文本分类作业实验报告

数据集选择

1. 本次文本分类任务选择的是外卖平台用户评论的情感分类数据集【正面情感 4000+；负面情感 8000+】，数据集地址如下：
   a) https://raw.githubusercontent.com/SophonPlus/ChineseNlpCorpus/master/datasets/waimai_10k/waimai_10k.csv
2. 数据描述：本数据集共两个字段。label 表示标签，1 表示正面，0 表示负面；review 则是用户评论文本。

a)

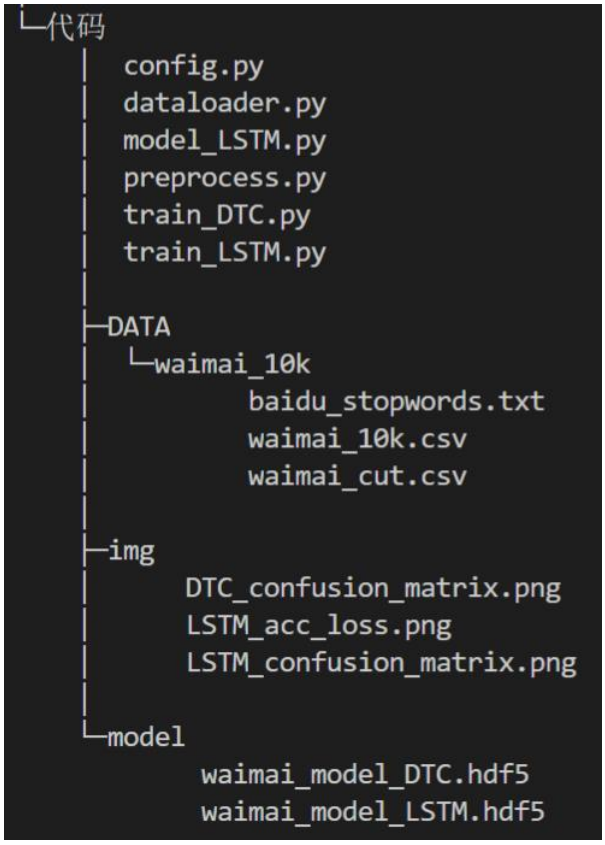
```
label,review
1,很快，好吃，味道足，量大
1,没有送水没有送水没有送水
1,非常快，态度好。
1,方便，快捷，味道可口，快递给力
1,菜味道很棒！送餐很及时！
1,今天师傅是不是手抖了，微辣格外辣！
1,"送餐快,态度也特别好,辛苦啦谢谢"
1,超级快就送到了，这么冷的天气骑士们辛苦了。谢谢你们。麻辣香锅依然很好吃。
1,经过上次晚了2小时，这次超级快，20分钟就送到了.....
1,最后五分钟订的，卖家特别好接单了，谢谢。
1,量大，好吃，每次点的都够吃两次
```

实验环境

1. Python 3.8.10
2. 在 www.autodl.com 中的 JupyterLab 进行实验，具体实验环境如下：

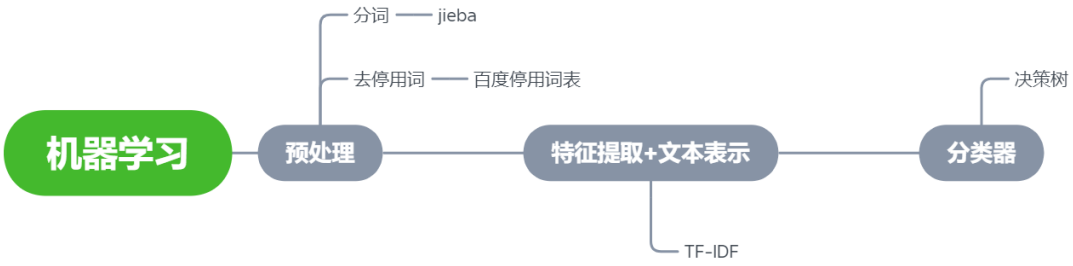
镜像	TensorFlow 2.5.0　Python 3.8(ubuntu18.04)　Cuda 11.2
GPU	RTX 2080 Ti * 1　显存: 11GB
CPU	12核 Intel(R) Xeon(R) Platinum 8255C CPU @ 2.50GHz　内存: 43GB
默认硬盘	系统盘: 25 GB　数据盘: 免费:50GB SSD 付费:0GB
附加磁盘	无
端口映射	无
网络	上行宽带: 10MB/s　下行宽带: 10MB/s

