“智慧政务”中的文本挖掘应用研究

摘要

本文针对“智慧政务”中的文本挖掘问题，建立了**卷积神经网络分类模型**，对留言内容的进行了一级标签的分类学习，并用**F-Score**对分类方法进行了评价；再利用**Density And K-means 聚类算法**对留言进行了汇聚和提取，得出了热点排名前 5 的热点问题；最后**建立答复评价指标体系**，从答复的信息系、完整性、相关性等角度对答复意见的质量做出一套评价方案。

针对问题一：首先通过分析大量留言信息，归纳文本的特点，在通过**隐马尔科夫模型**，借助附件 1 中各级别标签类别和附件 2 中已经分好一级标签对留言进行词意的划分，然后参考中文文本分类的一般流程，并考虑缺陷文本的特点，建立了一种基于**卷积神经网络**的留言文本分类模型；最后通过 **F-Score** 对分类模型进行评价。算例结果表明，所提出的留言文本分类模型能显著降低分类错误率，在分类效率上也比较可观。查准率、查全率和 F 分数如下所示：

指标	查准率	查全率	F 分数
分类情况	81%	86%	84%

针对问题二：首先对文本的时间、空间和内容进行初次聚类，然后通过基于统计的分词方法，体现文本之间结合关系的紧密程度，然后利用信息增益方法，对文本中的所有分词进行它自身所含的信息值的计算和分析，使文本的特征达到降维的目的，进行特征提取，在使用**向量空间模型**，将文本转化为空间向量，然后采用新的二次聚类算法—**DAK（Density And K-means）**，将**划分聚类方法**与**基于密度**的聚类方法相结合，不仅降低了时间消耗，还提高了聚类的准确性和广度，最后通过簇的热点提取，得到了排名前 5 的热点问题，并做出热点问题留言明细。

针对问题三：为保证文本评价时的准确性和客观性，本文将答复意见文本分为数值型和文本型两类,二者结合可获得完整性指标;从两类文本类型中提取评分数据,然后通过 **WRC 评价**和 **1R2C 评价**建立信息量、相关度、一致性、完整性、有效性和可解释性六项评价指标的**答复质量评价指标体系**，通过对附件中的答复意见进行评价分析，得出各指标的评价标准，结合文本指标的打分权重，计算出各答复的指标值，然后计算平均值作为本文评价指标的权重，建立**答复质量评价模型**，得出一套准确性较高的评价方案。

关键词： 卷积神经网络分类模型　DAK 聚类模型　答复质量评价模型

目录

一、挖掘目标

1.1 挖掘背景

近些年来，随着政府的有作为和网络问政平台的完善，让相关部门通过“智慧政务”等网络平台更多的了解民声、民意、名气。社情民意调查是采用有组织的统计学调查方法收集在一定时期、一定范围内社会公众对社会现实的主观反映，起到反映民意、引导舆论、为党政部门决策提供数据支撑和参考依据、检验政策实效等作用。随着各平台的完善，各类社区情况以及民众意愿相关的文本数据量不断上升，给以往依靠手工划分留言和整理热点的工作人员带来了挑战。

现在是一个大数据、 云计算、人工智能等技术的发展蓬勃的时代，拥有自然语言处理技术及人工智能分类系统的“智慧政务”是维护社会治安，共建和谐家园的新趋势，推动相关工作人员的施政效率和管理水平。

此题需要根据收集自互联网公开来源的群众问政留言记录，及相关部门对部分群众留言的答复意见等数据，运用数据挖掘方法，建立有效的数学模型，进而解决群众留言分类，热点问题挖掘及答复意见的评价等问题。

1.2 挖掘目标

根据附件 1 的分类标准，对留言进行分类，以便后续将群众留言分派至相应的职能部门处理。

根据附件 2 给出的数据，建立关于留言内容的一级标签分类模型，并用 F-Score 对分类方法进行评价。

根据附件 3 将某一时段内反映特定地点或特定人群问题的留言进行归类，定义合理的热度评价指标，并给出评价结果，列出排名前 5 的热点问题。

根据附件 4 相关部门对留言的答复意见，从答复的相关性、完整性、可解释性等角度对答复意见的质量给出一套评价方案，并尝试实现。

二、模型假设

假设一：不考虑情绪、失误等其他因素对留言准确度的影响。

假设二：附件中给出的时间和空间准确，没有错误和干扰影响。

假设三：热点事件在一定范围和一定时间内，不存在突然消失，并且留言均处于事件发生期内。

假设四：政府的答复意见真实可靠，不存在虚假现象。

三、符号说明

符号	描述
n	即向量的维数
I	形成矩阵
$I_{i:i+h-1}$	表示 I 从上到下的第 i 个 h 行 n 列矩阵块
b	为偏置项
p_j	卷积窗口对应的特征值
p	即为代表句子全局特征的向量
N	表示整个文本集的数目
n_i	指的是在整个文本集合中包含有分词 i 的文本数目
$TF_{i,j}$	指的是分词 i 在文本 d_j 里面出现的次数
$w_{i,j}$	是分词 i 在文本 d_j 里面的权重
W	表示文本的权重
a_j	表示第 i 句中第 j 个分词的权重，
f_j	表示此分词在第 i 句中出现的次数，
l_j	表示第 i 句中第 j 个分词的长度。
$\log(\frac{N}{n_i})$	反文档频率

四、问题分析

4.1 问题一的分析

问题一要求本文根据附件 2 给出的数据，建立关于留言内容的一级标签分类模型，群众问政留言记录与一般的中文文本相比，存在白话较多与信息关键词散乱等问题，因此首先对文本进行提取关键词、剔除垃圾词的预处理，然后建立可解释性较强并且层次较深的基于卷积神经网络的文本分类模型。模型采用分布式表示，即以词为单位进行文本表示形成词向量，再将词向量按照词在句子中出现的顺序进行拼接，形成代表句子的矩阵。在词表示的过程中，可以学习到词之间的语义相关性与三种等级分类的连续性，同时词向量的拼接也考虑了句子中词出现的顺序，最终得到完整的一级标签分类模型，并用 F-Score 对分类方法进行评价。流程如下图所示：

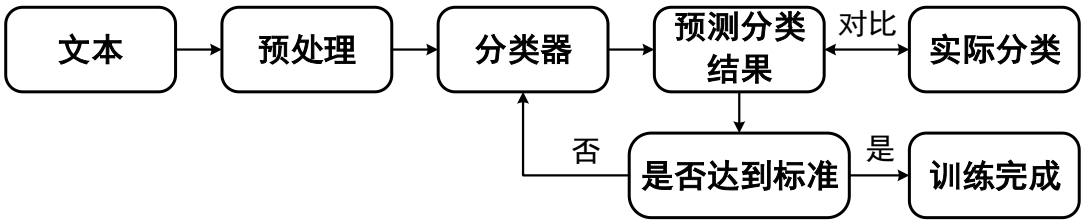

图 1：文本分类训练流程

4.2 问题二的分析

热点问题的发现与识别就是将关于同一个事件的文本信息聚扰到同一个话题簇中,其本质就是一个文本聚类的过程，但在此问题的聚类过程中需要考虑实践、地点、事件、点赞数和反对数四个指标，这个过程主要由文本预处理、文本聚类和热点取出三个步骤组成。本文先通过文本去重、分词、特征提取等一系列文本预处理，然后使用 *DAK*（Density And K-means）二次聚类算法对文本集进行聚类，得出多个聚类中心，再分别对各个簇的标题进行分词，并计算每个分词的权重，最后据分词的权重去计算各个标题的权重，分别从各个簇中选取若干个权重最高的标题作为热点话题输出，得出 5 个热点问题，并得出热点问题的明细。

