Lab1 使用了 Loglinear 模型和 TFIDF 特征完成了给定的文本分类任务。本实验报告解释了在 Lab1 中所实现的具体方法、进行实验和参数调优的细节、以及最终的性能评价。

1 方法

1.1 数据清洗

任务给出了训练与测试的新闻数据集。首先对每条数据的标题与内容进行分词和原形处理。这个过程中借助了 NLTK 工具包。分词前清除了数据集中带有的分隔符（例如，"/"）。原形处理需要借助词性标记，因此这一过程使用了 NLTK 提供的 POS 标注工具，这一工具利用了基于 WordNet 的模型进行标注。

同时，还在训练集上统计了词汇表，以及词汇表中每个词出现的次数，供后续特征提取使用。

1.2 特征提取

基于训练集上统计的词汇表信息，提取新闻的 TFIDF 词频特征。

首先，在词汇表中选取出现次数多于 500 的词，作为 unigram。然后，排除其中不适合被利用进行特征提取的 unigram。其中一部分通过 NLTK 工具包下 wordnet 的 stoplist 模块排除，这部分主要是没有区分意义的代词，冠词等。另一部分则是通过手动去除，例如人名，媒体信息 (例如普遍出现的 Reuters)，数字，被意外分开的连接词前缀 (例如 ex, anti)。

这样产生了 1024 个可用的 unigram，记为 $U = \{u_i\}_{i=1}^{1024}$。对于 u_i，记 $N = 120000$ 为样本总数，标题与内容包含 u_i 的训练集样本个数为 m_i。由此得到逆文档频率

$$\mathrm{IDF}_i = \log_{10} \frac{N}{m_i}$$

对于一条给定的新闻数据实例 $I = (\mathrm{Label}, \mathrm{title}, \mathrm{description})$，统计其标题与内容中每个 unigram 的出现次数 t_i。由此得到词频率

$$\mathrm{TF}_i = 1 + \log_{10} t_i$$

I 的 TFIDF 特征定义为 $F = [f_i]_{i=1}^{1024},\ f_i = \mathrm{TF}_i \times \mathrm{IDF}_i$。其特征长度为 1024。

按照课程中的定义方式，上述 TFIDF 特征应定义为 $F = [f_{i1}, f_{i2}, f_{i3}, f_{i4}]_{i=1}^{1024}$，其中 $f_{i1} = f_{i2} = f_{i3} = f_{i4}$。其长度为 4096。为了实现的方便，Lab1 使用了前一种定义，而四个类别分别设置模型参数。显然两种方式是等价的，前一种有助于减少运算量，并更容易实现。

1.3 Loglinear 模型

模型的参数为 $W = \{w_{ij}\}_{1024\times 4}$。对于样本 I 及其特征 $F = [f_i]_{i=1}^{1024}$，模型预测其属于第 c 类的概率为

$$p(c|F) = \frac{\exp q_c}{\sum_{j=1}^{4} \exp q_j}, c = 1, 2, 3, 4$$

$$q_j = \sum_{j=1}^{1024} w_{ij} \cdot f_i,\ j = 1, 2, 3, 4$$

1.4 性能度量

使用准确率与 Macro-F1 度量模型在目标数据集上的表现。记 $T = \{t_{ij}\}_{4\times 4}$，其中 t_{ij} 代表标签为 i 并被模型预测为 j 的样本个数。准确率定义为

$$\text{Accuracy} = \frac{\sum_{i=1}^{4} t_{ii}}{\sum_{i=1}^{4}\sum_{j=1}^{4} t_{ij}}$$

Macro-F1 为，

$$\text{Macro-F1} = \frac{1}{4}\sum_{i=1}^{4} \text{F1}_i$$

$$\text{F1}_i = \frac{2P_iR_i}{P_i + R_i}, P_i = \frac{t_{ii}}{\sum_{a=1}^{4} t_{ai}}, R_i = \frac{t_{ii}}{\Sigma_{a=1}^{4} t_{ia}}$$

2 实验

2.1 训练方法

Loglinear 模型的参数初始化是随机的，使用 batch 梯度下降方法训练模型参数。使用交叉熵损失函数进行训练，并使用了 L2 正则化项，即

$$L = \sum_{i=1}^{B} -\log p_{t_i} + \lambda||W||_2$$

其中 p_{t_i} 为模型预测其属于所属类别的概率。

Lab1 没有实现自动微分过程，也没有使用自动微分工具包。由于训练函数是固定的，其负梯度为

$$-\frac{\partial L}{\partial w_{nc}} = \sum_{i:t_i=c} f_n - \sum_{i=1}^{B} f_n p_c - \lambda w_{nc}, \quad n = 1, \cdots, 1024,\ c = 1, 2, 3, 4$$

因此训练时直接按照这一负梯度更新参数。

为了优化训练性能，Lab1 实现了学习率指数衰减方法。在每一 epoch 训练结束后以固定衰减率 γ 减小学习率。这有助于使模型在训练前期较快优化性能，并在训练后期拥有较小的学习率，防止出现性能震荡现象。

2.2 训练参数

训练过程所需的全部参数包括，训练 epoch 数 M，Batch 大小 B，学习率 α，正则化系数 λ，衰减率 γ。

Lab1 将数据集分为 3 部分，训练集，测试集，验证集。大小分别为 112400，7600，7600。使用训练集训练模型，在验证集上验证性能，并据此性能调优上述参数。

具体来说，初步调优固定 $B = 32,\ M = 50,\ \alpha = 0.01,\ \gamma = 0.99$。在此基础上，在 $\{0.001, 0.01, 0.1, 1\}$ 范围内尝试正则化系数，并最终选取 $\lambda = 0.001$

3 结果

在测试集上测试模型性能。考虑到模型初始化，以及训练过程中的随机性，以相同参数训练模型 10 次并在测试集上测试性能。以下分别为在验证集和测试集上的效果。

在验证集上，平均准确率为 0.869，平均 F1 为 0.869。二者方差均小于 10^{-6}。在测试集上，平均准确率为 0.867，平均 F1 为 0.867，二者方差均小于 10^{-6}

可以发现，尽管模型初始化与数据集迭代过程存在随机因素，但模型的最终性能波动很小，稳定在 86% 以上。

在测试集上的性能数据按照要求与 sklearn 中的 F1 计算模块结果数据比对确认，结果在 result 目录下。

4 运行脚本

由于 Lab1 进行数据清洗和预处理时使用了第三方库，这需要另外配置环境；以及进行了特征选择时经过了人工筛选，这不方便用脚本完成。出于以上两点因素，提交的压缩包里直接包括了预处理后的数据，脚本不执行数据清洗的过程。

项目根目录下有 train.sh 与 test.sh 两个脚本。train.sh 脚本基于预处理后的数据训练十次模型，并将模型保存在 model 目录下，结果输出在 result 目录下的文件中。

这一训练过程在 CPU 上完成，一次完整的训练约耗时 5 小时，脚本运行完成约耗时两天。因此，提交的压缩包还包括了训练得到的模型数据，在 model 目录下。test.sh 脚本基于清洗过的测试数据和模型数据运行，耗时较短，其结果同样输出到 result 目录的文件中。