“智慧政务”中的文本挖掘应用

摘要：

“智慧政务”是指政府相关部门充分利用物联网、云计算、大数据分析、移动互联网等新一代信息技术，以用户创新、大众创新、开放创新、共同创新为特征，强调作为平台的政府架构，并以此为基础实现政府、市场、社会多方协同的公共价值塑造，实现政府管理与公共服务的精细化、智能化、社会化。本文利用互联网公开来源的群众问政留言记录以及相关部门对留言答复的数据，开展了文本挖掘的数据分析，以提高政府相关部门的工作效率。

对于问题一，本文首先对留言详情进行文本分词，根据流行的通用词库以及本文自定义的停用词库完善分词结果。随后在特征工程中采用 TF-IDF、WordCountVector、TF-IDF 后再奇异值分解、Word2Vec 四种不同的文本数值化方式，将特征为文本的数据集转换为特征为数值的数据集，另外，本文还关注了一些基于文本自身的可挖掘的特征，如词语数、句子数、各词性词频等特征。在分类模型的建立中，本文首先在 Logistic 回归，朴素贝叶斯，SVM，XGBoost 这些基础模型上，分别去尝试本文上面不同手段得到的不同特征，并采用网格搜索算法寻找如罚项系数、学习率、批处理数等最优超参数，随后本文将本文的特征数据集在复杂网络模型上去训练，分别在常规神经网络、CNN、LSTM、Bi-LSTM、以及 CNN 的变种 Kmax-CNN、AVCNN 上进行训练，寻求最优的网络结构。最后，本文搭建一个集成模型，第一层选取取得相对最优结果的罚项系数为 2.3 的 Logistic 回归模型、基于 WordCountVector 特征数据集的朴素贝叶斯模型、以及分别基于 Word2Vec 特征数据集的 XGBoost 模型，第二层选用 CNN 网络模型。本文的集成模型在测试集上取得了 91.6%的分类准确率。在最后，本文尝试了迁移学习的方法，通过迁移已有的 BERT 模型和 Word2Vec 模型来预训练本文的网络，已达到快速收敛和减轻过拟合的效果；尝试了 VAE 生成式模型，获得大约 50000 条新的训练数据，在结果上略微提升，最终本文的模型达到了 92.1%的分类准确率。

对于问题二，本文对群众留言的数据中提取其中的热点问题。第一，本文对所给的留言数据进行分词、去噪、去停用词等预处理操作；第二，利用 VSM 模型对群众留言进行文本表示，构建留言的词频矩阵，利用 TF-IDF 特征权重提取留言数据中的特征词并将群众留言在这些特征词上的数值权重来表示每一条群众留言，将每一条留言文本数据转化为相同维度的数值向量；第三，本文借助余弦相似性，利用自适应 Kmeans 聚类的方法对留言数据进行文本聚类，自适应地选择适当的聚类数，进行群众问题的提取；第四，本文利用了群众留言和网络问政的特性，利用所给的数据，定义了包含市民关注度、热点突发性、热点时效性的热点评估方法，进而对所提取的问题进行热点评估，得到所给数据中群众集中反映的热点问题排行。

对于问题三，本文建立答复评价模型，从相关性、整体性、专业性、及时性四个角度对答复意见进行等级划分。第一，通过 TF-IDF 算法将群众留言及对应答复意见转化为向量，再计算其余弦相似度，作为答复意见的相关性指标。第二，通过 TF-IDF 算法挑选出答复意见的特征词后计算信息熵，作为答复意见的完整性指标。第三，搭建 LSTM 模型对答复进行情感分析，得到各答复情感值，衡量专业性。第四，计算每个留言日期与答复日期的工作日天数，作为衡量及时性的标准。最后，结合 Kmeans 聚类的方法与常用日期划分方法对以上四个指标进行五等级划分，得到对答复意见的评价方式。

关键词：文本挖掘；集成模型；LSTM；CNN；迁移学习；VAE；自适应聚类；评价模型

目录

1 挖掘目标

政府相关部门利用互联网相关问政平台的群众留言数据，开展群众留言的文本挖掘任务，以提高政府的工作效率以及群众问题能够得到及时的突出和反馈。利用深度学习、VSM 文本表示、自适应 Kmeans 文本聚类、生成式网络等算法，达到以下三个目标：

1）利用附件 2 数据，训练文本有监督分类模型，根据群众的留言主题以及留言详情的文本信息，输出该留言的一级标签。

2）利用附件 3 数据，对群众留言进行无监督文本分类，建立热点评估指标，提取某段时间内群众留言中所集中反映的热点问题。

3）利用附件 4 数据，对相关部门对留言的答复意见，建立对答复意见的质量评价体系。

2 符号说明

符号	说明
TF	文档频率
IDF	逆文档频率
Follow	市民关注度
Comment	评论指标
Outburst	热度突发性
Decay	热度衰弱系数
R	问题热度
H	信息熵

3 问题一

3.1 问题分析

本文现在需要解决的是一个 7 分类问题，数据量适中，接近 10000 条，不存在明显的类别不均衡现象。本文将从传统的机器学习文本分类模型到现今流行的基于深度学习的文本分类模型，最后给出一个模型集成，达到效果最优。本文会分别采用 TF-IDF、WordCountVector、SVD、Word2Vec 等手段将文本数据转换为数值数据。本文覆盖的文本分类方法有:Logistic Regression、Naive Bayes、SVM、Xgboost、Grid Search、Dense Network、LSTM、CNN、GRU、Ensembling 等。并且在最后，本文会尝试借助迁移学习，调用现成的已经训练好的 Word2Vec 模型和 BERT 模型，试图预训练本文的网络，此外将尝试适中量级的数据集变为大量级的数据集，观察是否对结果有提升。

$$IDF(t_i)=\log(\frac{\text{文本总数}}{\text{包含特征词}t_i\text{的文本数}})$$

综合考虑两个因子，得出本文需要的 IF-IDF 因子为：$TFIDF(t_i)=TF(t_i)*IDF(t_i)$。同时因为市民留言中包含主题和内容两项，而留言主题更能突出一个问题的关键所在，所以为了综合考虑市民问政中标题和留言详情的重要性与否，本文将对市民留言的标题给予更高的权重。在此本文定义权重为 3:1。即关键词出现在标题便增加权重 3，出现在内容便增加权重 1，都出现则增加权重 4，否则取值 0。

最终本文得到：8289x64501 的稀疏矩阵，并存有 692332 个元素。

（2）词汇计数

词汇计数使用 Count Vectorizer 来 fit 训练集和测试集（半监督学习）计数向量是数据集的矩阵表示，其中每行代表来自语料库的文档，每列表示来自语料库的术语，并且每个单元格表示特定文档中特定术语的频率计数。首先创建一个向量计数器对象，再使用向量计数器对象转换训练集和验证集。

最终本文得到 8289x64254 的稀疏矩阵，共存有 696748 个元素。

（3）奇异值分解降维

在 TF-IDF 得到的数值数据集矩阵基础之上，再进行奇异值分解降维。奇异值分解能够保存矩阵的大多数的信息，而又由于本文这里是一个大型稀疏矩阵，非常适合通过奇异值分解来降维。本文设置降维特征数为 200，所以将 8289x64501 的矩阵变为 8289X200 的矩阵，即每个文本的特征数被压缩到 200 维。

（4）基于 word2vec 的词嵌入

在不深入细节的情况下，本文将解释如何创建语句向量。Word2Vec 有 3 个值得关注的参数，iter 是模型训练时迭代的次数，本文参与训练的文本量较少，需要把这个参数调大一些，设置为 100；本文选择 CBOW 模式；CBOW 模式示意图如下

3.2 文本数据预处理

3.2.1 文本分词

值得注意的是，分词是任何中文文本分类的起点，分词的质量会直接影响到后面的模型效果。本文采取下面三步去完成本文的分词操作。

1.设置可靠的自定义词典，以便分词更精准；本文通过遍历词库，提取地点，作为词典添加进分词库中，保证这些词不会被分开，分词库中包括如“A3 区”、“华庭小区”、“A1 区”、“城中区”、“人民街道”……等。

2.采用 jieba 分词中的精准模式，方便且省时省力。

3.停用词库+编写预处理类，首先通过哈工大停用词表对分词结果进行一次筛选，随后浏览分词结果，对部分未能妥善解决的不需要的词，建立预处理类。诸如部分特殊格式的字符并没有被筛掉，自定义函数通过 ASCLL 码进行处理即可。

最终分词结果如下所示，可以看出如 A3 区，华庭小区等地名并没有被分词或删去：

	分类	正文	文本分词
0	城乡建设	A3区大道西行便道，未管所路口至加油站路段，人行道包括路灯杆，被圈西湖建筑集团燕子山安置房项...	A3区 大道 西行 道 未管 路口 加油站 路段 人行道 包括 路灯 杆 圈 西湖 建筑 集...
1	城乡建设	位于书院路主干道的在水一方大厦一楼至四楼人为拆除水、电等设施后，烂尾多年，用护栏围着，不但占...	位于 书院 路 主干道 在水一方 大厦 一楼 四楼 人为 拆除 水 电等 设施 烂尾 多年 ...
2	城乡建设	尊敬的领导：A1区苑小区位于A1区火炬路，小区物业A市程明物业管理有限公司，未经小区业主同意...	尊敬 领导 A1 区苑 小区 位于 A1 区 火炬 路 小区 物业 市程明 物业管理 有限公...
3	城乡建设	A1区A2区华庭小区高层为二次供水，楼顶水箱长年不洗，现在自来水龙头的水严重霉味，大家都知道...	A1 区 A2 区 华庭小区 高层 二次 供水 楼顶 水箱 长年 不洗 自来水 龙头 水 霉...

