Lab1 预处理 实验报告

10152130122 钱庭涵

preprocess.py

主函数中，创建需要用到的文件夹。

xml_form 中存放原 query/document 文件修改成 xml 格式后的文件，这个文件夹只是个暂存文件夹，预处理完毕后会删除。

Answer 文件夹中对应存放 query 和各种类型的 document 文件预处理后的结果。

```
168   if (not (os.path.exists('./xml_form'))):
169       os.mkdir('./xml_form')
170   if (not (os.path.exists('./Answer'))):
171       os.mkdir('./Answer')
172   if (not (os.path.exists('./Answer/query'))):
173       os.mkdir('./Answer/query')
174   if (not (os.path.exists('./Answer/document'))):
175       os.mkdir('./Answer/document')
176   if (not (os.path.exists('./Answer/document/AP'))):
177       os.mkdir('./Answer/document/AP')
178   if (not (os.path.exists('./Answer/document/DOE'))):
179       os.mkdir('./Answer/document/DOE')
180   if (not (os.path.exists('./Answer/document/FR'))):
181       os.mkdir('./Answer/document/FR')
182   if (not (os.path.exists('./Answer/document/WSJ'))):
183       os.mkdir('./Answer/document/WSJ')
184   if (not (os.path.exists('./Answer/document/ZIFF'))):
185       os.mkdir('./Answer/document/ZIFF')
```

读取 query 文件。

```
187   file1 = open('./query&document/topics.151-200', 'r')
188   Query_Lines = file1.read()
189   file1.close()
```

将 query 文件转换成 xml 格式的文件。

```
191   change_query_form(Query_Lines)
```

跳转到 change_query_form 函数，创建并打开 topics.xml 文件，先写入一个根元素，再在 QueryLines 中对应加上其他子元素的结尾，query 文件的格式比较统一，每个文件都有一对<top></top>根元素，并有这样的子元素：一个<num>、一个<title>、一个<desc>、一个<narr>（按顺序出现），因此使用字符串的 replace 函数，给对应的子元素加上结尾标志，再写入文件中，最后写入根元素的结尾。

```
12  def change_query_form(QueryLines):
13      fp = open('./xml_form/topics.xml', 'w')
14      fp.write('<query>\n\n')
15      QueryLines = QueryLines.replace('<title>', '</num>\n\n<title>').replace('<desc>', '</title>\n\n<desc>')\
16          .replace('<narr>', '</desc>\n\n<narr>').replace('</top>', '</narr>\n\n</top>')
17      fp.write(QueryLines)
18      fp.write('\n</query>\n\n')
19      fp.close()
```

跳转回主函数，对 xml 格式的 query 文件进行预处理，query_num 列表中存放 query 中各个子文件的名称。

```
192   query_num = []
193   query_process()
```

跳转到 query_process 函数，调用 xml.etree.ElementTree 模块，用 xml 文件的方式打开 query 文件，num 列表中存放每个子文件的名称，同理，title、desc、narr 列表中存放每个子文件的 title、desc、narr。

使用 findall 的方式找到所有 top 标签，在每个标签中，寻找一个子标签 num，获取该标签的文本，存入 num 列表，再去除该文本前后的空行、"Number:"、空格，得到该子文件的标题，添加到 query_num 列表中。

然后在每个 top 标签中，寻找一个子标签 title，获取该标签的文本，存入 title 列表，desc 和 narr 同理。

```
59  def query_process():
60      root = ET.parse('./xml_form/topics.xml')
61      num = []
62      title = []
63      desc = []
64      narr = []
65      for file in root.findall('top'):
66          i = file.find('num').text
67          num.append(i)
68          temp = i.strip('\n').replace('Number:', '').strip(' ')
69          query_num.append(temp)
70          title.append(file.find('title').text)
71          desc.append(file.find('desc').text)
72          narr.append(file.find('narr').text)
```

对各列表中的文本进行预处理，存入新的列表中。

```
74      stem_num = pre_process(num)
75      stem_title = pre_process(title)
76      stem_desc = pre_process(desc)
77      stem_narr = pre_process(narr)
```

