语义槽抽取

学号：1143710207

姓名：宋治勋

本实验采用双向LSTM作为基础模型，实现对话系统中语义槽抽取的系统。编程语言为python，采用pytorch作为深度学习框架。

**一、任务分析**

语义槽抽取本质上属于自然语言处理中的序列标注任务，采用循环神经网络对其进行训练，可以使用交叉熵损失函数作为学习目标，将每个语义槽看作一个类别。因此，对每个单词进行序列标注预测，也可以当成对每个单词进行分类，类别总数是语义槽名称的个数。

综上，本实验可以映射为序列标注任务，通俗来讲，即单词分类任务。

**二、模型构建**

首先读取atis\_slot\_names.xlsx，共计83个语义槽名称，再加上非语义槽名称“O”,总共有84个类别。读取atis.train.txt，atis.test.txt，构建词典，过滤低频词，并在词典中添加<UNK>，<PAD>字符，并且以每个句子为单位，根据设定的max\_len值对句子进行<PAD>补齐，之后将输入数据数字化处理，句子中的每个单词替换为字典索引，语义槽名称替换为类别索引。最后构建数据批处理加载器，使得每次传入模型的数据大小为（batch\_size, max\_len）。

模型的数据输入已经构建完成，接下来要对模型的逻辑进行实现。模型包括三个组件，embedding层，biLSTM层，fully connect层。Embedding层为每个单词初始化一个大小为embedding\_size的向量，将模型的输入从（batch\_size, max\_len）扩大为（batch\_size, max\_len, embedding\_size）。之后将数据输送到biLSTM层，biLSTM可以是堆叠多层，但是基础模型只设置一层biLSTM，取最后一层的biLSTM的output做数据输出，数据的大小为（batch\_size, max\_len, hidden\_size \* 2）。最后将数据输送到fc层，fc层的权重矩阵为（hidden\_size \* 2，num\_class），因此最后模型的输出是大小为（batch\_size, max\_len, class）的数据。并将其与真实的label数据输送到交叉熵损失函数，进行反向传播。为了计算损失函数，需要将数据从句子粒度转换为单词粒度。因为pytorch提供的交叉熵损失函数自带了softmax层选择类别，所以不需要对模型的输出进行进一步的类别提取。

最后是模型输出的后处理，需要去除数据中<PAD>字符，并且将数据中索引转换为真实的token以及真实的语义槽。构建输出文件，应用评测脚本conlleval.pl对模型进行评测。

本实验模型的参数设置如表2-1所示：

表2-1 参数设置

|  |  |  |
| --- | --- | --- |
| 参数名称 | 参数值 | 参数意义 |
| epoch | 20 | 训练周期 |
| batch\_size | 64 | 批处理大小 |
| embedding\_size | 300 | 词向量大小 |
| hidden\_size | 150 | biLSTM的隐状态大小 |
| num\_layers | 1 | biLSTM的层数 |
| max\_len | 50 | 句子的最大长度 |
| learning\_rate | 0.001 | 除embedding之外的模型参数初始学习率 |
| embedding\_learning\_rate | 0.001 | Embedding的初始学习率 |
| lr\_decay | 0.75 | 学习率下降 |
| dropout\_rate | 0.5 | dropout |

