QINMIN HU
2018年5月11日

LAB 4 查询扩展

一. 实验目的

1. 用Lab3的检索结果(tfidf,bm25) or VSM 检索结果和Qrel_disk12
2. 确定相关or不相关文档

   2.1.RF:根据qrel，在自己的检索结果的top-k文档中找到相关文档和不相关文档

   2.2.PRF: top-k文档为相关文档
3. 采用Rocchio算法更新查询向量
4. 用新向量q_m进行查询，并评测
5. 对query中的单词，在WordNet中找到其同义词

   5.1.同义词数目?

   5.2.是不是每个词都要找同义词?
6. 将同义词加入query中
7. 进行检索并评测

二. 实验步骤

1. 用户提交原始查询

```
def loadQuery():
        print('loadquery')
        xmldoc = ET.parse('topics_new.xml')
        i=0
        for doc in xmldoc.findall('top'):
                print(i)
                i=i+1
                t_dic={}
                test=''
                for s2 in doc.findall('desc'):
                        test=test+' '+s2.text.replace("Descript :","").strip()
        for s3 in doc.findall('narr'):
            test=test+' '+s3.text.replace("Narr :","").strip()
        for s4 in doc.findall('title'):
            test=test+' '+s4.text.replace("Topic :","").strip()
        while '_' in test or '`' in test or '.' in test or '-' in test or '!' in test or ';' in test or ',' in test or '$' in test or '//' in test or '/' in test or '{' in test or '}' in test or '*' in test or '#' in test or '^' in test or '|' in test or '~' in test or '=' in test or '\' in test or '+' in test or ':' in test or '?' in test:
            test = test.replace("_", "").replace('`', "").replace(".", "").replace('-', "").replace("!","").replace(";","").replace(",", "").replace("$", "").replace("//", "").replace('/', "").replace('{', "").replace('}',"").replace('*',"").replace('^', "").replace("|", "").replace('~', "").replace('=', "").replace('\', "").replace('+',"").replace(':',"").replace('?', "")
        if re.findall(r'\d+', test) or test == "":
            continue
        for word in test.split():
            if word in t_dic.keys():
                t_dic[word]=t_dic[word]+1
            else:
                t_dic[word]=1
        for s1 in doc.findall('num'):
            test_docid=s1.text.replace("Number:","").strip()
            query_id[test_docid]=len(test.split())
            query[test_docid]=t_dic
```

2. 根据原始查询返回初始查询结果

```
def loadResDic():
    （原始检索结果(bm25)的提取）
    print("loadresdic")
    f_myres=open('BM25Search.res')
    j=0
    for line in f_myres:
        if j%200<50:
            test=line.split()
            test1=[]
            if test[0] in myres_dic.keys():
                test1=myres_dic[test[0]]
            test1.append(test[2])
            myres_dic[test[0]]=test1
        j=j+1
    （标准答案相关检索结果）
    for i in range(5):
            f_qrel=open("qrels_for_disk12/qrels.
151-200.disk1.disk2.part"+str(i+1))
            for line in f_qrel:
                test=line.split()
                test1=[]
                if test[0] in qrel_dic.keys() and test[3]=='1':
                    test1=qrel_dic[test[0]]
                if test[3]=="1":
                    test1.append(test[2])
                    qrel_dic[test[0]]=test1
```

3. 确定相关or不相关文档并标记

```
        for key,value in myres_dic.items():
            for filename in value:
                if filename in qrel_dic[key]:
                    test=[]
                    if key in Cor_dic.keys():
                        test=Cor_dic[key]
                    test.append(filename)
                    Cor_dic[key]=test
                else:
                    test1=[]
                    if key in Uncor_dic.keys():
                        test1=Uncor_dic[key]
                    test1.append(filename)
                    Uncor_dic[key]=test1
```

