政务留言的分类及挖掘分析

摘要：大数据时代的到来，令群众反馈信息的方式发生了变化。为更高效、更直观地处理群众们通过网络问政平台反馈意见而产生的大量文本数据，本题结合自然语言处理和文本分类、文本挖掘技术，建立了合适的模型以达到上述目标。

为使数据更适合于分析，本组对数据进行了清洗、分词、去停用词等预处理操作以构建合适的词向量。其中，在分词操作中，本组根据字模型和词模型对字典的不同需求分成不同的方法并列进行：一方面对数据进行 jieba 分词，利用所有词语建立一个词字典；另一方面是将所有字组成一个字字典。本组根据上述两个字典，建立停用词表，去除停用词，为每个字或词在相应字典中找到对应的索引作为替代，从而构成词向量输入模型。

为了能够根据群众留言主题和内容，对群众留言进行分类，本组建立了基于深度学习模型 FastText 的文本分类模型。在 FastText 模型仅有的一个隐含层中，本组实现了将词向量分为 word-embedding 、 2-gram-embedding 、 3-gram-embedding 三种模式并取平均；在输出层中，通过 softmax 层将数据映射为在 7 个类别上的概率来实现对留言内容根据政府职能部门体系的多分类。

为了挖掘群众留言中的热点问题，本组抓住留言三要素：时间、地点、话题（留言内容），建立 Word2Vector 模型，计算不同话题之间的相似度。其中，对话题关键词和关键地点的搜索主要依赖正则表达式实现。

为了建立公平有效的答复评价体系，对答复内容进行定量评价，本组使用 Word2Vector 模型，计算留言与回复之间的相似度，构建星氏评价指标，实现了对答复内容的量化。

此外，为优化模型、寻求最佳参数，本组对比分析了 SGD、Adam、Momentum 三种优化算法，并选用了对本组模型效果最好的 Adam 算法对模型进行了优化。

关键词：FastText 模型、jieba 分词、Word2Vector 模型、Adam 优化算法

目录

图表目录

1、简介

1.1 挖掘意义

诸多网络平台的兴起、大数据时代的到来，反馈信息的方式也发生了变化。不同于传统的问政方式，群众们也开始通过以微信、微博、市长信箱为代表的网络问政平台来表述自己的想法。而使用这些网络平台必定会产生大量的文本数据，其会给相关部门中从事留言划分整理职业的职员们带来巨大的工作量，同时人工分类还存在着低效率、高差错等缺点。于是使用应运而生的新兴工具，对这些繁杂的文本数据归类整理就显得尤为重要。

文本挖掘作为近年来信息挖掘的新兴分支，它将大量的文本数据进行划分整理并且准确的提取到工作人员所需的信息主干部分，大大缩减了人们在整合信息上面所花费的时间与精力。利用文本挖掘对文本数据进行处理，这在减少政府工作量的同时也对于政府部门的决策有着重要的作用，它带来的作用不仅仅是政府工作效率上的提高，更是对政府提升管理水平和优化重大决策有着至关重要的推动作用。

1.2 挖掘目标

本组将建立合适的模型，对获得的群众留言数据进行自然语言处理和文本挖掘，希望达到以下目标：

- 根据群众留言主题和内容，按照政府职能部门体系，对群众留言进行分类，以代替传统的人工识别分类，是大规模的留言信息能够快速、高效、差错少得反映给相关部门，提高政府工作效率。
- 对群众留言中一定时间范围内的相似问题进行归类，建立合适的热度评价指标，挖掘群众留言中的热点问题，加强对特定地点、特定人群的针对性，提高特定部门的办事效率。
- 建立公平有效的答复评价体系，从相关性、完整性、可解释性等角度，评价政府政务服务质量，以便完善政务服务体系，获得更高的群众满意度。

1.3 挖掘流程

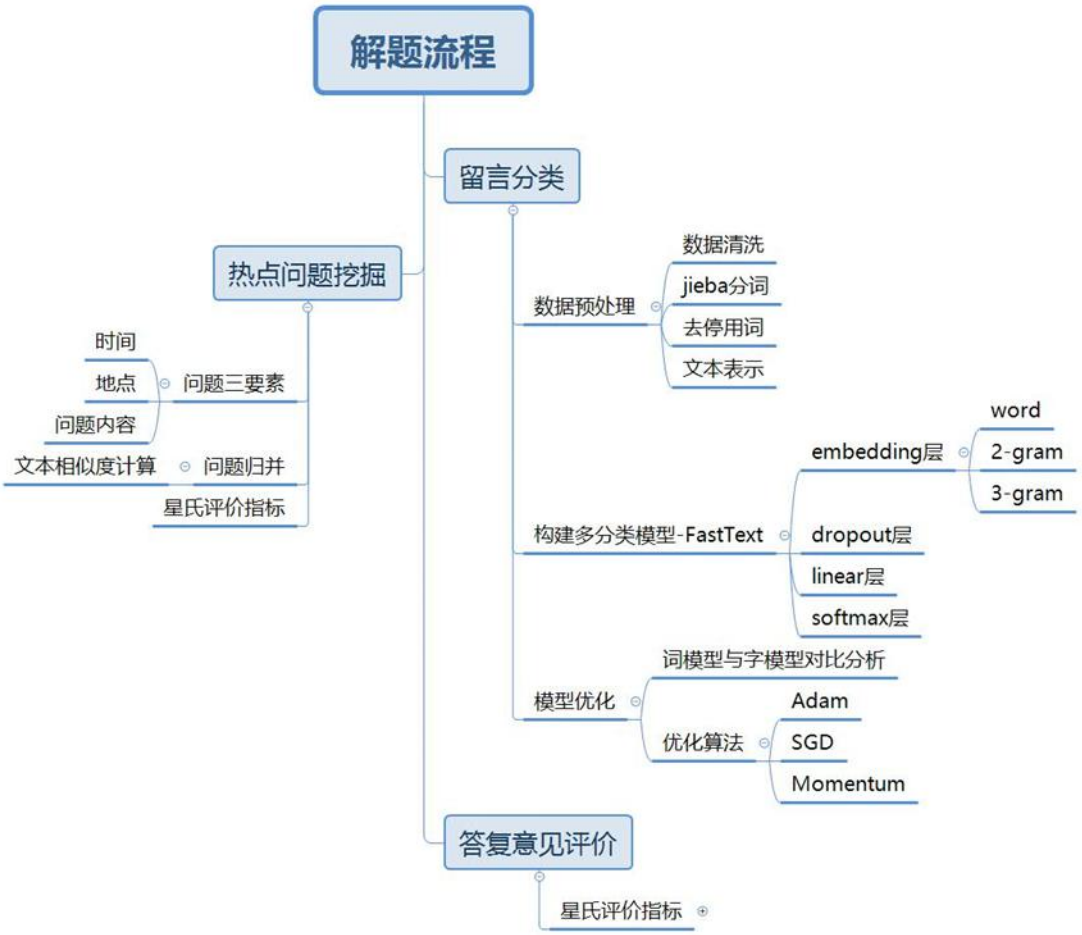

图 1.1 解题流程

2、预处理

2.1 数据清洗

由于本组用于分析的留言数据是文本格式，数据中会伴随着大量无用的字符和词，因此本组在对数据进行分析之前要对数据进行清洗，筛选出所需要的文本数据。本组使用正则表达式来清洗数据。

正则表达式也称规则表达式，是计算机科学的一个概念。正则表达式可以检索、替换符合某些模式的文本。利用正则表达式可以在事先定义好的特定字符及其组合，组成一个“规则的字符串”，以此来实现对文本数据的过滤。