图 1：分词结果示意

3.2.2 文本数值化

（1）TF-IDF 方法

TF-IDF 是一种用于信息检索与文本挖掘的常用加权技术。TF-IDF 是一种统计方法，用以评估一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降。TF-IDF 的主要思想是：如果某个单词在一篇文章中出现的频率 TF 高，并且在其他文章中很少出现，则认为此词或者短语具有很好的类别区分能力，适合用来分类。

一个词在文本中出现的频率用词频表示，TF 方法的基本思想是：如果一个词在文本中的 TF 值较高，则表明该词对于文本来说是重要的。计算公式如下：

$$TF(t_i)=\frac{\text{特征词}t_i\text{在文本}D\text{中出现的次数}}{\text{文本}D\text{中词语总数}}$$

但是有一些词会频繁地出现于很多文本中，在这种情况下，虽然词的 TF 值很高，但实际上这类词对于文本的重要性却很低。则考虑逆文档频率 IDF，如果一个词很少在大量文本中出现，则说明该词具有很强的标识性，可以用于快速地确定一个文本，其权重应较大。反之，如果一个词频繁出现于很多文本中，则说明该词并不具有标识文档的作用，其权重也应较低。由此得出其计算公式如下：

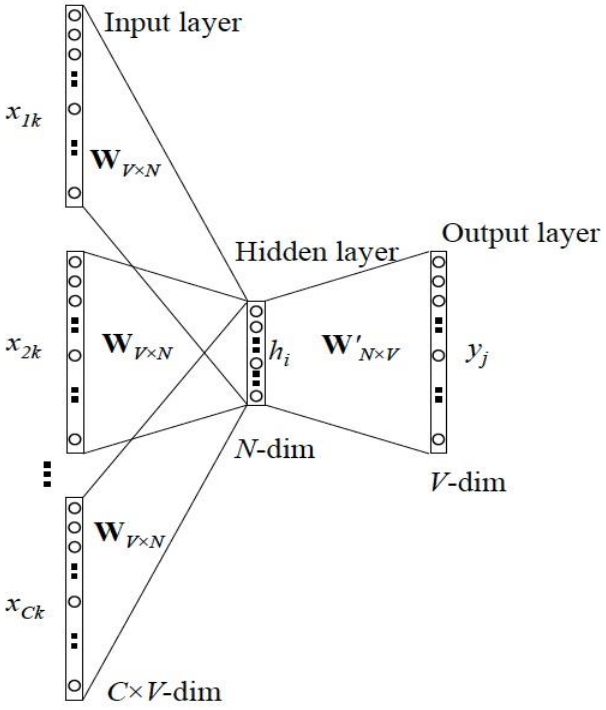


图 2：Word2Vec-CBOW 模型示意图

设置 window 控制窗口设置为 8，它指当前词和预测词之间的最大距离，如果设得较小，那么模型学习到的是词汇间的功能性特征（词性相异），如果设置得较大，会学习到词汇之间的相似性特征（词性相同）的大小。词嵌入是使用稠密向量代表词语和文档的一种形式。向量空间中单词的位置是从该单词在文本中的上下文学习到的，词嵌入可以使用输入语料本身训练，也可以使用预先训练好的词嵌入模型生成，接下来介绍如何在模型中使用预先训练好的词嵌入模型，主要有四步：

1. 加载预先训练好的词嵌入模型
2. 创建一个分词对象
3. 将文本文档转换为分词序列并填充它们
4. 创建分词和各自嵌入的映射

最终本文将数据集中出现的词语全部转换为 100 维的向量表示，如：查看“汽车”的词向量如下所示：

```
array([-0.21903884, -0.06766183, -0.5181544 , -0.07016603,  0.39717054,
        0.41125023, -0.66236126,  0.57061577, -0.00759434, -0.17310078,
        0.2788591 , -0.4718603 ,  0.6120221 , -0.24344605, -0.2003771 ,
       -0.4677188 ,  0.1364089 , -0.5970649 ,  0.22028811, -0.24435623,
        0.6033185 ,  0.57561684, -0.38641387, -0.15079185, -0.22847909,
       -0.3385108 ,  0.8041264 ,  0.33294478, -0.6075502 ,  0.03735462,
        0.21125093, -0.85933495,  0.819982  , -1.0566941 , -0.4906564 ,
       -1.0069236 , -0.49431568, -0.72750574, -0.24773137, -1.0340067 ,
        0.00485106, -0.82373166, -1.2222313 ,  0.6965263 , -0.42226195,
        0.3220477 ,  0.10207474, -0.06779543, -0.1122162 ,  0.17151849,
       -0.524049  ,  0.4907113 ,  0.24107367,  1.0205344 ,  0.1334752 ,
       -0.73179567, -0.85948753, -0.0200808 ,  0.5247138 , -0.51536554,
        0.9195719 ,  0.00268296,  0.00864221,  0.06362939, -0.18447924,
       -0.31025404, -1.0081354 ,  0.48542136,  0.3556386 , -0.7304114 ,
       -0.3413708 , -0.51785433, -0.4239142 , -0.8701061 ,  0.02287479,
        0.46970803, -0.55133474, -0.17805062, -0.16835544, -0.10841725,
       -0.9310898 , -0.06347245, -0.6638834 , -0.17923109, -0.10515457,
       -0.7451159 ,  1.1108774 ,  0.05636939,  0.86764467,  0.50815266,
        0.3169565 , -0.33780682,  0.35737082, -0.1563303 , -0.43920156,
        0.60935944, -0.4101953 , -0.47730157, -0.04290425,  0.54319835],
      dtype=float32)
```

3.2.3 基于文本的额外特征挖掘

创建额外的基于文本的特征有时可以提升模型效果。比如下面的例子：

- 文档的词语计数—文档中词语的总数量
- 文档的词性计数—文档中词性的总数量
- 文档的平均字密度--文件中使用的词语的平均长度
- 完整文章中的标点符号出现次数--文档中标点符号的总数量
- 完整文章中标题出现的次数—文档中适当的主题（标题）的总数量

这些特征有很强的实验性质，在后续实际输入模型中时会对比试验。

3.2.4 分类数据集预处理

（1）定义损失函数：

这是一个典型的文本多分类问题，需要将文本划分到给定的 7 个主题上。由题意设定 F1-Score 指标，为各类别 F1-Score 的不加权平均。

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

其中 P_i 为第 i 类的查准率，R_i 为第 i 类的查全率。

（2）标签数值化：

将文本标签转化为数字。如标签为“城乡建设”则对应变为 6。其余以此类推。

（3）训练集验证集划分：

在进一步研究之前，本文必须将数据分成训练和验证集。 按照 9:1 的比例，最终本文得到 8289 训练集，921 测试集。

3.3 文本分类基础模型

本文先创建一些非常基础的模型。

（1）逻辑斯蒂回归（Logistic Regression）

贝叶斯进行超参数调优。本文依次创建 pipeline、搜索参数设置、网格搜索模型初始化、训练网格搜索模型。相比于之前的朴素贝叶斯，本次得分都有所提高。

Model	Change_Term	Change	F1-Score
LogisticRegression	罚项系数 C	1.0 → 2.3	0.902 → 0.914
Naïve Bayes	拉普拉斯平滑系数 Alpha	1.0 → 0.5	0.855 → 0.890
SVD	降维特征数 n_components	120 → 200	
SVM	罚项系数 C	1.0 → 5.0	0.879 → 0.907
XGBoost	树最大深度 Max_depth	8.0 → 9.0	0.869 → 0.878

表 2：参数优化后，模型准确率提升对比表

3.5 复杂网络模型

这是一个深度学习大行其道的时代，文本分类问题在它的指引下得到了突飞猛进的发展，本文将从最简单的全连接层网络开始，再依次尝试 CNN 系列、LSTM 系列、GRU 系列，并选择其中表现最好的一个系列做深入研究。

（1）全连接网络：

创建 1 个 3 层的序列神经网络（Sequential Neural Net），在多次对神经网络的参数调优，添加或删减连接层参数，增加 Dropout 等操作后，获得预测分类结果。F1-Score 为 0.819。

（2）CNN 网络：

为了探究 CNN 的结构是否适合该数据集的效率，本文首先设计一个简单的 CNN 网络观察其效果，同理对 LSTM 系列网络以及 GRU 系列网络。网络结构详细和参数数量如下所示：

Layer (type)	Output Shape	Param #
input (InputLayer)	[(None, 453)]	0
layer_embedding (Embedding)	(None, 453, 100)	2803200
conv1d (Conv1D)	(None, 449, 128)	64128
global_max_pooling1d (Global	(None, 128)	0
dense (Dense)	(None, 64)	8256
dense_1 (Dense)	(None, 7)	455

Total params: 2,876,039
Trainable params: 2,876,039
Non-trainable params: 0

本文设置 batch_size 为 64，epochs 为 10 次，最终取得的 F-score 为 0.8694。

（3）LSTM 系列网络：[1]

为了更进一步，笔者使用 LSTM，本文需要对文本数据进行 Tokenize，对文本序列进行 zero 填充，基于已有的数据集中的词汇创建一个词嵌入矩阵（Embedding Matrix），基于前面训练的 Word2vec 词向量，使用 1 个两层的 LSTM 模型，使用 early stopping 来停止在最佳的迭代节点。最终取得 0.8149 的 F1-score。让本文看看基于 attention 机制的双向长短时记忆(Bi-Directional LSTM)是否可以给本文带来更好的结果。 对于 Keras 来说，使用 Bilstm 小菜一碟，基于前面训练的 Word2vec 词向量，构建 1 个 2 层的 Bidirectional-LSTM ，在模型拟合时，使用 early stopping 这个回调函数（Callback Function）。最终取得 0.8259 的 F1-score。

（4）GRU 系列网络：

本文首先尝试下面这种网络结构，即将 CNN 与 GRU 相结合的简单网络结构，最终 F1-Score 为 0.8023。

这个非常基础的模型基于 TF-IDF 变换后得到的特征数据集，最终本文得到的 F1-Score 为 0.902。随后本文基于相同模型采用不同的特征，使用 WordCountVector 获得的特征数据集作为输入，而不再使用 TF-IDF。 利用提取的 word counts 特征来训练一个简单的逻辑斯蒂回归。F1-Score 为 0.805，效果不佳。

（2）朴素贝叶斯模型

仿照在逻辑斯蒂回归中采用的 TF-IDF 和 Word-Count 文本特征数据集分别使用另外一个非常有名且简单的模型 Naïve Bayes，分别得到 F1-score0.824 和 0.899。