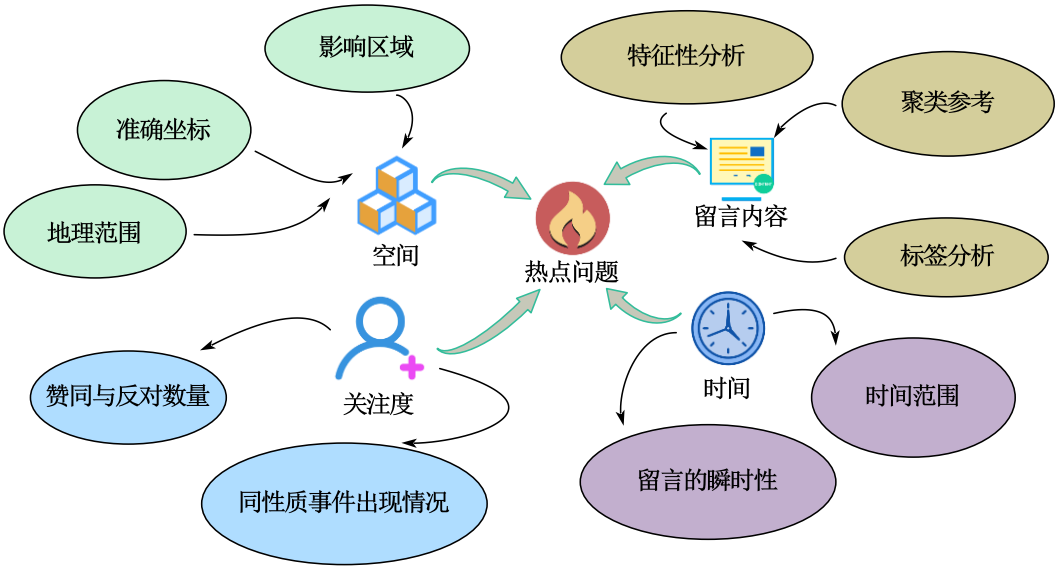

图 2：问题二思路图

4.3 问题三的分析

为探究政府部门对留言的答复质量情况,本文主要从下几个步骤进行分析与处理,首先选取评论数据特征,使用改进的自动标法算法和 K-means 算法对答复意见进行有效性标注，然后提取其他主要持征指标,并通过各种计算或编程得到适用模型的指标变量，其次,构建文本的回复质量评价指标体系，最后建立本文的答复意见质量评价模型。

具体流程如下图所示：

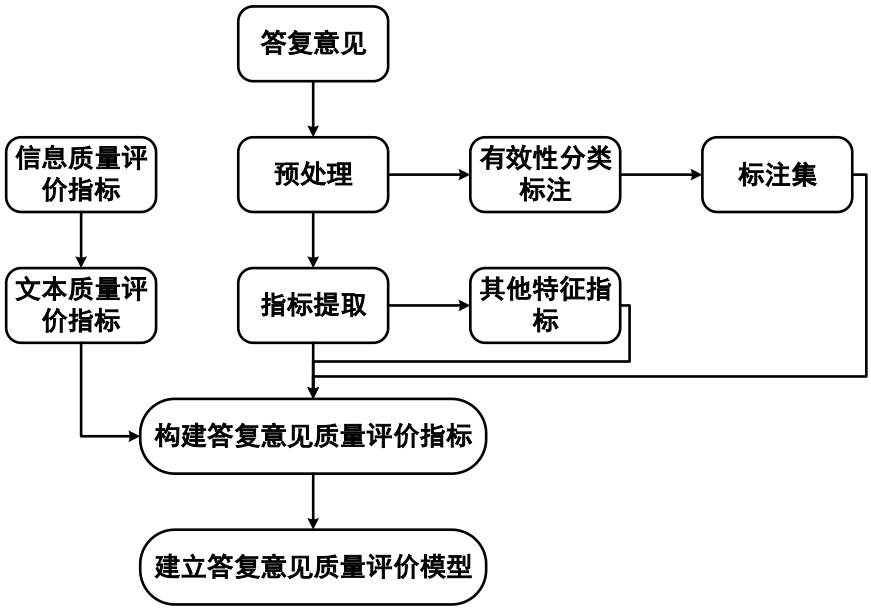

图 3：答复意见质量评价模型分析流程图

五、数据分析

5.1 附件 1 数据分析

根据附件 1 一级标签各类别所占数量，绘制如下雷达图：

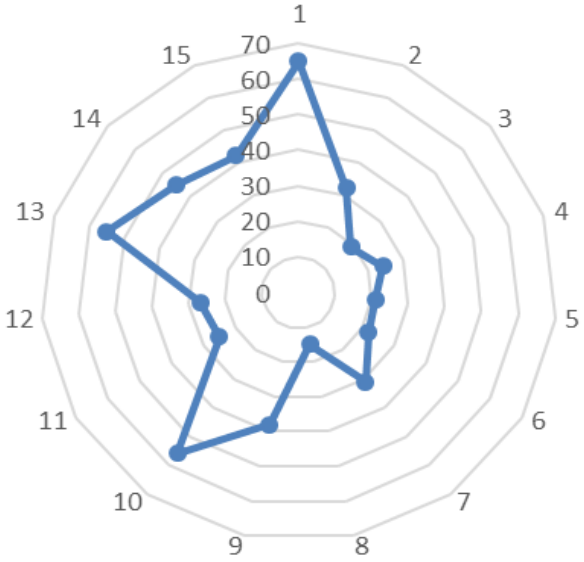

图 4：一级标签总数占比图

如上图所示，一级标签分为 15 类，各类数量不同。其中城乡建设总数最多，为 65 项，科技与信息产业总数最少，为 15 项。近年来，城乡建设广泛受到关注，对应的文本分类数多，因此在考虑第二问热点问题挖掘时，可以着重观察城乡建设对应的民众留言。

5.2 附件 2 数据分析

根据附件 2 中一级标签和高频词出现次数绘制如下图：

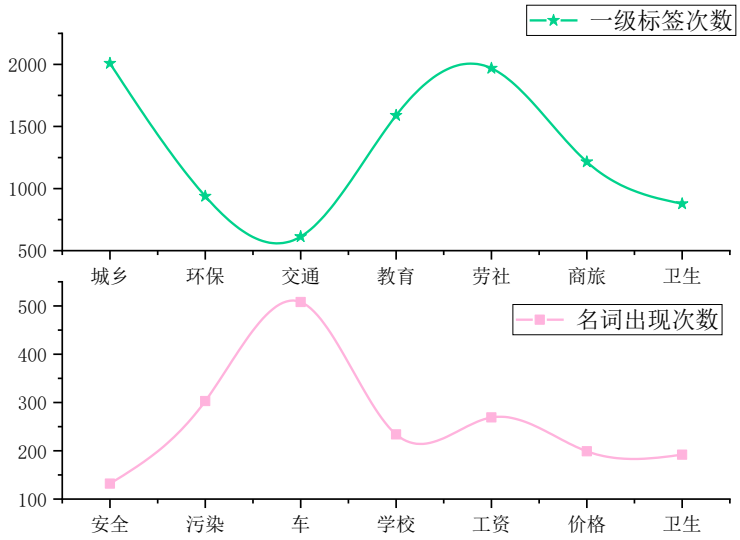

图 5：附件 2 数据分析

通过上图可以发现，一级标签中城乡建设，劳动和社会保障出现的次数最多，交通运输出现的次数最少。说明群众对城乡建设和自身劳动社会保障关注较多，留言内容也最多。附件 2 中出现的高频词汇：安全，卫生，学校，车，污染，价格，工资中，车，污染出现的频率最高。

5.3 附件 3 数据分析

根据附件 3 信息，绘制出点赞和反对总数排名前 8 的留言数据如图所示：

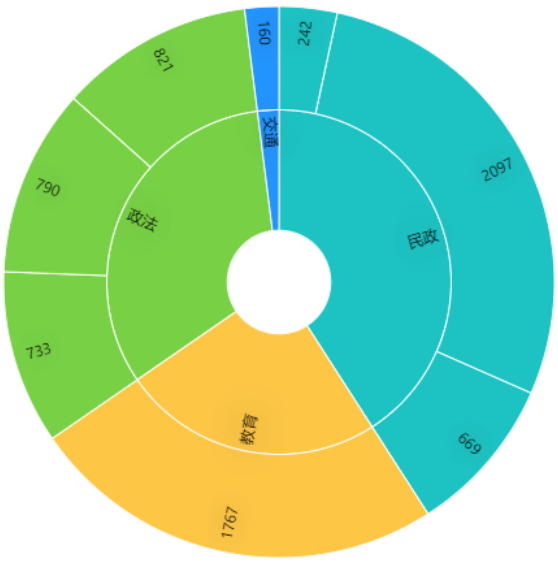

图 6：点赞，反对总数前 8 数据分析图

由上图所示，点赞，反对总数排名前 8 的留言中，有关民政的留言点赞反对总数最多，为 3008 次。分析得社区建设划分在民政这个一级标签内，而群众日常生活大多与社区有关，自身维权维利也与社区有关。其中有关交通的留言点赞，反对总数最少。

