汉语词语自动切分方法的实现与分析

2021K8009937003 骆敏言

一、 FMM 性能测试与分析

1.1 文件

主文件：FMM.py
前置处理：
1.vocabulary.py 处理得到与语料库的所有词汇。
2.pre_processdata.py 处理得到去除所有空格和日期的语料。
后续处理：
存入 FMM_result.txt 文件和 FMM_result.json 文件，便于后续浏览子词压缩处理。

1.2 算法

代码实现了中文文本的正向最大匹配分词算法：

首先接收一个词典作为参数，并将其转换成集合以加速查找。遍历输入文本，每次选择最大长度的词进行匹配，直到找到词典中的词或长度减为 1。将匹配到的词添加到结果列表中，并截取剩余文本进行下一轮匹配。

```python
class ForwardMaximumMatching:
    def __init__(self, dictionary):
        self.dictionary = set(dictionary)

    1 个用法
    def cut(self, text):
        result = []
        max_len = max(len(word) for word in self.dictionary)

        while text:
            # 选择最大长度的词进行匹配
            word = text[:max_len]
            while word not in self.dictionary and len(word) > 1:
                word = word[:-1]

            result.append(word)
            text = text[len(word):]

        return result
```

图 1：FMM 算法

1.3 结果分析

我们以 1998.01.01 这天的新闻为例，以下是 FMM 处理的结果：

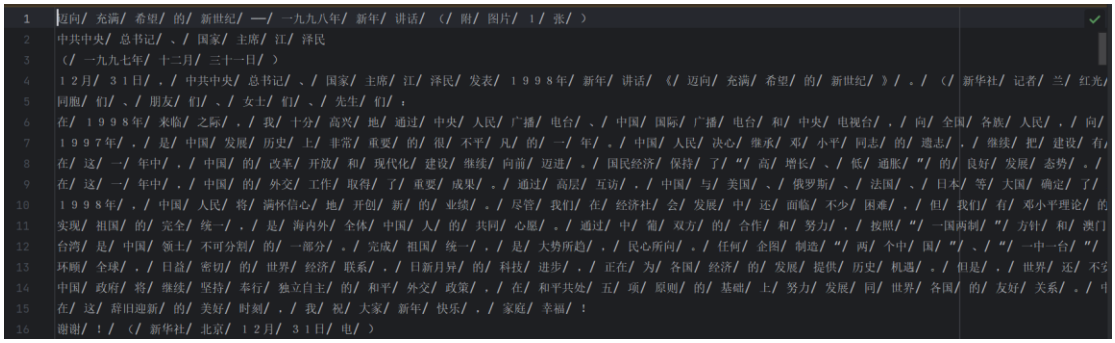

图 2：FMM 处理结果

可以看到，分词结果表现非常好。相较于模型更加复杂的 LSTM 方法，表现更好，可能的原因如下：

1. 分词前的语料是人工精确标注的，得到的词汇表质量很高。此外，由于文本数量很大，词典的完备性很高。
2. 每一行的句子长度较短，最大匹配长度设置为了词典中最长词语的长度，因此不存在遗漏，模型表现很好。

二、 BiLSTM-CRF 性能测试与分析

2.1 文件

主文件：LSTM.py、LSTMgpu-all-100.py
前置处理：

1. segment.py 处理得到 segmentes_text.json 文件得到单独每个字的列表
2. label.py 处理得到 label.json 文件，得到每个字的 BMES 标签

后续处理：

1. 使用 logging 将每轮 epoch 训练的 loss 和最终测试集的预测标签存入 LSRM-all-100.log 文件
2. 将每轮 epoch 训练后的模型存入 path_to_save_model-all-100-epoch.pth 文件中。

2.2 算法

代码实现了一个基于 BiLSTM-CRF 的中文分词模型：

BiLSTM_CRF 类定义了一个继承自 nn.Module 的模型类，包含了一个双向 LSTM 层和一个条件随机场 (CRF) 层。使用了 Embedding 层将词语映射为向量，LSTM 层接收 Embedding 层的输出，得到 LSTM 的输出特征。将 LSTM 输出映射到标签空间的线性层，将转移参数矩阵用于 CRF。[1]