4. 采用Rocchio算法更新查询向量

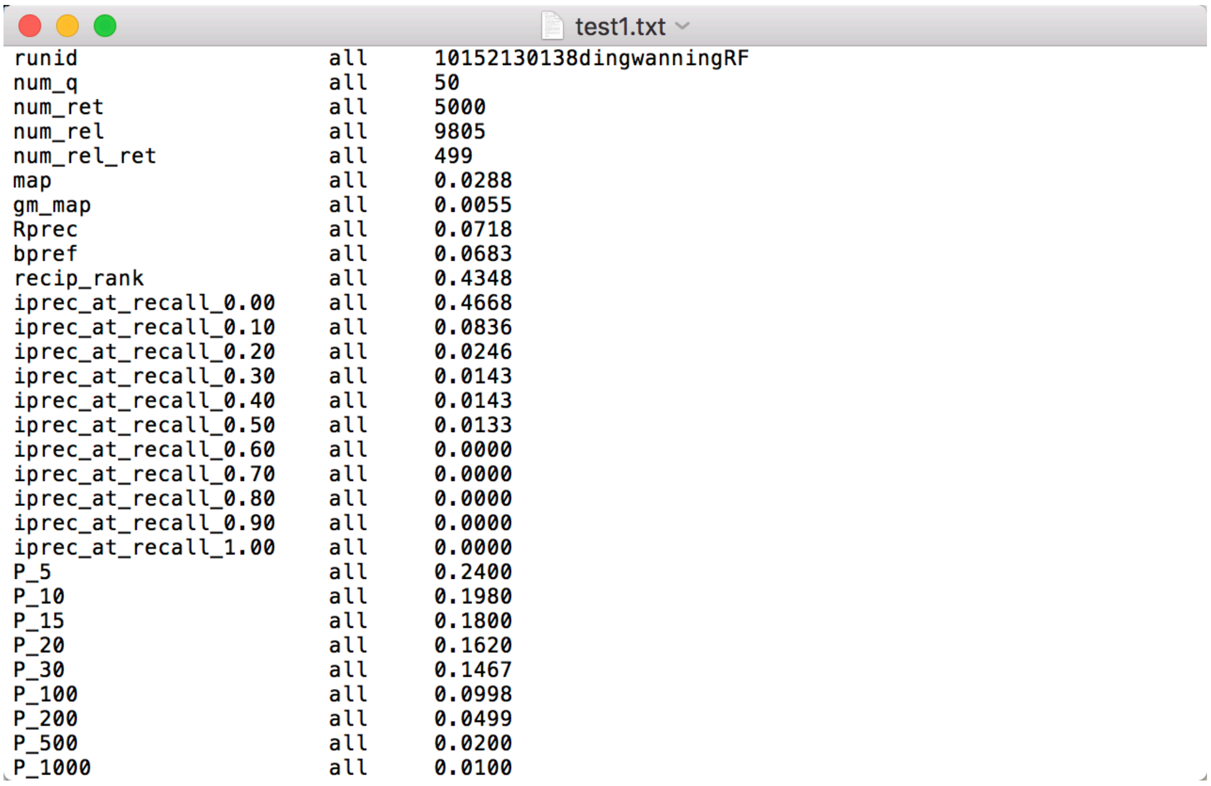
```
    def getRocchio():
        print('rocchio')
        for title,vector in matQueDic.items():
            print("rocchio"+title)
            if title in Cor_dic:
                Dr=len(Cor_dic[title])
                Dnr = len(Uncor_dic[title])

                Drv = mat([0 for i in range(len(query[title].keys()))])

                Dnrv = mat([0 for i in range(len(query[title].keys()))])
                for file in Cor_dic[title]:
                    docvector = getDocVector(query[title].keys(),
file.strip())
                    print(query[title].keys())
                    print(len(docvector))
                    Drv = Drv + mat(docvector)
                for file in Uncor_dic[title]:
                    docvector = getDocVector(query[title].keys(), 
file.strip())
                    Dnrv = Dnrv + mat(docvector)
                new_v = mat(matQueDic[title]) + 0.75 * (1 / Dr) * Drv - 0.15 * 
(1 / Dnr) * Dnrv
                newSeVector[title] = new_v.tolist()[0]
            else:
                Dnr=len(Uncor_dic[title])
                Dnrv=mat([0 for i in range(len(query[title].keys()))])
                for file in Uncor_dic[title]:
                    docvector=getDocVector(query[title].keys(),file)
                    Dnrv=Dnrv+mat(docvector)
                new_v=mat(matQueDic[title])-0.15*(1/Dnr)*Dnrv
                newSeVector[title]=new_v.tolist()[0]
```