5.4 附件 4 数据分析

根据附件 4 中，留言的时间段分析如下图所示：

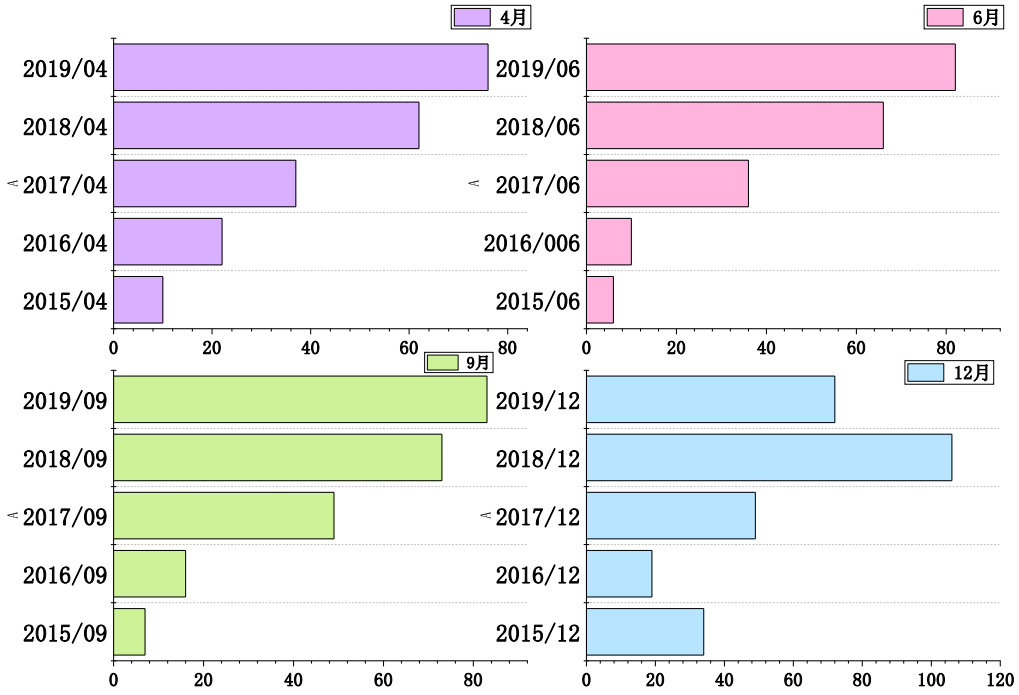

图 7：各时段留言总数

由上图可得：2015~2019 年，4 月，6 月，9 月的总留言数都逐年递增，2015 和 2016 年的新媒体，人工智能还不够发达，因此留言数不多。2018 年 12 月与 2019 年 12 月对比发现，2019 年 12 月总留言数要少，充分说明执政工作人员的办事效率得到了提高，更多的在实际生活中就解决了群众的问题，在网络上留言的数量就少了。

六、问题一模型建立与求解

6.1 模型的准备

针对群众问政留言记录文本的特点，文本预处理包括划分词意和剔除无用词。由于中文文本不同于英文文本，词与词之间没有空格的自然分界，并且句子之间的符号分割对文本的分类影响较小，因此在文本表示之前需要对中文文本进行划分词义。本文采用隐马尔科夫模型，借助附件 1 中各级别标签类别和附件 2 中已经分好一级标签对留言进行词意的划分，对于一些无法表征所在标签类别的词，如时间、相关的地名、连词等，需要作为停用词在划分词意后从文本中剔除，以增加文本的高效性。

6.2 模型的建立

6.2.1 分布式文本表示

运用分布式词向量对词进行表示。首先，参考 Bengio 神经网络语言模型(NNLM)，以大量经过预处理和标签分类的文本为语料库，训练出每个词的词向量表示，词向量的各个维度代表通过 NNLM 学习到的词的语义特征。以维度为 3 的词向量为例，将部分缺陷文本的词向量在特征空间中进行表示，如图下图所示。

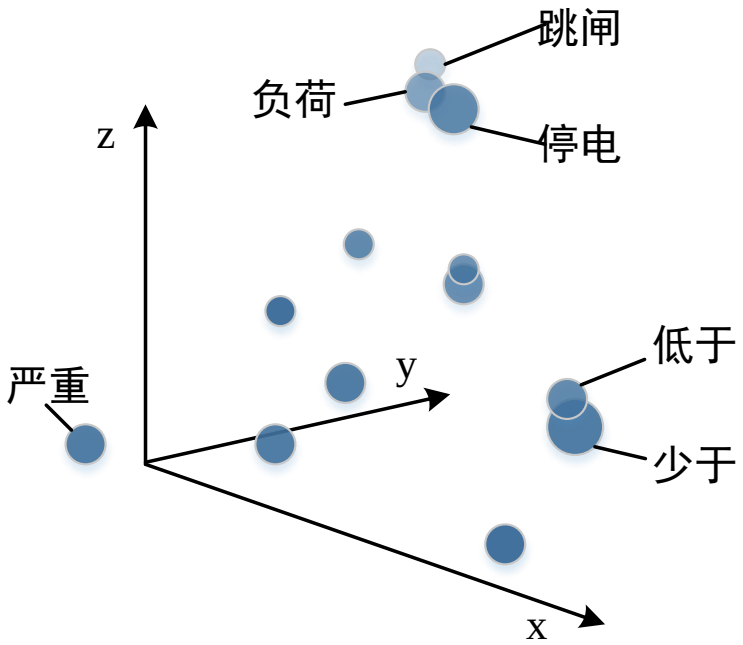

图 8：特征空间中的词向量

其中，每一个圆点表示一个词向量，x、y、z 轴分别表示词向量的 3 个语义特征维度。由上图可见，词义相近的词对应的词向量在特征空间中距离比较接近，而词义相差较大的词对应的向量距离比较远，其中与各一级标签和所属一级标签下的二级、三级标签的距离越近，代表相关度越高，即可以通过词向量对词义特征进行刻画。在实际应用时，词向量维度大小可根据语料库大小指定，通常取 100~300 维，每个维度代表机器自动学习到的一个词特征，没有实际的物理意义。

然后，对同义词的词向量进行合并。例如，通过对语料库的训练，得到了“停电”、“跳闸”和“负荷”的词向量，且三者在特征空间中很接近(接近程度通过向量的欧式距离进行刻画)。因此将词向量统一改为“停电”的词向量，从而实现同义词的词向量合并。

6.2.2 卷积神经网络分类器

卷积神经网络具有局部感知、权值共享的特点，可以大大减少训练参数的数目，提高了复杂网络的计算效率。卷积神经网络可作为分类器，对向量化后的缺陷说明文本进行分类，并输出相应的分类结果。本文通过对所知文本进行分析，构建了一个四层的卷积神经网络，如下图所示所示