（3）SVM 模型

传统文本分类算法里还有支持向量机（SVM）。由于 SVM 需要花费大量时间，因此在应用 SVM 之前，本文将使用奇异值分解（Singular Value Decomposition ）来减少 TF-IDF 中的特征数量。同时，在使用 SVM 之前，本文还需要将数据标准化（Standardize Data）使用 SVD 进行降维，由于 components 设为 200。所以本文的新的特征数据集的特征数为 200. 调用 SVM 模型得到准确率为 0.895。SVM 在这些数据上表现不错。

（4）XGBoost

本文分别在上述建立的四种特征数据集上使用 XGBoost 模型，参数使用默认参数，分别获得如下 F1-Score：

特征数据集	F1-Score
基于 TF-IDF 特征	0.898
基于 WordCountVector 特征	0.843
基于 TF-IDF 的 SVD 特征	0.851
基于 Word2Vec 特征	0.841

表 1：参数在 XGBoost 上不同数据集训练结果

3.4 参数优化

网格搜索是一种超参数优化的技巧。本文通过该方法获取最优的参数组合来产生良好的文本分类效果。在本节中，我将讨论使用基于逻辑回归模型的网格搜索。在开始网格搜索之前，本文需要创建一个评分函数，这可以通过 scikit-learn 的 make_scorer 函数完成。接下来本文需要一个参数网格。对于 SVD，本文评估 120、150、180、200 个分量（Components），对于逻辑回归，本文评估三个不同的罚项系数。对这些参数进行网格搜索。最终得分跟本文之前的 SVM 的结果相近。这种技术可用于对 xgboost 甚至多项式朴素

Layer (type)	Output Shape	Param #
input (InputLayer)	[(None, 453)]	0
layer_embedding (Embedding)	(None, 453, 100)	2803200
conv1d_19 (Conv1D)	(None, 453, 32)	9632
max_pooling1d_1 (MaxPooling1	(None, 226, 32)	0
gru_1 (GRU)	(None, 100)	39900
dense_14 (Dense)	(None, 7)	707

Total params: 2,853,439
Trainable params: 2,853,439
Non-trainable params: 0

本文再尝试 GRU 的变种 Bi-GRU 模型，取得了 F1-score 为 0.8022。

可以看出，LSTM 系列的网络结构和 GRU 系列的网络结构在该数据集上效果都不太好，所以本文继续研究 CNN 的变种 AVCNN_Model 和 KMax_CNN_Model，看看是否能够有突破。

（5）KMax_CNN_Model 网络[2]

在该网络上，本文获得了 0.889 的 F1-Score，该网络结构可视化如下：

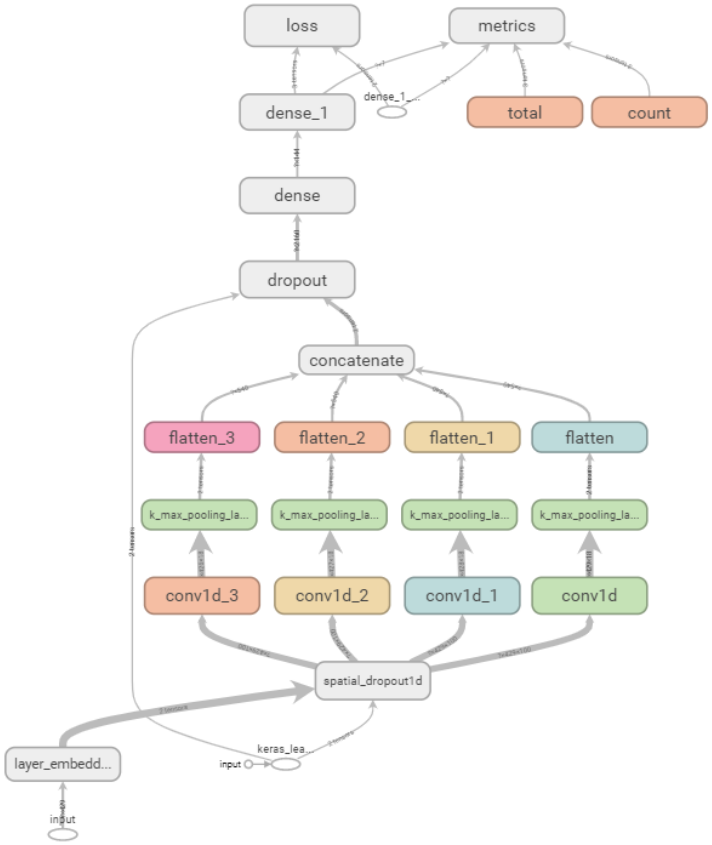


图 3：KMax_CNN_Model 模型示意图

Layer (type)	Output Shape	Param #	Connected to
input (InputLayer)	[(None, 453)]	0	
layer_embedding (Embedding)	(None, 453, 100)	2803200	input[0][0]
spatial_dropout1d_2 (SpatialDro	(None, 453, 100)	0	layer_embedding[0][0]
conv1d_10 (Conv1D)	(None, 453, 300)	30300	spatial_dropout1d_2[0][0]
conv1d_11 (Conv1D)	(None, 452, 300)	60300	spatial_dropout1d_2[0][0]
conv1d_12 (Conv1D)	(None, 451, 300)	90300	spatial_dropout1d_2[0][0]
conv1d_13 (Conv1D)	(None, 450, 300)	120300	spatial_dropout1d_2[0][0]
global_max_pooling1d_3 (GlobalM	(None, 300)	0	conv1d_10[0][0] conv1d_11[0][0] conv1d_12[0][0] conv1d_13[0][0]
attention_weighted_average_laye	(None, 300)	300	conv1d_10[0][0] conv1d_11[0][0] conv1d_12[0][0] conv1d_13[0][0]
global_average_pooling1d_2 (Glo	(None, 300)	0	conv1d_10[0][0] conv1d_11[0][0] conv1d_12[0][0] conv1d_13[0][0]
concatenate_4 (Concatenate)	(None, 1200)	0	global_max_pooling1d_3[0][0] global_max_pooling1d_3[1][0] global_max_pooling1d_3[2][0] global_max_pooling1d_3[3][0] attention_weighted_average_layer_ attention_weighted_average_layer_ attention_weighted_average_layer_ attention_weighted_average_layer_ global_average_pooling1d_2[0][0] global_average_pooling1d_2[1][0] global_average_pooling1d_2[2][0] global_average_pooling1d_2[3][0]
concatenate_5 (Concatenate)	(None, 3600)	0	concatenate_4[0][0] concatenate_4[1][0] concatenate_4[2][0]
dropout_2 (Dropout)	(None, 3600)	0	concatenate_5[0][0]
dense_8 (Dense)	(None, 144)	518544	dropout_2[0][0]
dense_9 (Dense)	(None, 7)	1015	dense_8[0][0]

Total params: 3,624,259
Trainable params: 3,624,259
Non-trainable params: 0

本文同样进行 10 次迭代，设置的批处理数为 64，设定早停参数，训练结果如下所示：

损失函数和准确率随着迭代变化如下图：

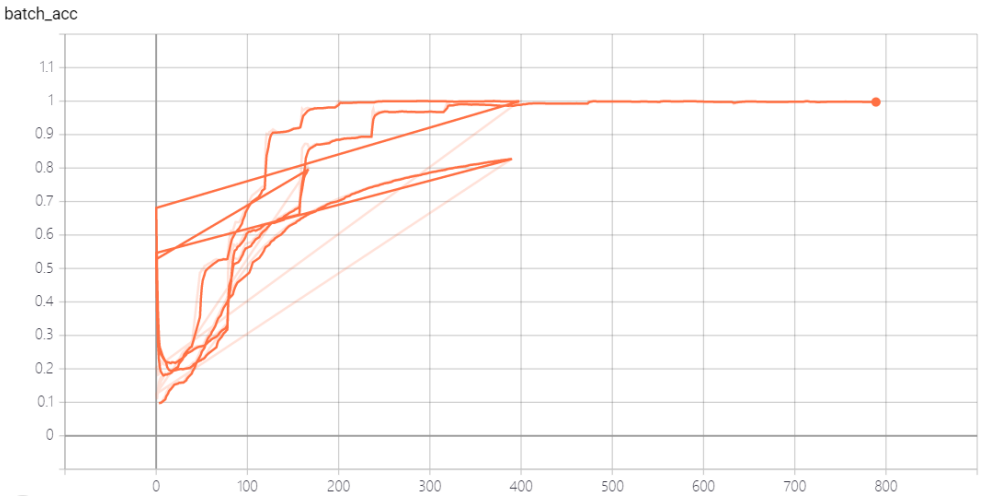


图 4：准确率与迭代次数的关系图

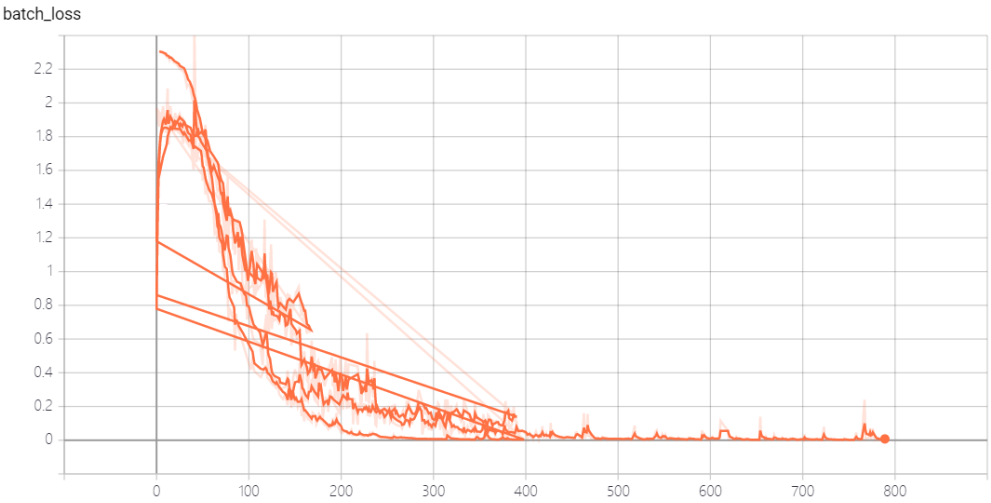


图 5：损失值与迭代次数的关系图

（6）AVCNN_Model 网络：[3]