3. 代码文件层次如下



模型构建

1. 本次分类使用了机器学习和深度学习两种方法。
   a) 机器学习：决策树模型



   b) 深度学习：LSTM 模型

c) 数据预处理：preprocess.py

i. 使用 jieba 库进行分词

ii. 使用自行修改后的百度停用词表（baidu_stopwords.txt）处理停用词。

iii. 预处理前后的结果如下

```
       label                                          review
0          1                                  很快，好吃，味道足，量大
1          1                                 没有送水没有送水没有送水
2          1                                    非常快，态度好。
3          1                               方便，快捷，味道可口，快递给力
4          1                                 菜味道很棒！送餐很及时！
...      ...                                            ...
11982      0                          以前几乎天天吃，现在调料什么都不放，
11983      0  昨天订凉皮两份，什么调料都没有放，就放了点麻油，特别难吃，丢了一份，再也不想吃了
11984      0                                凉皮太辣，吃不下都
11985      0                                本来迟到了还自己点！！！
11986      0        肉夹馍不错，羊肉泡馍酱肉包很一般。凉面没想象中好吃。送餐倒是很快。

[11987 rows x 2 columns]
Building prefix dict from the default dictionary ...
Loading model from cache /tmp/jieba.cache
Loading model cost 0.603 seconds.
Prefix dict has been built successfully.
       label                                          review
0          1                                 很快 好吃 味道 足 量 大
1          1                                      送水 送水 送水
2          1                                       快 态度 好
3          1                               快捷 味道 可口 快 递给 力
4          1                                 菜 味道 很棒 送餐 很
...      ...                                            ...
11982      0                                  天天 吃 调料 都 不放
11983      0  昨天 订 凉皮 两份 调料 都 放 放 点 麻油 特别 难吃 丢 一份 再也 不想 吃
11984      0                                凉皮 太辣 ， 吃不下 都
11985      0                                   本来 迟到 还 点
11986      0    肉夹馍 不错 羊肉 泡馍 酱肉 包 很 凉面 没 想象 中 好吃 送餐 倒 很快
```

d) 划分测试集训练集：

i. 决策树：训练集 80%，测试集 20%，随机划分。

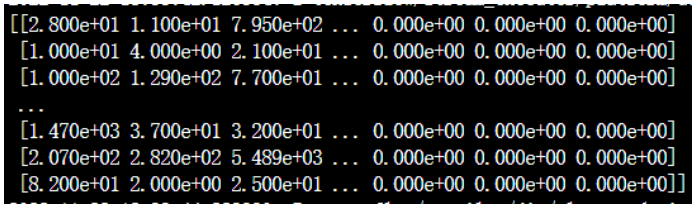
ii. LSTM：训练集 80%，验证集 5%，测试集 15%，随机划分。

e) 特征抽取+文本表示：

i. 决策树：采用 TF-IDF 进行特征抽取，使用 sklearn 中的 TfidfVectorizer 模型实现。部分结果如下：

```
[11987 rows x 2 columns]
  (0, 7724)     0.8452482868957322
  (0, 4535)     0.4438518522257745
  (0, 2916)     0.2975749767295281
  (1, 7214)     0.6254971181590783
  (1, 4519)     0.780226476848029
  (2, 7696)     0.2631841771358394
  (2, 7389)     0.1997432438728141
  (2, 5912)     0.5524924132181089
  (2, 5499)     0.2928838585884865
  (2, 3286)     0.34843153948648964
  (2, 3183)     0.21642811910804852
  (2, 2940)     0.46980716163601666
  (2, 2916)     0.19957947864659972
  (2, 196)      0.2664798271470743
  (3, 7274)     0.11961125477224328
  (3, 6019)     0.1302203761640057
  (3, 5503)     0.19141769519363794
```

