

人工智能基础-PROJECT

思路1——对可能的三元组做分类

- ▶ Sentence提供了组成三元组所需要的times, attributes和values, 因此可以将可能产生的三元组处理出来, 然后对这些三元组进行分类
- ▶ 遇到的问题
 - ▶ 对sentence打多标签
 - ▶ 类别不一致
 - ▶ 输出难以处理
 - ▶ 对三元组分类

思路1——对可能的三元组做分类

► 数据预处理

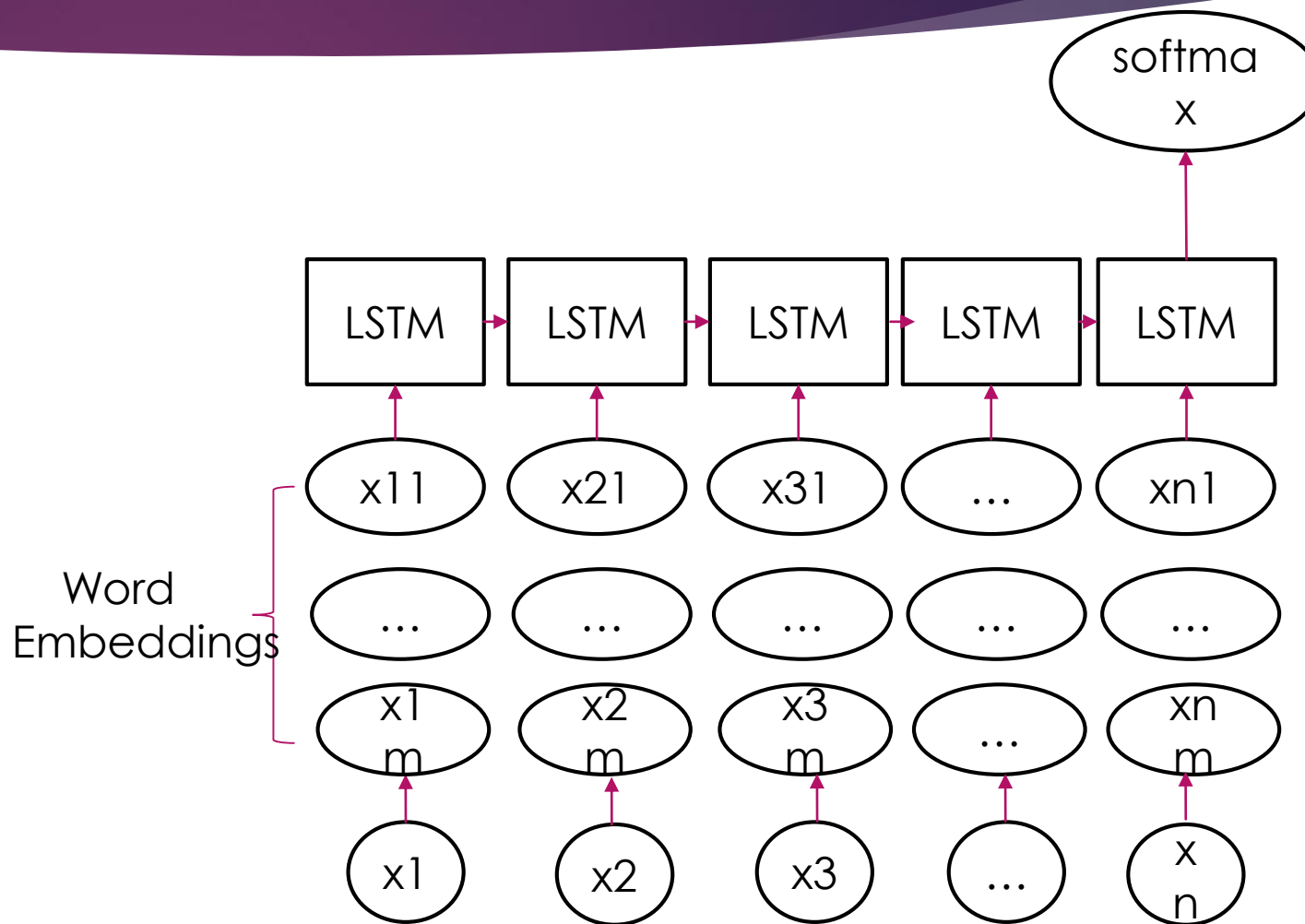
- 将times之前的attributes和values都删除。类似得处理times, attributes, values三个list (wrong)
- 此时三元组的提取不依赖于times之前和values之后的信息，删除这些indexes
- 将每个词的indexes加1，使index从1开始，视0为无效填充，放在每一个sentence的indexes的最前面同时，在每一个sentence的indexes最后三个填充为可能出现的三元组



