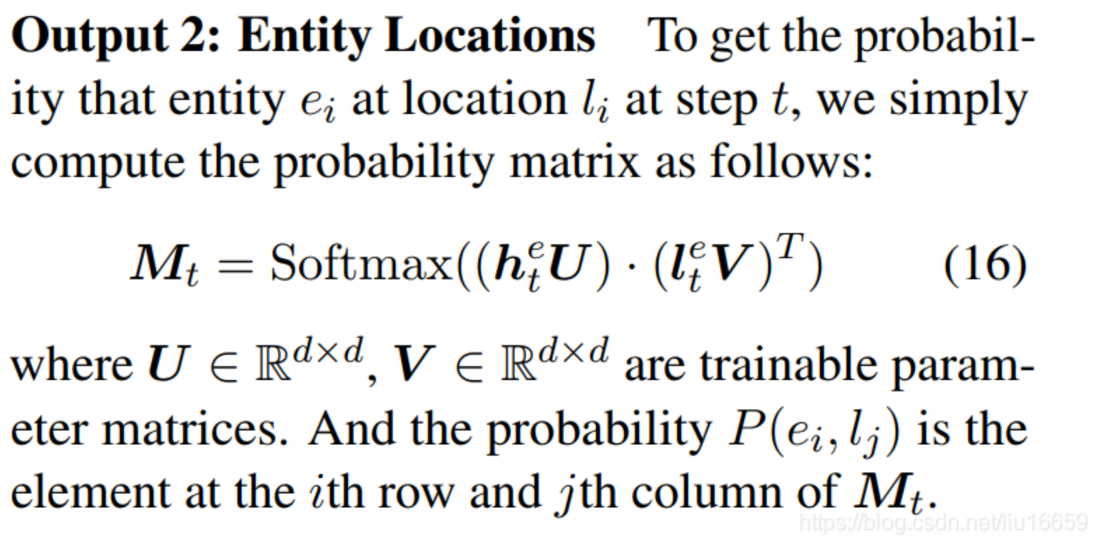
Output 1: Entity Actions Following NCET (Gupta and Durrett, 2019), We use CRF to model the dependency of actions. We use $\boldsymbol{h}_t^{e_i}$, which is the ith row of $\boldsymbol{h}_t^e$, to generate emission potentials for each action tag y_t at each time step t with respect to entity e_i:

$$\phi(y_t, t, e_i) = \boldsymbol{h}_t^e \cdot \boldsymbol{H} \tag{14}$$

where $\boldsymbol{H} \in \mathbb{R}^{d\times 1}$ is a trainable matrix. Additionally, we train a transition matrix to get the transition potentials among the 4 action tags and two extra tags (”START” and ”END”) which we denote by $\psi(y_{t-1}, y_t)$. Finally for an action sequence $\boldsymbol{y}$, we get the probability as:

$$P(\boldsymbol{y}|e_i) \propto \exp(\sum_{t=0}^{T} \phi(y_t, t, e_i) + \psi(y_{t-1}, y_t)) \tag{15}$$

2.2.3.2 Entity Locations



Output 2: Entity Locations To get the probability that entity e_i at location l_i at step t, we simply compute the probability matrix as follows:

$$\boldsymbol{M}_t = \mathrm{Softmax}((\boldsymbol{h}_t^e \boldsymbol{U}) \cdot (\boldsymbol{l}_t^e \boldsymbol{V})^T) \qquad (16)$$

where $\boldsymbol{U} \in \mathbb{R}^{d\times d}$, $\boldsymbol{V} \in \mathbb{R}^{d\times d}$ are trainable parameter matrices. And the probability $P(e_i, l_j)$ is the element at the ith row and jth column of $\boldsymbol{M}_t$.

参考资料

- GRU 的结构图

图6.8给出了 GRU 网络的循环单元结构.

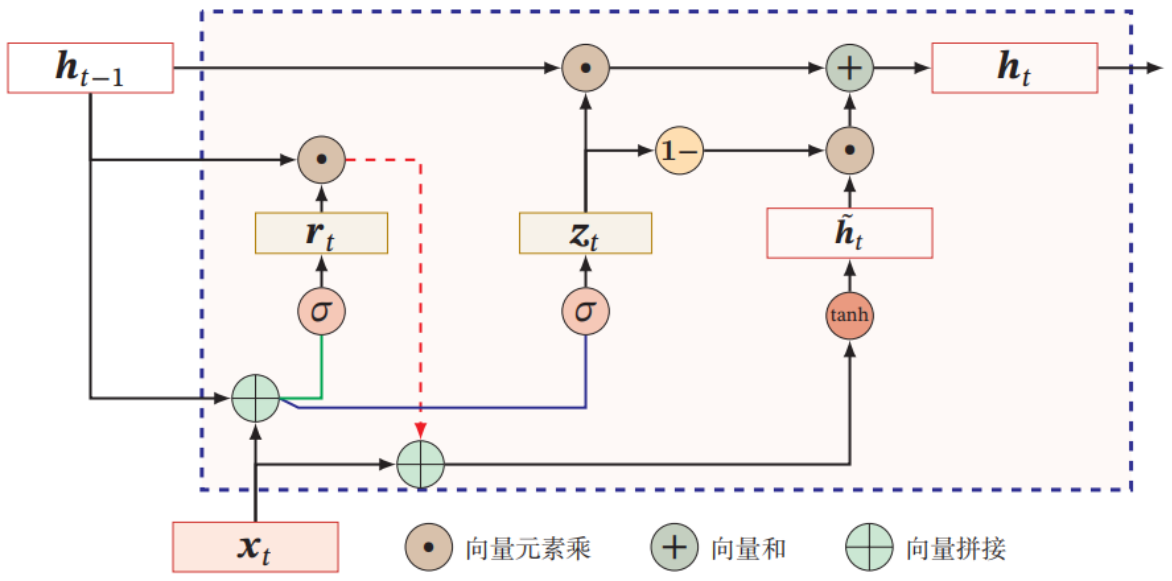


图 6.8 GRU 网络的循环单元结构