读取预处理过的文本和标签数据得到数据集。将数据集以 9：1 的比例划分为训练集和测试集。进行多轮训练，每轮都遍历训练集，计算损失并进行梯度更新。每次训练完以后应当保存模型。

使用训练好的模型对测试集进行分词预测。使用维特比算法解码出最优的分词结果。将真实标签和预测标签记录在日志中。

2.3 结果分析

（1）1000 句

数据集规模过大，由于算力的限制，我先选取了前 1000 个句子进行训练和测试（训练时间 18h）。运行 log 文件见 LSTM-1000.log，模型见 path_to_save_model-1000.pth。

Loss 从 40.72 下降至 5 左右，loss 的变化趋势如下图所示。可以看到在 50 至 250 轮之间，loss 保持在 10 附近。猜测可能的原因是出现了梯度消失问题，但催着 LSTM 模型本身遗忘机制的参与，可能在 250 轮以后解决了这一问题。

在 10 个句子这一较小的数据集上预测时知，loss 需下降到 1 以下，预测结果才准确。继续增大 epoch 的轮数可以提高预测准确度。适当改进初始模型参数，也许有助于模型更快收敛。

[1] 参考自 https://pytorchchina.com/2019/04/12/advanced-making-dynamic-decisions-and-the-bi-lstm-crf/

