2020 年“泰迪杯”数据分析职业技能大赛 C 题“智慧政务”中的文本挖掘应用

摘要

在政府留言问题中，传统的人工分类存在效率低、错误率工作量大等缺点，

本文主要研究利用自然语言处理和文本挖掘的方法解决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的相关问题。整个过程涉及到文本读取、数据预处理、数据清洗、特征词库建立、特征加样本测试等一系列过程。

我们利用 TD-IDF 算法进行特征权重加权算法，充分挖掘政府留言的热点问题。使用了一些譬如 jieba 分词，特征向量的计算。本题基于抽出的空间向量数据来拟合 CNN 模型，优化维度明显。

同时我们挑选了 4800 个留言进行了有监督学习法，利用卷积神经网络进行双向学习，改进了传统单向信息学习的不足之处，提高了政府留言分类的正确率，算法极大优化。

为简化算法，提高运行效率，我们通过卡方校验算法判定特征词距离，在大量样本测试后合理地提出距离阈值，将某些特征词汇积累回归，顺利达到了特征降维的目的。因此这种算法在满足分类正确率的情况下极大的简化了算法，更有利于政府政务的高效处理。

将政府留言特征词汇向量化后，对特征向量进行样本训练得到 CNN 模型，，代入到此模型中，从而对群众留言进行提前分类。在解决问题的过程中，我们使用停用词实现了一个分词程序，对附件二的已知分类进行了裁剪，挑选了 4800 个群众留言进行了有监督学习，用 F-Score 评价方法对结果进行分析，最后实现的分类算法的正确率在 84%左右。

关键词:机器学习 文本分类 朴素贝叶斯分类法 TF-IDF 算法 Python

目录

一、问题的提出和重述

本文是在大数据、云计算、人工智能等技术发展的大背景下，研究建立基于自然语言处理技术的智慧政务系统。利用自然语言处理和文本挖掘的方法解决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的相关问题。

（1）群众留言分类

为解决人工根据经验处理工作量大、效率低、差错率高等问题。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。

（2）热点问题挖掘

热点问题即为在某一时段内群众集中反映的某一问题。政府为提升服务效率，有针对性地对问题进行处理，需要根据附件 3 将某一时段内反映特定地点或特定时间人群问题的留言进行归类，并定义合理的热度评价指标，给出评价结果。我们需要给出排名前 5 的热点问题，并给出相应热点问题对应的留言信息。

（3）答复意见的评价

针对附件 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。

一、问题的分析

2.1 问题一分析

在这个问题中，首先从附件二中读取留言主题、留言详情以及一级分类等有效文本，进行了群众留言预处理，使用了中文文本 python 库的 jieba 分词来将实现文本切割。洗去了不良数据，对下一步的群众文本分类。计算出留言重点词频时应用 TF-IDF 算法，定义并去除停用词，最后利用贝叶斯分类器进行种类预测。

2.2 问题二分析

问题一解决的是针对不同内容的留言对留言进行归类聚类，从而分数具体部门。但在实际问题中，要解决群众相关的问题，就必须进而考虑时间与空间两个要素。

在某一时段、某一地点居民集中反映的问题，包括时间、地点与频率三个要素同时考虑，词频越高，点赞数越多，地点越集中，群众对事件的关注度越高，进而给出热点问题的热力排名，筛选前 5 名。

同时考虑时间，如果一次留言后在合理的时间内未得到答复，那这被搁置的问题同样属于热点问题。

2.3 问题三分析

问题三是基于问题一、二的后续问题。是对政府相关部门对热点问题答复意见的评价。分别从答复的相关性、完整性、可解释性等角度进行评价。

相关性可对留言及答复进行关键词频统计，参考相关语义算法评价相关性；

完整性可从留言与答复映射关系的角度评价；

可解释性是从答复的有效性来考虑，答复内容有具体步骤、具体时间节点、确定性词汇等可解释性好。

二、符号及变量说明

符号	表示含义
TP	预测为有，实际也有
FP	预测为有，实际无
TN	预测无，实际也无
FN	预测无，实际有
R	查全率
P	查准率

三、模型与算法

4. 1 基于朴素贝叶斯算法的贝叶斯分类器

文本分类是处理自然语言的典型方法，主要包括文本预处理[1]、文本特征的提取及分类器的设计等过程。朴素贝叶斯法是基于贝叶斯定理与特征条件独立假设的分类方法。最为广泛的两种分类模型是决策树模型(Decision Tree Model，DTM)和朴素贝叶斯模型(Naive Bayesian Model，NBM)。