5. 用新向量q_m进行查询，并评测

   5.1.查询

```
    def makeNewSearch(words,num):
        print('newsearch')
        scoreArr = {}
        for file in doc_filaname.keys():
            vectorList=[]
            flag=0
            for word in words.keys():
                if word in word_doc_freq.keys():
                    idf = math.log(N / word_doc_freq[word])
                else:
                    idf = 0
                if word in word_dic.keys():
                    if file in word_dic[word].keys():
                        freq = word_dic[word][file] / 
doc_filaname[file]
                        vectorList.append(idf * freq)
                        flag=1
                    else:
                        vectorList.append(0)
                else:
                    vectorList.append(0)
            if flag==0:
                continue
            t = 0
            t1 = 0
            t2 = 0
            for i in range(len(words)):
                t = t + newSeVector[num][i] * vectorList[i] * 1.0
                t1 = t1 + newSeVector[num][i]** 2 * 1.0
                t2 = t2 + vectorList[i] ** 2 * 1.0
            scoreArr[file] =t/(math.sqrt(t1)*math.sqrt(t2))*1.0
        score[num] = scoreArr
```

5.2.评测



```
runid                   all     10152130138dingwanningRF
num_q                   all     50
num_ret                 all     5000
num_rel                 all     9805
num_rel_ret             all     499
map                     all     0.0288
gm_map                  all     0.0055
Rprec                   all     0.0718
bpref                   all     0.0683
recip_rank              all     0.4348
iprec_at_recall_0.00    all     0.4668
iprec_at_recall_0.10    all     0.0836
iprec_at_recall_0.20    all     0.0246
iprec_at_recall_0.30    all     0.0143
iprec_at_recall_0.40    all     0.0143
iprec_at_recall_0.50    all     0.0133
iprec_at_recall_0.60    all     0.0000
iprec_at_recall_0.70    all     0.0000
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.2400
P_10                    all     0.1980
P_15                    all     0.1800
P_20                    all     0.1620
P_30                    all     0.1467
P_100                   all     0.0998
P_200                   all     0.0499
P_500                   all     0.0200
P_1000                  all     0.0100
```

6. 对query中的单词，在WordNet中找到其同义词，并加入其中

```
def loadQuery():
    print("loadquery")
    xmldoc = ET.parse('topics_new.xml')
    for doc in xmldoc.findall('top'):
        test = ''
        testList = []
        for s1 in doc.findall('num'):
            query_id.append(s1.text.replace("Number:", "").strip())
        for s2 in doc.findall('desc'):
            test = test + ' ' + s2.text.replace("Descript :", "").strip()
        for s3 in doc.findall('narr'):
            test = test + ' ' + s3.text.replace("Narr :", "").strip()
        for s4 in doc.findall('title'):
            test = test + ' ' + s4.text.replace("Topic :", "").strip()
```

```
                    while '_' in test or '`' in test or '.' in test or '-' in test or '!' in test or ';'
in test or ',' in test or '$' in test or '//' in test or '/' in test or '{' in test or '}' in test or '*' in
test or '#' in test or '^' in test or '|' in test or '~' in test or '=' in test or '\'' in test or '+' in
test or ':' in test or '?' in test: test = test.replace(
                            "_", "").replace('`', "").replace(".", "").replace('-',
"").replace("!", "").replace(";", "").replace(",",


                     "").replace(
                            "$", "").replace("//", "").replace('/', "").replace('{',
"").replace('}', "").replace('*', "").replace('^',


                     "").replace(
                            "|", "").replace('~', "").replace('=', "").replace('\'',
"").replace('+', "").replace(':', "").replace('?',


                     "")
                    if re.findall(r'\d+', test) or test == "":
                            continue
                    for w in test.split():
                            if len(wn.synsets(w)) == 0:
                                    testList.append(w)
                                    continue
                            else:
                                    testList.extend(wn.synsets(w)[0].lemma_names())
                            if w in testList:
                                    continue
                            else:
                                    testList.append(w)
                （from nltk.corpus import wordnet as wn  找到同义词并加入query中）
                    query.append(testList)
```