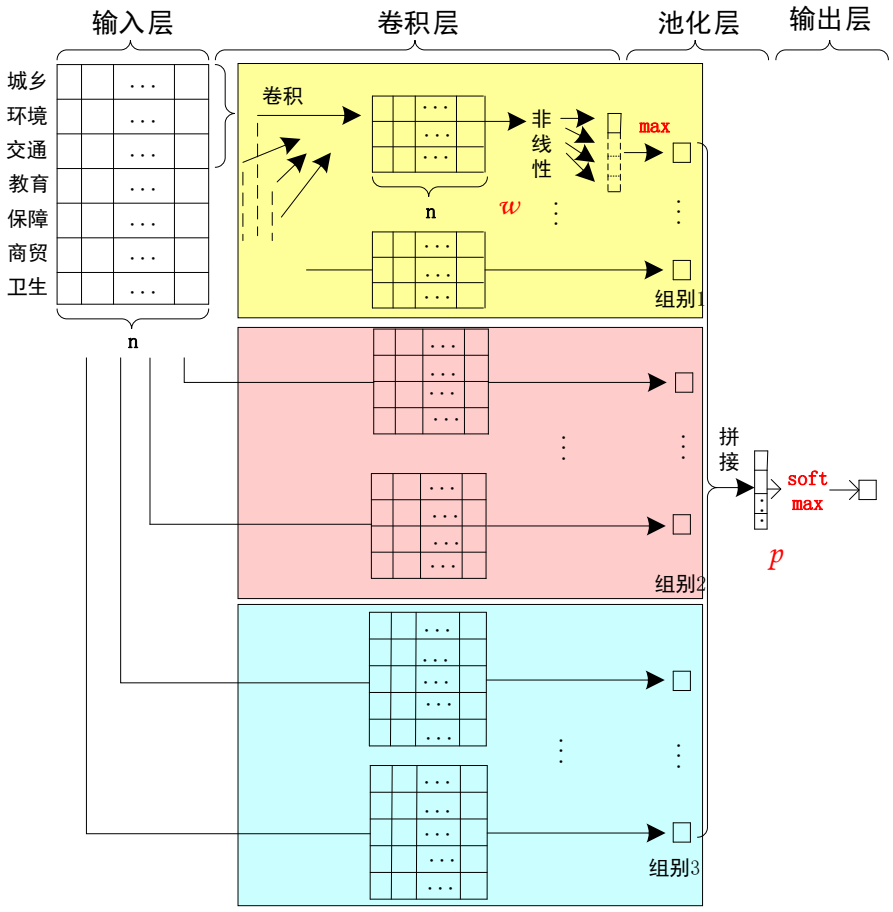

图 9：卷积神经网络结构

（1）输入层

第一层为输入层。输入层为一条待分类缺陷说明句子对应的矩阵，即

$$I \in R^{s \times n}$$

矩阵的每一行代表句子中每个词对应的向量，行数 s 即句子的词数，列数 n 即向量的维数。以“突然断电居民小孩经常被困电梯”这一缺陷说明为例，采用基于词向量的模型时，句子按词为单位被切分为“断电+居民+经常+困+电梯”，每个词再经过向量化转换为维数相等的词向量，从而形成矩阵 I，作为卷积神经网络的输入层。本文在训练过程中，保持数字的词向量固定，对其他词向量进行微调，以提高模型的泛化能力

（2）卷积层

第二层为卷积层。采用列数与 I 相同(为 n)、行数为 h 的卷积矩阵窗口 $W \in R^{h \times n}$，与输入层矩阵 I 的每个 h 行 n 列矩阵块由上到下依次进行卷积运算，得到卷积结果 r_i，即

$$r_i = W \cdot I_{i:i+h-1}$$

其中 $i = 1, \cdots, s-h+1$；$I_{i:i+h-1}$ 表示 I 从上到下的第 i 个 h 行 n 列矩阵块；“$\cdot$”表示点乘运算，即将 2 个矩阵所有相同位置的元素相乘后再求和。因此一共进行 $s-h+1$ 次卷积，每次卷积后再进行非线性化处理，得到非线性化后的结果 c_i，

即：

$$c_i = \mathrm{Re}LU(r_i + b)$$

其中b为偏置项，其值可以在训练过程中自动调整；$\mathrm{Re}LU$为修正线性单元函数。最终共得到$s-h+1$个实数c_i，将这些实数依次排列就构成卷积层的向量$c \in R^{s-h+1}$。

本文将卷积窗口分为多个组别，图 9 用 3 种背景色示意了 3 种组别，不同组别采用不同尺寸(不同行数)的卷积窗口与I进行卷积运算，以获取不同词数级别的语义特征。另外，每个组别中的卷积窗口有多个，且各个卷积窗口(矩阵)之间的元素值不同，以便多方面地提取特征。

（3）池化层

第三层为池化层。本文采用最大池化的方法，即取每个卷积窗口卷积得到的卷积层向量c中最大的元素$\max\{c\}$作为特征值，从而提取各个卷积窗口对应的特征值$p_j(j=1,2\cdots,w,$其中w为卷积窗口总数)，并将所有特征值p_j依次拼接构成池化层

的向量$p \in R^w$，p即为代表句子全局特征的向量。池化的过程既实现了特征的进一步提取，也降低了特征的维度，提高了分类效率。

（4）输出层

输出层与池化层全连接，以池化层向量p为输入，采用*softmax*分类器对向量p进行分类，并输出最终的分类结果。

6.3 模型的检验

在大规模数据集合中，查准率和查全率 2 个指标往往是相互制约的。理想情况下做到两个指标都高当然最好，但一般情况下，查准率高，查全率就低，查全率高，查准率就低。所以在实际中常常需要根据具体情况做出取舍，例如一般的搜索情况，在保证召回率的条件下，尽量提升精确率。因此，很多时候我们需要综合权衡这 2 个指标，这就引出了一个新的指标$F-score$。这是综合考虑查准率和查全率的调和值。

$$F_1 = \frac{1}{n}\sum_{i=1}^{n}\frac{2P_iR_i}{P_i+R_i}$$

其中：p_i为第i类的查准率，即真实正样本的比例；R_i为为第i类的查全率即正确预测的比例；通过得出精准的$F-score$来评价所建立的分类方法。

6.4 结果分析

用 python 的“pandas” 库读取附件 2 的信息，选取留言主题那一列文本数据作为特征，一级标签那一列数据作为目标，使用 “jieba” 库将文本中的停用词去除，再使用“sklearn”库将数据按照三七开分为测试集和训练集，并将文本内容进行特征提取，最后调用 “sklearn”中的卷积神经网络分类器对训练集进行训练，将测试集放入训练结果中测试。

表一：部分用户留言分析与实际结果对比

留言编号	留言用户	分析结果	实际结果
54346	U0005345	商贸旅游	商贸旅游
97181	U0006288	劳动和社会保障	劳动和社会保障
123974	U0002980	城乡建设	城乡建设
⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮
92850	U0008038	卫生计生	卫生计生
126880	U0003918	环境保护	环境保护
127222	U000861	交通运输	商贸旅游

根据一级标签分类模型，使用 python 编译结果显示：编译分析结果与实际结果完全一致的概率为 81.4%。在一定程度上实现了人工智能，云计算，解决了依靠人工处理留言工作量大和效率低的问题。但还有极少部分的智能分析与实际结果不符。如编号：127222。根据留言内容：L5 县至桐木溪客运车抱在手里的小孩乘车不免费，分析造成智能识别错误的原因为：留言关键字模糊，交通运输和商贸旅游之间本来就存在一定的关联。因此问题一模型还需要进一步改进，完善缺陷，来达到更高百分比的准确度。

七、问题二模型建立与求解

7.1 模型的建立

7.1.1 数据预处理

通过分词、特征提取、向量空间模型三部分，将文本转化为空间向量，为后面的基于向量空间模型的聚类做铺垫。

（1）基于统计的分词方法

通过对语料中相邻共现的各个字的组合的频度进行统计，计算它们的互现信息，然后定义两个字的互现信息，计算两个汉字 X、Y 的相邻共现概率。共现信息体现了汉字之间结合关系的紧密程度。当紧密程度高于某一个阈值时，便可认为此字组可能构成了一个词。同时本文使用统计方法识别一些白话词，即将串频统计和串匹配结合起来，既发挥匹配分词切分速度快、效率高的特点，又利用了无词典分词结合上下文识别生词、自动消除歧义的优点。

（2）特征提取

对于本题在进行特征选取时选择最高代表性的特征，利用信息增益方法，对文本中的所有分词进行它自身所含的信息值的计算和分析，通过选取其值最高的一些作为文本的特征达到降维的目的。流程如下图所示：