这给 NBC 模型的正确分类带来了一定影响。

该算法包括以下几个步骤：

1.收集数据；

2.准备数据（数值型或 bool 型数据。若是文本文件，要解析成词条向量）；

3.分析数据（统计数据特征众多时，直方图分析效果更好）；

4.算法训练（通过样本训练得出各独立特征的条件概率）；

5.测试算法（计算错误率）；

6.使用算法（文本分类）。

导出贝叶斯：

B 发生事件 A 发生的概率为：

$$P（A|B）=\frac{P(A\cap B)}{P(B)}$$

A 发生条件下 B 发生概率：

$$P（B|A）=\frac{P(A\cap B)}{P(A)}$$

结合上述两式，可以得到概率乘法规则：

$$P(A|B)P(B)=P(A\cap B)=P(B|A)P(A)$$

当 P（A）不为零时，上式两边同时除以 P（A）得到贝叶斯定理：

$$P（B|A）=\frac{P(A|B)P（B）}{P(A)}$$

4.2 TF-IDF 算法的特征加权（特征权重）

它由 TF 和 IDF 两部分组成：

某个词在所有留言中出现的概率称作 TF。其计算公式为：

$$TF_{ij}=\frac{n_{ij}}{\sum_k n_{kj}}$$

其中，nij 为特征词在政府留言中出现的次数，∑knkj 是政府留言中所有特征词的个数。计算的结果即为某个特征词的词频。

逆向文本频率（IDF），整个语料库中的特征词频，目的是剔除掉留言预处理后产生的一些经常遇到但却不重要的词语同时也能够保留一些特征明显的词语。其计算公式为：

$$IDF_{ij}=\log\frac{N}{1+N_{mi}}$$

N 表示语料库中全部文章的数量，$1+N_{mi}$ 表示文章中具有明显特征的词语 mi 的数量。

其计算公式如下：

$$TF-IDF_{ij}=TF_{ij}*IDF_{ij}$$

TF-IDF 关系到 TFij 和 IDFij。TF-IDF 越大，说明留言特征热度愈高。从而排序留言问题中的热点事件。

4.3 卷积神经网络

卷积神经网络主要由提取特征的卷积层、压缩结果、保留重要特征的池化层以及将文本数据映射到样本标记空间的全连接层组成。应用于文本的卷积神经网络，卷积层输入常是嵌入向量，因此通常会有嵌入矩阵层，将词语索引转换为嵌入向量。本体的模型如下。