网络结构和参数数量如下所示：

不然会出现拖后腿的情况，导致模型的整体性能还不如单个模型的要好。所以本文都精心挑选的准确率全都接近 90%或 90%以上的基本模型。

训练网络过程中发现第二层梯度在训练 30 轮之后偶尔上扬幅度会过大，会导致后续训练的损失函数值来回震荡，因此需要在每轮的每批数据训练之后对梯度进行裁切。将该层所有梯度连接成一个向量g，给定裁切阈值$\theta = 1.5$，令新的梯度g^*为：

$$g^* = \min\left(\frac{\theta}{||g||}, 1\right)g$$

梯度裁切后训练的收敛效果显著改善，参数的更新梯度以理想的方式趋于平缓，不再反复震荡。裁切前后梯度的 TSNE 流形降维分布图如图所示：

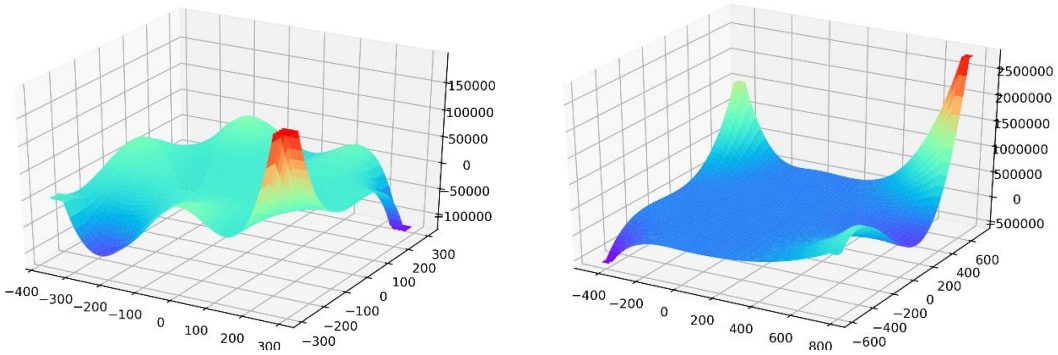


图 6：裁切前后集成模型第二层梯度的 TSNE 流形降维分布图

本文取得了 F1-Score 为 0.919 的最终成绩。本文看到集成模型在很大程度上提高了分数。分数的随着迭代变化的示意图如下所示：

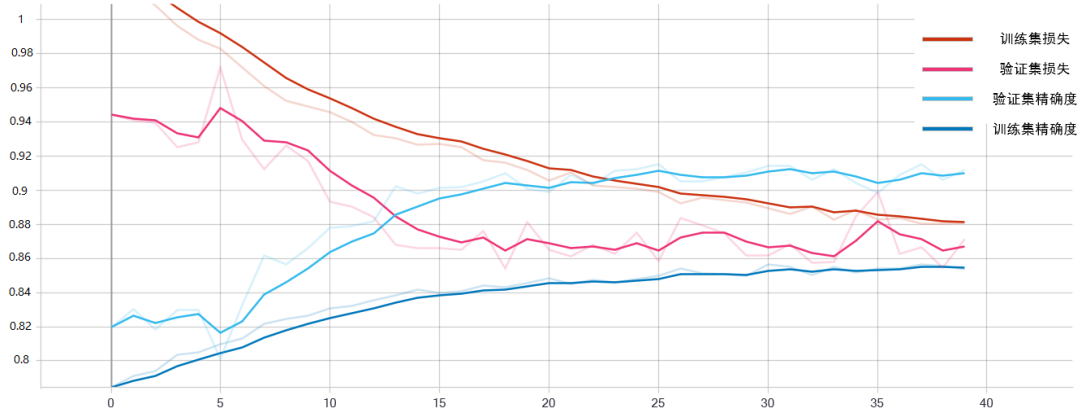


图 7：集成模型在验证集合训练集上的准确率以及损失函数的变化图

3.7 模型提高

（1）迁移学习

```
Epoch 1/10
78/79 [============================>.] - ETA: 0s - loss: 1.8012 - acc: 0.2899Epoch 1/10
79/79 [==============================] - 16s 198ms/step - loss: 1.8000 - acc: 0.2900 - val_loss: 1.4818 - val_acc: 0.4776
Epoch 2/10
78/79 [============================>.] - ETA: 0s - loss: 1.1874 - acc: 0.5719Epoch 1/10
79/79 [==============================] - 13s 162ms/step - loss: 1.1799 - acc: 0.5722 - val_loss: 0.8650 - val_acc: 0.7225
Epoch 3/10
78/79 [============================>.] - ETA: 0s - loss: 0.5990 - acc: 0.8093Epoch 1/10
79/79 [==============================] - 13s 161ms/step - loss: 0.5933 - acc: 0.8096 - val_loss: 0.5202 - val_acc: 0.8432
Epoch 4/10
78/79 [============================>.] - ETA: 0s - loss: 0.2870 - acc: 0.9189Epoch 1/10
79/79 [==============================] - 13s 161ms/step - loss: 0.2837 - acc: 0.9190 - val_loss: 0.3845 - val_acc: 0.8893
Epoch 5/10
78/79 [============================>.] - ETA: 0s - loss: 0.1247 - acc: 0.9649Epoch 1/10
79/79 [==============================] - 13s 160ms/step - loss: 0.1259 - acc: 0.9648 - val_loss: 0.4144 - val_acc: 0.8854
Epoch 6/10
78/79 [============================>.] - ETA: 0s - loss: 0.0673 - acc: 0.9808Epoch 1/10
79/79 [==============================] - 13s 160ms/step - loss: 0.0669 - acc: 0.9808 - val_loss: 0.4838 - val_acc: 0.8914
Epoch 7/10
78/79 [============================>.] - ETA: 0s - loss: 0.0354 - acc: 0.9904Epoch 1/10
79/79 [==============================] - 13s 161ms/step - loss: 0.0388 - acc: 0.9902 - val_loss: 0.5159 - val_acc: 0.8867
Epoch 8/10
78/79 [============================>.] - ETA: 0s - loss: 0.0268 - acc: 0.9920Epoch 1/10
79/79 [==============================] - 13s 162ms/step - loss: 0.0267 - acc: 0.9920 - val_loss: 0.6257 - val_acc: 0.8836
Epoch 9/10
78/79 [============================>.] - ETA: 0s - loss: 0.0232 - acc: 0.9956Epoch 1/10
79/79 [==============================] - 13s 159ms/step - loss: 0.0229 - acc: 0.9956 - val_loss: 0.6521 - val_acc: 0.8819
Epoch 10/10
78/79 [============================>.] - ETA: 0s - loss: 0.0195 - acc: 0.9966Epoch 1/10
79/79 [==============================] - 13s 158ms/step - loss: 0.0193 - acc: 0.9966 - val_loss: 0.6700 - val_acc: 0.8849
```

本文在该 7 分类问题上取得的最终结果如下所示，达到 0.8824 的 F1-Score：

	precision	recall	f1-score	support
交通运输	0.7712	0.7521	0.7615	121
劳动和社会保障	0.9531	0.9196	0.9361	398
卫生计生	0.9045	0.8994	0.9020	179
商贸旅游	0.8719	0.8845	0.8781	277
城乡建设	0.8558	0.8558	0.8558	423
教育文体	0.9270	0.9419	0.9344	310
环境保护	0.8894	0.9296	0.9091	199
accuracy			0.8925	1907
macro avg	0.8818	0.8833	0.8824	1907
weighted avg	0.8927	0.8925	0.8925	1907

随着本文的持续优化，模型的性能将不断提高。

3.6 模型集成

模型集成（Model Ensembling）。集多个文本分类模型之长，合成一个很棒的分类融合模型。同时为每个 level 的集成模型指定特定的使用数据。

第一层模型以及参数设置	数据（基于）
LogisticRegression(C=2.3,solver='lbfgs',multi_class='multinomial',max_iter=500)	TF-IDF
LogisticRegression(C=1.0,solver='lbfgs',multi_class='multinomial',max_iter=500),	WordCountVector
SVC(C=5.0, probability=True)	SVD
MultinomialNB(alpha = 0.5)	TF-IDF
第二层模型	**数据（基于）**
XGBClassifier(max_depth=9, n_estimators=200, subsample=0.8, learning_rate=0.1)	Word2Vec

表 4：双层集成模型结构

但要注意，集成模型只有在参与集成的模型表现都不差的情况下才能取得良好的效果，

迁移学习近年来在图形领域中得到了快速的发展，主要在于某些特定的领域不具备足够的数据，不能让深度模型学习的很好，需要从其它领域训练好的模型迁移过来，再使用该模型进行微调，使得该模型能很好地拟合少量数据的同时又具备较好的泛化能力（不过拟合）。迁移学习方法在文本分类中的应用主要集中于基于 Word2Vec、BERT 等预训练模型来提取高质量的文本特征，再喂给分类器，本文采用的亿训练好的模型如下：

训练语料	微信公众号的文章，多领域，属于中文平衡语料
语料数量	800 万篇，总词数达到 650 亿
模型词数	共 352196 词，基本是中文词，包含常见英文词
模型结构	Skip-Gram + Huffman Softmax
向量维度	256 维
分词工具	Jieba 分词，加入了有 50 万词条的词典，关闭了新词发现
训练工具	Gensim 的 Word2Vec，服务器训练了 7 天
其他情况	窗口大小为 10，最小词频是 64，迭代了 10 次

表 5 模型概况

文本分类迁移学习整体流程如下：

•前人在海量数据集上训练语言模型。之后发布这个模型，而不只是词嵌入向量的表达结果；

•普通用户拿到这个模型后，把它在自己的训练文本上微调，这样一来，就有了符合自己任务问题领域上下文的语言模型；

•把这个语言模型的头部，加上一个分类器，在训练数据上学习，这就有了一个针对当前任务的完整分类模型；

•如果效果还不够好，可以把整个分类模型再进行微调。

本文以 CNN 文本分类任务为例进行描述，总结一下迁移学习在 NLP 领域文本分类任务中的一些经验。

在文本分类任务中的迁移学习，例如源数据集合为新闻文本的分类（数据量大），目标数据集合为短视频标题分类（标注的数据少），通过预先训练的新闻分类模型，在短视频标题分类任务上进行模型（Embedding 层、卷积层、全连接隐层、输出层）的微调，使得模型既能完成对少量有监督数据的拟合，又具备相应的泛化能力。下边将针对本文在试图迁移预训练 CNN 网络文本分类任务时的准则。