2.2 分词

由于中文的每个词之间不像英文那样有空格做分割，所以在处理中文文本时首先要做的就是找到一种好的方法，根据实际要表达的语义对中文文本进行分词。分词大致分为两种：基于词典的机械切分，基于统计模型的序列标注切分两种。

（1）基于词典的机械切分

基于词典的方法本质上就是字符串匹配的方法，将一串文本中的文字片段和已有的词典进行匹配，文字片段匹配成功，则作为一个分词结果。但是该方法存在诸多问题，如：词典里匹配不到相应的词，这种问题被称为“未登录词识别”，可以通过将新词人为加入词典解决；另一个常见问题是“歧义切分”，比如“结婚的和尚未结婚的人”通过机械切分会有两种切分结果：（1）“结婚/的/和/尚未/结婚/的/人”，（2）“结婚/的/和尚/未/结婚/的/人”，第二种切分明显有歧义。因此，单纯的机械切分很难避免这样的情况发生。

（2）基于统计模型的序列标注

针对基于词典的机械切分所面临的问题，尤其是未登录词识别，使用基于统计模型的分词方式能够取得更好的效果。基于统计模型的分词方法，简单来说就是一个序列标注问题。

在一段文字中，我们可以将每个字按照他们在词中的位置进行标注，常用的标记有一下四种 label：

表 2.1 词语位置标注的四种 label

B	Begin	表示这是一个词的首字
M	Middle	表示这是一个词中间的字
S	Single	表示这是单字成词
E	End	表示这是一个词的末尾字

分词的过程就是将一段字符输入模型，然后得到相应的标记序列，再根据标记序列进行分词。举例来说：“泰迪科技专注于大数据技术研发”，经过模型后得到的理想标注序列是：“BMMEBESBMEBEBE”,最终还原的分词结果是“泰迪科技/专注/于/大数据/技术/研发”。在自然语言处理领域，解决序列标注问题的常见模型主要有 HMM 和 CRF。

针对本次比赛的数据，我们采用不同于上述两种方法的 python 的第三方分词库：jieba 分词。以下是分词的具体操作：

1、观察附件 2 中的数据，本组认为所给数据特征中的“留言编号”、“留言用户”是留言和留言者信息的标记，“留言时间”是对留言时间的标记，“留言主题”、“留言详情”、“一级标签”是基于留言内容的标记，且又“一级标签”是对留言内容的标记。为实现对留言内容的分类，本组选择留言数据的“留言主题”、“留言详情”、“一级标签”三类特征，

作为主要研究对象。

考虑到“留言主题”与“留言详情”均能表达留言者留言的内容，而“留言主题”更具概括性，且在进行分词操作时更为高效，因此，本组选择“留言主题”来代替留言内容。

在对附件 2 所给特征信息进行筛选后，本组留下“留言主题”、“一级标签”、以及能够代表留言本身信息的“留言编号”三列数据，作为下一步操作的数据

留言编号	留言主题	一级标签
24	A市西湖建筑集团占道施工有安全隐患	城乡建设
37	A市在水一方大厦人为烂尾多年，安全隐患严重	城乡建设
83	投诉A市A1区苑物业违规收停车费	城乡建设
303	A1区蔡锷南路A2区华庭楼顶水箱长年不洗	城乡建设

图 2.1 筛选特征后数据示例

2、因为本次赛题是针对整体而言，某市，某区等地域词不属于本赛题的重要元素，因此在统计整合的过程中要利用正则表达式将将脱敏后的具体地址如“A 市”“K1 区”去掉。

留言编号	留言主题	一级标签
24	西湖建筑集团占道施工有安全隐患	城乡建设
37	在水一方大厦人为烂尾多年，安全隐患严重	城乡建设
83	投诉苑物业违规收停车费	城乡建设
303	蔡锷南路华庭楼顶水箱长年不洗	城乡建设

图 2.2 正则表达式脱敏后数据示例

3、jieba 分词。根据 jieba 分词的原理，本组分以下步骤实现 jieba 分词操作：

A.基于前缀词典实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图（DAG）；

B.采用动态规划查找最大概率路径，找出基于词频的最大切分组合；基于前缀词典实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图（DAG）；

C.对于未登录词，采用基于汉字成词能力的 HMM 模型，使用了 Viterbi 算法。

完成 jieba 分词得到如下结果：

留言编号	留言主题	一级标签
24	[西湖, 建筑, 集团, 占, 道, 施工, 有, 安全隐患]	城乡建设
37	[在水一方, 大厦, 人为, 烂尾, 多年, , , 安全隐患, 严重]	城乡建设
83	[投诉, 苑, 物业, 违规, 收, 停车费]	城乡建设
303	[蔡锷, 南路, 华庭, 楼顶, 水箱, 长年, 不洗]	城乡建设

图 2.3 jieba 分词结果示例

4、去除停用词，停用词表来自四川大学机器智能实验室停用词表。这一步值得注意的是有的留言主题的词语全部出现在了停用词表中，对后续的分类造成了极其隐藏性的 bug，所以对极少数这种数据做删除处理。

留言编号	留言主题	一级标签
24	西湖 建筑 集团 占 道 施工 安全隐患	城乡建设
37	在水一方 大厦 人为 烂尾 多年 安全隐患	城乡建设
83	投诉 苑 物业 违规 收 停车费	城乡建设
303	蔡锷 南路 华庭 楼顶 水箱 长年 不洗	城乡建设

