基于 LSTM-CRF 的序列标注

谢子陬 19307130037

2021 年 12 月 12 日

1 实验简介

1.1 命名实体识别任务 NER

在本次实验中，我们需要完成命名实体识别任务 (Named entity recognition)，即识别文本中有意义的实体类型从而为其他的 NLP 下游任务服务。对于该任务我们目前采用 BIO 标注方法，即将每个元素标注为 “B-X”、“I-X” 或者 “O”。其中，“B-X” 表示此元素所在的片段属于 X 类型并且此元素在此片段的开头，“I-X” 表示此元素所在的片段属于 X 类型并且此元素在此片段的中间位置，“O” 表示不属于任何类型。我们可以用通用的序列标注模型来完成该任务。

1.2 模型主体

模型主体使用 LSTM 来提取序列特征搭配 CRF (Conditional Random Field) 对上下文特征的提取来完成 NER 任务。LSTM 网络结构和 Glove 词嵌入在上一次报告中已经详细介绍，在此就不过多赘述。本次实验模型采由三部分组成，分别是 Word Embedding 层 (Glove/Glove+CharRep/ELMo) 、Encoder 层 (LSTM)、Decoder 层（CRF）。

1.3 条件随机场 CRF

条件随机场 (Conditional Random Field) 是一个判别式模型，其可以看做是最大熵模型在序列问题上的推广。关于判别式和生成式模型在课上已经多次讲过。下图 1 可以很好的概括我们学过的判别式和生成式模型之间的关系。

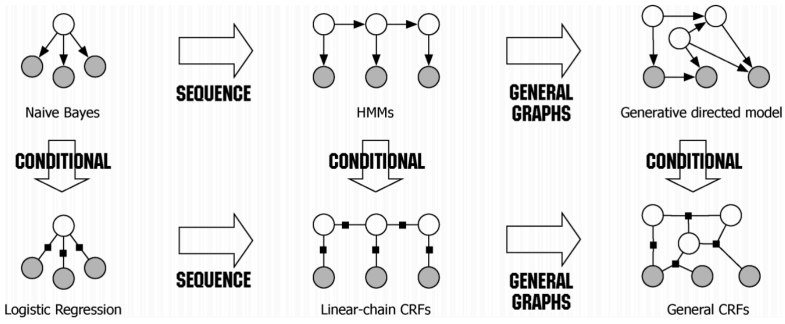

图 1: 常用生成式与判别式模型及其关系

对于实验中的 CRF 而言，由于我们是 Seq2Seq 任务，因此使用的是 Linear-chain CRF，其输入是序列 $X = (x_1, x_2, ...x_i)$，输出序列 $Y = (y_1, y_2, ...y_i)$ 对应在词性标注任务中，输入序列为一串单词，输出序列就是相应的词性。

Linear-CRF 作为序列化模型，和 HMM 十分类似，不同的是 CRF 是判别式模型，我们需要计算的是条件概率 $P(Y|X)$，$Y = (y_1, y_2, ..., y_n)$，$X = (x_1, x_2, ..., x_n)$，该条件概率可通过极大似然求得。

$$P(y_1, y2..y_n|X) = \frac{1}{Z(x)} exp \sum_{k=1}^{n} f_k(y, x) \tag{1}$$

$$f_k(y, x) = t(y_k, y_{k+1}) + h(y_{k+1}, x) \tag{2}$$

(1) 中 $Z(x)$ 是规范因子，(2) 中 $t(y_k, y_k + 1)$ 为转移矩阵

可以看到 CRF 构建的模型相比 HMM 去除了齐次马尔科夫链假设（即当前状态仅取决于上一状态），标签序列的联合概率依赖于全体序列，从而可以构建更加全局的序列特征。同样的由于直接计算的复杂度过高，CRF 也采用了和 HMM 类似的前向后向算法来简化复杂度加快计算，篇幅限制就不过多赘述。