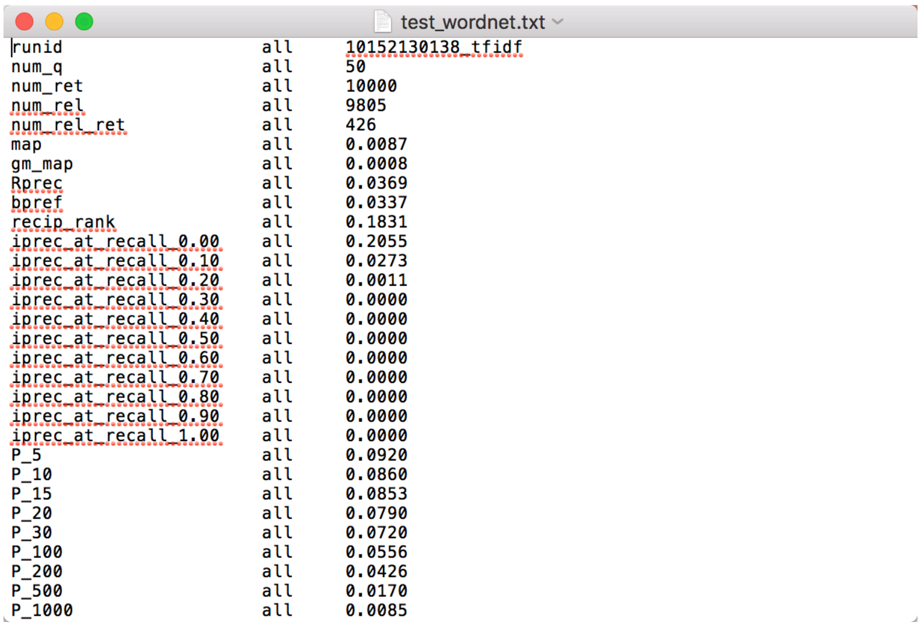
7. 进行检索并评测

（用tfidf来检索）

```
def gettfidf(words,doc_id):
        tfidf_scoreArr={}
        for file in doc_filaname.keys():
                s = 0
                for word in words:
                        if word in word_doc_freq.keys():
                                idf=math.log(N/word_doc_freq[word])
                        else:
```

```
                    idf=0
                if word in word_dic.keys():
                    if file in word_dic[word].keys():
                        freq = word_dic[word][file]/
doc_filaname[file]
                        s = s + idf * freq
                    else:
                        s= s + 0
            tfidf_scoreArr[file]=s
        tfidf_score[doc_id]=tfidf_scoreArr
```

评测：



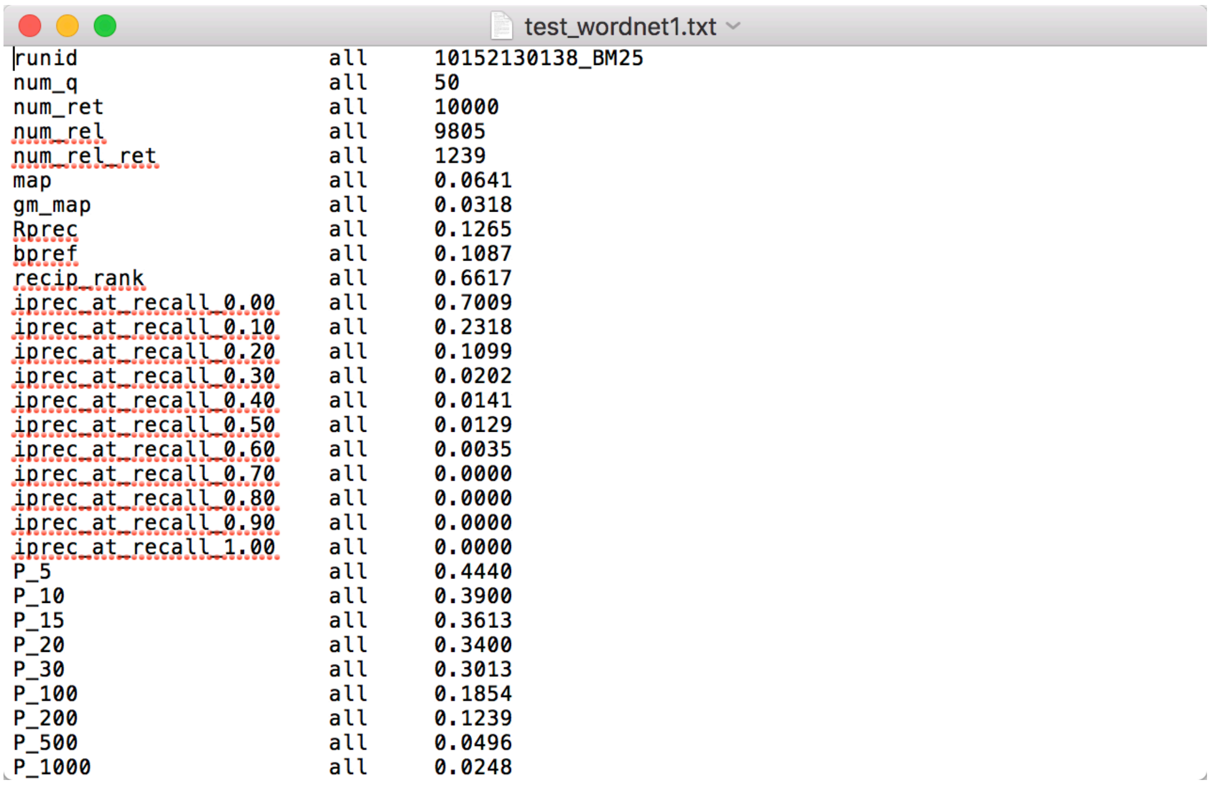
test_wordnet.txt

```
runid                   all     10152130138_tfidf
num_q                   all     50
num_ret                 all     10000
num_rel                 all     9805
num_rel_ret             all     426
map                     all     0.0087
gm_map                  all     0.0008
Rprec                   all     0.0369
bpref                   all     0.0337
recip_rank              all     0.1831
iprec_at_recall_0.00    all     0.2055
iprec_at_recall_0.10    all     0.0273
iprec_at_recall_0.20    all     0.0011
iprec_at_recall_0.30    all     0.0000
iprec_at_recall_0.40    all     0.0000
iprec_at_recall_0.50    all     0.0000
iprec_at_recall_0.60    all     0.0000
iprec_at_recall_0.70    all     0.0000
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.0920
P_10                    all     0.0860
P_15                    all     0.0853
P_20                    all     0.0790
P_30                    all     0.0720
P_100                   all     0.0556
P_200                   all     0.0426
P_500                   all     0.0170
P_1000                  all     0.0085
```