图 2.4 去停用词后数据示例

5、本组将不同一级标签用数字标记，如下表：

表 2.1 一级标签标记转换表

0	1	2	3	4	5	6
城乡建设	环境保护	交通运输	教育文体	劳动和社会保障	商贸旅游	卫生计生

6、将全部数据按 6：2：2 的比例分为训练集、测试集、验证集。这是为了能够选出效果最好的、泛化能力最佳的模型。

图 2.5 训练集数据示例

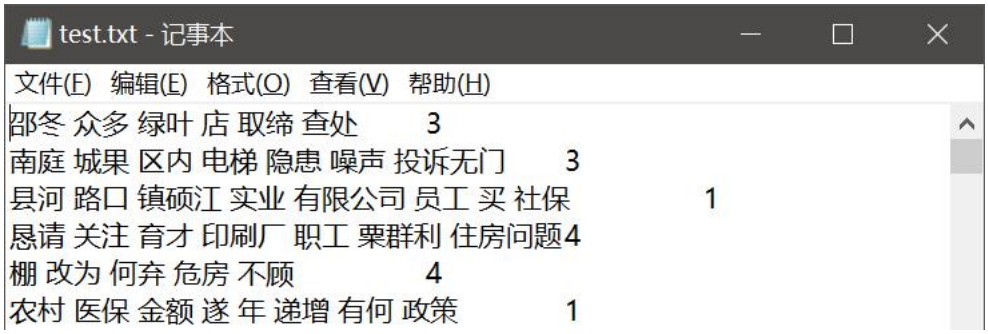

图 2.6 测试集数据示例

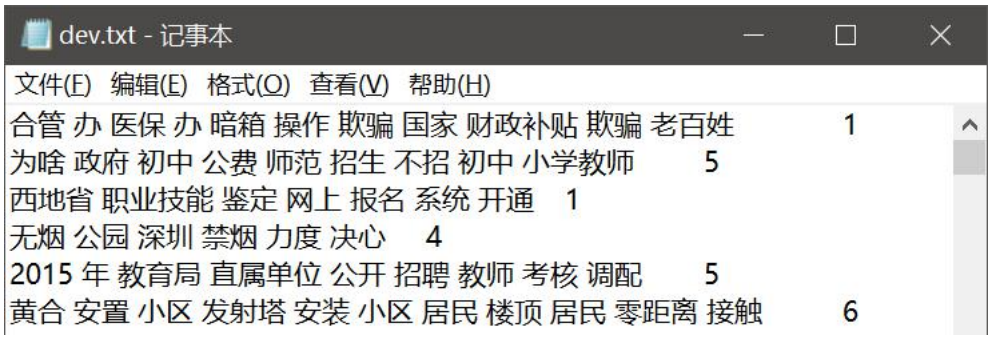

图 2.7 验证集数据示例

7、根据全部数据为每个出现过的词语分配一个索引，生成词表 word_vocab.pkl。

{'请': 0, '污染': 17, '医保': 34, '路': 51,
'西地省': 1, '违规': 18, '建设': 35, '相关': 52,
'咨询': 2, '学校': 19, '工作': 36, '公积金': 53,
'小区': 3, '人员': 20, '学生': 37, '违法': 54,
'解决': 4, '电梯': 21, '质量': 38, '扰民': 55,
'建议': 5, '居民': 22, '补课': 39, '管理': 56,
'请求': 6, '收费': 23, '教育': 40, '中心': 57,
'年': 7, '退休': 24, '市场': 41, '补贴': 58,
'镇': 8, '投诉': 25, '员工': 42, '未': 59,
'公司': 9, '办理': 26, '企业': 43, '中': 60,
'教师': 10, '乱收费': 27, '出租车': 44, '噪音': 61,
'医院': 11, '有限公司': 28, '县': 45, '待遇': 62,
'政策': 12, '传销': 29, '相关': 46, '乡镇': 63,
'职工': 13, '希望': 30, '教育局': 47, '拖欠': 64,
'村': 14, '小学': 31, '非法': 48, '\xa0': 65,
'请问': 15, '中学': 32, '公积金': 49, '发放': 66,
'社保': 16, '工资': 33, '农村': 50, '垄断': 67,

图 2.8 词表

8.、根据全部数据为每个字都分配一个对应的索引，生成字表：char_vocab.pkl。

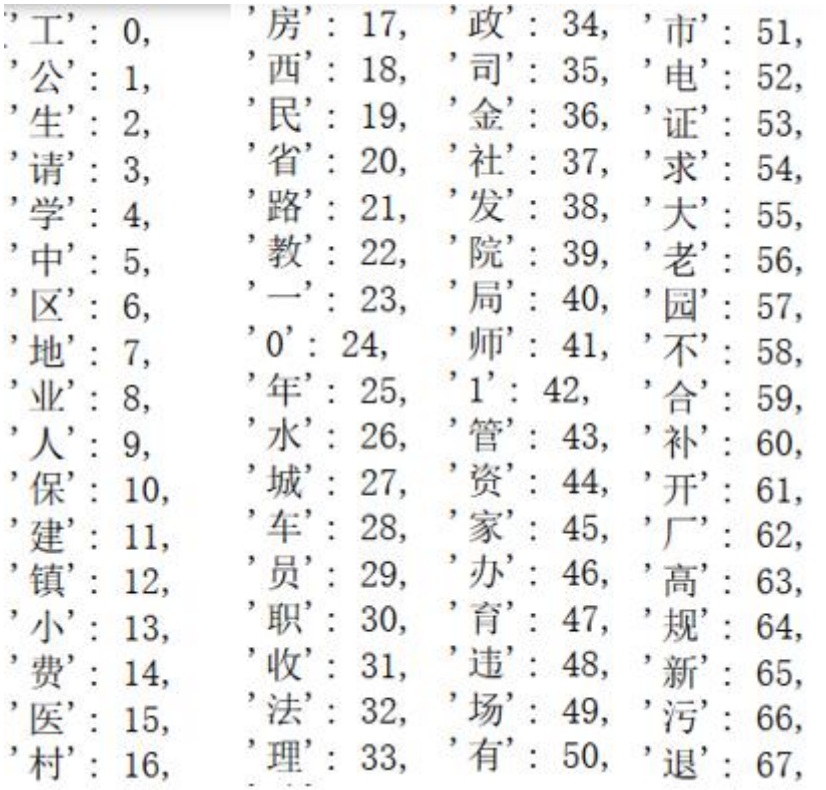

图 2.9 字表

2.3 数据规约

在数据集成与清洗后，我们可以得到整合了多个数据源同时数据质量完好的数据集。但由于数据清洗与数据集成无法改变数据集的大小，因此我们仍需要利用某些方法来降低数据的规模。数据规约是利用编码方案通过小波变换或者组成成分分析有效的压缩原始数据，或者通过特征提取技术进行属性子集的选择与重造。

这里我们可以使用数据聚集、抽样或者维归约的方式来实现数据规约。

其中数据聚集是将多个数据集合成为一个数据集，从而降低了数据计算量，同时还得到了更稳定的特征。但是这种方法也有一些缺点。例如，进行数据聚集的过程中可能会导致某些数据的消失，而这些数据可能恰巧是我们所需要的数据，因此具有风险。数据抽样则是对数据样本中的抽选，减少计算的负担，但是由于是抽选，减少计算量的同时也会导致最终结果精准性的降低。维归约则是通过减少属性的个数来对数据进行压缩。

3、基于 FastText 的文本分类模型

3.1Hidden 层

根据 FastText 模型原理，建立模型：

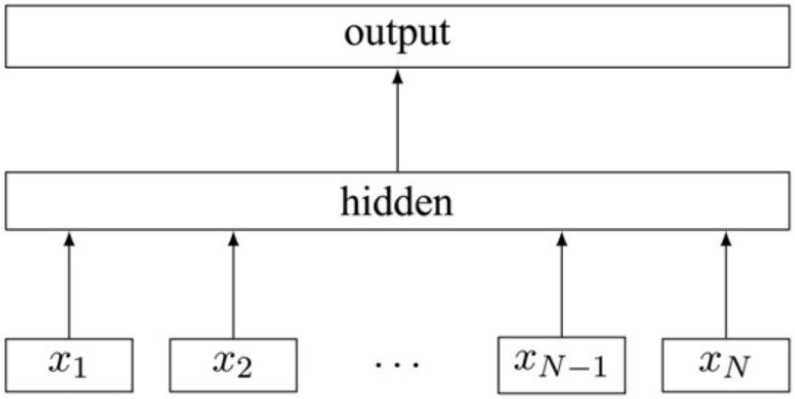

图 3.1 FastText 模型原理图

Hidden 层里面具体是 embedding 层→dropout 层-→线性层。具体见下图：

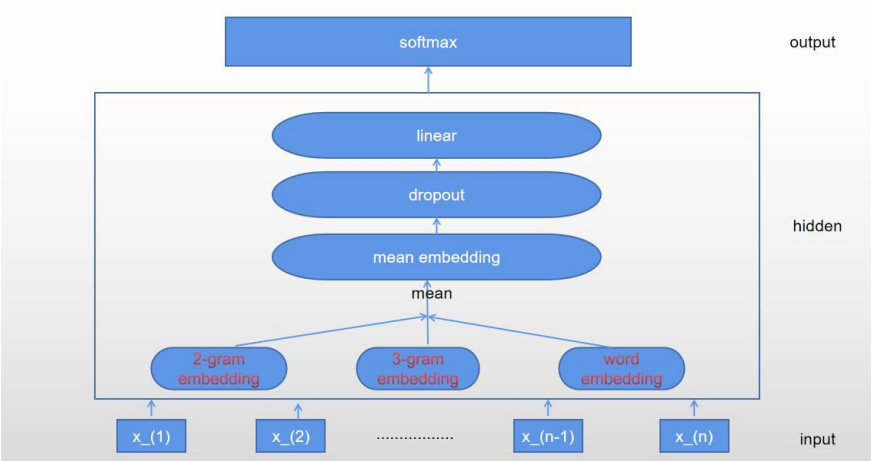

图 3.2 Hidden 层

总模型中 X1,x2…xn 是代表文本的数字，由于本组使用的是字字典，所以没有对训练集分词，（去除停用词还可以考虑）。字模型：例如：一个文本是‘为何 X 县 13 个月发放工资两个样‘转变为数字表示是：30，2，4，5，0，9，6，65，34，666，33，22，54，355，789 这个列表，假如 30 对应‘为’，2 对应‘何’…….,这 15 个数字就对应模型输入的 x1,x2,…xn。但是我们的文本不是每条数据都有一样的字数，所以还要指定一个最大长度，超过这个长度的部分会被去掉，不足的用零补充占位。同理，词模型。