最后的学习过程我们需要最大化条件概率 $P_w(Y|X)$ 即极大化其对数似然函数，添加符号转为最小化负对数损失函数，即定义 NegLog-loss 为 $L(w) = log\Pi_{x,y} P_w(y|x)^{\hat{P}(x,y)} = \sum_{x,y} \hat{P}(x, y) log P_w(y|x)$ 类似最大熵。其中 $\hat{P}(x, y)$ 为经验分布。解码过程同样采用维特比算法（Viterbi）得到最大条件概率来进行判别。

1.4 ELMo 和 Char Representation

在之前的工作中，我们接触了 GloVe Embedding，相比 Word2Vec 更好的获取了全局的信息。但是由于其词和向量对应的关系导致无法很好的处理多义词的情况。对此 ELMo 提出使用训练好的模型来做词嵌入，每个词的词向量会受到上下文信息的影响从而解决多义词的问题。

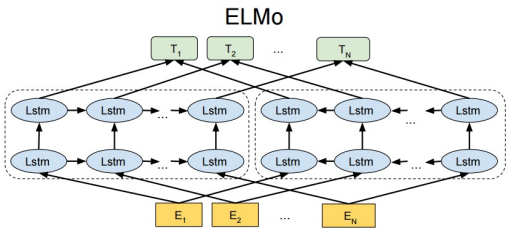

图 2: ELMo 模型结构

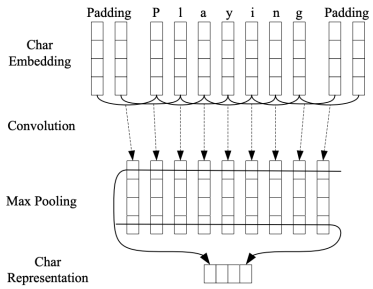

图 3: right picture

其模型主体是两个预训练好的 Bi-LSTM，分别输入前向语言和后向语言，双向提取信息进行融合，得到对应的词向量。

Char Representation 则在词向量基础上以字符为单位，将一个词拆成字符表示，先给每个字符一个 Embedding，再经过 dropout 和 CNN 层，最后做一个 Max Pooling 在字符角度上得到该词的特征表示，再和原先的 Word Embedding 连接在一起送入 LSTM 提升模型性能。

2 实验内容

2.1 CRF 效果对比

基础模型是单层的 Bi-LSTM，使用 300d GloVe 词嵌入表示进行 BIO 标注训练。对比添加 CRF(Conditional Random Field) 前后模型的性能。

参数统一为隐层大小 512，dropout 0.5。优化器选用 AdamW，学习率 2e-4，Batch_Size 64。对于非 CRF 采用交叉熵损失函数，计算损失时 mask 掉句子中 padding 的部分。对于 CRF 采用 NegLog-Loss。统计的 F1 采用 micro F1。