（用BM25来检索）

```
def getBM25(words,doc_id):
    BM25_scoreArr={}
    for file in doc_filaname.keys():
        s=0
        for word in words:
            if word in word_doc_freq.keys():
                idf=math.log((N-word_doc_freq[word]+0.5)/
(word_doc_freq[word]+0.5))
            else:
                idf = math.log((N - 0+ 0.5) / (0+ 0.5))
            if word in word_dic.keys():
                if file in word_dic[word].keys():
                    freq = word_dic[word][file]
                    s = s + idf * freq * 2.5 / (freq + 1.5 * (0.25 + 0.75
* doc_filaname[file] / avgLen))
            else:
                s= s + 0
        BM25_scoreArr[file]=s
BM25_score[doc_id]=BM25_scoreArr
```

评测：

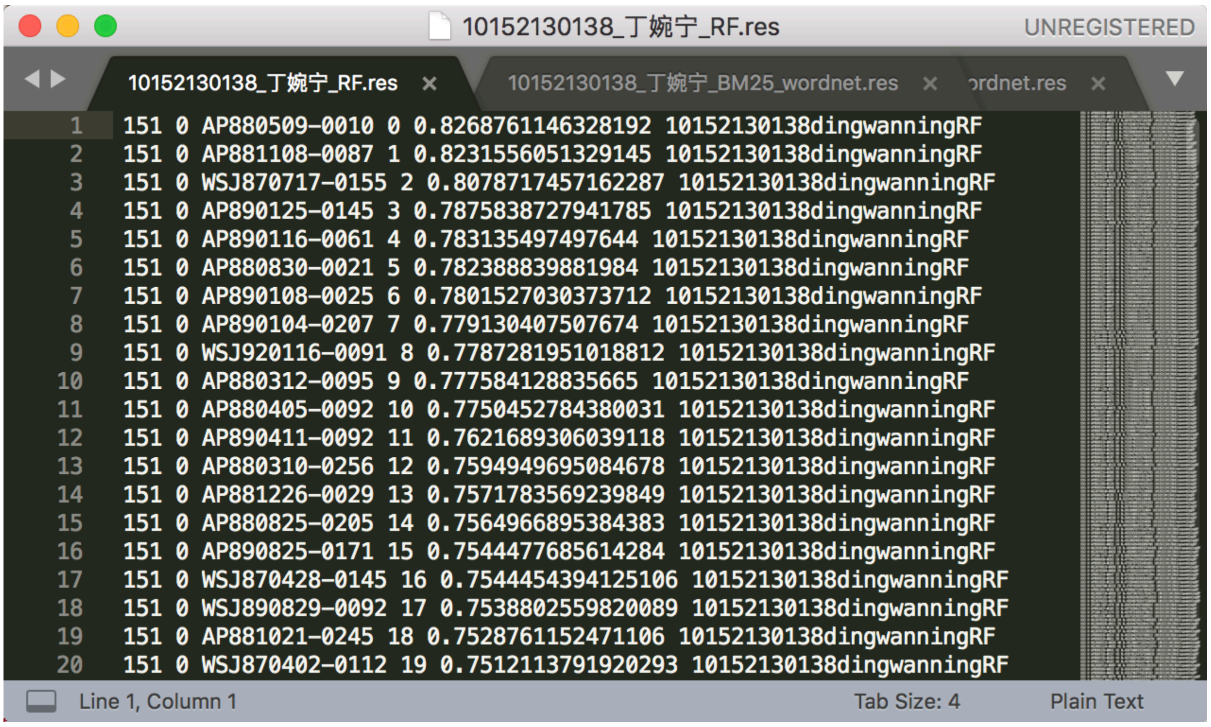


test_wordnet1.txt

```
runid                   all     10152130138_BM25
num_q                   all     50
num_ret                 all     10000
num_rel                 all     9805
num_rel_ret             all     1239
map                     all     0.0641