思路1——对可能的三元组做分类

网络

- ▶ Word Embeddings
- ▶ 单层LSTM
- ▶ Softmax分类
- ▶ Loss: softmax交叉熵
- ▶ 梯度下降: Adam



思路1——对可能的三元组做分类

► 训练

- 7:3划分训练集和测试集
- test error, train error很低
- 计算的Precision、Recall和F1很低

```
epoch is 0, precision: 0.317014, recall: 0.151167, F1 score: 0.204716
epoch is 1, train error 0.060000, test error 0.097125
epoch is 1, precision: 0.277822, recall: 0.134711, F1 score: 0.181443
epoch is 2, train error 0.060000, test error 0.089786
epoch is 2, precision: 0.391246, recall: 0.222350, F1 score: 0.283553
epoch is 3, train error 0.060000, test error 0.094465
epoch is 3, precision: 0.280037, recall: 0.115959, F1 score: 0.164005
epoch is 4, train error 0.060000, test error 0.092844
epoch is 4, precision: 0.320306, recall: 0.144279, F1 score: 0.198945
epoch is 5, train error 0.060000, test error 0.091713
epoch is 5, precision: 0.343598, recall: 0.162266, F1 score: 0.220432
epoch is 6, train error 0.060000, test error 0.094495
epoch is 6, precision: 0.341106, recall: 0.195943, F1 score: 0.248906
(tensorflow3) [root@localhost djf]#
(tensorflow3) [root@localhost difl]# ls
```

反思

- ▶ 样本不均衡
 - ▶ 即便全判0，训练的错误率都会非常低
- ▶ 思路也有问题
 - ▶ 对网络不理解，数据预处理，构造的特征很没有道理
 - ▶ 想让LSTM根据前面的句子学到最后三元组的匹配的pattern

进一步思考

- ▶ 对三元组匹配，一定是一个分类的问题，没法构造回归问题
- ▶ 分类：
 - ▶ 对三元组分类
 - ▶ 一个句子对应多个三元组，如果句子中没有三元组信息，那么同一个句子经过训练得出的结果会在多个三元组上分类为1，但是在哪里加入三元组是一个问题，如果利用LSTM输出和三元组向量之间的距离去判断，感觉也不能学到什么规律；如果句子中有三元组信息，这样构造我也解释不通，效果也不好
 - ▶ 对句子分类
 - ▶ 利用多标签，没有尝试，但是类别问题等，感觉不好处理
- ▶ 根据我们语言习惯，我们找三元组根据的是语法的信息
 - ▶ 对句子做句法分析，然后进行规则匹配或者学习匹配