这些数字到了 embedding 会发生如下变化：

如果我们指定 embedding(100,300)那么就是把‘为’对应的输入 30 转变为一个 300 维度的向量，可以采用预训练初始化这个向量，也可以随机初始化。例如[-0.01,0.33,0.3,…]

100 表示词汇表大小（就是这条数据分词以后一共有几个词），除了这个 embedding 还有 n_gram embedding(分词方式不同，导致一条数据可以被分为多种词的组合)如：

如果按照字模型划分：2_gram embedding 将‘武汉加油!’这句话分为：[‘武汉’,’汉加’,’加油’,’油!’],词汇表大小为 4，而 3_gram embedding 将这句话分为:[‘武汉加’,’汉加油’,’加油!’],词汇表大小为 3。同理，如果按照词模型划分，就是将分割的最小粒度由字变为词。

3.2 数据流

Batch_size = 64，代表每个小循环采用的数据数量。

Seq_len = 32，代表每条文本的最大长度(以字或词为单位)。

Embed_size = 300，代表每个字（字模型）或词（词模型）被一个 300 维的向量表示。

下面是张量（tensor）在 FastText 中的每一层的具体形状：

1.模型输入：[batch_size, seq_len]

2.embedding 层：随机初始化，词向量维度为 embed_size，2-gram 和 3-gram 同理：
word：[batch_size, seq_len, embed_size]
2-gram：[batch_size, seq_len, embed_size]
3-gram：[batch_size, seq_len, embed_size]

3.拼接 embedding 层：
[batch_size, seq_len, embed_size * 3]

4.求所有 seq_len 个词的均值
[batch_size, embed_size * 3]

5.全连接+非线性激活：隐层大小 hidden_size
[batch_size, hidden_size]

6.全连接+softmax 归一化：
[batch_size, num_class]==>[batch_size, 1]

3.3‘xavier'网络参数初始化

参数初始化的目的是为了让神经网络在训练过程中学习到有用的信息，这意味着参数梯度不应该为 0。而我们知道在全连接的神经网络中，参数梯度和反向传播得到的状态梯度与激活值有关——激活值饱和会导致该层状态梯度信息为 0，然后导致下面所有层的参数梯度为 0；激活值为 0 会导致对应参数梯度为 0。所以如果要保证参数梯度不等于 0，那么参数初始化应该使得各层激活值不会出现饱和现象且激活值不为 0。我们把这两个条件总结为参数初始化条件：

初始化必要条件一：各层激活值不会出现饱和现象。

初始化必要条件二：各层激活值不为 0。

而 Glorot 认为：优秀的初始化应该使得各层的激活值和状态梯度的方差在传播过程中的方差保持一致：

$$\forall(i,j),\ \mathrm{Var}\left(h^i\right)=\mathrm{Var}\left(h^j\right)$$

$$\forall(i,j),\mathrm{Var}\left(\frac{\partial cost}{\partial z^i}\right)=Var\left(\frac{\partial cost}{\partial z^j}\right)$$

这两个条件称为 Glorot 条件。满足 Glorot 条件的初始化方法称为 xavier 初始化。

根据前人的研究成果，xavier 初始化对 sigmoid 激活函数效果较好，所以我们采用了 xavier 初始化方法。

3.4 Softmax 回归

Softmax 回归是对 logistic 回归的延拓，针对多分类问题。softmax 回归跟线性回归一样将输入特征与权重做线性叠加。与线性回归的一个主要不同在于，softmax 回归的输出值个数等于标签里的类别数，举个例子：一共有 4 种特征和 3 种输出动物类别，所以权重包含 12 个标量（带下标的 w）、偏差包含 3 个标量（带下标的 b），且对每个输入计算 o_1,o_2,o_3 这 3 个输出：

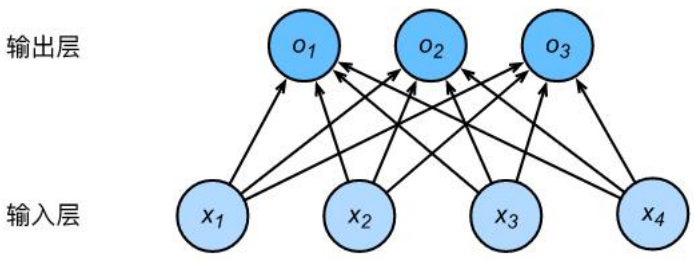

图 3.3 softmax 回归模型

一个简单的做法是将值最大的输出所对应的类作为预测输出，但是会存在问题：一方面，由于输出层的输出值的范围不确定，我们难以直观上判断这些值的意义。另一方面，由于真实标签是离散值，这些离散值与不确定范围的输出值之间的误差难以衡量。softmax 运算符（softmax operator）解决了以上两个问题。它通过下式将输出值变换成值为正且和为 1 的概率分布：

$$\hat{y_1}, \hat{y_2}, \hat{y_3}=\text{softmax}(o_1, o_2, o_3)$$

其中，$\hat{y_1}=\frac{exp(o1)}{\sum_{i=1}^{3}exp(o_i)}$，$\hat{y_2}=\frac{exp(o2)}{\sum_{i=1}^{3}exp(o_i)}$，$\hat{y_3}=\frac{exp(o3)}{\sum_{i=1}^{3}exp(o_i)}$。容易看出$\hat{y_1}+\hat{y_2}+\hat{y_3}=1$ 且 $0\leq \hat{y_1}, \hat{y_2}, \hat{y_3} \leq 1$，因此$\hat{y_1}, \hat{y_2}, \hat{y_3}$是一个合法的概率分布，并且$\text{argmax}o_i = argmax\hat{y}_i$，因此 softmax 运算不改变预测类别输出。

3.5FastText 总结

从隐含层输出到输出层输出，它是一个 softmax 线性多类别分类器，分类器的输入是一个用来表征当前文档的向量；模型的前半部分，即从输入层输入到隐含层输出部分，主要在做一件事情：生成用来表征文档的向量。具体是这样实现的：叠加构成这篇文档的所有词及 n-gram 的词向量，然后取平均。叠加词向量背后的思想就是传统的词袋法，即将文档看成一个由词构成的集合。

简言之，就是将整篇文档的词及 n-gram 向量叠加平均得到文档向量，然后使用文档向量做 softmax 多分类。这中间主要用到了两个技巧：字符级 n-gram 特征的引入以及分层 Softmax 分类。

3.6 训练过程

```
Epoch [26/50]
Iter:   2200,  Train Loss:  0.11,  Train Acc: 96.88%,  Val Loss:  0.51,  Val Acc: 85.50%,  Time: 0:03:32
No optimization for a long time, auto-stopping...
```