**三、实验结果**

模型运行情况如图3-1所示：

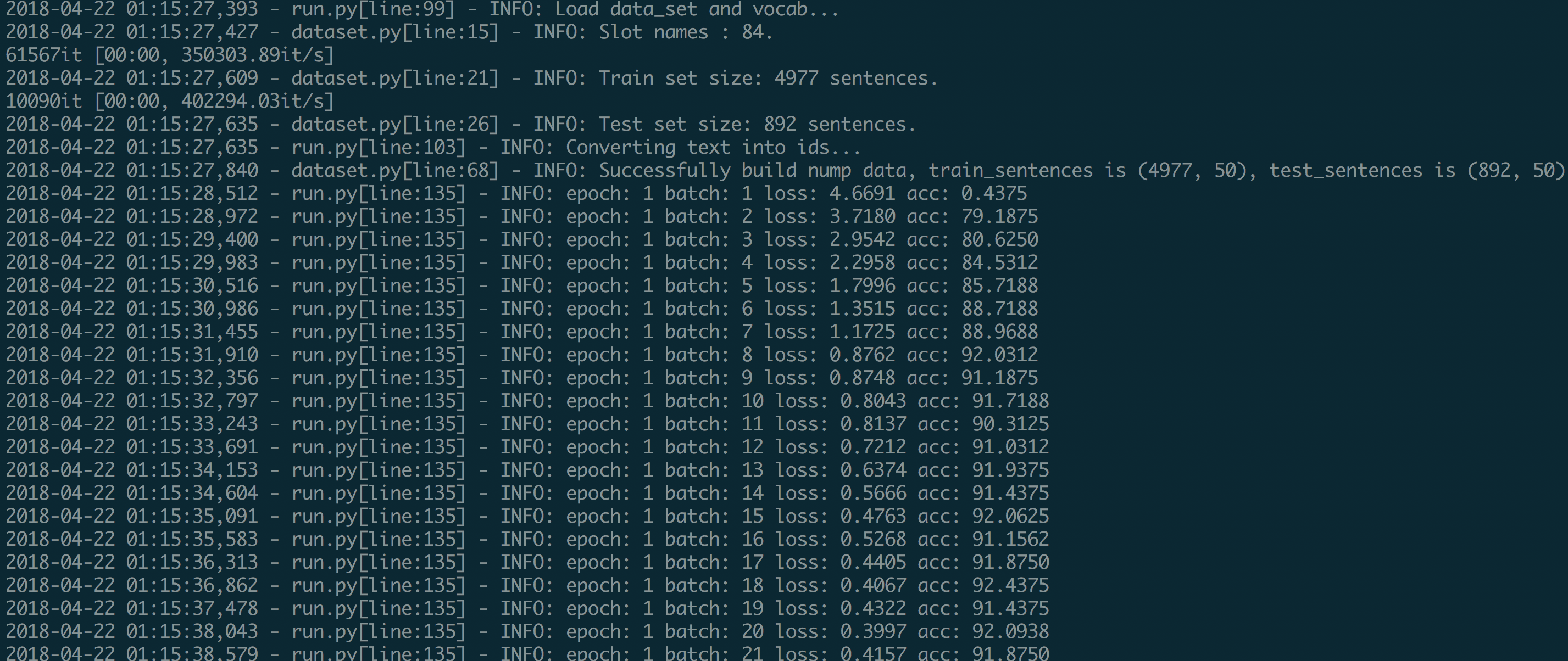


图3-1 模型运行情况截图

模型的收敛情况如图3-2所示：

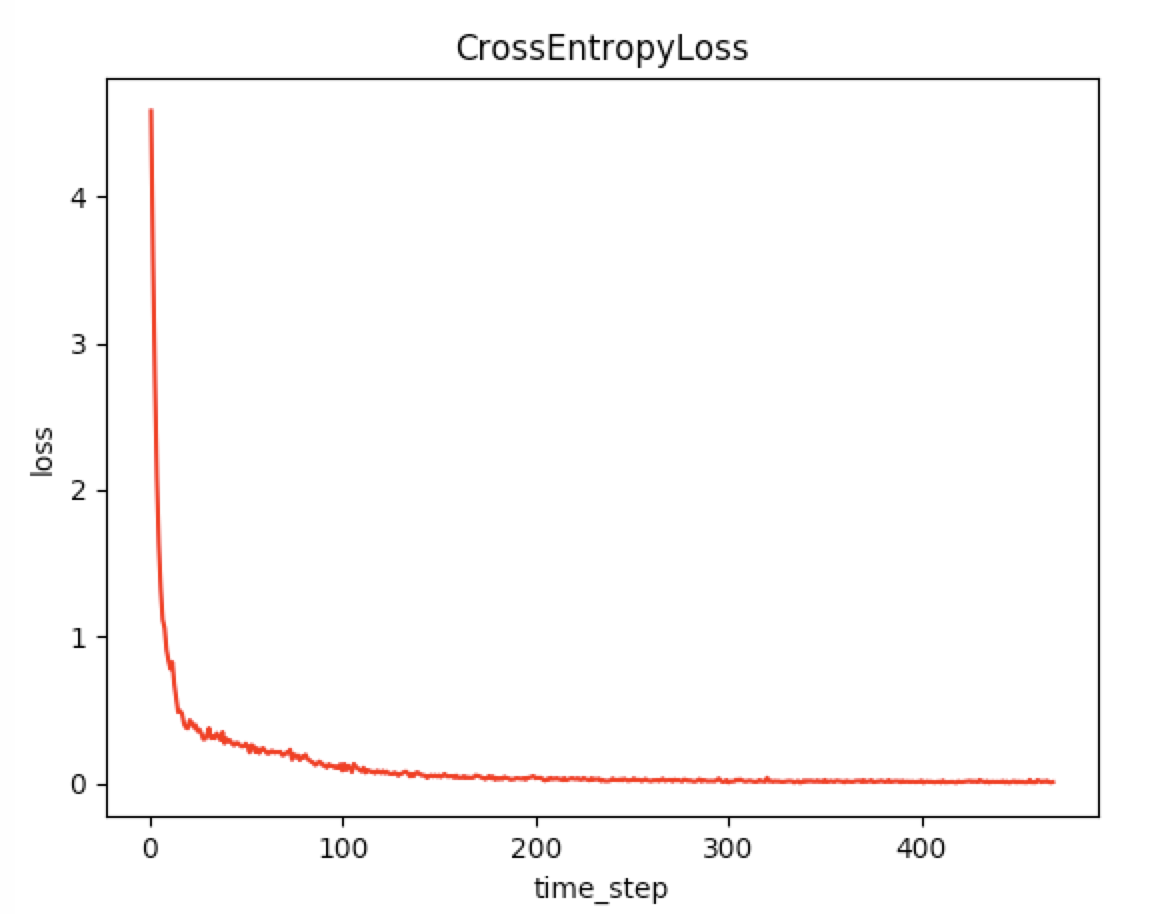


图3-2 模型收敛情况绘图

测试集的得分情况如图3-3，图3-4所示