2.1.1 不加 CRF

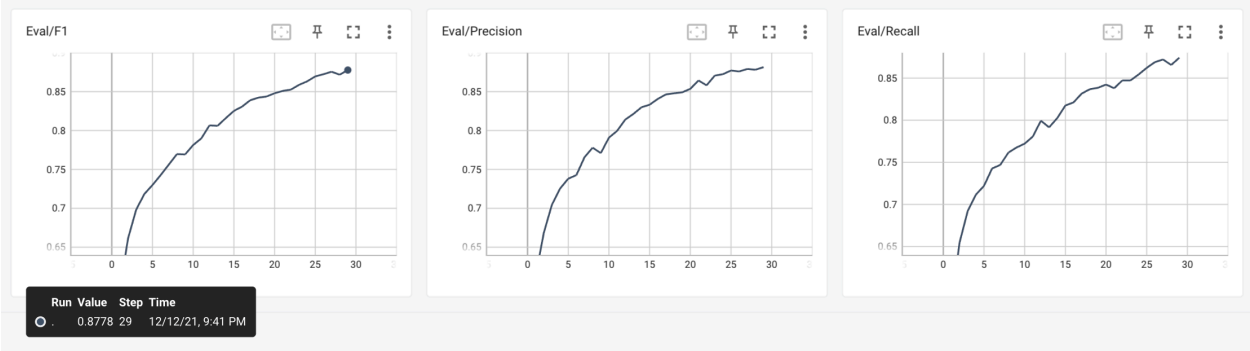

图 4: 无 CRF 结果

结果: F1 Score 87.78%，Precision 88.15%，Recall 87.41%

2.1.2 加 CRF

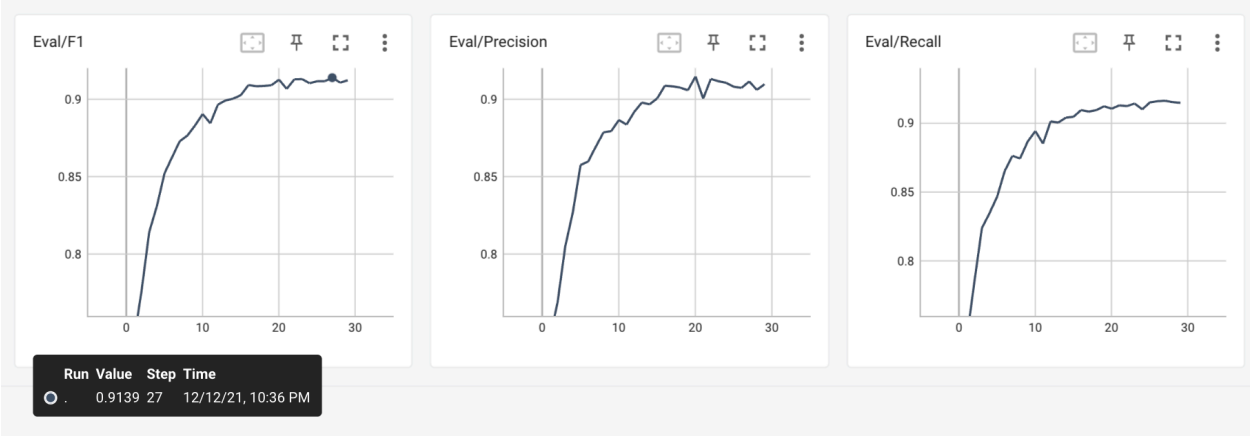

图 5: 添加 CRF 结果

结果: F1 Score 91.39%，Precision 91.14%，Recall 91.64% 最好性能在第 27 个 Epoch

对比可以发现，添加 CRF 的模型其表达能力和提取上下文信息的能力更强，从而提升了模型性能。在 F1 Score、精准率、召回率上都有一定的提升。

2.2 超参数搜索

在本实验中采用 Ray Tune 来进行超参数搜索。超参数搜索可以通过 grid/random search 以及贝叶斯搜索，我使用了类似 grid search 的方法确定参数范围，同时进一步采用了贝叶斯搜索来更加精细化的搜索参数范围。由于 grid search 要求每个样本点都进行测试，而由于计算资源有限，所以通过随机选取了部分样本点进行测试。

贝叶斯搜索采用高斯过程，考虑之前的参数信息，不断地更新先验；网格搜索未考虑之前的参数信息。相比网格搜索迭代次数少，速度快，对于非凸优化效果更好。实验中主要对 Batch Size，Learning Rate 和 DropOut 参数进行了搜索。搜索目标优化是最大化 F1 Score。

2.2.1 Grid Search

类 Grid 搜索设置搜索 Batch Size 在 [16, 32, 64] 中选取，Learning Rate 是在 1e-3 到 1e-5 之间按正态取值，DropOut 在 [0.1, 0.2, 0.5] 中选取。（由于 Learning Rate 是按正态分布挑选，所以不算非常严格的 grid search，类似 Random 和 Grid 的结合）

Trial ID	Show Metrics	ray/tune/f1	learning_rate	batch_size	drop_out	embed_dropout
launch_wrapper...	☐	0.92179	0.00056222	32.000	0.50000	0.50000
launch_wrapper...	☐	0.91700	0.00028228	32.000	0.50000	0.50000
launch_wrapper...	☐	0.91486	0.00056272	64.000	0.20000	0.50000
launch_wrapper...	☐	0.90924	0.00028335	32.000	0.10000	0.20000