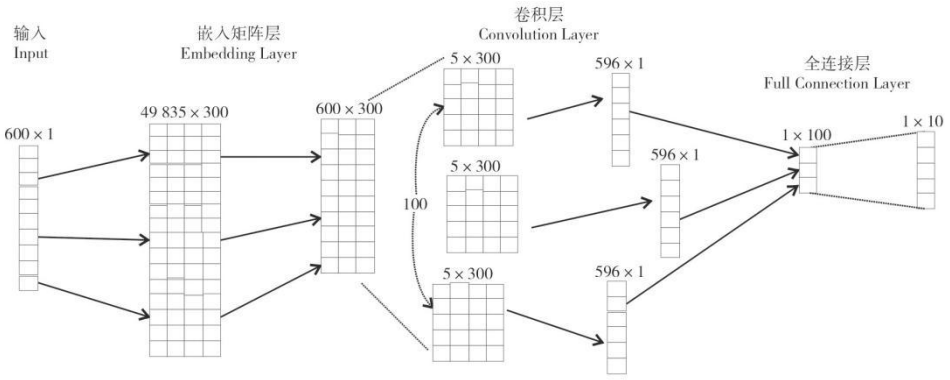


图 1 模型结构图

4.3.1 嵌入矩阵层

根据嵌入向量是词语或字符级别，将嵌入矩阵分为词嵌入矩阵或字符嵌入矩阵。由于本文工作是基于词语级别的输入，因此后文中嵌入矩阵特指词嵌入矩阵。

模型的输入一般为词语在嵌入矩阵中的行索引，如“1,3 021,234,11”，在嵌入矩阵层，根据行索引在嵌入矩阵中找到对应的嵌入向量，再堆叠形成卷积层的输入。

4.3.2 卷积层

应用于留言内容的卷积神经网络的卷积核(文本卷积核)大小通常表示为2,3,5 等，假设嵌入向量维度为 dim，那么其实际含义等价于图像卷积核中2×dim,3×dim,5×dim。后文中卷积核特指文本卷积核。

一个卷积核一次卷积为:

$$c_i = f(X_{i,i+h-1} \cdot W_h + b_i)$$

加上偏移参数 b_i 后经过激活函数 f 转换，得到一次卷积结果 c_i，假设卷积层输入矩阵包含 n 个行向量，进行多次卷积操作，便可以得到:

$$C_j = [c_1, c_2, c_3, \ldots, c_{n-h+1}]^T$$

卷积层输出为:

$$M = [c_1, c_2, c_3, \ldots, c_m]$$

4.4.3 全连接层

将文本数据映射到留言特征库后，在此层，将卷积、池化得到的局部特征通过权值矩阵进行整合，并映射到样本标记空间。全连接层为重要的分类器。

四、问题求解结果

读取数据后进行数据预处理。政府群众留言的提前处理的方法的优劣。现在定义包括下面的步骤：

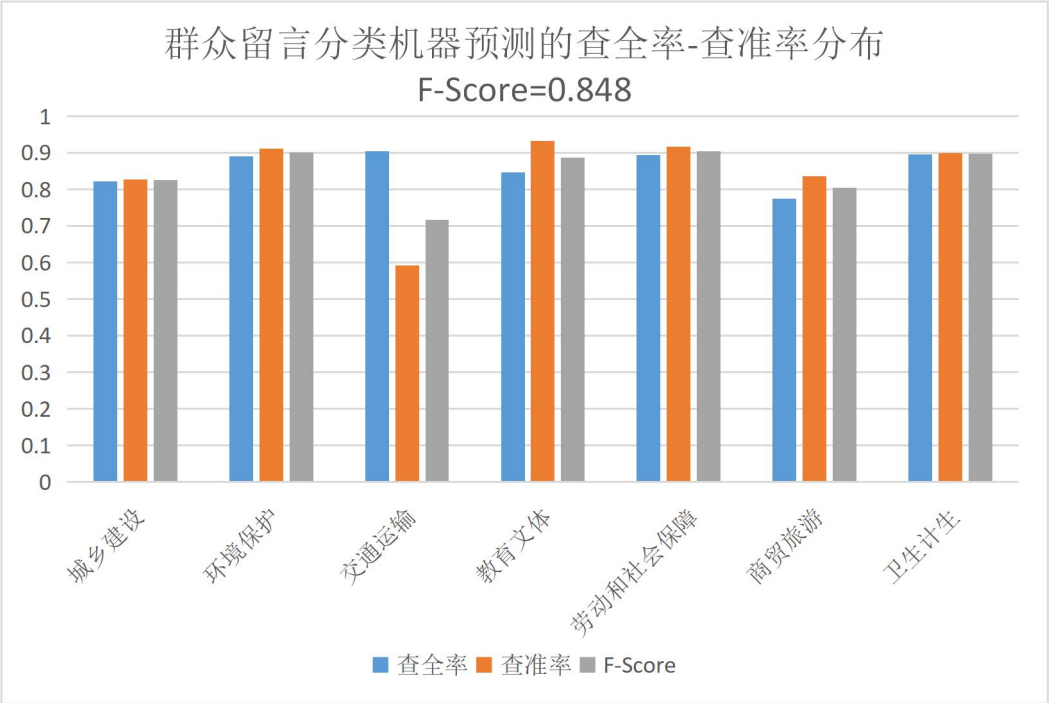
我们为每个类选出最具代表性的十个词语发现程序提取的特征词还是很具有类别区分度的，但也有部分分类较为相似，可能在分类上会比较难区分。于是我们又从以下三个方面进行了如下改进：