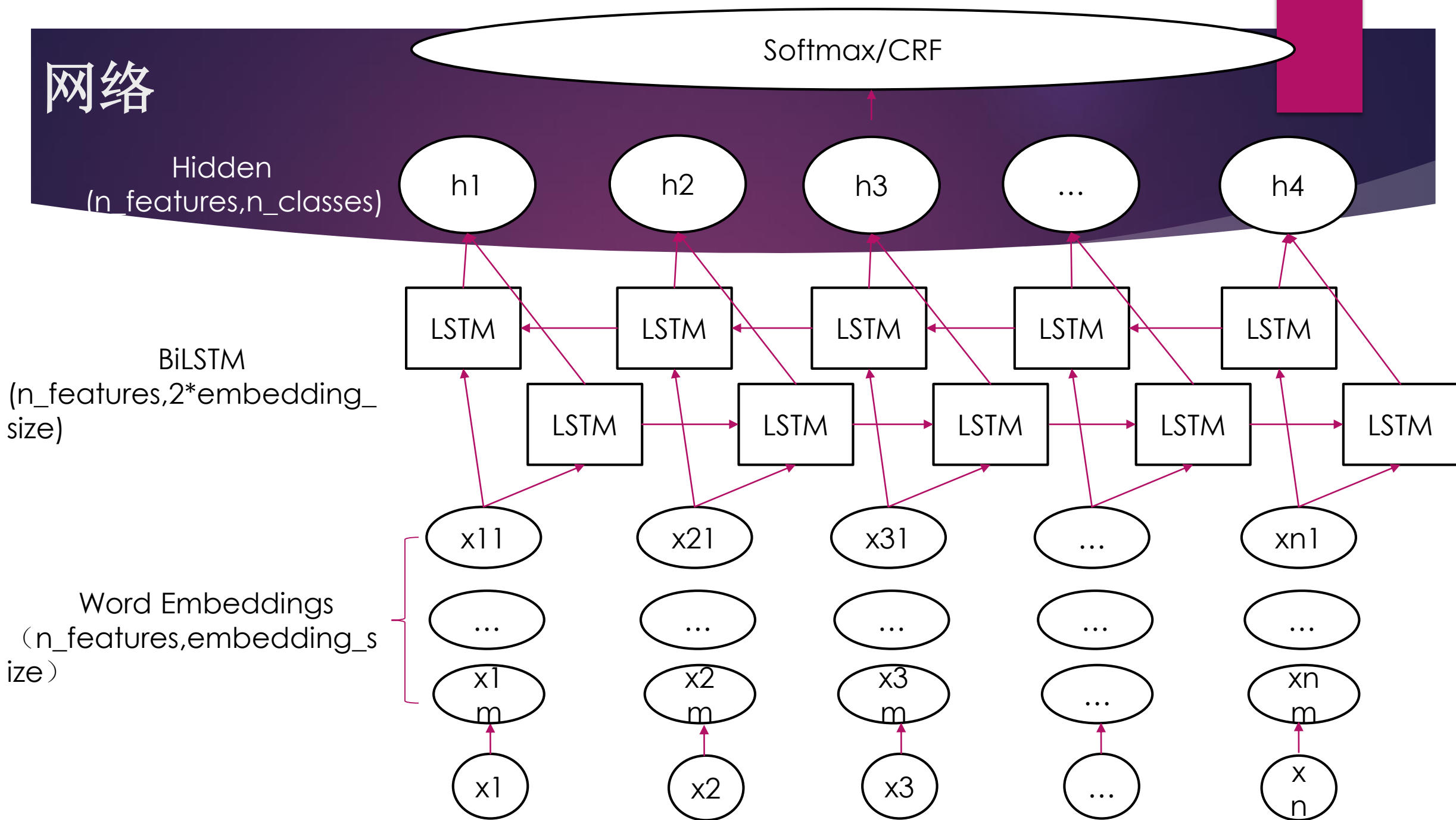
进一步思考

- ▶ 查阅资料
 - ▶ LSTM用于分词、序列标注效果很好
 - ▶ BiLSTM+CRF做序列标注
- ▶ 和同学交流
 - ▶ 问了几个同学发现好像对三元组分类都没什么思路...
 - ▶ 通过对每个词进行分类，然后进行规则匹配，正确率很高
- ▶ 最终确定的思路
 - ▶ WordEmbeddings+BiLSTM+Softmax/CRF

数据预处理

- ▶ 对每个词打标签，如果在有效的三元组里面，标为1，否则标为0
- ▶ 输入的句子长度为200，对句子做padding，无效的padding标为2
- ▶ Padding的数据是0，每个index加1
- ▶ 7:3划分训练集合测试集
- ▶ 将预处理好的数据保存为.npy文件

网络

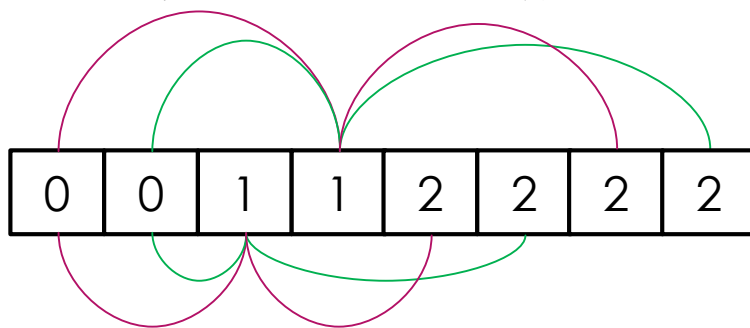
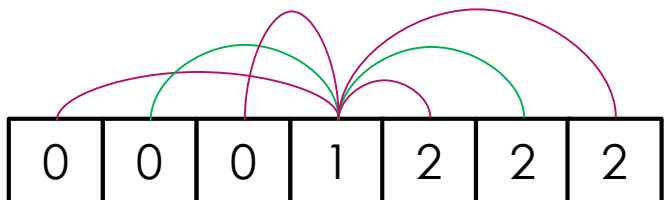