•选择源数据集合时，尽量保证数据量大、语义上与目标数据集合部分相似；

•完全迁移 Embedding 层；

•迁移卷积层和隐层时考虑微调，不使用固定参数。

（2）生成式模型

本文拟通过变分自编码器（VAE）来扩充数据集试图进一步改善本文模型的 F1-Score。VAE 是一类生成式网络结构，通过编码器将输入数据降维到低维的隐含向量编码，再由解码器还原出原数据，在理想情况下，输入数据与输出数据应该极为接近，训练完毕后可通过输入中间编码来生成与训练数据相似的数据集。但在实际应用中，由于很难自行构造合适的中间隐含向量，所以无法生成合适的数据，所以常通过编码器先输出两个低维向量，分别作为正态分布的期望和标准差，采样得到服从正态分布的隐含向量，训练目标除了最小化输入与输出的差异之外，还要使得此正态分布尽可能接近标准正态分布，即期望和标准差分别尽可能接近 0 和 1。如此，训练完毕后，即可通过采样标准正态分布构造隐含向量，从而解码得到与输入数据相似的数据集。

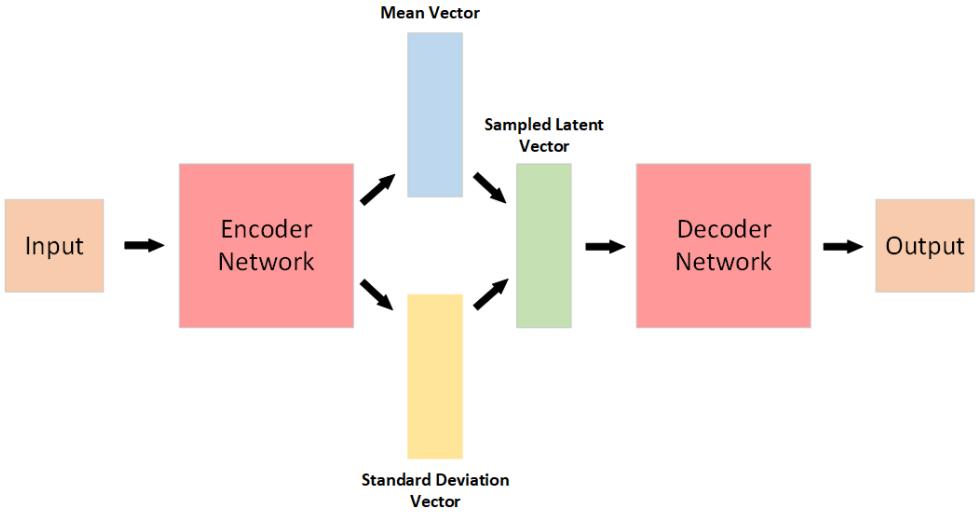


图 8：VAE 网络结构

分别尝试线性编码解码器和 LSTM 编码解码器，发现对语言数据处理效果一直较好的 LSTM 网络反而表现较差，虽然比线性网络要收敛得更快，但损失函数值最终收敛到的水平也明显高出线性网络一大截，反而线性网络的损失函数收敛值极为可观。因此仅用线性网络作为编码解码器即可，期望和标准差向量长度均设为 20，对各类别数据分别训练 900 轮。

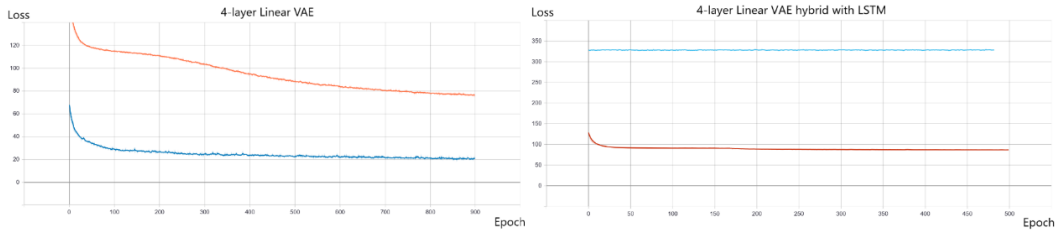


图 9 线性编码解码器训练效果左，LSTM 编码解码器训练效果右，蓝色为“城乡建设”。橙色为“卫生计生”（便于观察仅画出两个类别示例）

期望和标准差的每轮训练后的分布如下，可见最终都较好得分别收敛到了 0 和 1 附近，由此可从随机采样的标准正态分布，分别生成想要数量的新数据向量，并加入到各个分类中。

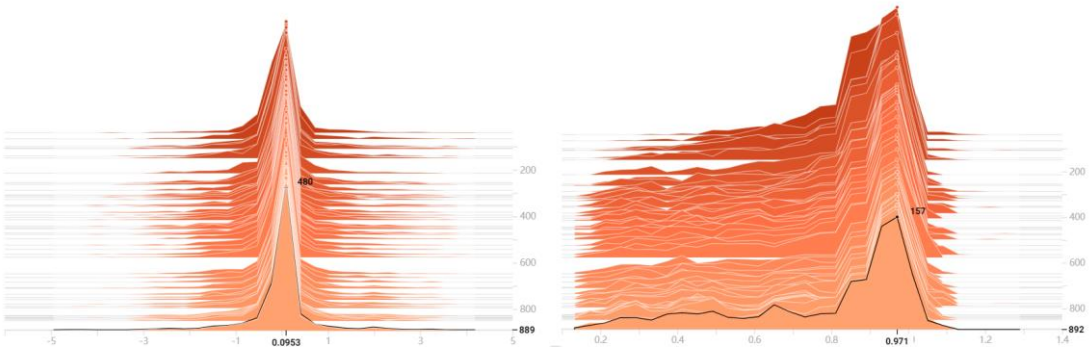


图 10 期望和标准差分布的重叠直方图

本文在新数据集上（数据量为 50000 条）上应用本文的集成模型，最终结果略有提升，至 0.921，效果相对不明显。

4 问题二

4.1 问题分析

某一时段内群众集中反映的某一问题可称为热点问题。目前本文所持有的数据为市民问政的留言详情，因此，为了提取出市民集中反映的相同问题，首先本文需要做的便是将市民留言依据某种相似性度量进行聚类，将相似的留言归为一类，这么一来就可以将市民反映的相似的问题归在同一类中，起到问题提取的作用。进而，为了提取出某一时段的热点问题，本文还需定义合理的热度评价指标，由此完成热点问题挖掘任务。具体的处理流程如下图所示。

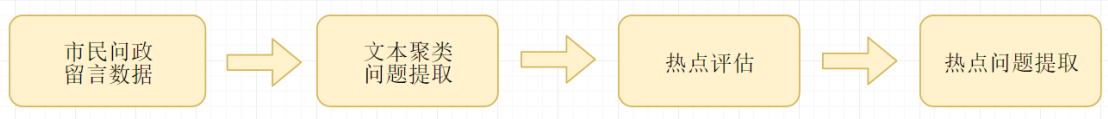


图 11 问题二处理流程

将文本进行向量化后，本文便可以对文本进行数值计算，因为向量化这一过程相当于说将文本统一映射到一个内积空间当中，而在内积空间中，本文可以定义两两之间的距离和角度，所以本文便可以得到文本之间的相似性度量，在此本文选用欧式空间中的距离作为文本之间的相似性度量。即若两个文本之间的距离小，说明两个文本比较相似，反之说明两个文本差别较大。

于是本文便采取机器学习中的 Kmeans 聚类方法。其步骤是，预将数据分为 K 组，则随机选取 K 个对象作为初始的聚类中心，然后计算每个对象与各个种子聚类中心之间的距离，把每个对象分配给距离它最近的聚类中心。聚类中心以及分配给它们的对象就代表一个聚类。每分配一个样本，聚类的聚类中心会根据聚类中现有的对象被重新计算。这个过程将不断重复直到满足没有聚类中心再发生变化。而在本题所提供的数据当中，由于无法事先确定市民反映的“问题”总数，并且大部分市民的问题还是不尽相同了，为了解决这个问题，本文首先将问题按照地区划分，因为在不同地区的问题肯定是不一样的问题，因为问题的构成为时间，地点，事件，若事件发生的地点不同，说明事件不同，因此本文的这个做法是合理的，进而采取数值离散化取最优解的思想，设留言总数为 n，本文在 0.6n 到 0.9n 之间离散取点，依次聚类，选取其中误差最小的聚类数，进行 Kmeans 聚类。最后归为一类的市民留言反映同一个问题。

4.3 热点问题评价方法：

目前本文已经了解了市民网络问政留言的问题情况，政府最关注的问题是市民集中反映的问题，于是本文便需要从众多问题中提取出热度最高的问题。故而本文需要建立一个合理的指标体系去衡量一个问题的热度。

（1）市民关注度

一个问题的严重程度可以从市民的关注程度来衡量，而在网络问政中，市民分为两类人：一类是意见提出者，另一类是意见评论者。在数据集中，前者反映为一个问题的留言数量，留言数越多，即提出该问题的市民数量较多，说明这个问题是较为严重的，反之问题是比较轻微的；后者反映为群众对该意见的点赞数和反对数，点赞数和反对数较多的问题，说明群众积极地参与到该问题的讨论，而点赞数多可以突出该问题的严重性，而参与讨论的人数多的话，说明该问题具有较大的争议性，需要政府出面调解。于是这两类量都可以衡量市民群众对该问题的关注程度，问题 q_i 的意见关注度定义为如下的计算方法：

$$\text{Follow}(q_i)=\frac{\text{该问题留言数}}{\text{留言总数}}$$

4.2 市民问政反映的问题提取

4.2.1 数据预处理

在这里本文采用的数据预处理为分词和去停用词。分词是根据一定的分词规则，把完整的语句切割成由有限个字词组成的词序列，其 是在中文文本中的词汇之间添加边界。中文文本不同于英文文本，英文文本由天然的词与词之间的空格分割，而中文则没有明显分界线，只有字、句、段之间可以用分界符划分。因此本文对中文文本（市民留言）进行分词的操作是必要的。文中采取的是 Python 中 jieba 库所带的分词方法进行中文分词。而停用词在一句话中可能无法提供过多的有效信息，为了节约计算机的内部储存并且提高运算速率，去除停用词也是一个必要的操作。