gm_map                  all     0.0318
Rprec                   all     0.1265
bpref                   all     0.1087
recip_rank              all     0.6617
iprec_at_recall_0.00    all     0.7009
iprec_at_recall_0.10    all     0.2318
iprec_at_recall_0.20    all     0.1099
iprec_at_recall_0.30    all     0.0202
iprec_at_recall_0.40    all     0.0141
iprec_at_recall_0.50    all     0.0129
iprec_at_recall_0.60    all     0.0035
iprec_at_recall_0.70    all     0.0000
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.4440
P_10                    all     0.3900
P_15                    all     0.3613
P_20                    all     0.3400
P_30                    all     0.3013
P_100                   all     0.1854
P_200                   all     0.1239
P_500                   all     0.0496
P_1000                  all     0.0248
```

三. 实验结果

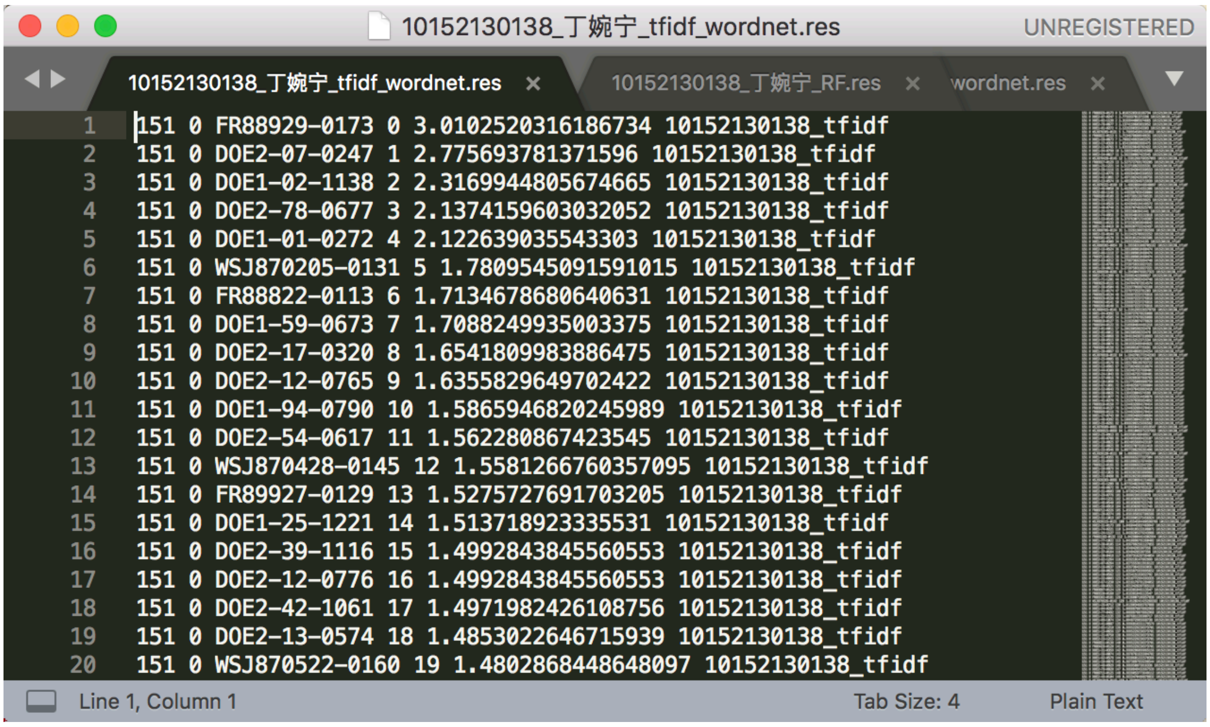
10152130138_丁婉宁_RF.res：



10152130138_丁婉宁_RF.res
UNREGISTERED
10152130138_丁婉宁_RF.res
10152130138_丁婉宁_BM25_wordnet.res
ordnet.res

