作业四报告

Requirement 1：

本次作业实现很简单，只要参照官方文档中《详细指南》部分即可。

本次实验我使用了 The 20 newsgroups text dataset 中全部训练数据并按 9:1 划分训练集和验证集。数据的预处理基本等同于 assignment-2， 这里不再赘述。总共训练集大小为 10183，验证集大小为 1131，测试集大小为 7532，词表大小为 19679。

RNN 文本分类

我采用了多层 BiLSTM，将每层最终的 hidden_state concatenate 作为句子的特征，过全连接层进行分类。其基本结构如下：

Embedding->LSTM->Dropout->FC

实验中我采用了一下模型参数进行训练：

embedding dim	hidden size	dropout rate	Epochs
128	128	0.5	20

考虑到在该数据集上深度网络很容易过拟合，因此尝试加入 L2 正则化，最终训练结果如下：

Layers	Weight_decay	Dev_Accuracy	Test_Accuracy
1	0	0.837	0.622
2	0	0.822	0.630
1	1e-4	0.833	0.632
2	1e-4	0.832	0.650

可以看出模型在数据集上存在明显的过拟合，而使用 L2 正则化可以一定程度上缓解该问题。

CNN 文本分类

CNN 文本分类我采用了经典的 TextCNN 模型[1]，其本质是在句子上使用不同 size 的 kernel 做一维卷积来提取特征（虽然 embedding 层后的输入是二维的，但在 feature 维做卷积并没有意义）。实验中我使用 kernel_size=(3,4,5)。

TextCNN 有许多改进版本，有时会采用 K-Max Pooling：原先的 Max Pooling 从 Convolution 层一系列特征值中只取最大的那个值，而 K-Max

[1] Yoon Kim. 2014. Convolution Neural Networks for Sentence Classification.

Pooling 可以取所有特征值中得分在 Top-K 的值，在 models.py 中我实现了使用 K-Max Pooling 的 TextCNN 并测试效果。

训练参数与 RNN 基本类似，采用 L2 正则化，最终训练结果如下：

K-Max Pooling	Kernel_num	Dev_Accuracy	Test_Accuracy
1	(25,25,25)	0.851	0.749
1	(50,50,50)	0.880	0.755
1	(100,100,100)	0.870	0.744
2	(25,25,25)	0.880	0.763
2	(50,50,50)	0.869	0.775

可以看出 K-Max Pooling 要稍好于直接 Max Pooling。同时在这个问题上，CNN 表现更好，且比 RNN 训练速度快很多，是一个不错的方法。

Requirement 2:

相比于其它 NLP 框架如 Allennlp、HanLP，我觉得 fastNLP 最大优点是有一个非常清晰全面的文档，这使得上手比较容易。在迅速搭建一个供深度网络训练平台方面，我觉得 fastNLP 做得很好，但是 fastNLP 似乎还缺少一些 NLP 传统工具例如繁简转化、快捷分词等。另外非常期望能看到 fastNLP 能支持更多的 state of the art 的 pretrained model 以及 contextualized embedding（似乎现在只支持 BERT）。

此外，还有一些关于 fastNLP 的细节建议如下：

1. 在封装好的 Trainer 中，当使用验证集时，如果在训练过程中发生如 CUDA out of memory 等异常，程序仍然会继续执行，最后会抛出如下异常：

```
Traceback (most recent call last):
  File "main.py", line 65, in <module>
    trainer.train()
  File "C:\Users\dell\AppData\Local\Programs\Python\Python36\lib\site-packages\fastNLP\core\trainer.py", line 537, in tr
ain
    self.tester._format_eval_results(self.best_dev_perf), )
  File "C:\Users\dell\AppData\Local\Programs\Python\Python36\lib\site-packages\fastNLP\core\tester.py", line 182, in _fo
rmat_eval_results
    for metric_name, metric_result in results.items():
AttributeError: 'NoneType' object has no attribute 'items'
```

在没有使用 callback 的情况下，遇到这样的错误会令新手困惑。

2. 建议在封装好的 trainer 中对应 print_every(iteration)设置 print_every_epoch，这样比较方便（个人感觉一般都是每个周期打印一次平均 loss）。

3. 封装好的 trainer 似乎只支持 fastNLP 内部的 Loss 函数，我觉得兼容一下对 pytorch loss function 的支持。

4. 建议在 Vocabulary 里增加开始符<sos>结束符<eos>的保留位。