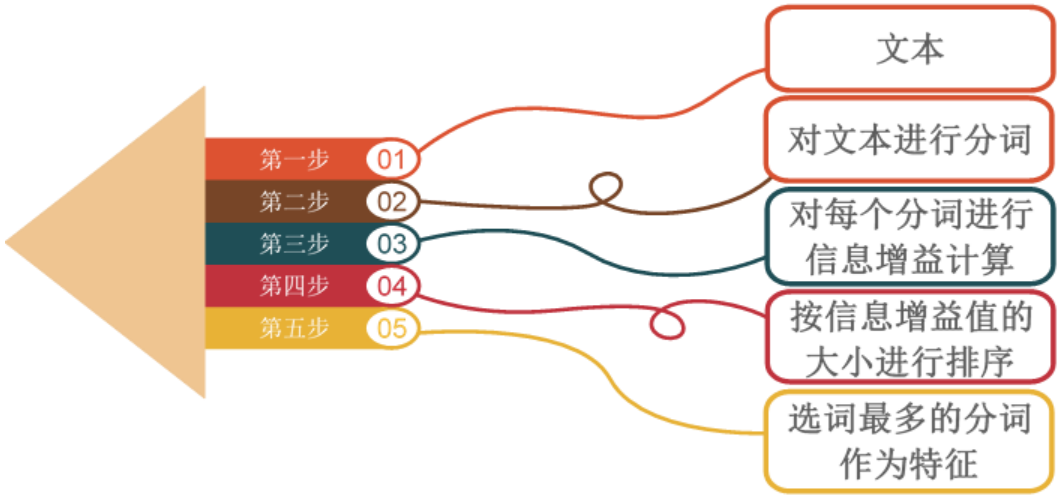

图 10：特征提取过程

（3）向量空间模型构建

通过特征提取，从中选取分词所含信息量高的来作为向量空间的一维，在一定程度上达到了降维的目的。向量空间模型中的每一个向量中的一维对应的就是一个分词，对应分词权重的计算，采用的是经典的权重计算公式

$$w_{i,j} = TF_{i,j} \cdot \log(\frac{N}{n_i})$$

其中 $w_{i,j}$ 是分词 i 在文本 d_j 里面的权重，$TF_{i,j}$ 指的是分词 i 在文本 d_j 里面出现的次数，而 $\log(\frac{N}{n_i})$ 反文档频率，N 表示整个文本集的数目，而 n_i 指的是在整个文本集合中包含有分词 i 的文本数目。

文本相似度的计算是文本挖掘方面中重要的一环，根据向量空间模型，本文采用余弦相似度计算。余弦相似度的计算公式是由内积与夹角余弦的关系转变过来的，公式如下：

$$Sim(X,Y) = \frac{X \cdot Y}{|X| \cdot |Y|} = \frac{\sum_{i=1}^{n}(x_i \cdot y_i)}{\sqrt{\sum_{i=1}^{n} x_i^2} \cdot \sqrt{\sum_{i=1}^{n} y_i^2} \cdot}$$

得出余弦相似度的计算结果，其值是在区间（0,1）里面，可以比较方便的通过设置阀值来判断文本之间在相似度是否达到了要求。

7.1.2DAK（Density And K-means）聚类算法

本文在用 K-Means 聚类分析时，不是直接使用，而是在得出了一个粗糙聚类结果的情况下，再进行第二次聚类。所以在第一次聚类时需要做的就是确定 k 的值，以及 k 个初始中心点。在通过分析了层次聚类、基于密度的聚类、网格聚类等方法后，最终选择了基于密度的聚类方法做为首次聚类得到一个粗糙的结果。它把簇中的数据空间从低密度区域分割成高密度对象的区域，最大的优势是在于能够从空间数据库中发现任意形状的聚类。它是根据数据空间中密度的差异，把具有相似密度的点作为聚类，流程图如下：

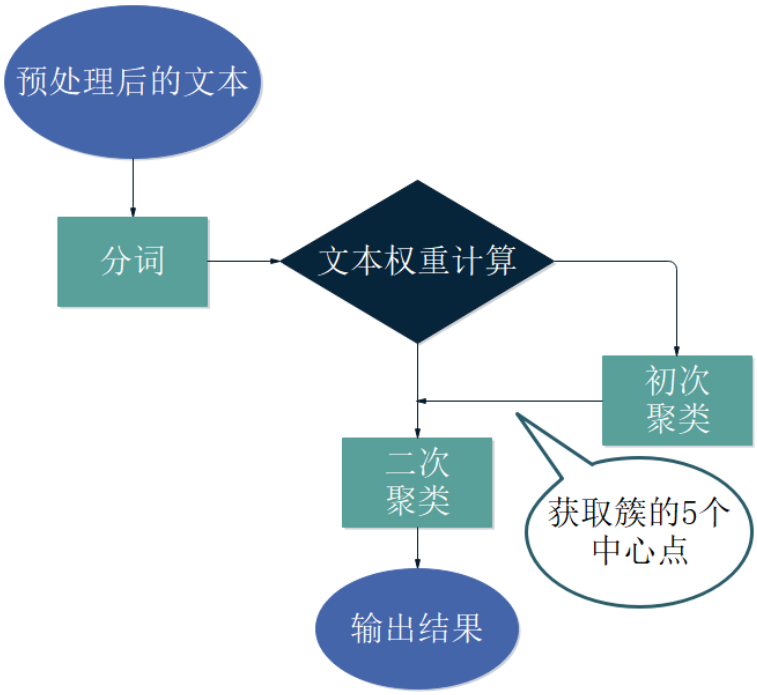

图 11：DAK 流程图

具体实现步骤如下所示：

●计算空间 N 个点之间的距离，共有 m 对 $m(l_1,l_2\cdots,l_m)$，$m=C_N^2$，可以得到一个球半径：

$$r=\frac{\sum_{i=1}^{m} l_i}{m}$$

●分别以这 N 个点为中心计算在半径 r 的球体里面包含别的点的个数 $(n_1,n_1\cdots,n_l)$，其中 n_i 表示第 i 个点在半径 r 里面包含的点数。

●对 $(n_1,n_1\cdots,n_l)$ 按从大到小的顺序排序，依次选取来作为簇中心，如果 n_i 这个点已经出现在了已选出的簇中心所在的球体里面，则放弃该点，继续生成簇中心，直到所有的点都被判定过，可以得出 K-means 所需要的最后聚类簇的个数，以及初始聚类中心。

●计算剩余的每个文档到各个簇中心的距离，再把它们归到最近的那个簇中心。

●重新计算各个类的簇中心。

●重复上面两次步骤直到簇中心不再发生变化。

此 DAK（Density And K-means）算法不需要通过个人的经验来对初始参数进行设置，而是通过文本数据分布的特点来实现分类和统计，使最后的聚类结果更加客观，不再呈现多样性，而是稳定的结果。

7.1.3 热点提取

由于每个簇都含有巨大的文本数量，因此在进行热点生成时考虑这些所有文本的信息是不现实的。所以，本文在做热点生成时，针对的对象不是该簇的整个文本集，而是文本集所对应的关键词，具体操作流程如下所示：