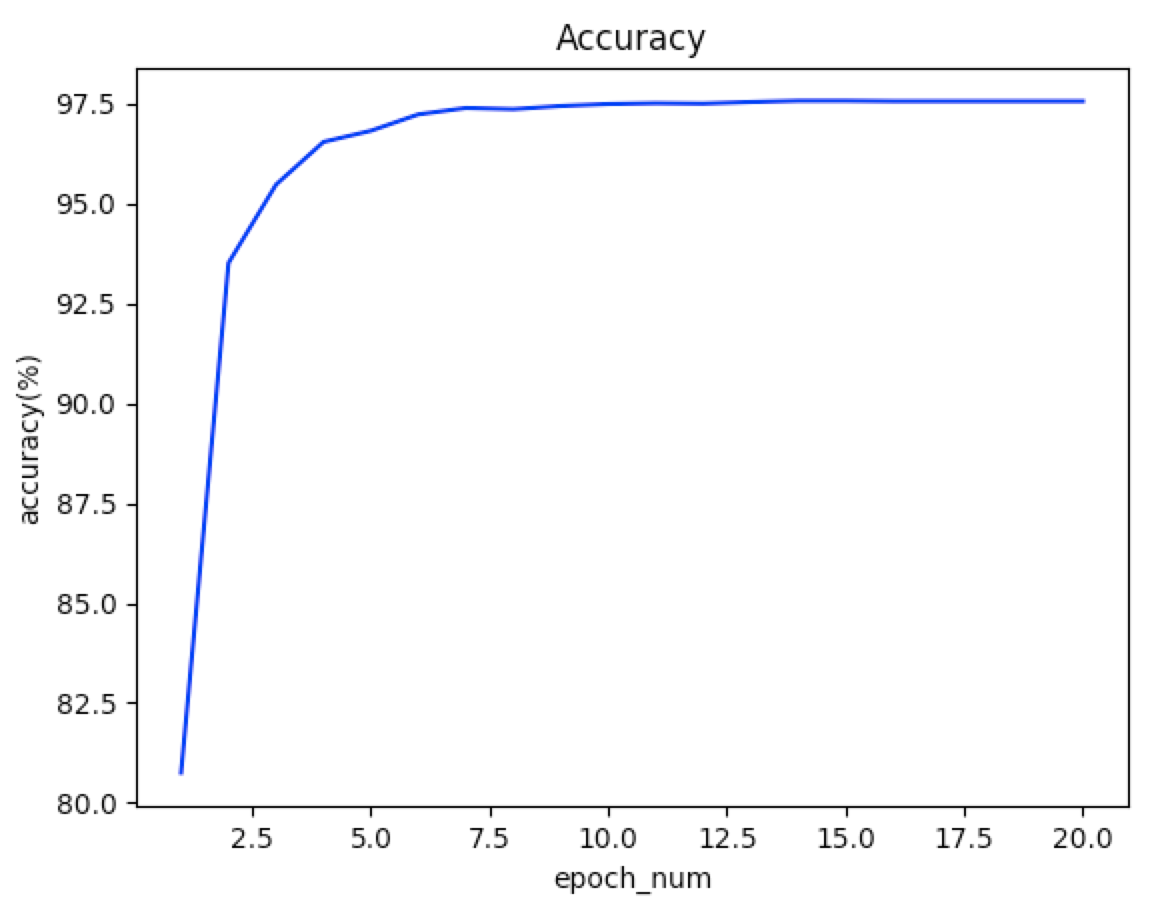


图3-3 测试集得分情况绘图

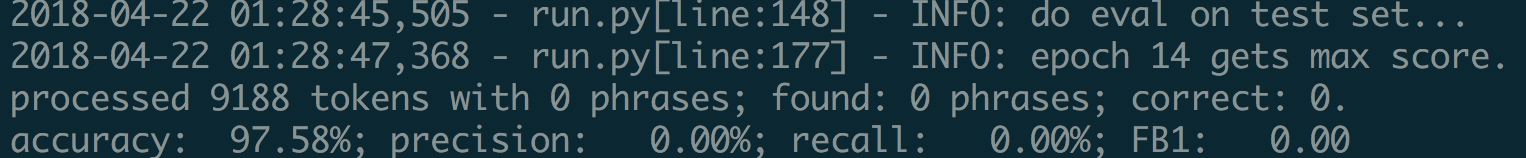


图3-4 测试集最高得分截图

由此可以，随着epoch的增加，biLSTM模型迅速收敛，收敛速度逐渐降低，并在第14个epoch后，模型在测试集的分数达到最高，后边的epoch得分趋于平缓，最高准确率为97.58%。。