由于计算和时间资源限制，进行了 4 次搜索。可以看到当学习率在 5e-4，batch_size 在 32，drop_out 为 0.5 时，F1 Score 达到最大 92.179%

2.2.2 Bayes Search

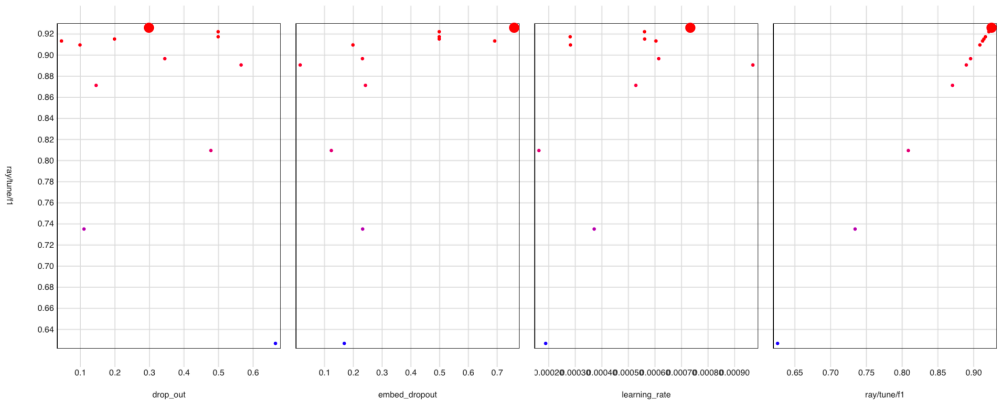

图 6: 贝叶斯搜索散点图

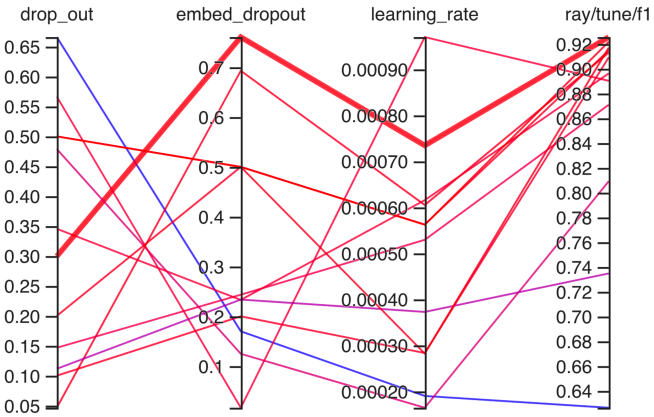

图 7: 贝叶斯参数分布

Trial ID	Show Metrics	ray/tune/f1	drop_out	embed_dropout	learning_rate
lstm_crf_bay/la...	☐	0.92556	0.29963	0.76057	0.00073467
lstm_crf_searc...	☐	0.92179	0.50000	0.50000	0.00056222
lstm_crf_searc...	☐	0.91700	0.50000	0.50000	0.00028228
lstm_crf_searc...	☐	0.91486	0.20000	0.50000	0.00056272
lstm_crf_bay/la...	☐	0.91299	0.046467	0.69294	0.00060510
lstm_crf_searc...	☐	0.90924	0.10000	0.20000	0.00028335
lstm_crf_bay/la...	☐	0.89624	0.34556	0.23298	0.00061573
lstm_crf_bay/la...	☐	0.89026	0.56646	0.016468	0.00097021
lstm_crf_bay/la...	☐	0.87094	0.14672	0.24339	0.00052951
lstm_crf_bay/la...	☐	0.80919	0.47893	0.12481	0.00016443
lstm_crf_bay/la...	☐	0.73479	0.11160	0.23372	0.00037270
lstm_crf_bay/la...	☐	0.62639	0.66595	0.16987	0.00019001

图 8: 贝叶斯搜索结果

可以看到在学习率 7e-4, Dropout(FC)=0.7/Embed_DropOut=0.2 的时候模型有最好的效果，F1 Score 达到 92.556%。实验中可以发现，大学习率和大 dropout 有利于模型的学习与收敛。在后续实验中都将学习率设为 6e-4 且 Dropout 为 0.5，经过实验可以达到较为理性的效果。