图 3.4 训练过程图示

这里本组为了准确的评估模型，设置了训练分类模型。但是由于容易过拟合，于是本组又另外设置了验证集以便找出效果最佳的模型，使用各个模型对验证集数据进行预测，并记录模型准确率。通过验证集和训练集的准确率和错误率的计算，可以判断出该模型是否最大程度的拟合原始数据。

在验证集和训练集的准确率如下图：

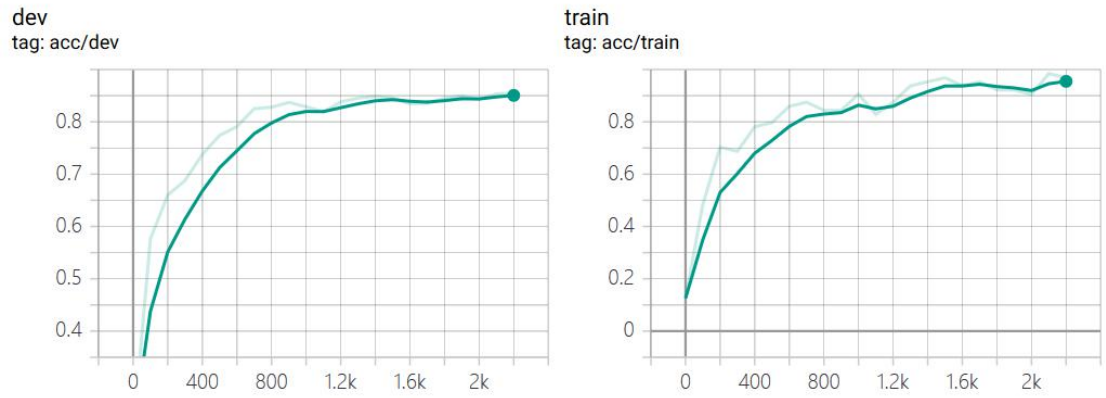

图 3.5 在验证集和训练集上随迭代次数变化的准确率图

在验证集和训练集的错误率如下图：

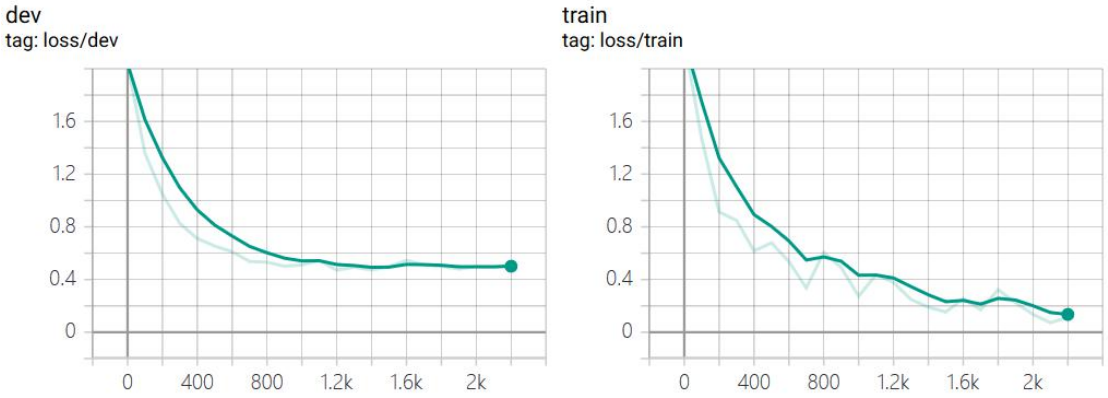

图 3.6 在验证集和训练集上随迭代次数变化的错误率图

由上述两个图可以看出验证集和训练集具有很高的准确率和较低的错误率，因此可以说明本组所建立的模型较高程度上拟合了原始数据。

4、基于 Word2voctor 的热度与答复评价模型

针对 Task2 部分挖掘热点问题和 Task3 的对答复内容进行定量评价的部分，这里需要用到 Word2voctor 模型以及文本相似度的计算问题。Task2 需要计算不同话题之间的热度，其中 3 点要素是时间，地点，话题。Task3 中要想对留言的答复做出一套完整的评价体系，主要从答复的完整性、答复与留言的语义相关性、留言与答复的时间三个方面着手。

4.1 统计词频

Task2 中首先要根据全部数据（分好词后的）统计词频,(从高到底排序为：

v’_1,v’_2,v’_3....)，而 Task3 中需要统计答复的完整性，完整性要求答复包含留言的主要问题。这里就需要提取留言与答复的关键特征词，然后将这些关键的特征词统计出词频。于是这两种问题都需要利用到 MulCounter 和 WordCounter 两种统计词频的方法。

4.1.1MulCounter

MulCounter 完成的是根据单词数组来完成统计词频的工作。 这是一个继承自 Counter 的类。之所以不直接用 Counter 是因为它虽然能够统计词频，但是无法完成过滤功能。

4.1.2WordCounter

WordCounter 完成的是根据文本来统计词频的工作。确切的来说，对完整的文本进行分词，过滤掉停用词，然后将预处理好的单词数组交给 MulCounter 去统计。

4.2 构建 Huffman 树结构

构建如图所示的 Huffman 树结构。首先输入文本，这里有输入层（input），映射层（projection）和输出层（output）三层结构。输入层即为某个单词 A 周围的 n-1 个单词的词向量。如果 n 取 5，则词 A(可记为 w(t)) 前两个和后两个的单词为 w(t-2),w(t-1),w(t+1),w(t+2) 。相对应的，那 4 个单词的词向量记为 v(w(t-2)),v(w(t-1)),v(w(t+1)),v(w(t+2))。从输入层到映射层比较简单，将那 n-1 个词向量相加即可。从映射层到输出层需要从根节点开始，映射层的值需要沿着 Huffman 树不断的进行 logistic 分类，并且不断的修正各个中间向量和词向量。

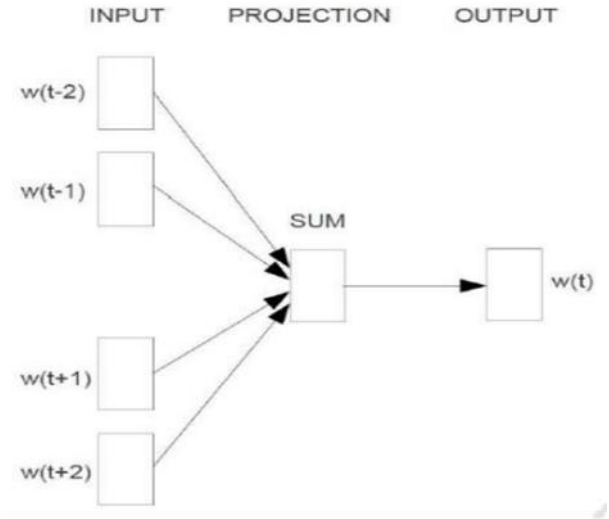

图 4.1 Huffman 树结构

4.3 文本相似度

Task2 中我们需要找出热点话题，Task3 中由于对答复的完整性的定量描述粒度较大，会带来较大的误差，所以我们不仅仅是做机械式的特征词条对比，还要让这套评价体系真正地具有“灵魂”——语义相似度。相似度的衡量需要一个具有先验知识的模型，也就是适用于这些留言和答复的训练好的模型，在 NLP 领域词的数学表示就是向量，在 task1 中我们详细的介绍了词嵌入，与之相似的 Word2Vector 模型可以方便地对词的语义相似度进行计算。所以现在就有了对留言与答复之间语义相似度的方法：将所有答复与留言的词特征喂给模型，然后就可以通过这个模型计算任意一条数据的留言与答复的相似度了。这里我们利用了基于 word2voctor 与编辑距离的文本相似度的计算。