规则匹配

- ▶ 根据出现在最后结果的三元组中的times, attributes和values组合出预测的三元组
- ▶ 具体规则(0,1,2分别代表times, attributes, values)
 - ▶ 0001222
 - ▶ 00012221222
 - ▶ 012012
 - ▶ 00112222
 - ▶ ...

规则匹配

- ▶ 先将对应的有效的times, attributes和values用0,1,2表示, 然后将这些按indexes从小到大排序
- ▶ 分别将0,1,2读入一个list, 遇到被1,2分隔的0时停止, 然后检查 $\text{len}(\text{times}) * \text{len}(\text{attributes})$ 和 $\text{len}(\text{values})$, 如果相等, 说明可以组成有效三元组, 否则不可以组成
- ▶ 可以组成有效三元组, 那么就按照0001222,00111222222进行匹配



规则匹配

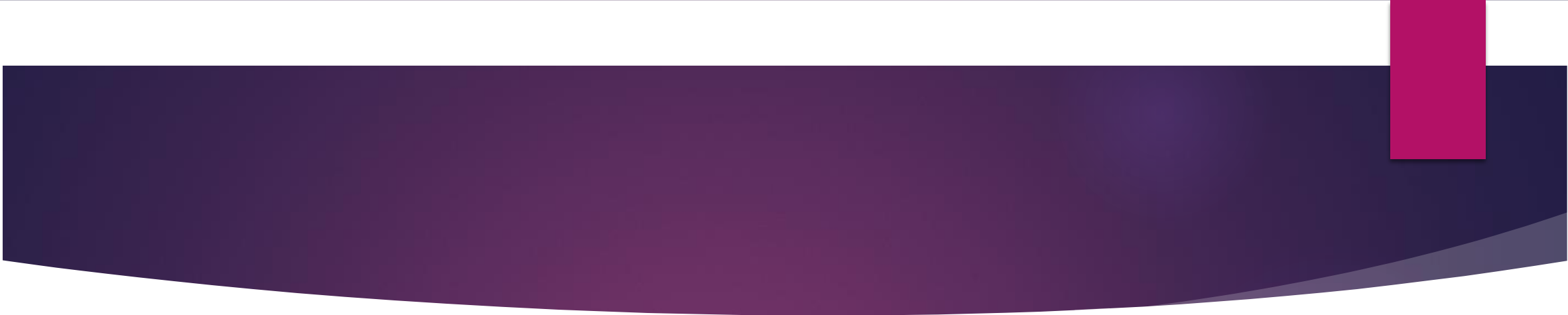
- ▶ 在给定的真实标签下，这个规则匹配的正确率极高，只错了一个句子
- ▶ 错误句子的类型：20101
- ▶ sentence: '由于2012年至2014年(2012年、2013年、2014年)公司业务发展情况良好，**总资产**规模呈稳步上升趋势，由**2012年12月31日**的**1,731.46亿元**增长至**2014年12月31日**的**2,432.94亿元**，年复合增长率为18.54%。'
- ▶ 训练集的3000个句子其他的都能正确匹配（训练集样本种类还是比较单一）

Softmax or CRF

- ▶ 最后一层如果接softmax就是直接依据当前的信息进行分类，而CRF会考虑前后的转移信息，因此感觉上会更可靠一些
- ▶ 经过验证发现，在验证集上softmax的F1比CRF的F1低了百分之几，效果相差不大
- ▶ 最后采用了CRF
- ▶ 模型：BiLSTM+CRF+规则匹配

改进

- ▶ K-fold交叉验证
- ▶ 调参：
 - ▶ 学习率, embedding size, batch size
- ▶ 尝试不同的梯度下降算法



谢谢！