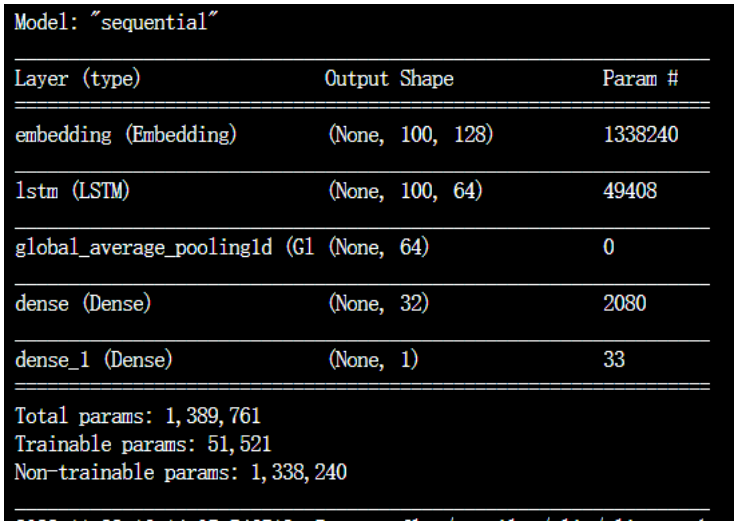
ii. LSTM：采用 word2vec 构建词向量，使用 gensim 中的 Word2Vec 模型实现，限制词向量维数为 128。进行填充及截断后,每句话表示为 50 个向量。部分结果如下：



f) 分类模型：

i. 决策树：设置树的最大深度为 10，使用基尼系数划分节点。

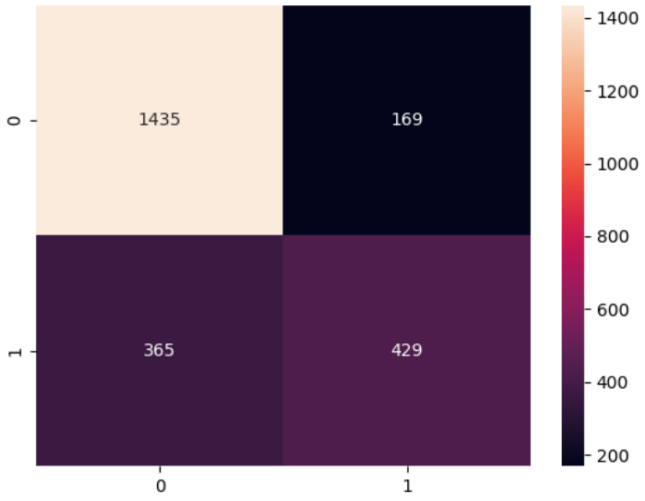
ii. LSTM 模型(model_LSTM.py)：模型结构如下所示，网络结构由嵌入层、LSTM 层、全局平均池化层及两个全连接层构成。此外，针对深度学习模型的优化器和损失函数，我分别使用了 Adam 优化器和 binary_crossentropy 二分类交叉熵损失函数。最后设定评价指标为准确率 accuracy。



模型评估

1. classification_report 解释：
   a) accuracy 准确率 即正确预测样本量与总样本量的比值。
   b) precison 表示精确率，precison=TP/(TP+FP)
   c) recall 表示召回率或者敏感度，sensitivity=TP/(TP+FN)
   d) F-score 就是 precision 和 recall 的 harmonic mean（调和平均值）
   e) support 意思为支持，也就是说真实结果中有多少是该类别
   f) macro avg 表示宏平均，表示所有类别对应指标的平均值
   g) weighted avg 表示带权重平均，即类别样本占总样本的比重与对应指标的乘积的累加和。
2. 机器学习——决策树

a)



b)

```
Sum time is 0.105804
在训练集上的准确率：0.79476
              precision    recall  f1-score   support

           0       0.80      0.89      0.84      1604
           1       0.72      0.54      0.62       794

    accuracy                           0.78      2398
   macro avg       0.76      0.72      0.73      2398
weighted avg       0.77      0.78      0.77      2398

在测试集上的准确率：0.77731
[[1435  169]
 [ 365  429]]
Accuracy:  77.7314%
Precision: 89.4638%
Recall:    79.7222%
F1:        84.3126%
```