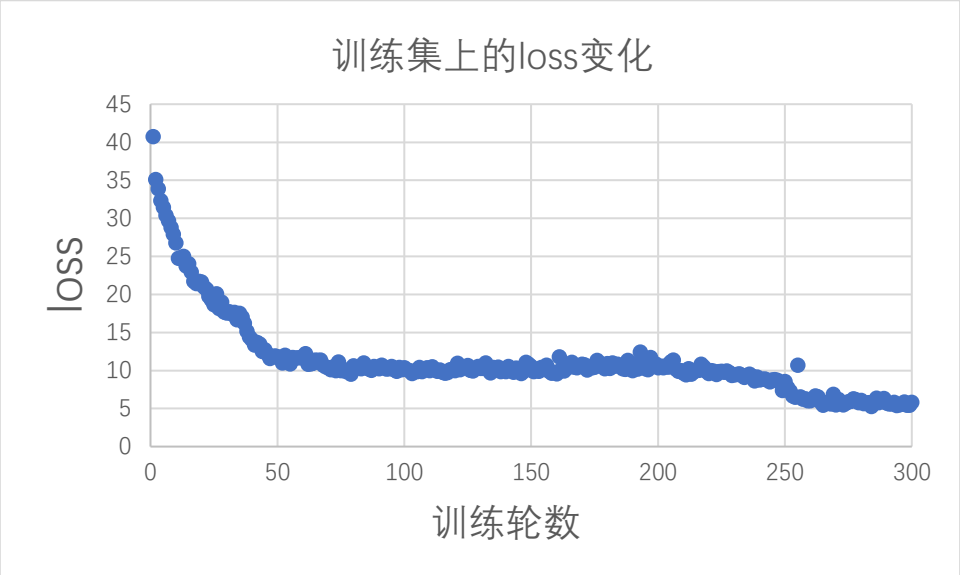

图 3：1000 个句子训练集上的 loss 变化

我们挑选一个分词结果进行分析:

```
INFO:root:True Tags: ['B', 'E', 'B', 'E', 'B', 'E', 'B', 'E', 'S',
'B', 'E', 'B', 'M', 'M', 'M', 'E', 'B', 'M', 'E', 'B', 'M', 'E', 'S',
'B', 'M', 'E', 'B', 'E', 'B', 'E', 'S', 'B', 'M', 'M', 'E', 'B', 'E',
'B', 'E', 'B', 'E', 'B', 'E', 'B', 'E', 'B', 'E', 'S', 'B', 'E', 'S',
'S', 'B', 'M', 'M', 'E', 'S', 'S', 'B', 'E', 'S', 'S', 'B', 'E', 'B',
'E', 'S', 'S', 'B', 'E', 'B', 'E', 'B', 'E', 'S', 'S', 'B', 'M', 'M',
'E', 'B', 'E', 'S', 'B', 'M', 'M', 'E', 'B', 'E', 'B', 'M', 'M', 'M',
'M', 'M', 'E', 'S', 'S', 'S', 'B', 'E', 'S']↓
INFO:root:Predicted Tags: 本报/ 昆明/ 1月/ 2日/ 电/ 截至/ 1 9 9 7年/ 1 2
月/ 3/ 1日/ ，/ 云南省/ 全面/ 完成/ 了/ 1 9 9 8年度/ 党报/ 党刊/ 订/ 阅发/ 行
/ 任务/ ，/ 实现/ 了/ “/ 稳中/ 有/ 升/ ”/ 的/ 目标/ 。/ 具初/ 步统计/ ，/ 在/
上年/ 订数/ 基础/ 上/ ，/ 1 9 9 8年度/ 的/ 人民/ 日报/ 完成/ 1 0/ 0. 6 2%
/ 。/ （/ 云宣/ 新/ ）/ ↵
```

图 4：1000 个句子训练集上的训练结果分析

训练结果:

本报/昆明/1 月/2 日/电，截至/1997 年/12 月/3/1 日/，/云南省/全面/完成/了/1998 年度/党报/党刊/订/阅发/行/任务，/实现/了/“/稳中/有/升/”/的/目标/。/具初/步统计/，/在/上年/订阅/基础/上/，/1998 年度/的/人民/日报/完成 10/0.62%/。/（/云宣/新/）/

该句的正确划分应为:

本报/昆明/1 月/2 日/电，截至/1997 年/12 月/31 日/，/云南省/全面/完成/了/1998/年度/党报/党刊/订阅/发行/任务，/实现/了/“/稳中有升/”/的/目标/。/具/初步/统计/，/在/上年/订阅/基础/上/，/1998/年度/的/人民日报/完成 100.62%/。/（/云/宣新/）/

对比可知，训练结果基本可以反应句意，但在一些数字、时间和专有名词的处理上略欠。模型目前存在欠拟合的问题，继续训练可以提高分词精度。

注意：由于我们分割了训练集和测试集，部分测试集中的句子字不在词典中，采用<UNK>标签替代。实验表明，未被词典包含的字较少，对分词准确度影响不大。

（2）全部语料

数据集规模过大，由于算力的限制，当选取了所有句子进行训练和测试时，即使使用 gpu 训练速度仍然较慢。我训练了 10 轮（每轮训练时间 1.5h）。运行 log 文件见 LSTM-all-100.log，训练模型见 path_to_save_model-all-100-epoch10.pth。

由于训练数据较大，模型的初始 loss 较小，因此训练的轮数应少于前一例子。巡练的 loss 从 4.2 下降至 3.2，结果如下：

```
INFO:root:Epoch 1, Batch Loss: 4.2493133544921875
INFO:root:Epoch 2, Batch Loss: 3.7276153564453125
INFO:root:Epoch 3, Batch Loss: 3.375579833984375
INFO:root:Epoch 4, Batch Loss: 3.317230224609375
INFO:root:Epoch 5, Batch Loss: 3.256561279296875
INFO:root:Epoch 6, Batch Loss: 3.171844482421875
INFO:root:Epoch 7, Batch Loss: 3.2442169189453125
INFO:root:Epoch 8, Batch Loss: 3.35736083984375
INFO:root:Epoch 9, Batch Loss: 3.5800018310546875
INFO:root:Epoch 10, Batch Loss: 3.273193359375
```

图 5：全部语料的训练 loss

可以看到 Loss 总体趋势逐渐下降，存在小范围内的抖动，可能是由于学习率过大跳过了最优解，或者是部分部分数据噪声扰动。随着训练轮数的增加，loss 会逐渐下降，分词精度逐渐提高。