（1）特征词维度的选择：只定义一百左右的特征的维度；

（2）特征权重的计算方式：使用 TF-IDF 作为权重；

分类	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
查全率	0.822299652	0.891257996	0.903752039	0.847073631	0.894362621	0.774485597	0.895096921
查准率	0.828070175	0.911668484	0.593147752	0.932132964	0.916710047	0.835701599	0.899198167
F-Score	0.825174825	0.901347709	0.716224952	0.887570063	0.905398458	0.803929945	0.897142857

可以看出交通运输一级分类的查全率最高，但是查准率最低，可以看出，其他分类容易被误认为该分类，该分类与其他分类有逻辑交叉的地方，经过不断的优化，仍然存在低达 0.6 的查准率，可以看出其他大类的未识别的大部分就属于交通运输部分。



该模型对群众留言分类及其预测打到了一个很好的查全率和查准率，这种分布情况让我们发现最对于除了交通运输以外的其他分类都达到了较好的分类影响，所以说我们可以说我们的我们的分类器以及机器学习模型达到一个比较好的状况。该分类使用了，基于四级神经网络的逻辑分析，从而在在基础分类上打到了一个很好的参数依据，使我们的模型由于网络上大多数的其他普通模型，我们针对了群众留言分类及政府信息的特征，进行了一些细节上的优化，比如说提高的主题的全中，然后增加了主题和和评论留言内容的一种比例，通过提高这个比例以及合理的运算出这个比例的全职大小，我们得到一个最佳的比例，这个比例是 5:1。通过 5.18 比 1 的实现，我们发现通过改进这种比例，在 F-Score 得分方面有了一个 0.26 的分数提高。提高如下。

经过基于主题词和内容语句的权重比例提高，以及优化调整之后，我们发现在于城乡建设环境保护，教育文本及劳动和社会保障商贸有和卫生计生方面得到了明显的提高，而在教育文体方面，起到一个持平的作用，所以说我们可以发现在教育文体方面，它的题目对于文本的一个明显的中心词的作用是不明显的。

综上我们对问题一的实现得到比较好的一个情况，F-score 达到了 0.848，我们已经很好的完善了问题一的要求及赛方的标准。

五、评价与改进

6. 1 方案的优点

1、利用 TF-IDF 算法加权赋值，通过贝叶斯分类器分类，提高了文本的分类效率，从而表现出了该模型的优势。无论是在准确率还是运行效率上都优于单一传统的神经网络模型，该模型具有更强的特征提取能力；