4.2.2 模型简介[4]

向量空间模型（VSM：Vector Space Model）由 Salton 等人于 20 世纪 70 年代提出，并成功地应用于著名的 SMART 文本检索系统。把对文本内容的处理简化为向量空间中的向量运算，并且它以空间上的相似度表达语义的相似度，直观易懂。

4.2.3 模型建立

相直接地对文本进行数据分析是十分困难的一件任务，因此本文希望将文本转化成一个向量，通过向量化文本的操作使得将文本转化为本文熟知的数据类型，进而采取数据分析的手段进行操作。而 VSM 正是一个这样的模型其基本思想是将文本内容转化为包含特征及其权重的向量表示，即使用向量空间中的向量运算来代替对文本内容的处理，这样文本就表示为：

$$D=\{t_1:w_1,t_2:w_2,\cdots,t_n:w_n\}$$

其中，D 表示文本，w_k 表示特征项 t_k 的权重,n 为向量空间的维数。当 w_k 的值比较小时，就说明其对应的特征项（词）t_k 表示文本 D 的能力较差，反之说明就说明其对应的特征项（词）t_k 表示文本 D 的能力较好。

VSM 的思想便是把文本都转化为相同维数的特征向量上，而要完成此步则需要寻找到同样多的特征（关键词），但是出于维数的考虑，不能将所有词语都视作特征词。因此 VSM 就是把所有文本投影到一个子空间里面（全空间是所有单词构成的）。这个投影需要解释最多的样本的信息，那么每一个空间的维数其实就是一个词语对应的一个数值，进而对文本进行数值表示。

（1）特征词加权（IF-IDF）

具体方法在第一问中已介绍，在此省略。

而市民的留言主题以及内容经过 TFIDF 特征选择以及权重计算后，每一条留言详情都转化为一个空间中的向量。

（2）文本聚类挖掘

问题 q_i 的评论关注度定义为如下的计算方法：

$$\text{Comment}(q_i)=\frac{\text{该问题点赞数+该问题反对数}}{\text{点赞总数+反对总数}}$$

进而综合考量两个指标，定义用户关注度为：

$$\text{U}(q_i)=\lambda*\text{Follow}(q_i)+(1-\lambda)*\text{Comment}(q_i),\lambda\in(0,1)$$

从这个指标中本文可以知道 $\text{U}(q_i)$ 越大的问题说明群众更加关注，也就是说政府应该尽快着手解决或者回应的问题，反之 $\text{U}(q_i)$ 越小的问题说明是个性化比较强的问题，可以稍微缓一缓，先解决最为严重的问题。

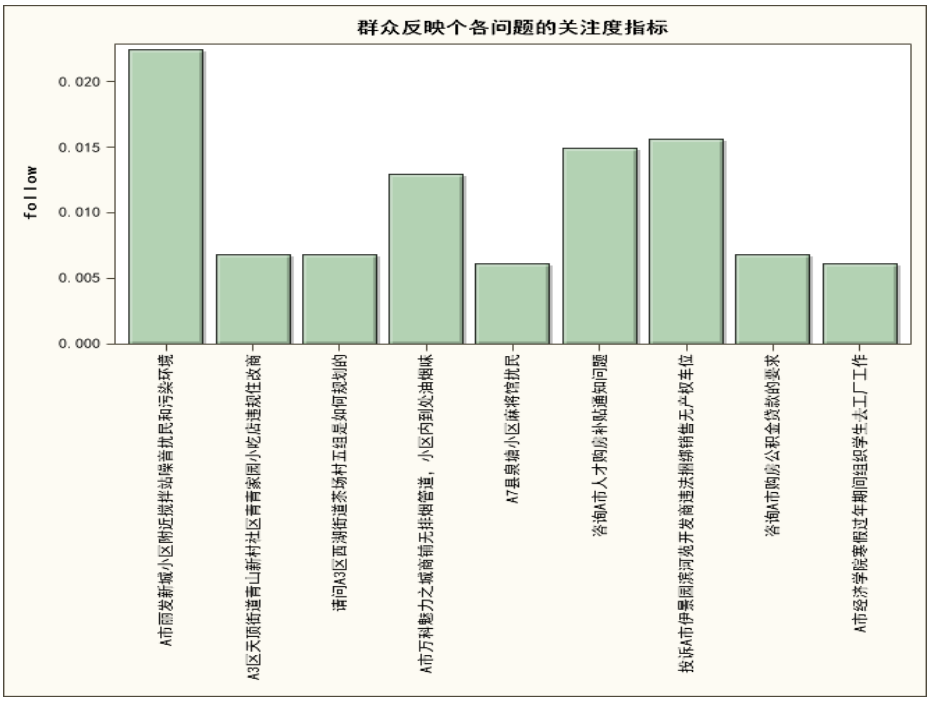


图 12 市民关注度计算案例

仅从市民关注度上看市民反映的问题，得到如上图所示的几个最受关注的市民问题，如 A 市丽发新城的搅拌站噪音扰民问题、A 市魅力新城商铺油烟扰民问题、A 市伊景园捆绑销售车位问题等是市民集中关注的几类问题。

（2）热点突发性

如果一个问题被长期地悬而未决，而受到市民群众的长期关注，那么该问题的曝光性会越来越大，随之其热度会单调递增直到一个稳定的数值，并且由于人类的记忆具有短时性，故热度发生递增的速率应是随着时间递减的逐渐累积或者达到一个趋于平稳的热度值。而当具有较大社会影响力的突发性话题出现时，由于前面所列举的例子的问题具有高

以下就丽发新城搅拌厂问题的留言评论的时效性进行了计算，时效性曲线如下图所示，可以发现的确是随着时间的推移，从前的留言的时效性是有逐渐减弱的趋势的。

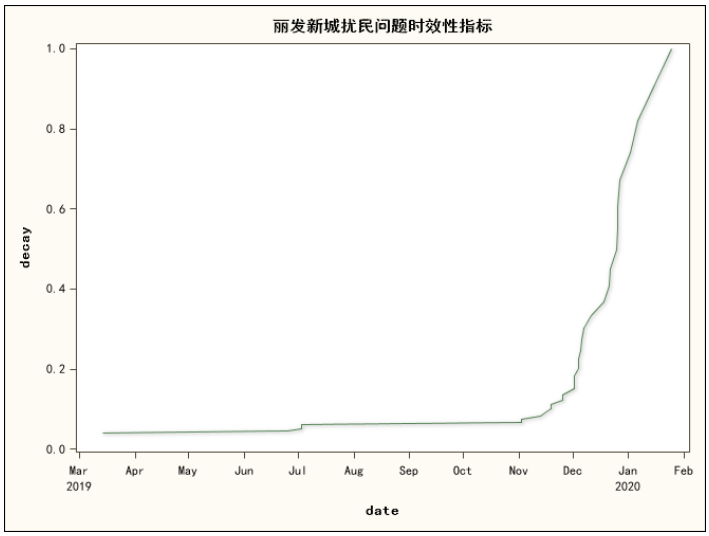


图 13 热点时效性计算案例

（4）热度计算公式

综合市民关注度、事件突发性和时效性三个因素，定义出累积热度值为：

$$R=\sum_{i}(1-e^{-\varepsilon \text{Follow}(q_i)})*\text{Outburst}(q_i)*\text{decay(t-i)}$$

这个公式首先是由以上本文定义的三个指标综合得出的热度值，然后对时间进行一个求和，本文可以发现：如果只是计算一天的热度值，只需要把对时间的求和符号去掉即可，其计算公式$(1-e^{-\varepsilon \text{Follow}(q_i)})*\text{Outburst}(q_i)*\text{decay(t-i)}$。可以看出，如果市民关注度越高，突发性越强，时效性越大，说明该天该问题的热度值越高，反之热度值越低。直观地符合本文对于热度一词的理解。

计算出各问题热度，画出热度前四问题的热度变化趋势如下：

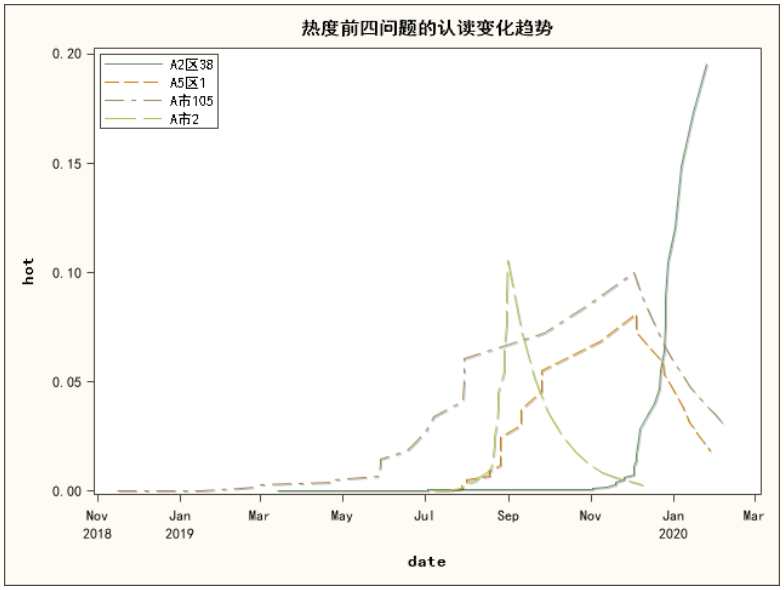


图 14 热度趋势图

4.4 热点问题挖掘

根据上述公式，本文可以计算出每个问题的热度值，最后按照热度值排行，得出市民集中反映的最重要的八个问题分别为：

表 6 热点问题提取表

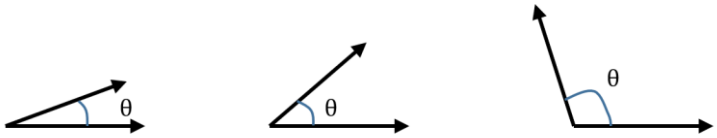
热度排名	地点/人群	问题描述
1	A 市 A2 区丽发新城小区	小区附近违建搅拌厂污染环境噪音扰民
2	A 市伊景园滨河苑	小区捆绑销售无产权车位
3	A 市 A5 区魅力之城小区	小区一楼夜宵摊油烟噪音扰民
4	A 市	人才购房补贴发放申请咨询
5	A 市经济学院	寒假组织学生外出实习
6	A 市 A3 区青青家园小区	社区商店长期违规住改
7	A 市 A3 区西湖街道茶场村	茶场村五组拆迁规划
8	A 市 A7 县泉塘街道	麻将馆夜夜扰民