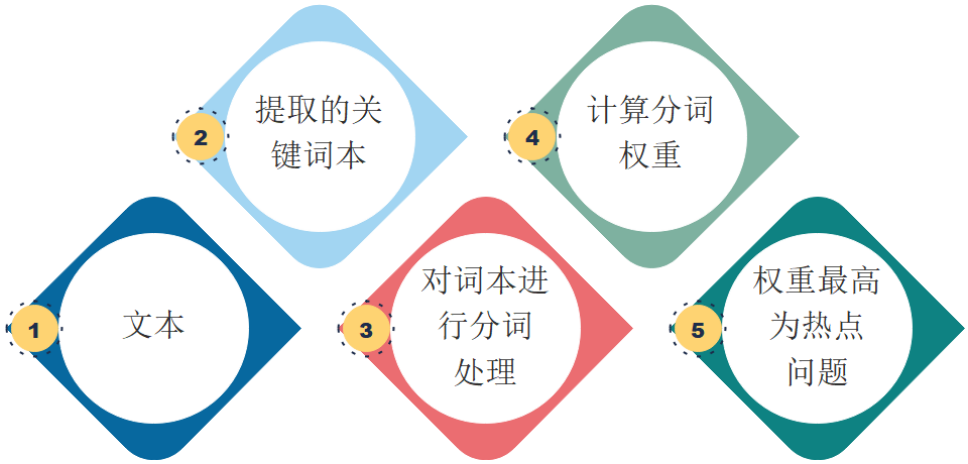

图 12：热点问题提取过程

由于不需要考虑位置等情况，最后的计算公式如下：

$$W = \frac{\sum a_j \cdot f_j \cdot l_j}{\sum l_j}$$

其中，W 表示文本的权重，a_j 表示第 i 句中第 j 个分词的权重，f_j 表示此分词在第 i 句中出现的次数，l_j 表示第 i 句中第 j 个分词的长度。依据文本里面所含分词权重的高低，权重高的分词越多，然后通过聚类统计，得到 5 个热点问题，也就是该簇的热点话题所在。

7.2 结果分析

表二：热点问题表

热度排名	问题 ID	热度指数	时间范围	地点/人群	问题描述
1	1	480.6	2019/1/11 至 2019/12/5	A 市 A5 区湖镇区居民	存在车贷和诈骗案件
2	2	429.0	2019/1/15 至 2019/11/11	A 市 A5 区圭塘路居民和行人	垃圾站对水岸周边产生污染
3	3	368.2	2019/1/7 至 2019/10/30	A 市 A5 区御府路区明路学生	学生入学困难，反映问题没有得到解决
4	4	188.6	2019/1/3 至 2019/12/31	A 市 A1、A2、A4、A7、A8 区小区居民	存在房产与扰民问题
5	5	144.2	2019/1/30 至 2019/9/6	A 市 A4 区距渝长厦小区居民	存在高铁等噪音的扰民

其中热度最高的是 2019 年 1 月 11 日至 2019 年 12 月 5 日的车贷诈骗案。查阅相关新闻，确实在此时间段发生过几起车贷诈骗案，因此问题二的模型精确度

较高。

使用 python 的 pandas 库读取附件 3 的信息，选取留言主题那一列文本数据，使用 “jieba”库将文本中的停用词去除，再把处理后的文本转化为 TF-IDF 的特征矩阵，使用 “sklearn”库中的 K-Means 方法对文本进行聚类，类别数为三级分类中标签的数量 390；分类完成后以每类的数量和其中所有反对数、点赞数作为热度数据来源；将热度最高的 5 类数据提取出来生成热点问题留言明细表，再将热点问题留言明细表中每类文本提取关键词生成地点、人群和问题描述，并将每类数据中的时间提取出来生成范围。

通过问题二建立的指标和得出的热度指数，对照附件 3 列举出各热度排名，得出下表：

表三：部分热点问题留言明细表

问题 ID	留言编号	留言用户	留言主题	留言时间	留言详情	反对数	点赞数
1	220711	A00031682	请书记关注 A 市 A4 区 58 车贷案	2019/2/21 18:45:14	A4 区 p2p 公司 58 车贷，非法经营近四年……	0	821
2	208636	A00077171	A 市 A5 区汇金路五矿万境 K9 县存在……	2019/8/19 11:34:04	我是 A 市 A5 区汇金路五矿万境 K9 县 24 栋的……	0	2097
3	223297	A00087522	反映 A 市金毛湾配套入学的问题	2019/4/11 21:02:44	书记先生：您好！我是梅溪湖金毛湾的一名业主……	5	1762
……	……	……	……	……	……	…	……
5	263672	A00041448	A4 区绿地海外滩小区距长……	2019/9/5 13:06:55	您好，近日看到了渝长厦高铁最新的红线征地……	0	669

由总表可得与热度指数排名前 5 相关的留言总数为 228 条，其中点赞数最高的留言编号是：208636，留言详情是与社区居住环境有关的反应。社区建设包含在民政一级标签中，大部分网上留言也是与生活息息相关的问题反应。因此有关部门需对留言热度高的事件积极处理。

八、问题三模型的建立与求解

8.1 模型准备

本文在数据预处理、指标提取、相关度分析、聚类评价等方面使用与问题一和问题二相同的方法，得到比较成熟的神经网络优化算法和聚类结果。

8.2 模型的建立

8.2.1 构建答复评价指标体系

（1）WRC 评价指标

本文从信息质量评价角度出发，将信息特质分为内在特质、内容特质和描述特质三个方面,针对研究文本具有的情感丰富、内容随意与形式多变特点,进行重新整合相应指标,抽取文本特征并分别归属到主客观、信息量和一致性三个类别,说明如下：

表四：答复质量评价 WRC 指标

指标	解释
信息量	从内容上确保评论质量,W 评论长度衡量
相关度	评论内容与评论产品主题的相关性
一致性	衡量评论质量的标准之一,判断评分和情感得分是否一致

（2）1R2C 评价指标

从数据角度出发,建立完整性、有效性和可解释性三类指标，从答复与问题的相关联系来分析文本的答复质量，说明如下：

表五：答复质量评价 1R2C 指标

指标	解释
完整性	衡量评论是否包含数值型和文本型数据
有效性	表示某评论对读者是否可信或有效
可解释性	用于检测评论的被理解程度

8.2.2 构建在线评论质量评价模型

通过上述的所建立的 WRC 评价指标和 1R2C 评价指标，建立文本指标的打分权重，然后计算平均值作为本文评价指标的权重，其中需要符合：

$$\sum_{j}^{n} w_j = 1, n, j = 1,2,\cdots 6,\ \ w_j \in (0,1)$$

上述从多角度.多视角对答复意见的质量进行评价,该方法具有操作性强,效果好等优点，具有良好的权重比较性。

8.3 结果分析

本文通过对所建立的 6 个答复质量评价指标进行分权，其中，文本就是按长度判断时间，15 天以内 1 分，30 天以内 0.5 分，30 天以外 0.1 分，文本长度 85 字以内 0 分，85 到 100 字 0.7 分，100 字以上 1 分，最终求和后与总分相比得到百分数，这样的多目标评价模型使得评价时更具有科学性和准确性，各指标的权重如下表所示：

表六：各指标权重

指标	信息量	相关度	一致性	完整性	有效性	可解释性
权重	14%	25%	20%	13%	12%	16%

该方法更易于确定评价等级和标准，并且直观性强、计算方法简单、可将能够进行定量与无法定量计算的评价项目都考虑等的特点，并且指标包含范围广，影响能力强，是良好的评价因素。

通过所建立的答复质量评价模型，根据所得个指标权重对答复意见进行评价计算，得到答复合格率和时间跨度合格率如下表所示：

表七：评价合格率

答复合格率	84.30%
时间跨度合格率	74.70%

通过所得的合格率，发现在时间跨度上面，政府的答复合格情况有 74.7%，回复速度较慢，但从回复质量来看，整体满意情况比较高，合格率超过 84%，所以此政府的政务能力比较强，具有良好的处理民生能力。