4.3.1 编辑距离

编辑距离方法是指两个句子间，由一个句子转换到另一个句子所需的最少的编辑操作次数。这里的编辑操作共有“插入”、“删除”、“替换”三种。例如：

“你是中国人”--（把“你”替换成“我”）

“你是中国人”--（把“是”替换成“爱”）

“你是中国人”--（把“你”删除）

利用这种方法对两个句子进行相似度的比较。优点是简单，速度快。缺点也很明显，编辑操作无法进一步的判断语义，因此这种方法只适合句子间的模糊匹配。

4.3.2word2voctor

Wordvoctor 可以对句子进行分词，然后将词汇映射成 N 维向量，这样就将两个词汇的相似度转化为两个向量之间的相似度比较，可以利用 cosine 相似度，欧氏距离等数学工具对词汇进行语义分析，根据词频用 Huffman 编码技术将相似词频词汇的隐藏层激活的内容出于大致相同的位置，如果哪个词汇出现的频率很高，那么它所激活的隐藏层的数目就很少，这样处理使得计算的复杂程度大幅降低。

4.3.3 基于 word2voctor 和编辑距离计算相似度的方法

先利用 word2voctor 训练好的模型将词汇生成一个与之对应的词向量，计算两个词汇生成的词向量可以知道两个词汇的相似度，如果为 1 则两个词汇一致，如果为 0 则完全没有关系。

这里考虑到一种情形，当利用替换操作进行两个词汇之间的替换时，如果两个词汇意思是相近的，那么它的替换价值会相应的低一些，反之，则会相应的高。为了将词语的相似度考虑进去，这里引入 word2voctor 的词向量来改进替换操作的系数：假设两个词汇的向量距离为 k，k 是 0 与 1 之间的数字，考虑到 k 的值与编辑距离的大小是相反的，这里将更新后的替换操作的系数设定为 1/（1+k）。这样更新后的替换操作会根据不同词汇之间的距离发生变化，变化范围在 0.5 与 1 之间。而且这个值的范围不会打破编辑操作里面的平衡，即替换=插入+删除，更新后的编辑距离公式 L=a+1/（1+k）*b+c。建立了理论支撑，下面我们来解决问题。

4.4 解决热度与答复评价问题

4.4.1 挖掘热点问题

本组认为热点问题有以下组成因素：时间，地点，留言内容。时间的热度较难统计，所以最后考虑，热点地带有热点问题和普通问题，热点问题可以在热点地带，也可以在非热点地带，

综上，为了尽可能的不遗漏数据，做如下处理：

首先根据全部数据（分好词后的）统计词频,(从高到底排序为：v’_1,v’_2,v’_3....)；然后将相似度极高的词语归一（保留一个词，其他的词的词频求和赋予这个词），所有词还剩下一部分（对应的词频从高到底排序为 v_1,v_2, v_3.....）组成集合 V’；进而保留词频大于某个阈值(v_i)的词组成集合 V；其次根据全部数据（分好词后的）统计高重复性的地点，每个地点的重复次数为：p_1,p_2,p_3...,这些地点作为集合 P，对 V 和 P 的数据取交集，剩下最后的数据表 T；最后将 T 根据不同的关键词（表征热点问题）分类为 c_1,c_2,c_3....c_n；还要统计每个类的时间跨度 t_1,t_2,t_3...t_n。考虑到去量钢化，需要归一化每个类的属性：词频，时间，地点频次：本组使用如下公式进行归一化：

$$x = \frac{x - \min}{\max - \min}$$

定义热度系数为：

$$\text{heat\_coff}_{\text{i}} = \frac{\frac{1}{2}v'_i + \frac{1}{2}p'_i}{t'_i}$$

有了这个公式就可以分别计算并找出前五个热度系数最大的问题了。

为了建立直观上的认识，将以上涉及到词频的部分，分别做出了词云图。

图 4.2 全部留言数据的词云图

图 4.3 热点话题词云图

图 4.4 高频地点词云图

4.4.2 建立答复评价体系

对留言的答复提出一套评价体系，主要从以下三个方面对回复的质量进行评价

答复的完整性(c)：完整性要求答复要包括留言的主要问题，具体操作上可分别提取留言与回复的关键特征词，若答复的特征包括留言的所有特征，则认为该答复具有完整性，若

进一步要求定量评价，则可用答复的包含在留言中的特征除以留言的特征数量，这个比值作为该答复在完整性这个指标上的评分。当然，这会对一些特征做出误判，因为相同的语义可以用不同的词来表达，仅仅这样做对一些特征词是“不公平的”，所以这样就引出了第二个指标。

答复与留言的语义相关性(s)：对答复的完整性的定量描述粒度较大，会带来较大的误差，所以我们不仅仅是做机械式的特征词条对比，还要让这套评价体系真正地具有“灵魂”——语义相似度。相似度的衡量需要一个具有先验知识的模型，也就是适用于这些留言和答复的训练好的模型，在 NLP 领域词的数学表示就是向量，在 task1 中我们详细的介绍了词嵌入，与之相似的 Word2Vector 模型可以方便地对词的语义相似度进行计算。所以现在就有了对留言与答复之间语义相似度的方法：将所有答复与留言的词特征喂给模型，然后就可以通过这个模型计算任意一条数据的留言与答复相似度了。

留言与答复的时间(t):如果这两个时间相距比较近，那么回复就很及时，容易令留言的人满意，反之，则不易令人满意。但是如果要定量地比较的话，时间距离的远近就要有明确的变量来衡量。只有单纯的一条数据就让一个行外人来评价回复及时与否显然不科学，所以要从批量的数据来定指标。具体做法是：把每对留言与回复的时间差（都为正数）作为一个样品，这些样品组成一个样本，然后按从小到大排序构成了一个顺序统计量，设最小值为 $\mathrm{x}_{(1)}$ 最大值为 $\mathrm{x}_{(n)}$，如果这时有一个待评价的样品 $\mathrm{x}_{(i)}$

那么这个样品在回复及时性这个指标上的评分为：$\frac{\mathrm{x}_{(n)}-x_i}{x_{(n)}-x_{(1)}}$，解释：当 $\mathrm{x}_{(i)}$ 离最长耗时 $\mathrm{x}_{(n)}$ 越近时，$\mathrm{x}_{(n)}-x_i$ 就越小分母是常量，所以评分就越低，这符合常识。

根据已知数据还应有一个评价指标：点赞数与反对数(m)，这是最为直观的评价指标。但考虑到每个独立的公民都有可能做出偏激的举动，所以要综合来看，去掉这种“极数”，但从现有数据来看并不能区分出这种“极数”，所以现实一点的做法是一个点赞数抵消一个反对数（规定一个点赞为+1，一个反对为-1），最后留下净值。这个净值是个整数，可能是正整数，也可能是负整数，所以要将它归一化化为-1 到 1 之间的数，这与批量数据的大小有关，所以采用与指标 3 一样的做法，归一化，这里不再赘述。

最后综合四个指标还需要为各自分配权重才可得出综合评分，很明显指标 4 是最客观最可靠的，应分给最大的权重，考虑到 4 个指标权重和为 1，所以暂且分给指标 4 的权重为 0.4，其他三个指标各占 0.2。最后的评分公式为：

$$Score = 0.2 \times (c + s + t) + 0.4 \times m$$

5、实验评估

5.1 实验平台

本组在建立模型、进行实验的过程中，实验配置如图：

CPU	Inteli7
操作系统	windows10
内存	8GB
显卡	NVIDIA GeForce MX150
python	3.6.0
CUDA	9.2
Pytorch	1.2.0