由热度值排行可以看出，这八个问题是目前政府应该极力关注的热点问题，因为这几个问题的热度值高，代表了这段时间以来市民最关注的社会动向。由此本文便完成了热点问题的文本挖掘任务。

5 问题三

5.1 问题分析

“从人民群众中来，到人民群众中去”，对于政府单位，积极查看群众留言，能够从多角度了解社会的需求以及治理上的漏洞；积极回应群众的留言，能够赢得群众的信任和支持，



因此，对留言转化的向量$X=(x_1,x_2,\ldots,x_n)$与答复转化的向量$Y=(y_1,y_2,\ldots,y_n)$计算余弦值，得到它们之间的相似度，公式如下

$$\cos(\theta)=\frac{\sum_i x_i\times y_i}{\sqrt{\sum_i x_i^2}\times\sqrt{\sum_i y_i^2}}$$

计算群众留言和答复意见的向量的余弦相似度作为答复意见的相似度，相似度越高，答复意见的相关性越高。

5.2.2 完整性

群众留言中含有群众对不同社会现象的质疑、对相关政策的询问、对政府寻求帮助等较为复杂的内容，甚至一条留言包含较多的问题，都需要在答复意见中一一解决。因此，工作人员在对留言进行回复时，需要对群众的每一点疑惑进行详尽的解释，事无巨细得讲解清楚造成问题的原因、问题的发展阶段及解决方案，或一个政策制定的流程、实施的程度。并且，答复意见需对留言中不同人的实际情况给出合理的、全面的、可行的建议，从根本上解决群众的问题。因此，答复意见的内容中需要囊括较多的信息才能为留言者提供帮助，在评价答复内容时，答复完整性也是一个重要的指标。答复的完整性越高，能够提供的有效信息量越多，从而质量也越高。

完整性即答复含有的有效信息量，有效信息量越多，完整性越高。因此，可以通过计算答复内容的信息熵来量化答复意见的信息量，并对有效信息进行筛选。筛选后的信息熵越高，答复意见包含的有效信息量越多，答复完整性越高。

信息熵是结合物理热力学中的热熵得到的概念，用来度量信息混乱程度。一般来说，某个信源发出的符号内容是不确定的，但可以根据符号出现的概率来度量它们。一个符号出现的概率越大，出现机会多，不确定性小；反之不确定性就大。统计信源所有可能出现的情况的平均不确定性，得到信息熵的公式如下。

$$H(X)=-\sum_x p(x)\log_2 p(x)$$

在答复内容中，存在部分字词，它们具有一定信息量，但对于解决留言的问题没有实际的帮助，例如“感谢来信”，“留言已收悉”等。若在计算信息熵时包含这些字词，会导致答复的信息熵偏高，但整体的有效信息较低。因此，将答复内容通过 TF-IDF 算法筛选出文本

热度值，突发性问题在问题发生起初可能并不受到太多人的关注，但是本文如果没有考虑突发性这一因素的话则有可能会因此忽略掉这种问题。为了避免这种情形再现，本文将热点突发性作为影响市民提出问题的热度的指标。根据统计学中 2*2 列联表卡方独立性检验的思想，因为新出现的话题应该和之后才发生的话题相互独立，所以定义如下突发性指标：

$$\text{Outburst}(q_i)=\frac{(a+b+c+d)*(ad-bc)^2}{(a+b)*(a+c)*(c+d)*(b+d)}$$

其中，在时间 i 内，该问题中包含的留言数数用 a 表示，不属于该问题的的留言数用 c 表示。在其他时间段内，该问题涉及到的留言数用 b 表示，其余问题的留言数用 d 表示。

为了使得该指标正则化，本文规定：如果$ad < bc$，则令$\text{Outburst}(q_i)=0$，并且同时对其进行单位化处理，最后，为了使得数乘不改变热度的意义，本文令问题的突发性权重为：

$$\sigma=\exp(\text{Outburst}(q_i))$$

由此本文可以分析该指标，σ 越大，说明在规定某段时间内，该问题具备的突发性越强，越应该受到政府相关部门的关注，而当σ 取最小值 1 时，说明该话题是一直以来都存在的，但不代表不需要解决，该指标的目的是为了增大突发性问题的权重以达到让相关部门趁早重视的效果，但不会改变原本就非常重要的问题的热度。由此可以看出该指标的定义还是较为合理的。

（3）热点时效性

由热度的定义，本文知道在一个特定时间段内同一问题的讨论程度越高，那么该为的热度就应该较高，而同样讨论程度的话题如果分布在很长的一个时间段内，只能说明这个问题持续的被人关注，但不具有很强的社会影响力。就犹如新闻报道通常会随着时间的推移淡出人们的视线。也就是说，一个问题的热度会随着时间的推移不断衰减，过去时间的讨论会逐渐累积到当今问题的热度，但是过去的时间越久，理应这种影响就越低，并且这个衰减速率应对时间呈一个单调递减的凸函数。并且越火爆的问题衰减的速度应该越慢，也就是说，人们越发关注的问题，这种衰减速率就越慢，于是本文利用指数函数的性质首先定义出如下的衰减函数：

$$\text{decay}(q_i)=\exp(-(1-\text{Comment}(q_i))*t)$$

提升政府公信力和执行力。在网络平台上对群众留言进行回复时，不同质量的答复意见有截然相反的效果。质量高的答复意见能够准确得为群众提供他们需要的有效信息，迅速答疑解惑，从而让群众感到满意。而质量低的答复意见不仅无法解决问题，还可能抹黑相关部门在群众心中的形象，导致更严重的后果。为了提高答复意见的质量，应当采取合理的评价方式对答复意见进行等级划分，推广质量高的答复模式，警惕质量低的答复模式。

考虑到答复内容、语言风格以及答复时间，通过量化答复意见的相关性、完整性、专业性和及时性建立答复评价模型，并对 2816 条答复意见进行五等级划分。

5.2 答复评价模型的建立

5.2.1 相关性

群众留言的原因是他们对各种生活中的不同社会现象存在质疑，而不同人对同一社会现象也会有不同角度的看法与见解。因此，为了解决群众留言中的问题，答复意见的内容应该与留言的话题高度相关，并针对留言中的特定问题结合实际情况进行说明，从而避免答非所问的情况。于是，对答复意见的质量进行评价时，首先需要考虑答复意见与群众留言的相关性。

答复相关性衡量答复内容与对应群众留言是否存在关联，相关性越高，答复的内容越能帮助群众解惑，答复意见的质量越高。本文采取 TF-IDF 算法计算每个字词在整体语句中的权重，从而提取答复意见中的特征词，通过对比群众留言与对应答复意见的 TF-IDF 值，再计算余弦相似度得到答复意见的相似度，从而判断相关性。

TF-IDF 算法已在 4.2.3 中进行了详尽的说明，本处不再重复。采用 TF-IDF 算法可以将群众留言与对应答复意见转化为两个维度相等的向量，向量每个位置上的值为该位置对应的字词的权重。

余弦相似度是由两个向量的夹角的余弦值计算得到的判断相关性的指标。在三角函数中，任意一个角的余弦值取-1 到 1，角度为 0 时余弦值为 1；角度越大，余弦值越小。并且，向量夹角的余弦值不受向量长度影响，仅与向量的方向有关。因此，两个向量的夹角的余弦值可以判断两个向量是否指向大致相同的方向，即余弦值越大，两向量指向的方向越接近，可以理解为两个向量越相似。

中的特征词后，再计算信息熵，得到答复意见的有效信息量。

在计算信息熵时，X为答复经过分词后的答复文本，x是该答复内容包含的所有字词，$p(x)$是x发生的概率，取为该字词在答复文本中的词频。答复文本的不确定性越大，信息熵也就越大，理解答复中的内容所需要的信息量也就越大，即答复完整性越高。

5.2.3 专业性

由于答复群众留言时，笔者是作为官方代表进行回复，答复意见除了内容上需要相关、完整，整体语言风格上也需要体现政府态度的专业性。答复风格应该是客观、热情的，文本而不能过于口语化，也绝不能是态度恶劣、随意得对群众回复。一个优质的答复意见应该考虑群众写下留言时的心态是焦虑、担忧的，因此在进行答复时，更应该耐心得讲解问题缘由，体现全心全意为人民服务的工作作风。因此，本文对答复意见进行情感分析，由此判断答复的情感倾向。

文本情感分类的传统思路存在着两个难以克服的局限性：一、精度问题，传统思路差强人意。当然，一般的应用已经足够了，但是要进一步提高精度，却缺乏比较好的方法；二、背景知识问题，传统思路需要事先提取好情感词典，而这一步骤，往往需要人工操作才能保证准确率。经过本文的测试，基于深度神经网络的情感分析模型，其准确率往往有 95%以上。在自然语言处理中，通常用到的方法是卷积神经网络或循环神经网络（RNNs）。它们的作用跟卷积神经网络是一样的，将矩阵形式的输入编码为较低维度的一维向量，而保留大多数有用信息。跟卷积神经网络的区别在于，卷积神经网络更注重全局的模糊感知，好比本文看一幅照片，事实上并没有看清楚某个像素，而只是整体地把握图片内容，而 RNNs 则是注重邻近位置的重构。并且，语言总是由相邻的字构成词，相邻的词构成短语，相邻的短语构成句子。因此，需要有效地把邻近位置的信息进行有效的整合重构。由此可见，对于语言任务，RNNs 更具有说服力，所以本文采用深度学习中的 LSTM 网络而非 CNN 来应用在该处。