为了进一步的探究答复内容和答复时间对总体评价质量产生的影响，本文文本长度和时间跨度的加分情况进行统计，具体情况如下图所示：

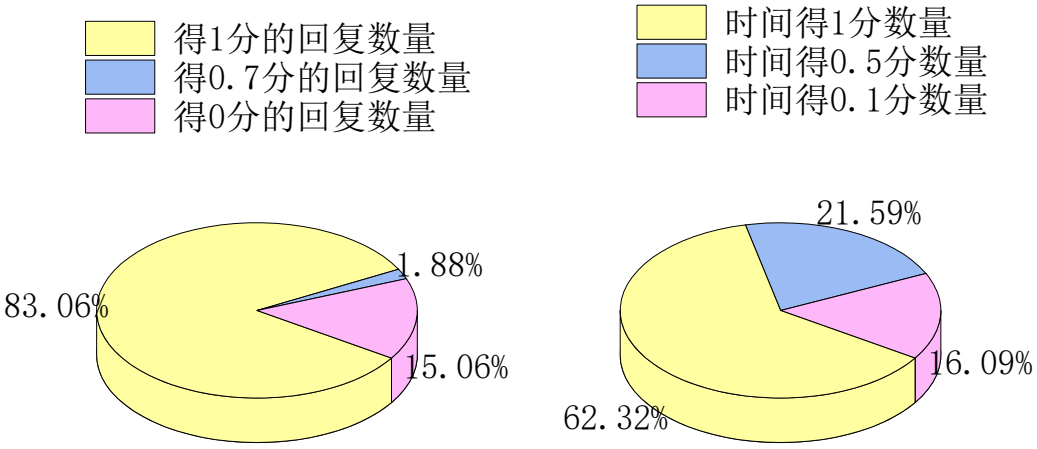

图 13：加分情况百分图

从上图可以得到得 1 分的回复数量和时间得 1 分的数量呈现主体分别站 83.06%和 62.32%，说明政府的政务工作处于积极的状态，在回复时间上有 21.59%的消息回复不及时，需要提高办公效率。0 分的数量占比相对较大，因此需要政府对自身进行严格监管，杜绝人员不作为的现象，使得“智慧政务”发挥更大的作用。

九、模型的优缺点评价

9.1 模型的优点

优点一：整体上卷积神经网络分类模型的错误率显著低于传统分类模型，严重偏差率也低于传统分类模型。

优点二：本文将划分方法与基于密度的聚类方法相结合的 DAK 算法，在算法时间消耗，以及对最后结果的查全率、查准率上都有了较大提高，可以帮助系统更加快速、准确的发现、生成评论信息热点，为相关部门处理好处理此事件的提供时间。

优点三：本文建立的评价模型，从信度分析和效度检验进行评价，可以准确的对答复意见的质量进行考量，具有较高的准确性和可行性。

9.2 模型的缺点

缺点一：卷积神经网络分类模型对于庞大数据和零散类数据的神经学习并不特别优秀，在处理混乱数据时存在分类情况查准率较差的缺点。

缺点二：本文建立的建立 VSM 模型，在向量空间中的维数没有进行确定，因此会对后面聚类的结果产生较大的影响，虽然准确性较高，但是缺乏平维性。

十、模型的改进与推广

10.1 模型的改进

1.本文构建的分类模型计算耗时比传统分类模型稍长，一方面是由于在词表示阶段要通过迭代方法学习词的语义特征，而传统分类模型只需进行简单统计；另一方面是由于卷积神经网络的结构比传统模型复杂，训练参数较多，因此计算过程耗时较多。但实际应用中训练耗时均为离线计算耗时，并且显著高于人工分类的效率，在实际群众留言文本分类中具有可行性与实用性。

2.除了将 DAK（Density And K-means）聚类算法应用于群众留言文本汇聚之外，如能将其与数据融合、知识推理和风俗调查等技术相结合，将在意见缺陷处理中得到更多应用，如热点问题的精确判断与识别、反馈缺陷处理意见的自动推荐等，进而促进“智慧政务”的智能化发展。

10.2 模型的推广

本文建立的模型能够有效提高文本分类和热点问题搜索的准确性，并且不需要过多的无效文本或复杂语句问题，其结果与定性分析结果比较吻合，所得的结论具有较好的参考性和实用性，此评估有一定的参考和现实意义，并且可以将模型推广到其他影响人文和信息化的领域，不止在群众留言文本分类，对社会，民生，经济，发展等领域都发挥着重大作用。

后继的研究还可以考虑文章结构以放宽位置独立假设，根据特征出现位置的不同给特征赋予不同的权重等方法，对分类方法进行改进，特别是对于医学、人文等具有相对固定文本结构的文档。此模型也可以考虑在当前个性化很强的时代，根据用户的信息情况来改进文本分类，根据用户的信息需求的不同，加以调整和规划，使展现的结果更满足用户需求。

十一、参考文献

[1]朱弘扬,马海斌,葛天祎.基于卷积神经网络的高精度文本分类方法[J].电脑知识与技术,2019,15(21):204-207.

[2]刘欣. 基于改进的朴素贝叶斯算法和 KNN 算法在招聘文本分类中的应用[D].河南大学,2019.

[3]陈巧红,王磊,孙麒,贾宇波.卷积神经网络的短文本分类方法[J].计算机系统应用,2019,28(05):137-142.

[4]刘梓权,王慧芳,曹靖,邱剑.基于卷积神经网络的电力设备缺陷文本分类模型研究[J].电网技术,2018,42(02):644-651.

[5]郭银灵. 基于文本分析的在线评论质量评价模型研究[D].内蒙古大学,2017.

[6]张晓娟,刘亚茹,邓福成.基于用户满意度的政务微信服务质量评价模型及其实证研究[J].图书与情报,2017(02):41-47+83.

[7]何跃,蔡博驰.基于因子分析法的微博热度评价模型[J].统计与决策,2016(18):52-54.

[8]王驰. 基于海量网络舆情信息的热点发现[D].电子科技大学,2011.

[9]毛伟,徐蔚然,郭军.基于 n-gram 语言模型和链状朴素贝叶斯分类器的中文文本分类系统[J].中文信息学报,2006(03):29-35.

[10]曹喆岫. 基于数据挖掘的 T 产品质量评价研究[D].南京理工大学,2018.

[11]毛伟. 基于统计语言模型的中文自动文本分类系统[D].北京邮电大学,2006.

[12]毛伟. 基于统计语言模型的中文自动文本分类系统[D].北京邮电大学,2006.

[13]唐立. 基于文本挖掘的网络舆情监控与分析系统的研究与实现[D].湖南大学,2016.

[14]杜鲁燕. 基于语言模型的中文文本分类系统[C]. 中国中文信息学会语音信息专业委员会、中国声学学会语言、听觉和音乐声学分会、中国语言学会语音学分会,2009:304-308.

[15]张舒,李慧,施珺,王成强.结合用户评论与评分信息的推荐算法[J].陕西师范大学学报(自然科学版),2020,48(02):84-91.

[16]赵泽青.网络评论观点挖掘综述[J].现代计算机(专业版),2019(07):49-53.

[17]陈曦.文本挖掘技术在社情民意调查中的应用[J].中国统计,2019(06):27-29.

[18]谌志群,张国煊.文本挖掘与中文文本挖掘模型研究[J].情报科学,2007(07):1046-1051.

[19]王超,彭湃,李波.舆情短文本挖掘的数学模型及其实现[J].数学建模及其应用,2018,7(03):29-36+43.

附录

问题一代码

```
import pandas as pd
import jieba
import re
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neural_network import MLPClassifier
from sklearn.externals import joblib
import csv

# 停用词
stopwords = []
with open('stopwords.txt', errors='ignore') as sf:
    for line in sf.readlines():
        stopwords.append(line.strip())

# 分词处理
def text_cut(in_text):
    in_text = re.sub('[a-zA-Z0-9]',",in_text)
    words = jieba.lcut(in_text)
    cut_text = ' '.join([w for w in words if w not in stopwords and len(w) > 1])
    return cut_text

data = pd.read_excel('附件 2.xlsx')
x = data.留言主题
y = data.一级标签

# 数据拆分
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,
random_state=0)
# 文本分词
x_train_cut = []
x_test_cut = []
for i in x_train:
    x_train_cut.append(text_cut(i))
for i in x_test:
    x_test_cut.append(text_cut(i))

# 特征提取
vectorizer = TfidfVectorizer(min_df=2, ngram_range=(1,2),
strip_accents='unicode', norm='l2', token_pattern=r"(?u)\b\w+\b")
X_train = vectorizer.fit_transform(x_train_cut)
X_test = vectorizer.transform(x_test_cut)

model = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(50,50))
model.fit(X_train, y_train)
```

```
score = model.score(X_test, y_test)
print(score)
```