图 6.1　实验配置图

特别地，本组在实验中使用了 Pytorch 机器学习库。

PyTorch 是 2017 年由 Facebook 人工智能研究院（FAIR）基于 Torch 推出的一个开源的 Python 机器学习库，用于自然语言处理等应用程序。它是一个基于 Python 的可续计算包，提供两个高级功能：

1、具有强大的 GPU 加速的张量计算（如 NumPy）；

2、包含自动求导系统的的深度神经网络；

本组在实验中用到了 pytorch 的深度学习训练模式，在这种模式的基础上实现词向量转化和 FastText 模型的构建，以及利用可视化工具 tensorboardX 对训练结果可视化。

5.2 实验评估指标

对于使用的模型，本组将采用精确率（Precision）、召回率（Recall）、两者的累计调和平均数 F-Score 三项性能评估指标对其表现效果进行评价。为明确说明以上指标，本组先定义混淆矩阵如下：

表 5.1　混淆矩阵表

		实际类别	
		正类	负类
预测类别	正类	TP	FP

	负类	FN	TN

- TP(True Positive): 正类样本被判定为正类
- FP(False Positive): 负类样本被判定为正类
- FN(False Negative): 正类样本被判断为负类
- TN(True Negative): 负类样本被判断为负类

5.2.1 查准率（Precision）

查准率也称精确率，考察模型对负例的识别能力，在分类问题中，这个指标尤为重要，可以直观地反映出模型训练后在验证集上检验出的性能好坏。

查准率 P（precision）定义为：

$$P = \frac{TP}{TP + FP}$$

其中，真正例 TP 表示被判定为正样本，事实上也是正样本，假正例 FP 表示被判定为正样本，但事实上是负样本。

5.2.2 查全率（Recall）

与查准率相对应的指标为查全率，也称召回率，考察模型对正例的识别能力；查准率和查全率都是模型性能评估中至关重要的指标。

召回率 R（Recall）定义为：

$$R = \frac{TP}{TP + FN}$$

其中，真正例 TP 表示被判定为正样本，事实上也是正样本，假正例 FP 表示被判定为正样本，但事实上是负样本，假反例 FN 表示被判定为负样本，但事实上是正样本。

5.2.3F-score

查准率与查全率是一对相互矛盾的度量，而 F-score 是一个在传统 F1 度量综合考虑查准率与查全率的基础上，基于查准率与查全率的累计调和平均。本组先给出传统的查准率与查全率的调和平均度量 F1 度量的定义：

$$\frac{1}{F1} = \frac{1}{2}(\frac{1}{P} + \frac{1}{R})$$

即：

$$F1 = \frac{2PR}{P+R}$$

为更好的适应本题，本组采用 F-score 度量，定义如下：

$$F = \frac{1}{n}\sum_{i=1}^{n}\frac{2P_iR_i}{P_i+R_i}$$

其中，P_i为第 i 类的查准率，R_i为第 i 类的查全率。

5.3 评估结果与分析

根据出题方给出的全部数据，本组对获得的所有留言数据的一级标签进行了分类，对比已给出的人工划分结果，本组得到了模型在全部数据集上实验后的查准率、查全率及 F-score，以此来描述使用的模型效果。

5.3.1 查准率

为更加贴近现实，本组根据不同的一级标签，计算出该一级标签内留言数据分类的查准率，即，现实层面中，准确属于某一类部门需要解决的留言问题，占使用模型分到该类部门的留言问题的比例，模型的查准率指标评估结果如下：

表 5.2 模型的查准率指标评估表

一级标签	查准率（P）	对应留言数
城乡建设	0.8629	124
环境保护	0.8786	392
交通运输	0.8065	179
教育文体	0.8109	245
劳动和社会保障	0.8054	421
商贸旅游	0.9029	300
卫生计生	0.8407	181

观察上述评估结果，在目前获得的全部留言数据中，本组使用的模型对各一级标签的分类查准率在 80%~90%之间，每个一级标签内分类查准率较高，各类直接差异较小。为更加直观的评估本组使用模型的分类效果，本组将分类结果中各一级标签对应的留言条数在全部的 1842 条留言数据中所占比例作为权重，对模型查准率进行了加权平均，得到查准率的加权平均值为 0.8450。

本组认为约为 85%的查准率说明选用的模型在该类留言数据集上效果较好，模型的分类结果中，分错类的情况较少，模型比较适合该数据集。

5.3.2 查全率

对于模型的查全率，本组同样按照一级标签，计算出该一级标签内留言数据分类的查全率，即，现实层面中，使用模型分到某一类部门的待解决留言问题数据，占实际应当属于该类部门工作范畴的留言问题的比例，模型的查全率指标评估结果如下：

表 5.3 模型的查全率指标评估表

一级标签	查全率（R）	对应留言数
城乡建设	0.8629	124
环境保护	0.8673	392
交通运输	0.8380	179
教育文体	0.7878	245
劳动和社会保障	0.8551	421
商贸旅游	0.8367	300
卫生计生	0.8453	181

观察上述评估结果，在目前获得的全部留言数据中，本组使用的模型对各一级标签的分类查全率在 78%~87%之间，每个一级标签内分类查全率较高，各类直接差异较小。为更加直观的评估本组使用模型的分类效果，本组将分类结果中各一级标签对应的留言条数在全部的 1842 条留言数据中所占比例作为权重，对模型查全率进行了加权平均，得到查全率的加权平均值为 0.8436。

本组认为超过 84%的查全率说明选用的模型在各一级标签留言类别上分类效果都较好，超过 84%的该类数据都能被准确的分到该类，选用的模型能够将属于该类的留言数据比较完全的概括进来，在该数据集上效果较好。

5.3.3F-score

结合上述得到的模型的查准率和查全率，本组根据公式计算得到了模型的 F1 指标，结果如下：

表 5.4 模型根据一级标签的 F1 指标评价表

一级标签	查准率（P）	查全率（R）	F1 指标	对应留言数
城乡建设	0.8629	0.8629	0.8629	124
环境保护	0.8786	0.8673	0.8729	392
交通运输	0.8065	0.8380	0.8219	179
教育文体	0.8109	0.7878	0.7992	245
劳动和社会保障	0.8054	0.8551	0.8295	421
商贸旅游	0.9029	0.8367	0.8685	300
卫生计生	0.8407	0.8453	0.8430	181

本组将分类结果中各一级标签对应的留言条数在全部的 1842 条留言数据中所占比例作

为权重，对模型 F1 指标进行了加权平均，得到题目要求的 F-score 指标为 0.8439。

F-score 指标综合评价了选用模型在解决此类问题上的性能，本组认为超过 84%的 F-score 指标说明选用的模型在解决留言数据一级标签分类问题上有较好的分类结果，适合留言数据集，在使用机器学习算法代替人工对庞大的留言数据分类的问题上有可预见的可实现性，能够较好的解决该类问题。

6、模型优化

6.1 词向量生成模型选择

在生成词向量模型时，本组考虑了字模型和词模型两种可能的模型。字模型是指将一句话中的汉字“翻译”成为对应的向量，将一句话输入为一个长向量。词模型则是指将语料库中的一个词语“翻译”成为对应的向量，也将一句话输入为一个长向量。即，由最小划分单元为字或词，生成词向量存在字模型与词模型。

字向量：例如文本留言为“为何 X 县 13 个月发放工资两个样”，假设‘为’在字字典中的索引为 30，‘何’为 2……则该文本会转变一列数字：30，2，4，5，0，9，6，65，34，666，33，22，54，355，789，将这个数字列表作为向量输入模型 $x = (x1,x2,\cdots xn)$，这个向量就是文本转换的字向量，输入 embedding 层。