下面本文搭建 LSTM 模型。尽管模型的结构较为简单，但是由于经测试后结果已经足够好，所以采纳。

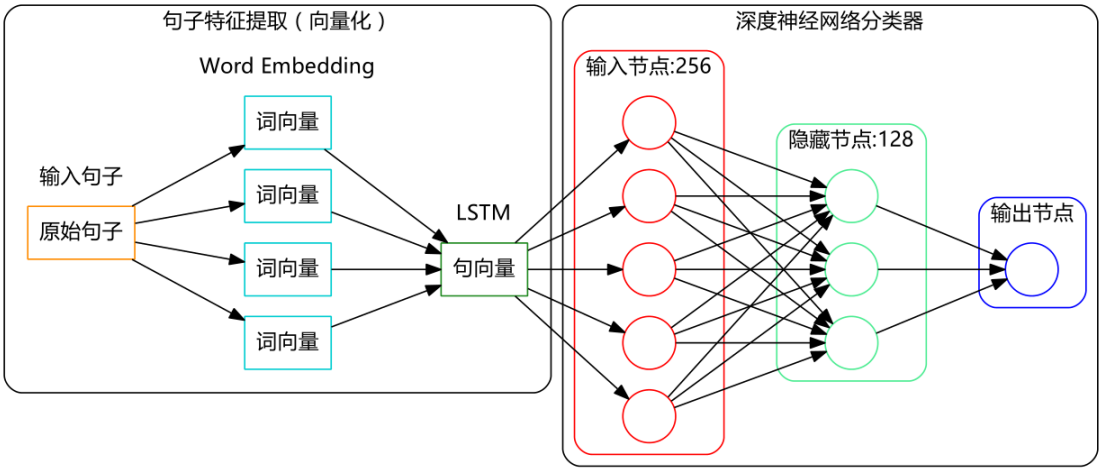


图 15 LSTM 网络示意图

随后本文进行标注语料的收集。要注意本文的模型是监督训练的，所以需要收集一些已经分好类的句子。本文收集了两万多条中文标注语料（涉及六个领域）用来训练模型。经过十次迭代最终在训练集上达到 99%的准确率，在验证集上达到了 95%的准确率。

```
Epoch 1/10
10553/10553 [==============================] - 293s 28ms/step - loss: 0.4249 - accuracy: 0.8060
Epoch 2/10
10553/10553 [==============================] - 281s 27ms/step - loss: 0.1614 - accuracy: 0.9423
Epoch 3/10
10553/10553 [==============================] - 280s 27ms/step - loss: 0.0649 - accuracy: 0.9782
Epoch 4/10
10553/10553 [==============================] - 254s 24ms/step - loss: 0.0385 - accuracy: 0.9873
Epoch 5/10
10553/10553 [==============================] - 255s 24ms/step - loss: 0.0259 - accuracy: 0.9919
Epoch 6/10
10553/10553 [==============================] - 262s 25ms/step - loss: 0.0256 - accuracy: 0.9915
Epoch 7/10
10553/10553 [==============================] - 249s 24ms/step - loss: 0.0206 - accuracy: 0.9928
Epoch 8/10
10553/10553 [==============================] - 250s 24ms/step - loss: 0.0242 - accuracy: 0.9940
Epoch 9/10
10553/10553 [==============================] - 258s 24ms/step - loss: 0.0110 - accuracy: 0.9971
Epoch 10/10
10553/10553 [==============================] - 255s 24ms/step - loss: 0.0043 - accuracy: 0.9991
```

为了便于本文进行量化分析，本文将最后一层的二分类函数修改为概率函数，从而可以将最终结果映射到 0-1 之间的概率分布，记为答复意见的情感值，越接近 1 则越积极，越接近 0 则越消极。

由情感分析的结果，本文额外建议在回复群众留言时能够将答复模板化，开头结尾按照情感分析得分较高的案例进行推广，例如答复开头时统一为“市民同志，您好！您的留言已收悉。现将有关情况回复如下”，答复结束时加入“感谢您对本文工作的支持、理解与监督！”或“感谢您的留言，祝您事事顺心。”，从而让市民们阅读答复时更加满意。

5.2.4 及时性

在衡量一个答复意见的质量高低时，除了判断答复内容的好坏之外，还需要考虑答复时

割点的小数位数为 1 或 2 位，便于实际输入。其中，等级为 1 到 5，等级越高，对应的指标越好。例如，对各答复意见计算信息熵并聚类后，完整性等级为 5 的分割点为 6.39231742277876，等级为 5 的留言有 205 条，若做微小调整使得分割点变为 6.5，等级为 5 的留言仅增加 11 条，而工作人员登记时更为便捷。对于及时性，由于极端值较多，聚类效果较差，于是采用常用的天数的五等级划分。通过三天、一周、两周、一个月四个节点对及时性进行评级。相关性、完整性、专业性及及时性等级划分标准如下。

表 相关性等级划分

相似度范围	相关性等级	答复个数
0.25 及以上	5	343
0.15~0.25	4	685
0.05~0.15	3	1201
0.01~0.05	2	387
0~0.01	1	200

表 完整性等级划分

信息熵范围	完整性等级	答复个数
8.2 及以上	5	216
7.2~8.2	4	774
6.5~7.2	3	928
5.75~6.5	2	593
0~5.75	1	305

表 专业性等级划分表

情感值范围	专业性等级	答复个数
0.85 及以上	5	278
0.75~0.85	4	753
0.65~0.75	3	1187
0.4~0.65	2	363
0~0.04	1	235

表 及时性等级划分

拖延工作日	及时性等级	答复个数
0~3	5	730
3~7	4	616
7~14	3	673
14~30	2	537
30 及以上	1	260

经过等级划分后，示例结果如下。

表 答复等价结果

答复编号	留言主题	相关性	完整性	专业性	及时性
6448	关于 A 市实施差别化购房措施的咨询	5	1	2	5
9128	关于 A 市和馨园租户相关回复的质疑	5	4	5	2
17290	关于 A7 县农村老房翻新事宜的咨询	2	2	5	5
104224	请求打除 K7 县桃川镇建安亭村多年来横行鱼肉乡邻百姓的村霸	4	5	5	1
107228	第二次请求在 K1 区萍州小学正大门设置人行横道线（斑马线）	1	3	3	2
107940	投诉 K 市职业技术学院中秋节不放假	2	1	2	1
124055	投诉 L5 县社保局违规收取滞纳金	3	2	5	5

距离留言日期的工作日总天数，即工作人员是否能耗费较少时间及时回复群众留言。就算答复内容质量非常高，但答复时间拖延过久，无法在群众最需要的时刻提供帮助，依旧不能为群众带来较好的留言体验，也是工作人员的失职。因此，在评价答复意见的质量时，需要考虑答复的及时性。

计算答复时间与留言时间之间间隔的天数，考虑工作人员周末休息，仅考虑二者之间间隔的工作日，称为拖延工作日。

5.2.5 整体结果

建立答复评价模型后，对所有附件 4 中所有留言进行五个指标的计算，部分结果展示如下。

表 答复评价结果

答复编号	留言主题	相关性	完整性	专业性	及时性
6448	关于A市实施差别化购房措施的咨询	0.628385513	5.169925	0.81284391	1
9128	关于A市和馨园租户相关回复的质疑	0.804519365	8.03891899	0.89347432	17
17290	关于 A7 县农村老房翻新事宜的咨询	0.059370736	6.507795	0.85348653	2
104224	请求打除 K7 县桃川镇建安亭村多年来横行鱼肉乡邻百姓的村霸	0.185090634	9.942514505	0.91493656	45
107228	第二次请求在 K1 区萍州小学正大门设置人行横道线（斑马线）	0.005733115	7.129283	0.67395682	15
107940	投诉K市职业技术学院中秋节不放假	0.029032707	4.95419631	0.47396346	42
124055	投诉 L5 县社保局违规收取滞纳金	0.16843513	6.409390936	0.87435965	3

5.3 等级划分及结果

通过以上五个方面对答复意见的质量进行衡量，本文初步得到了对答复意见的

优劣进行量化的方法。由于每个指标的数值范围差异较大，且实际工作中对答复评价时常用五个等级划分，本文采用 K-Means 聚类的方法对相关性、完整性、可解释性和专业性的数值进行五个类别的划分。

由于在实际应用中，对各等级划分的节点不需要精度太高，若小数点位数过多反而登记时较为麻烦。因此，在聚类后，对原有分类进行一定调整，使得各指标的一维数据的四个分

6 展望

6.1 模型的优点

1）在识别群众留言标签时，本文多角度，系统全面的，由简单至复杂建立起文本分类模型，模型拥有很高的准确率，对每一类文本都拥有普遍较高的辨识能力。

2）在热点问题的挖掘问题中，本文提出的自适应聚类方式不需要事先给定聚类数目，符合人们的直观感觉，同时热点评估的指标考虑全面，计算准确。

3）在相关部门答复意见的质量评价中，本文多角度衡量答复意见的质量，涵盖文本内容、行文风格以及答复时间，结果较为全面可靠。

6.2 模型的缺点

1）在识别群众留言标签时，最终的集成模型过于复杂，准确率提升有限，使用较为基本的基学习器已经能够取得不错的效果。如果综合考量时间复杂度和效率，可以适当简化。

2）在热点问题的挖掘问题中，由于自适应聚类方法需要通过搜索的方法找到最适合的聚类数目，因而当数据集扩大时，其聚类时间较长。

3）在相关部门答复意见的质量评价中，对答复意见进行评价时，不专业性除了词汇情感倾向，也体现在语序和语法上，但本文的情感分析模型无法识别。同时由于缺少留言中出现的问题的实际信息，无法判断答复真实性。

参考文献(References)

[1] BrikerMan, Kashgari, https://kashgari-zh.bmio.net/

[2] 清华大学数据科学研究院, 手把手教你在 Python 中实现文本分类, https://zhuanlan.zhihu.com/p/37157010

[3] NLP - 15 分钟搭建中文文本分类模型, https://eliyar.biz/nlp_chinese_text_classification_in_15mins/

[4] 王琦. 面向网络新闻的热点话题发现与极性分析[D].大连海事大学,2017.