跳转到 pre_process 函数, word_stem_list 是返回的预处理后的文本列表, 遍历 word_list 中的的文本，进行一系列处理（全部转换成小写字母，词条化，去除停用词，词性归并，词干还原，去除大部分无用符号），再添加到 word_stem_list 列表中。

词干还原采用 porter 算法，调用写好的 **porter.py** 来实现。

返回的预处理后的文本列表。

```
48  def pre_process(word_list):
49      word_stem_list = []
50      for sentence in word_list:
51          tokens = nltk.word_tokenize(sentence.lower())
52          lemmatizaer = WordNetLemmatizer()
53          stem_list = [(porter.Stemmer()).stem(lemmatizaer.lemmatize(w)) for w in tokens \
54                       if (w not in stopwords.words('english'))]
55          tmp = re.split('[][():;/|.,*"+=> \\\?!^@_$#%{}-]+', ' '.join(stem_list))
56          word_stem_list.append(' '.join(tmp))
57      return word_stem_list
```

回到 query_process 函数，使用 findall 的方式找到所有 top 标签，在每个标签中，寻找一个子标签 num，将预处理后的文本对应放入该 num 子标签的文本中，title、desc、narr 同理，最后再将文件写回到 query 文件中。

```
78      j = 0
79      for file in root.findall('top'):
80          elem = file.find('num')
81          elem.text = stem_num[j]
82          elem = file.find('title')
83          elem.text = stem_title[j]
84          elem = file.find('desc')
85          elem.text = stem_desc[j]
86          elem = file.find('narr')
87          elem.text = stem_narr[j]
88          j += 1
89      root.write('./xml_form/topics.xml')
```

跳转回主函数，开始切分 query 文件。

```
195  query_depart()
```

跳转到 query_depart 函数，打开并读入预处理后的 query 文件，对于每行，先去掉换行符，得到 temp1 字符串，然后进行 if 判断。

如果 temp1 是'<top>'，说明是一个子文件的开始，标记 start 为 1，根据 query_num 列表中的标题打开并创建一个 xml 文件。

如果 start 为 1，说明 xml 文件已经创建并打开，可以开始写入读取的行。

如果 temp1 是'</top>'，说明到达一个子文件的结尾（此时元素的结尾标志也已经写入了），关闭文件，print 对应的成功标志，标记 start 为 0，表示文件写入停止。

最后删除已经无用处的 xml 格式的 query 文件，节约内存。

```
91   def query_depart():
92       file = open('./xml_form/topics.xml', 'r')
93       Lines = file.readlines()
94       file.close()
95       k = 0
96       start = 0
97       for line in Lines:
98           temp1 = line.strip('\n')
99           if temp1 == '<top>':
100              start = 1
101              file = open('./Answer/query/Topic_' + query_num[k] + '.xml', 'w')
102          if start == 1:
103              file.write(line)
104          if temp1 == '</top>':
105              file.close()
106              print('Topic_' + query_num[k] + ' succeed')
107              start = 0
108              k += 1
109      os.remove('./xml_form/topics.xml')
```

跳转回主函数，创建 path 列表，添加所有 document 文件所在的文件夹路径。

```
197  path = []
198  path.append('./query&document/disk12/disk1/AP')
199  path.append('./query&document/disk12/disk2/AP')
200  path.append('./query&document/disk12/disk1/DOE')
201  path.append('./query&document/disk12/disk1/WSJ')
202  path.append('./query&document/disk12/disk2/WSJ/1990')
203  path.append('./query&document/disk12/disk2/WSJ/1991')
204  path.append('./query&document/disk12/disk2/WSJ/1992')
205  path.append('./query&document/disk12/disk1/ZIFF')
206  path.append('./query&document/disk12/disk2/ZIFF')
207  path.append('./query&document/disk12/disk1/FR')
208  path.append('./query&document/disk12/disk2/FR')
```