2.3 Char Representation 和 ELMo

进一步实验中，添加字符表示，来进一步提升模型性能。两个模型后都添加了 CRF decoder 模块。

2.3.1 Char Representation

按照论文 [1] 设置 Char Embedding 接 Dropout 和 Conv2d，其中 kernel size 设为 $(3, max\_word\_len)$，可以融合临近字符的信息，最后接 MaxPooling 和 GloVe Word Embedding 进行连接之后输入 LSTM。

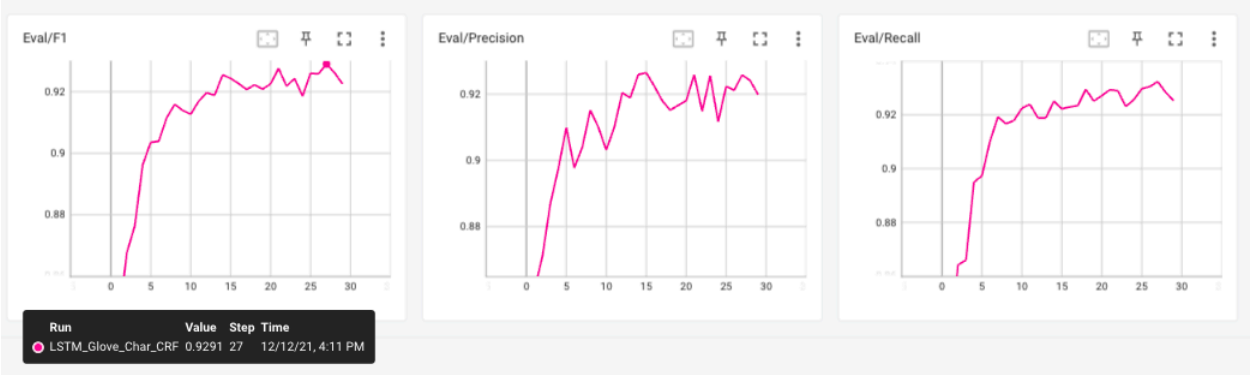

图 9: Char Representation

可以看到在添加了字符表示后，相比纯 CRF 模型性能有了进一步提升。F1 Score: 92.91%, Precision: 92.58%, Recall: 93.23%

2.3.2 ELMo

按照论文 [2]，采用预训练的参数权重，使用 fastnlp 框架搭建嵌入层，代替 Char Representation 和 GloVe，冻结预训练参数，同时开启多层 LSTM 之间输出的权值训练进一步提高模型性能。参数设置：BatchSize=64、LR=6e-4、DropOut=0.5

图 10: ELMo 模型结果

ELMo 预训练词嵌入使得 NER 模型性能得到了极大提升，F1 Score 最高达到了 94.62%，Precision 达到 94.26%，Recall 达到 94.43%。可以看到好的词向量提取到更多句子的语义信息对于 NER 任务模型性能的提升是十分关键的。

3 实验结果

模型名称	$F1\ Score(micro)/\%$	$Precision/\%$	$Recall/\%$
BaseLine	87.78	88.15	87.41
CRF	91.39	91.14	91.64
Char Rep	92.91	92.58	93.23
ELMo	94.62	94.26	94.43

结论

NER 任务作为 NLP 基础任务可以为其他众多 NLP 下游任务提供支持，在本次实验中我们探索了利用 LSTM-CRF 模型进行 NER-BIO 序列标注。我们进一步探索了 Encoder-Decoder 模型结构，采用 CRF(Condition Random Field) 进一步提取序列上下文信息来进一步提升模型性能。同时在实验中可以看到良好的词嵌入向量有助于模型提取句子词和字符中的语义信息和相互联系。即使是非常简单的单层 LSTM 模型，只要有良好的词嵌入（ELMo）也可以在序列标注任务上达到较高的性能。

参考文献

[1] Xuezhe Ma, Eduard Hovy. (2016). End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF.

[2] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer. (2018). Deep contextualized word representations.