问题二代码

```
import pandas as pd
import jieba
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.externals import joblib

data = pd.read_excel('附件 3.xlsx')
x = data.留言主题

# 停用词
stopwords = []
with open('stopwords.txt', errors='ignore') as sf:
    for line in sf.readlines():
        stopwords.append(line.strip())

# 分词处理
def text_cut(in_text):
    words = jieba.lcut(in_text)
    cut_text = ' '.join([w for w in words if w not in stopwords and len(w) > 1])
    return cut_text

x_change = []
for i in x:
    x_change.append(text_cut(i))

# 特征提取
vectorizer = TfidfVectorizer(min_df=2, ngram_range=(1,2),
strip_accents='unicode', norm='l2', token_pattern=r"(?u)\b\w+\b")
X = vectorizer.fit_transform(x_change)
num_clusters = 390
k_cluster = KMeans(n_clusters=num_clusters, max_iter=310, n_init=50,
                   init='k-means++', n_jobs=-1)
k_result = k_cluster.fit_predict(X)

data['类别编号'] = k_result
hot_data = {}
for i in range(0, len(data)):
    hot_data[data.iloc[i,7]] = hot_data.get(data.iloc[i,7], 0) + 0.8 +
data.iloc[i,5]*0.2 + data.iloc[i,6]*0.2

hot_top = sorted(hot_data.items(), key = lambda kv:(kv[1], kv[0]), reverse=True)
hot_list = []
for i in range(5):
```

```
        hot_list.append(hot_top[i][0])

    # 将热度前 5 的问题提取出来
    # 1
    one_dataframe = data[(data.类别编号==hot_list[0])]
    one_dataframe.pop('类别编号')
    one = np.ones((len(one_dataframe),1), int)
    one_dataframe.insert(0, '问题 ID', one)
    # 2
    two_dataframe = data[(data.类别编号==hot_list[1])]
    two_dataframe.pop('类别编号')
    two = np.ones((len(two_dataframe),1), int) * 2
    two_dataframe.insert(0, '问题 ID', two)
    # 3
    three_dataframe = data[(data.类别编号==hot_list[2])]
    three_dataframe.pop('类别编号')
    three = np.ones((len(three_dataframe),1), int) * 3
    three_dataframe.insert(0, '问题 ID', three)
    # 4
    four_dataframe = data[(data.类别编号==hot_list[3])]
    four_dataframe.pop('类别编号')
    four = np.ones((len(four_dataframe),1), int) * 4
    four_dataframe.insert(0, '问题 ID', four)
    # 5
    five_dataframe = data[(data.类别编号==hot_list[4])]
    five_dataframe.pop('类别编号')
    five = np.ones((len(five_dataframe),1), int) * 5
    five_dataframe.insert(0, '问题 ID', five)

    all_dataframe = pd.concat([one_dataframe, two_dataframe, three_dataframe,
four_dataframe, five_dataframe])
    pd.DataFrame(all_dataframe).to_excel('热点问题留言明细表.xls',
sheet_name='Sheet1', index=False, header=True)



    import pandas as pd
    import time, datetime
    import numpy as np
    from jieba.analyse import extract_tags


    # 时间跨度提取
    def time_out(time_in):
        t = []
        for i in time_in:
            try:
```

```
                    t.append(datetime.datetime.strptime(i,
'%Y/%m/%d %H:%M:%S'))
                except:
                    t.append(i)
        min_time = t[0]
        max_time = t[0]
        for i in t:
            if i < min_time:
                min_time = i
            if i > max_time:
                max_time = i
        return    str(min_time.year)+'/'+str(min_time.month)+'/'+str(min_time.day)+'
至'+str(max_time.year)+'/'+str(max_time.month)+'/'+str(max_time.day)

    # 关键词提取
    def word_out(text_in):
        t = ''.join([text for text in text_in])
        word = []
        for keyword, weight in extract_tags(t, topK=10, withWeight=True):
            word.append(keyword)
        return ' '.join([w for w in word])

    data = pd.read_excel('热点问题留言明细表.xls')

    # 热度计算及总结
    one_dataframe = data[(data.问题 ID==1)]
    score_1 = len(one_dataframe)
    for i in range(len(one_dataframe)):
        score_1 += (one_dataframe.iloc[i,6]*0.2 + one_dataframe.iloc[i,7]*0.2)
    two_dataframe = data[(data.问题 ID==2)]
    score_2 = len(two_dataframe)
    for i in range(len(two_dataframe)):
        score_2 += (two_dataframe.iloc[i,6]*0.2 + two_dataframe.iloc[i,7]*0.2)
    three_dataframe = data[(data.问题 ID==3)]
    score_3 = len(three_dataframe)
    for i in range(len(three_dataframe)):
        score_3 += (three_dataframe.iloc[i,6]*0.2 + three_dataframe.iloc[i,7]*0.2)
    four_dataframe = data[(data.问题 ID==4)]
    score_4 = len(four_dataframe)
    for i in range(len(four_dataframe)):
        score_4 += (four_dataframe.iloc[i,6]*0.2 + four_dataframe.iloc[i,7]*0.2)
    five_dataframe = data[(data.问题 ID==5)]
    score_5 = len(five_dataframe)
    for i in range(len(five_dataframe)):
        score_5 += (five_dataframe.iloc[i,6]*0.2 + five_dataframe.iloc[i,7]*0.2)

    all_dataframe = pd.DataFrame(columns=['热度排名', '问题 ID', '热度指数', '时
间范围', '地点/人群', '问题描述'])
    all_dataframe.loc[0] = ['1', '1', str(score_1), time_out(list(one_dataframe.留言时
```

```
间)), word_out(list(one_dataframe.留言主题)), word_out(list(one_dataframe.留言主题))]
    all_dataframe.loc[1] = ['2', '2', str(score_2), time_out(list(two_dataframe.留言时间)), word_out(list(two_dataframe.留言主题)), word_out(list(two_dataframe.留言主题))]
    all_dataframe.loc[2] = ['3', '3', str(score_3), time_out(list(three_dataframe.留言时间)), word_out(list(three_dataframe.留言主题)), word_out(list(three_dataframe.留言主题))]
    all_dataframe.loc[3] = ['4', '4', str(score_4), time_out(list(four_dataframe.留言时间)), word_out(list(four_dataframe.留言主题)), word_out(list(four_dataframe.留言主题))]
    all_dataframe.loc[4] = ['5', '5', str(score_5), time_out(list(five_dataframe.留言时间)), word_out(list(five_dataframe.留言主题)), word_out(list(five_dataframe.留言主题))]

    pd.DataFrame(all_dataframe).to_excel('热点问题表.xls', sheet_name='Sheet1', index=False, header=True)
```

问题三代码

```
import pandas as pd
import time
import datetime

data = pd.read_excel('附件 4.xlsx')
answer_idea = data.答复意见

# 答复合格率
answer_succed = 0.0
answer_1 = 0      # 1 分答复数量
answer_07 = 0     # 0.7 分答复数量
answer_0 = 0      # 0 分答复数量
for answer in answer_idea:
    if len(answer) > 100:
        answer_succed += 1
        answer_1 += 1
    if 85 < len(answer) < 101:
        answer_succed += 0.7
        answer_07 += 1
    if len(answer) < 86:
        answer_0 += 1
answer_percent = answer_succed/len(answer_idea)
print(answer_1, answer_07, answer_0)
print(answer_percent)

# 时间跨度合格率
def day_len(in_time, out_time):
```

```
        try:
            front = time.strptime(in_time, '%Y/%m/%d %H:%M:%S')
        except:
            return 17
        front = datetime.datetime(front[0], front[1], front[2])
        try:
            later = time.strptime(out_time, '%Y/%m/%d %H:%M:%S')
        except:
            return 17
        later = datetime.datetime(later[0], later[1], later[2])
        return (later - front).days
time_succed = 0.0
time_1 = 0       # 1 分时间数量
time_05 = 0      # 0.7 分时间数量
time_01 = 0       # 0 分时间数量
for i in range(len(data)):
        time_len = day_len(data.iloc[i,3], data.iloc[i,6])
        if time_len < 16:
            time_succed += 1
            time_1 += 1
        if 15 < time_len < 31:
            time_succed += 0.5
            time_05 += 1
        if time_len > 30:
            time_succed += 0.1
            time_01 += 1
time_percent = time_succed/len(data)
print(time_1, time_05, time_01)
print(time_percent)
```