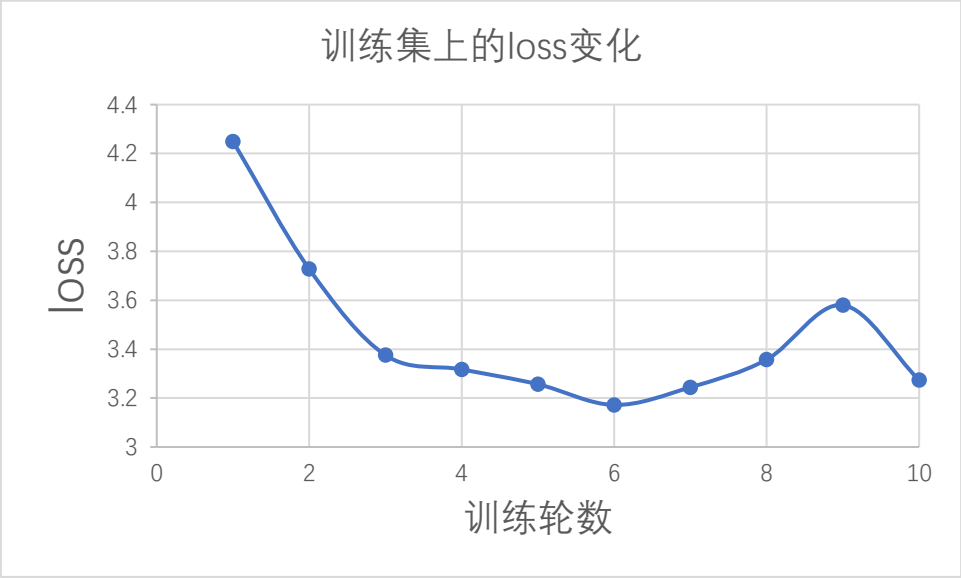

图 6：1000 个句子训练集上的 loss 变化

由于我们没有训练完成，需要将模型导入 beiyong.py 文件获取当前模型下的分词结果。我们挑选一个分词结果进行分析：（由于随机分配，上一例中的例句这一例中未在测试集中，否则可以纵向对比二者效果）

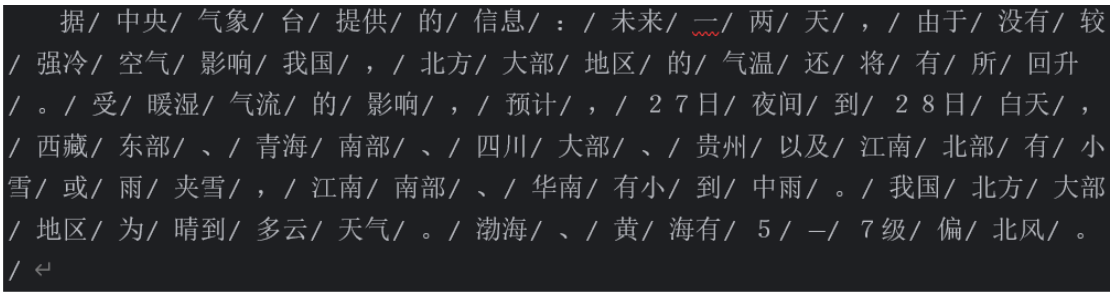

图 7：全部语料训练集上的训练结果分析

训练结果：

据/中央/气象/台/提供/的/信息/：/未来/一/两/天/，/由于/没有/较/强冷/空气/影响/我国/，/北方/大部/地区/的/气温/还/将/有/所/回升/。/受/暖湿/气流/的/影响，/预计/，/27 日/夜间/到/28 日/白天/，/西藏/东部/、/青海/南部/、/四川/大部/、/贵州/以及/江南/北部/有/小雨或/雨/夹雪/，/江南/南部/、/华南/有小/到/中雨/。/我国/北方/大部/地区/为/晴到/多云/天气/。/渤海/、/黄海/有/5/——/7 级/偏/北风/。/

