CopyMTL_Copy M... g.pdf
1.69MB

CopyMTL: Copy Mechanism for Joint Extraction of Entities and Relations with Multi-Task Learning

Daojian Zeng[*§], **Haoran Zhang**[*†] , **Qianying Liu**[‡]
[§]Changsha University of Science & Technology, Changsha, 410114, China
[†]University of Illinois at Urbana-Champaign, Illinois, 61820, USA
[‡]Kyoto University, Kyoto, 606-8501, Japan
zengdj916@163.com, haoranz6@illinois.edu, ying@nlp.ist.i.kyoto-u.ac.jp

AAAI 2020

这个其实就是对CopyRE的改进，分析了CopyRE的两个不足，之后对其进行改进提出了CopyMTL。
这个关系抽取应该是sentence-level的，没有额外的标注。
输入句子，输出三元组。

COpyRE其实上也是一个encoder-decoder with attention mechanism的设置，然后也加入了copy mechanism。2018年的模型了。
对于encoder，其实就是一个biLSTM
对于decoder，其实也就是一个LSTM+attention
然后呢，这个decoder会以三个时间步为周期：

- 第一个时间步预测出relation
- 第二个时间步预测出head entity
- 第三个时间步预测出tail entity

$$\boldsymbol{logit}_t = \begin{cases} [\boldsymbol{h}_t^D \cdot \boldsymbol{W}^r; q^{NA}], & \text{if } t\%3 = 1; \\ [\boldsymbol{q}_t; q^{NA}], & \text{if } t\%3 = 2; \\ [\boldsymbol{M} \otimes \boldsymbol{q}_t; q^{NA}], & \text{if } t\%3 = 0. \end{cases}$$

对于预测relation的，那就是在所有可能的relation和NA（无relation）中做softmax。

对于预测entity的，用的就是copy mechanism！就是在输入句子的所有词上以及NA上做softmax。
特别的，在预测tail entity的时候，会把已经预测为head entity的词语给mask掉，之后再做softmax。
在train的时候，copyRE会加入（NA，NA，NA）三元组padding

之后呢，作者就分析了copyRE的问题，主要集中在两个：

- 模型没有很好的表征头尾实体的顺序：实验发现对relation的预测F1高，但是对entity预测的F1低；去掉MASK之后F1接近0
- 模型对于entity只允许一个词，但是有的实体是多个词

对于第一个问题，详细解释如下。对于输入序列每个位置的得分如下：

$$
\begin{aligned}
q_i^t &= [\boldsymbol{h}_t^D; \boldsymbol{h}_i^E] \cdot \boldsymbol{W}^e \\
&= [\boldsymbol{h}_t^D; \boldsymbol{h}_i^E] \cdot [\boldsymbol{W}_1^e; \boldsymbol{W}_2^e] \\
&= \boldsymbol{h}_t^D \cdot \boldsymbol{W}_1^e + \boldsymbol{h}_i^E \cdot \boldsymbol{W}_2^e
\end{aligned}
$$

由此时decoder的hidden state和输入序列每个的hidden state concat之后经过矩阵。

$$
\begin{aligned}
p(y_t | y_{<t}, s) &= \frac{e^{q_i^t}}{\sum_j e^{q_j^t}} = \frac{e^{\boldsymbol{h}_t^D \cdot \boldsymbol{W}_1^e} \cdot e^{\boldsymbol{h}_i^E \cdot \boldsymbol{W}_2^e}}{e^{\boldsymbol{h}_t^D \cdot \boldsymbol{W}_1^e} \cdot \sum_j e^{\boldsymbol{h}_j^E \cdot \boldsymbol{W}_2^e}} \\
&= \frac{e^{\boldsymbol{h}_i^E \cdot \boldsymbol{W}_2^e}}{\sum_j e^{\boldsymbol{h}_j^E \cdot \boldsymbol{W}_2^e}}
\end{aligned}
$$

所以预测entity的时候，竟然跟decoder的t无关，所以预测头尾实体的分布是一样的。
尽管模型用一个mask来控制，但是mask并没有参与optimize，所以效果其实就是 最高的做头，第二高的做尾。

所以paper提出一个解决方案就是：

$$q_i^t = \sigma([\boldsymbol{h}_t^D; \boldsymbol{h}_i^E] \cdot \boldsymbol{W}^f) \cdot \boldsymbol{W}^o$$

加入一个非线形层，让分子分母不能约分，也就是每个时刻预测的分布与t有关。

第二个问题作者提出的方案就是multi-task learning。
其实就是加了一个biLSTM+CRF的NER任务来参与训练，之后尽管模型只能预测一个词，但是

- 'B'. a single token entity.
- 'I', an entity with multiple tokens, it will look for the token before the current token until it finds 'B'.
- 'O', a single token entity.

其实后面paper还提出了另一种方案，就是预测三元组变成五元组，增加了两个实体的长度项。

实验采用了New York Time数据集和WebNLG。
实验过程中，每个decoder设置最多解码出5个三元组，因为数据集平均每个就2个三元组。
然后解码器没有搞一个eos结束符号，而是当解码出（NA，NA，NA）的时候停止。

其实就是一个 发现问题 -> 发掘原因 -> 解决问题。