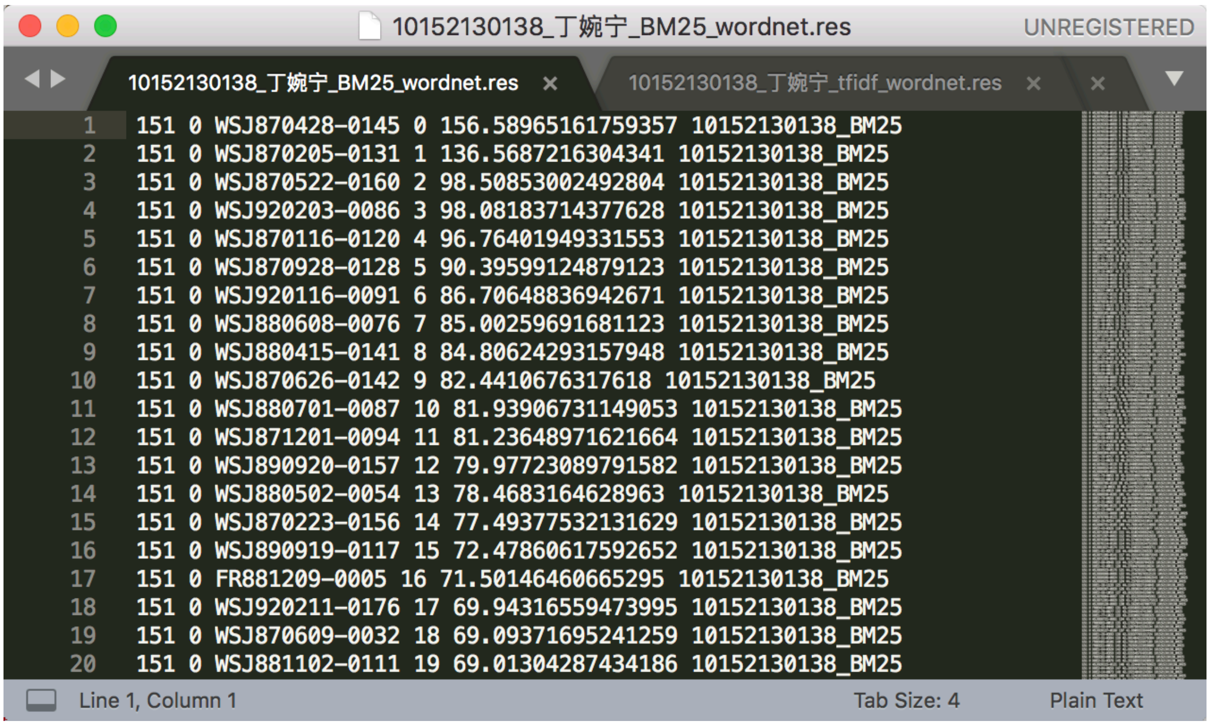
```
151 0 AP880509-0010 0 0.8268761146328192 10152130138dingwanningRF
151 0 AP881108-0087 1 0.823155605132914 5 10152130138dingwanningRF
151 0 WSJ870717-0155 2 0.8078717457162287 10152130138dingwanningRF
151 0 AP890125-0145 3 0.7875838727941785 10152130138dingwanningRF
151 0 AP890116-0061 4 0.783135497497644 10152130138dingwanningRF
151 0 AP880830-0021 5 0.782388839881984 10152130138dingwanningRF
151 0 AP890108-0025 6 0.7801527030373712 10152130138dingwanningRF
151 0 AP890104-0207 7 0.779130407507674 10152130138dingwanningRF
151 0 WSJ920116-0091 8 0.7787281951018812 10152130138dingwanningRF
151 0 AP880312-0095 9 0.777584128835665 10152130138dingwanningRF
151 0 AP880405-0092 10 0.7750452784380031 10152130138dingwanningRF
151 0 AP890411-0092 11 0.7621689306039118 10152130138dingwanningRF
151 0 AP880310-0256 12 0.759494969508467 8 10152130138dingwanningRF
151 0 AP881226-0029 13 0.7571783569239849 10152130138dingwanningRF
151 0 AP880825-0205 14 0.7564966895384383 10152130138dingwanningRF
151 0 AP890825-0171 15 0.7544477685614284 10152130138dingwanningRF
151 0 WSJ870428-0145 16 0.7544454394125106 10152130138dingwanningRF
151 0 WSJ890829-0092 17 0.7538802559820089 10152130138dingwanningRF
151 0 AP881021-0245 18 0.7528761152471106 10152130138dingwanningRF
151 0 WSJ870402-0112 19 0.7512113791920293 10152130138dingwanningRF
```