该句的正确划分应为：

据/中央/气象台/提供/的/信息/：/未来/一两/天/，/由于/没有/较/强/冷空气/影响/我国/，/北方/大部/地区/的/气温/还/将/有所/回升/。/受/暖湿气流/的/影响/，/预计/，/27 日/夜间/到/28 日/白天/，/西藏/东部/、/青海/南部/、/四川/大部/、/贵州/以及/江南/北部/有/小雪/或/雨夹雪/，/江南/南部/、/华南/有/小到中雨/。/我国/北方/大部/地区/为/晴到多云/天气/。/渤海/、/黄海/有/5/——/7/级/偏/北风/。/

可以看到整体的预测结果不错，但在如“强冷空气”、“暖湿气流”、“雨夹雪等”天气专有名词表现不佳。模型目前处于欠拟合状态，继续训练可以提高分词精度。

对比两个模型可以发现，当使用全部预料时，训练数据多，即使循环次数较少，loss 也比使用较小数据量的模型小。但训练时间过长，是一个需要改进的点。可以尝试从优化模型结构，减少层数等方面入手。

三、 子词压缩

3.1 文件

主文件：BPE-FNN.py
前置处理：
FMM.py 处理得到 FMM_result.json 文件
后续处理：
使用 logging 将每次合并的词语和最终合并效果输入 BFE-FNN.log 文件

由于 LSTM 处理得到的预测结果仅原语料的 1/10，我们选用 FNN_result 作为子词压缩的源文件。如果需要查看 LSTM 子词压缩结果，请查看 BFE-LSTM.py 文件和 BFE-LSTM.log 文件。（与 FNN 相比，对于文件读入处理略有不同）

3.2 算法

将输入的 context_split 中的文本按照字符拆分，并在每个字符之间插入分隔符 |，形成 context_raw 列表。在每次迭代中，统计字符对的频次，并找到频次最高的字符对。将找到的频次最高的字符对进行合并，形成新的字符，同时更新子词压缩词典。打印每次迭代的合并信息，以及子词压缩词典的当前状态。如果某次迭代中找到的字符对的频次为 1，打印提示信息并终止迭代。

使用 logging 模块记录每次迭代的合并信息和子词压缩词典的状态。在最后一次迭代结束后，将最终的子词压缩词典记录到日志。

我迭代了 1000 次，这里截取末尾 20 次迭代进行展示，子词压缩结果如下：

图 8：FMM 子词压缩

可以看到，合并结果能较好地符合语义。得到“共产党”、“你们”、“感谢”、“必要”等词语。整体来说，子词压缩找到了那些介于字和词之间的元素，有助于我们进一步的语言处理。

四、 补充功能：子句压缩

4.1 文件

主文件：BPE-phrase.py
前置处理：
FMM.py 处理得到 FMM_result.json 文件
后续处理：使用 logging 将每次合并的词语和最终合并效果输入 BFW-phrase.log 文件

4.1 算法及分析

一开始似乎理解错了老师的意思，我对分词得到的结果进行了词级别的“子词压缩”，得到一些出现频率较高的词组。

思路与第三部分类似，这里不再重复，运行 log 文件见 BFE-phrase.log。
我迭代了 1000 次，这里截取末尾 10 次迭代进行展示，子句压缩结果如下：

图 9：FMM 子句压缩

可以看到一些合并是很有道理的，如名词：高科技、新时期；人名：童志成；地名：巴以；时间：1 月 22 日等。

但标点符号的结合似乎有些奇怪。由于标点符号前后往往句意关系不够紧密，因此这样的结合应当想办法去除。这是模型的改进方向之一。

五、 不足与改进

总的来说，模型在分词任务上取得了不错的表现。存在以下几点需要改进：

1. 由于算力的限制，模型其实没有运行，应当继续训练模型，以获得更好的分词结果。
2. 人民日报的文本比较官方，语言措辞较为正式。覆盖性更强、更全面的语料有助于我们提高分词模型的性能。
3. 尝试简化模型，提高训练速度。