创建 elem_type 列表，添加不同种类 document 文件中，需要删除的子标签列表。

```
210  elem_type = []
211  AP_remove_elem = ['FILEID', 'NOTE', 'UNK', 'FIRST', 'SECOND', 'HEAD', 'DATELINE', 'BYLINE']
212  elem_type.append(AP_remove_elem)
213  DOE_remove_elem = []
214  elem_type.append(DOE_remove_elem)
215  WSJ_remove_elem = ['AN', 'AUTHOR', 'CO', 'DATELINE', 'DOCID', 'DATE', 'LP', 'DD', 'DO', 'G',
216                     'GV', 'HL', 'IN', 'SO', 'MS', 'ST', 'RE', 'NS']
217  elem_type.append(WSJ_remove_elem)
218  ZF_remove_elem = ['JOURNAL', 'TITLE', 'AUTHOR', 'SUMMARY', 'DESCRIPT', 'DOCID', 'PRODUCT',
219                    'ADDRESS', 'COMPANY', 'CATEGORY', 'SPECS', 'NOTE']
220  elem_type.append(ZF_remove_elem)
221  FR_remove_elem = ['DOCID']
222  elem_type.append(FR_remove_elem)
```

遍历 path 路径，调用 os.walk 模块遍历路径中的每一个文件，获取当前文件的文件名。

如果当前文件不是 README.gz，则为有效的 document 文件，开始进行处理。

调用 gzip 模块，直接打开该文件压缩包，读取其所有内容，

```
224    for count in range(len(path)):
225        for a in os.walk(path[count]):
226            document_files = a[2]
227            for document_file in document_files:
228                if document_file != 'README.gz':
229                    file2 = gzip.GzipFile(path[count] + '/' + document_file, 'r')
230                    document_lines = file2.read()
231                    file2.close()
232                    document_file_name = document_file.replace('.gz', '')
```

将当前读入的 document 文件转换成 xml 格式。

```
233                    change_document_form(document_lines, document_file_name)
```

跳转到 change_document_form 函数，创建并打开当前 document 文件的 xml 格式文件，先写入根元素，然后调用 chardet 模块，获取当前 document 文件的编码类型，根据不同编码类型，使用不同的方式，将当前 document 文件的内容改用 utf-8 编码，然后进行类型判断：

在五种 document 文件中（AP、DOE、WSJ、ZF、FR）中，AP、DOE 和 WSJ 种类 document 的<TEXT>标签中不存在嵌套子标签，因此可以将文件中影响 xml 格式的'&'符号去掉之后，直接写入 xml 格式文件。

在 ZF 种类 document 中，<TEXT>标签中嵌套有<ABSTRACT>子标签，因此除了去掉'&'符号之外，还去掉<ABSTRACT>和</ABSTRACT>这一对子标签，再写入 xml 格式文件。

在 FR 种类 document 中，<TEXT>标签中嵌套有多种类型子标签，因此先去掉'&'符号，再将需要用到的标签（包括在 elem_type 列表中记录的需要删除的标签）替换成不含'<>'字符的独一无二的纯文本，然后调用 re 模块去除读入内容中所有'<>'及其内部的字符串，这样就去掉了嵌套的多种类型子标签，再将之前替换的纯文本替换为带'<>'的标签，最后写入 xml 格式文件。

最后再写入根元素的结尾。

```
21  def change_document_form(DocumentLines, type):
22      fp = open('./xml_form/' + type + '.xml', 'w')
23      fp.write('<document>\n\n')
24      strCodingFmt = chardet.detect(DocumentLines)['encoding']
25      if strCodingFmt == 'ISO-8859-1':
26          lines_tmp = DocumentLines.decode('utf-8', 'ignore')
27      else:
28          lines_tmp = DocumentLines.decode()
29      if type[:2] == 'ZF':
30          fp.write(lines_tmp.replace('&', '').replace('<ABSTRACT>', '\n').replace('</ABSTRACT>', '\n'))
31      elif type[:2] == 'FR':
32          lines_tmp1 = lines_tmp.replace('&', '').replace('<DOC>', '(Document_FR_DOC1)')\
33              .replace('</DOC>', '(Document_FR_DOC2)').replace('<DOCNO>', '(Document_FR_DOCNO1)')\
34              .replace('</DOCNO>', '(Document_FR_DOCNO2)').replace('<DOCID>', '(Document_FR_DOCID1)')\
35              .replace('</DOCID>', '(Document_FR_DOCID2)').replace('<TEXT>', '(Document_FR_TEXT1)')\
36              .replace('</TEXT>', '(Document_FR_TEXT2)')
37          lines_tmp2 = re.sub(r'<.*>', ' ', lines_tmp1)
38          lines_tmp3 = lines_tmp2.replace('(Document_FR_DOC1)', '<DOC>').replace('(Document_FR_DOC2)', '</DOC>')\
39              .replace('(Document_FR_DOCNO1)', '<DOCNO>').replace('(Document_FR_DOCNO2)', '</DOCNO>')\
40              .replace('(Document_FR_DOCID1)', '<DOCID>').replace('(Document_FR_DOCID2)', '</DOCID>')\
41              .replace('(Document_FR_TEXT1)', '<TEXT>').replace('(Document_FR_TEXT2)', '</TEXT>')
42          fp.write(lines_tmp3)
43      else:
44          fp.write(lines_tmp.replace('&', ''))
45      fp.write('\n</document>\n\n')
46      fp.close()
```