c) 该模型训练用时约 0.1 秒，获得测试集上的准确率约为 78%。

d) 多指标综合来看，该模型在测试集上的表现尚可，准确率有待提高。

e) 从分类指标来看，负面评论的预测质量低于正面评论的预测质量，尤其在 recall 这个指标上表现明显，这可能是正负面评论数据量不均导致的差异。

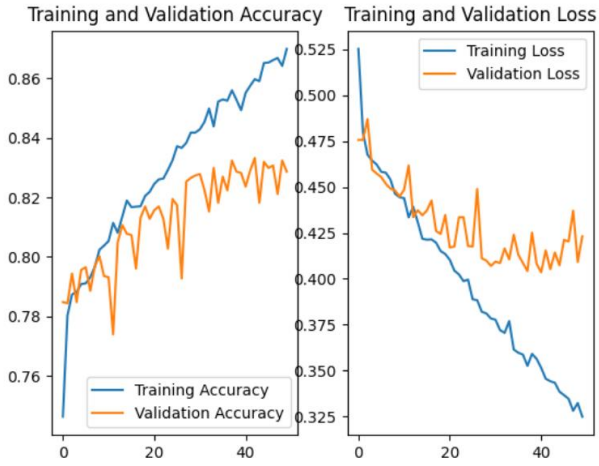
3. 深度学习——LSTM

a) 共训练 50 个 epoch，训练过程中，在训练集和验证集上的表现如下：可见在 30 轮所有，验证集的准确率和损失函数已趋于平稳。

i.

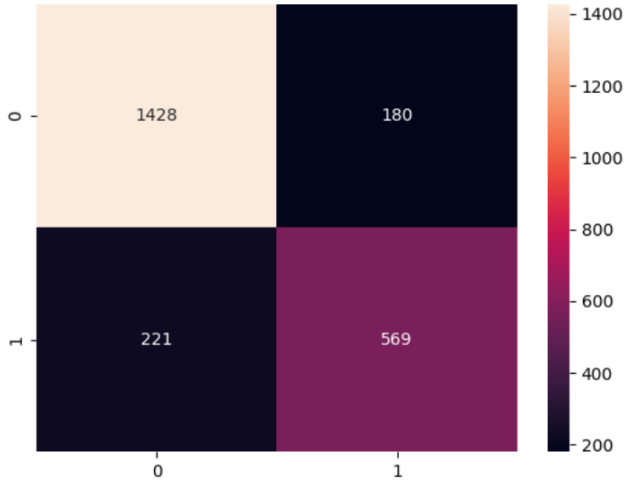
```
225/225 [==============================] - 1s 4ms/step - loss: 0.4466 - accuracy: 0.7944 - val_loss: 0.4594 - val_accuracy: 0.8011
Epoch 8/50
225/225 [==============================] - 1s 5ms/step - loss: 0.4456 - accuracy: 0.7972 - val_loss: 0.4596 - val_accuracy: 0.8044
Epoch 9/50
225/225 [==============================] - 1s 4ms/step - loss: 0.4420 - accuracy: 0.8035 - val_loss: 0.4572 - val_accuracy: 0.8078
Epoch 10/50
225/225 [==============================] - 1s 4ms/step - loss: 0.4469 - accuracy: 0.7944 - val_loss: 0.4627 - val_accuracy: 0.7990
Epoch 11/50
225/225 [==============================] - 1s 4ms/step - loss: 0.4299 - accuracy: 0.8112 - val_loss: 0.4551 - val_accuracy: 0.8053
```

ii.



b) 最终训练结果的混淆矩阵如下。可以看出将真实值为负面评论的数据中，被预测成正面评论的仍占有一定比例；而真实值为正面评论的数据的分类表现更好。

i.



c) 以下是该模型的评价指标：

i.

```
Sum time is 51.502190
75/75 [==============================] - 0s 2ms/step - los
损失: 0.40243983268737793, 准确率: 0.8327773213386536
WARNING:tensorflow:Model was constructed with shape (None,
='embedding_input', description="created by layer 'embeddi
              precision    recall  f1-score   support

           0       0.87      0.89      0.88      1608
           1       0.76      0.72      0.74       790

    accuracy                           0.83      2398
   macro avg       0.81      0.80      0.81      2398
weighted avg       0.83      0.83      0.83      2398

[[1428  180]
 [ 221  569]]
Accuracy:  83.2777%
Precision: 88.8060%
Recall:    86.5979%
F1:        87.6881%
```

ii. 该模型训练用时约 51.5s，测试集中准确率约为 83%。

d) 对比分析

i. 多指标综合来看，决策树模型在测试集上的表现尚可，准确率等指标均有待提高。LSTM 模型准确率显著高于决策树，Recall 和 F1 值也高于决策树，在精确率上两个模型基本持平。

ii. 从混淆矩阵来看，在两个模型上，负面评论的预测质量低于正面评论的预测质量，尤其在 recall 这个指标上表现明显，这可能是正负面评论数据量不均导致的差异。

4. 优化方向

a) 负面样本量较少，可以适当进行数据增强后重新训练。

b) 可尝试不同词向量进行数据改进，也许会有不同的训练效果。

c) 尝试更多模型，评估多模型表现。