10152130138_丁婉宁_tfidf_wordnet.res：


10152130138_丁婉宁_tfidf_wordnet.res
UNREGISTERED
10152130138_丁婉宁_tfidf_wordnet.res
10152130138_丁婉宁_RF.res
wordnet.res
Line 1, Column 1
Tab Size: 4
Plain Text

```
151 0 FR88929-0173 0 3.0102520316186734 10152130138_tfidf
151 0 DOE2-07-0247 1 2.775693781371596 10152130138_tfidf
151 0 DOE1-02-1138 2 2.3169944805674665 10152130138_tfidf
151 0 DOE2-78-0677 3 2.1374159603032052 10152130138_tfidf
151 0 DOE1-01-0272 4 2.122639035543303 10152130138_tfidf
151 0 WSJ870205-0131 5 1.7809545091591015 10152130138_tfidf
151 0 FR88822-0113 6 1.7134678680640631 10152130138_tfidf
151 0 DOE1-59-0673 7 1.7088249935003375 10152130138_tfidf
151 0 DOE2-17-0320 8 1.6541809983886475 10152130138_tfidf
151 0 DOE2-12-0765 9 1.6355829649702422 10152130138_tfidf
151 0 DOE1-94-0790 10 1.5865946820245989 10152130138_tfidf
151 0 DOE2-54-0617 11 1.562280867423545 10152130138_tfidf
151 0 WSJ870428-0145 12 1.5581266760357095 10152130138_tfidf
151 0 FR89927-0129 13 1.5275727691703205 10152130138_tfidf
151 0 DOE1-25-1221 14 1.513718923335531 10152130138_tfidf
151 0 DOE2-39-1116 15 1.4992843845560553 10152130138_tfidf
151 0 DOE2-12-0776 16 1.4992843845560553 10152130138_tfidf
151 0 DOE2-42-1061 17 1.4971982426108756 10152130138_tfidf
151 0 DOE2-13-0574 18 1.4853022646715939 10152130138_tfidf
151 0 WSJ870522-0160 19 1.4802868448648097 10152130138_tfidf
```

10152130138_丁婉宁_BM25_wordnet.res：


10152130138_丁婉宁_BM25_wordnet.res
UNREGISTERED
10152130138_丁婉宁_BM25_wordnet.res
10152130138_丁婉宁_tfidf_wordnet.res
Line 1, Column 1
Tab Size: 4
Plain Text

```
151 0 WSJ870428-0145 0 156.58965161759357 10152130138_BM25
151 0 WSJ870205-0131 1 136.5687216304341 10152130138_BM25
151 0 WSJ870522-0160 2 98.50853002492804 10152130138_BM25
151 0 WSJ920203-0086 3 98.08183714377628 10152130138_BM25
151 0 WSJ870116-0120 4 96.76401949331553 10152130138_BM25
151 0 WSJ870928-0128 5 90.39599124879123 10152130138_BM25
151 0 WSJ920116-0091 6 86.70648836942671 10152130138_BM25
151 0 WSJ880608-0076 7 85.00259691681123 10152130138_BM25
151 0 WSJ880415-0141 8 84.80624293157948 10152130138_BM25
151 0 WSJ870626-0142 9 82.4410676317618 10152130138_BM25
151 0 WSJ880701-0087 10 81.93906731149053 10152130138_BM25
151 0 WSJ871201-0094 11 81.23648971621664 10152130138_BM25
151 0 WSJ890920-0157 12 79.97723089791582 10152130138_BM25
151 0 WSJ880502-0054 13 78.4683164628963 10152130138_BM25
151 0 WSJ870223-0156 14 77.49377532131629 10152130138_BM25
151 0 WSJ890919-0117 15 72.47860617592652 10152130138_BM25
151 0 FR881209-0005 16 71.50146460665295 10152130138_BM25
151 0 WSJ920211-0176 17 69.94316559473995 10152130138_BM25
151 0 WSJ870609-0032 18 69.09371695241259 10152130138_BM25
151 0 WSJ881102-0111 19 69.01304287434186 10152130138_BM25
```