词向量：先对留言进行分词，“为何/X 县/13 个月/发放/工资/两个样/”，然后在词字典中搜索每个词的索引，假设是：33，20，13，45，67，87.则这六个数字组成的向量就作为这条文本的词向量，输入 embedding 层。

考虑到输入模型的文本并非相同字数，为简化向量计算、提高模型效率，本组规定向量最大长度，丢弃超过该长度的部分，不足该长度的向量部分用 0 补充占位。

上述由文本转换而成的数字向量在模型的 embedding 层又会发生变化。例如：按照字模型划分时，2_gram embedding 将‘武汉加油!’这句话分为：[‘武汉’,’汉加’,’加油’,’油!’],词汇表大小为 4；而 3_gram embedding 将这句话分为:[‘武汉加’,’汉加油’,’加油!’],词汇表大小为 3。

为找到最合适、最高效的词向量生成模型，本组对词向量生成模型的字模型与词模型两种模型进行了比较，如下图。图中横轴表示迭代次数，纵轴为该模型在测试集上的准确率，可以发现，字模型的效果普遍优于词模型，随迭代次数的增加，字词模型的准确率逐渐接近，但字模型仍略有优势。因此，对于使用的数据，本组倾向于使用字模型生成词向量模型。

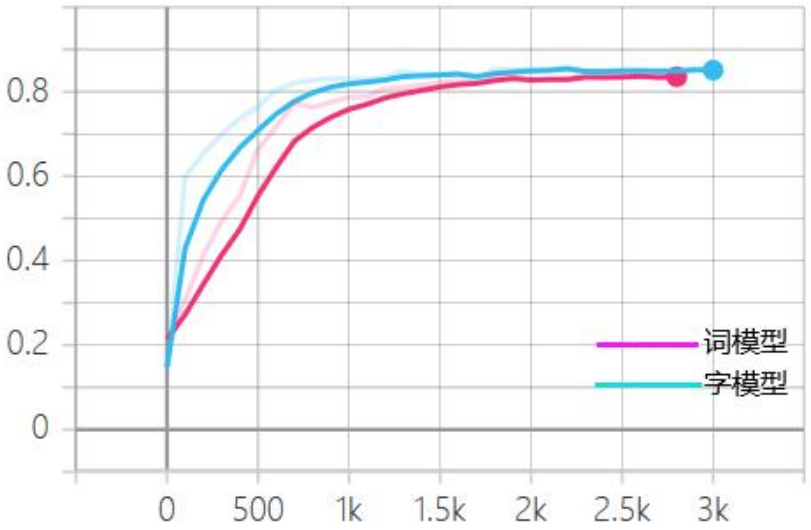

图 7.1 字词模型随迭代次数的准确率变化对比图

6.2 优化算法

模型的性能高度依赖于选择参数时优化算法的选择，找到合适的优化算法和最佳的模型参数是提高模型性能的关键步骤。为了使本组使用的分类模型在留言数据上有良好的分类效果，本组比较分析了随机梯度下降法（SGD）、Adam、Momentum 三种优化算法，对比如下：

表 6.1 SGD、Adam、Momentum 三种优化算法的比较分析表

名称	更新过程	特征
随机梯度下降法（SGD）	确定一个初始参数值θ_0，每次按$\theta_{t+1}=\theta_t-\alpha\,\partial_\theta f_t(\theta_{t-1})$的方式进行更新，其中，$t$ 为迭代次数，α为学习率；每个样本迭代更新一次。	优化速度快，但准确度下降，可能得到局部最优，而不是全局最优。
Adam 算法	确定初始化参数向量θ_0、一阶矩向量m_0、二阶矩向量v_0。按照$\theta_t=\theta_{t-1}-\alpha\,\widehat{m_t}/(\sqrt{\widehat{v_t}}+\varepsilon)$的方式进行更新，其中，$\alpha$为学习率；$\widehat{m_t}=[\beta_1 m_{t-1}+(1-\beta_1)g_t]/(1-\beta_1^t)$，$\beta_1$为一阶矩估计的指数衰减率，$g_t=\partial_\theta f_t(\theta_{t-1})$；$\widehat{v_t}=[\beta_2 v_{t-1}+(1-\beta_2){g_t}^2]/(1-\beta_2^t)$，$\beta_2$二阶矩估计的指数衰减率；ε 是极小的正数，防止出现除以零的情况。依据上式循环迭代到θ收敛。	可以为不同的参数维护不同的学习率；计算效率高；适合解决含大规模数据和参数的优化问题；收敛至局部解。
Momentum 算法	确定初始化参数向量θ_0。按照$\theta_t=\theta_{t-1}-\alpha\,v_{d\theta}$的方式进行更新，其中，$\alpha$为学习率；损失函数反向传播的梯度动量$v_{d\theta}=\beta v_{d\theta}+(1-\beta)d\theta$，通常$\beta=0.9$。	可以解决梯度更新幅度摆动大的问题，同时可以使得网络的收敛速度更快。

为探究以上优化算法对本组使用的模型及数据的优化效果，本组对随迭代次数增加、模型准确率的变化情况作图如下。图中可以明显看出，对于本组的模型和数据，Adam 算法在优化模型、得到高准确率方面，具有明显优势，且确实能够将模型准确率提升至 85%左右，达到较高水平。因此，本组在确定模型参数时，使用 Adam 算法对模型及参数进行优化，以

达到较好效果。

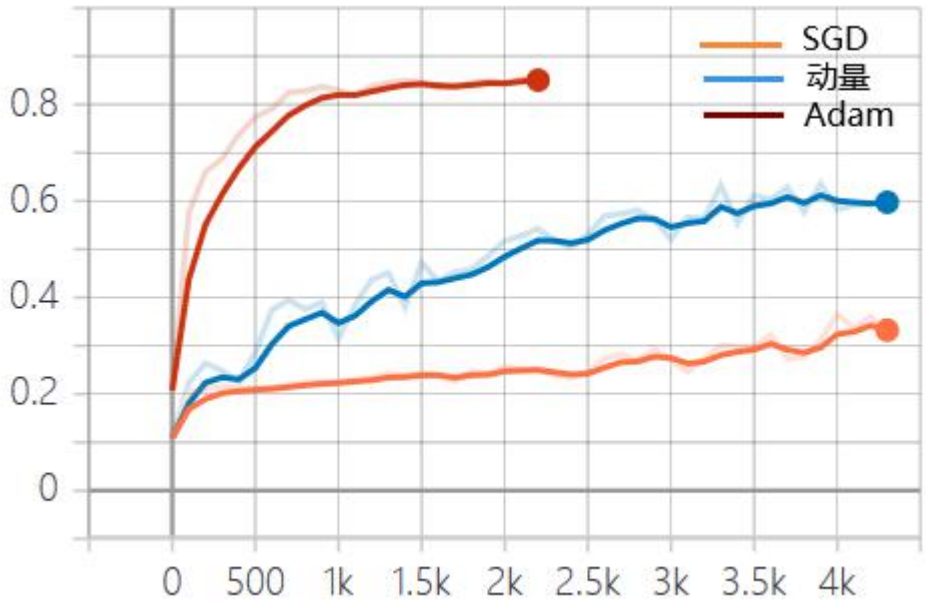

图 6.2 优化算法随迭代次数变化的准确率变化对比图

7、参考文献

[1] https://blog.csdn.net/willduan1/article/details/78070086
[2] https://blog.csdn.net/guoyuhaoaaa/article/details/78155007
[3] https://zhuanlan.zhihu.com/p/32965521
[4] https://www.cnblogs.com/cyandn/p/10891608.html
[5] Understanding the difficulty of training deep feedforward neural networks by Xavier Glorot, Yoshua Bengio in AISTATS 2010