2、该模型优化了算法，提高了识别效率，有利于提高政府服务效率。

6. 2 方案的缺点

1、使用完整词表构建的嵌入矩阵，训练参数多，占用内存多，训练时容易

出现内存不足。

2、随着留言问题的堆积，TF-IDF 留言中词频骤增，热度出现大幅降低。

3、压缩嵌入矩阵后，模型达到最优的参数是否发生变化尚待验证。模型训练时间影响因素中除了嵌入矩阵的训练参数外，全连接层占据了主要部分。

6.3 改进方案

由有时间有限，我们不能做到尽善尽美。但我们思考是不是可以寻找一种替代全连接的方式，更加优化模型，改进缺点。

六、参考文献

[1] 梁志剑.谢红宇、基于 BiGRU 和贝叶斯分类器的文本分类.中北大学大数据学院（222006），2018 年第 12 期

[2] 张波.黄晓芳、基于 TF-IDF 的卷积神经网络新闻文本分类优化.西南科技大学计算机科学与技术学院（223545），2019 年第 09 期

[3] 姚佳奇.徐正国.燕继坤、基于标签语义相似的动态多标签文本分类算法.盲信号处理重点实验室（294325），2013 年第 03 期

七、附录

一：

```
#coding=utf8
import jieba

def wwstp():
    wwstp_file = open('wwstp_ch.txt', u'r')
    stpww = []
    for line in wwstp_file.readlines():
        stpww.append(line[:-1])
    return stpww

def jiaba(raw, stpww):
    wrd_lL = list(jieba.cut(raw, cut_all=False))
    for wrd in wrd_lL:
        if wrd in stpww:
            wrd_lL.remove(wrd)
    # wwsdet 用于统计 A[nClass]
    """wrd_lL.remove('\n')"""
    wwsdet = set(wrd_lL)
    return wrd_lL, wwsdet


stpww = wwstp()
```

```
ress = []
qq = 0
with open('dat/testing.csv', u'r',encoding='GBK') as f:
    for line in f:
        content = ""
        for aa in line.split(',')[1:]:
            content += aa
        wrd_lL, wwsdet = jiaba(content, stpww)
        label = 2
        couu = 0
        for i, cla in enumerate(feature_wrds):
            temp = 0
            for wrd in wrd_lL:
                if wrd in cla:
                    temp += 1
            if temp > couu:
                couu = temp
                label = i+1
        if couu == 0:
            qq += 1
        ress.append(label)
print(qq)
with open('output/feature_match_output.csv', u'w') as g:
    for i, res in enumerate(ress):
        g.write(str(i+1))
        g.write(',')
        g.write(str(res))
        g.write('\n')
```

二：

```
#coding=utf8
import jieba

def wwstp():
    wwstp_file = open('wwstp_ch.txt', u'r')
    stpww = []
    for line in wwstp_file.readlines():
        stpww.append(line[:-1])
    return stpww

def jiaba(raw, stpww):
    # 使用*
    wrd_lL = list(jieba.cut(raw, cut_all=False))
```

```
    for wrd in wrd_lL:
        if wrd in stpww:
            wrd_lL.remove(wrd)
    # wwsdet 用于统计 A[nClass]
    """wrd_lL.remove('\n')"""
    wwsdet = set(wrd_lL)
    return wrd_lL, wwsdet


stpww = wwstp()

ress = []
qq = 0
with open('dat/testing.csv', u'r',encoding='GBK') as f:
    for line in f:
        content = ""
        for aa in line.split(',')[1:]:
            content += aa
        wrd_lL, wwsdet = jiaba(content, stpww)
        label = 2
        couu = 0
        for i, cla in enumerate(feature_wrds):
            temp = 0
            for wrd in wrd_lL:
                if wrd in cla:
                    temp += 1
            if temp > couu:
                couu = temp
                label = i+1
        if couu == 0:
            qq += 1
        ress.append(label)
print(qq)
with open('output/feature_match_output.csv', u'w') as g:
    for i, res in enumerate(ress):
        g.write(str(i+1))
        g.write(',')
        g.write(str(res))
        g.write('\n')
```

三

```
# -*- coding: utf-8 -*-
import jieba
import nltk
from math import log
```

```
import numpy as np
import json

N = 9000

# 读取停词表
def is_Chinese(wrd):
    for ch in wrd:
        if '\u4e00' <= ch <= '\u9fff':
            return True
    return False
def wwstp():
    wwstp_file = open('wwstp_ch.txt', 'r',encoding='gbk')
    stpww = []
    for line in wwstp_file.readlines():
        stpww.append(line[:-1])
    return stpww

def jiaba(raw, stpww):
    # 使用*
    wrd_lL = list(jieba.cut(raw, cut_all=False))
    for wrd in wrd_lL:
        if wrd in stpww:
            wrd_lL.remove(wrd)
    # wwsdet 用于统计 A[nClass]
    wrd_lL.remove('\n')
    wwsdet = set(wrd_lL)
    return wrd_lL, wwsdet

def peocxe(train_p, test_p):

    stpww = wwstp()
    # 用于*
    A = {}
    tf = []
    i=0
    bb=0
    # 存**
    couu = [0]*11
    trainss = []
    test_set = []
    with open(train_p, 'r',encoding='GBK') as f:
        for line in f:
            tf.append({})
```

```
            if len(line.split(',')[0])>2 or
is_Chinese(line.split(',')[0][0]) or line.split(',')[0]=='\n' or
line.split(',')[0]=='"\n':
                continue
            label = int(line.split(',')[0])-1

            if label not in A:
                A[label] = {}
            couu[label] += 1
            content = ""
            for aa in line.split(',')[1:]:
                content += aa
            wrd_lL, wwsdet = jiaba(content, stpww)
            trainss.append((wrd_lL, label))
            for wrd in wwsdet:
                if wrd in A[label]:
                    A[label][wrd] += 1
                else:
                    A[label][wrd] = 1
            for wrd in wrd_lL:
                if wrd in tf[i]:
                    tf[i][wrd] += 1
                else:
                    tf[i][wrd] = 1
            i += 1
        print("*据")

    tf2 = []
    j = 0
    with open(test_p, 'r',encoding='gbk') as g:
        for line in g:
            tf2.append({})
            if len(line.split(',')[0])>2 or
is_Chinese(line.split(',')[0][0]) or line.split(',')[0]=='\n' or
line.split(',')[0]=='"\n':
                continue
            label = int(line.split(',')[0])-1
            content = ""
            for aa in line.split(',')[1:]:
                content += aa
            wrd_lL, wwsdet = jiaba(content, stpww)
            test_set.append((wrd_lL, label))
            for wrd in wrd_lL:
                if wrd in tf2[j]:
```

```
                    tf2[j][wrd] += 1
                else:
                    tf2[j][wrd] = 1
            j += 1
    return A, tf, tf2, trainss, test_set, couu


def calculate_B_from_A(A):

    B = {}
    for key in A:
        B[key] = {}
        for wrd in A[key]:
            B[key][wrd] = 0
            for kk in A:
                if kk != key and (wrd in A[kk]):
                    B[key][wrd] += A[kk][wrd]
    return B

def feature_select_use_new_CHI(A, B, couu):

    wrd_dict = []
    wrdff = []
    for i in range(0, 7):
        CHI = {}

        M = N - couu[i]
        for wrd in A[i]:
            #print wrd, A[i][wrd], B[i][wrd]
            temp = (A[i][wrd] * (M - B[i][wrd]) - (couu[i] - A[i][wrd])
* B[i][wrd]) ** 2 / (
            (A[i][wrd] + B[i][wrd]) * (N - A[i][wrd] - B[i][wrd]))
            CHI[wrd] = log(N / (A[i][wrd] + B[i][wrd])) * temp
                a = sorted(CHI.items(), key=lambda t: t[1],
reverse=True)[:100]
        b = []
        for aa in a:
            b.append(aa[0])
        wrd_dict.extend(b)
        for wrd in wrd_dict:
            if wrd not in wrdff:
                wrdff.append(wrd)
    return wrdff
```

```
def document_features(wrdff, TF, dat, num):

    document_wrds = set(dat)
    features = {}
    for i, wrd in enumerate(wrdff):
        if wrd in document_wrds:
            features[wrd] =
1#TF[num][wrd]#*log(N/(A[cla][wrd]+B[cla][wrd]))
        else:
            features[wrd] = 0
    return features

A, tf, tf2, trainss, test_set, couu = peocxe('dat/training.csv',
'dat/testing.csv')
B = calculate_B_from_A(A)
print("***")
wrdff = feature_select_use_new_CHI(A, B, couu)
#print wrdff
print(len(wrdff))
for wrd in wrdff:
    print(wrd)

print ("**")
documents_feature = [(document_features
                      (wrdff, tf, dat[0], i), dat[1])
                     for i, dat in enumerate(trainss)]

print ("**")
test_do = [document_features(wrdff, tf2, dat[0], i)
                          for i, dat in enumerate(test_set)]

json.dump(documents_feature, open('temp/documents_feature.txt', 'w'))
json.dump(test_do, open('temp/test_do.txt', 'w'))
```