跳转回主函数的 for 循环内，判断 document 文件的类型，对于不同的文件，使用不同的列表处理。

如 AP 类型的 document 文件，AP_title 存放所有当前文件的子文件的标题，elem_type 列表则取对应的 AP 类型 document 中需要删除的子标签列表，document_depart 函数的最后一个参数为子文件应存放的文件夹名，即'AP'。

DOE、WSJ、ZF、FR 同理。

```
234                 if document_file[:2] == 'AP':
235                     AP_title = []
236                     document_process(document_file_name, AP_title, elem_type[0])
237                     document_depart(document_file_name, AP_title, 'AP')
238                 if document_file[:3] == 'DOE':
239                     DOE_title = []
240                     document_process(document_file_name, DOE_title, elem_type[1])
241                     document_depart(document_file_name, DOE_title, 'DOE')
242                 if document_file[:3] == 'WSJ':
243                     WSJ_title = []
244                     document_process(document_file_name, WSJ_title, elem_type[2])
245                     document_depart(document_file_name, WSJ_title, 'WSJ')
246                 if document_file[:2] == 'ZF':
247                     ZF_title = []
248                     document_process(document_file_name, ZF_title, elem_type[3])
249                     document_depart(document_file_name, ZF_title, 'ZIFF')
250                 if document_file[:2] == 'FR':
251                     FR_title = []
252                     document_process(document_file_name, FR_title, elem_type[4])
253                     document_depart(document_file_name, FR_title, 'FR')
```

跳转到 document_process 函数，调用 xml.etree.ElementTree 模块，用 xml 文件的方式打开 document 文件，DOCNO 列表中存放每个子文件的名称，TEXT 列表中存放每个子文件中有用的文本。

使用 findall 的方式找到所有 DOC 标签，在每个标签中，寻找一个子标签 DOCNO，获取该标签的文本，存入 DOCNO 列表，再去除该文本前后的空行和空格，得到该子文件的标题，添加到 title_list 列表中。

然后寻找所有 TEXT 标签，获取该标签的文本，存入 temp2 列表。

如果 temp2 列表不为空，则按照一个换行符作为间隔，将 temp2 列表中所有字符串连接在一起，并添加到 TEXT 列表中。

如果 temp2 列表为空，则将一个换行符添加到 TEXT 列表中。

再根据 remove_list，寻找所有需要删除的子标签，并删除该子元素。

```
111 def document_process(type, title_list, remove_list):
112     root = ET.parse('./xml_form/' + type + '.xml')
113     DOCNO = []
114     TEXT = []
115     for file in root.findall('DOC'):
116         i = file.find('DOCNO').text
117         DOCNO.append(i)
118         temp = i.strip('\n').strip(' ')
119         title_list.append(temp)
120         temp2 = [f.text for f in file.findall('TEXT')]
121         if temp2 != [None]:
122             TEXT.append('\n'.join(temp2))
123         else:
124             TEXT.append('\n')
125         for remove_elem in remove_list:
126             for k in file.findall(remove_elem):
127                 file.remove(k)
```

对 DOCNO 和 TEXT 列表中的文本进行预处理，存入新的列表中。

```
129         stem_DOCNO = pre_process(DOCNO)
130         stem_TEXT = pre_process(TEXT)
```

使用 findall 的方式找到所有 DOC 标签，在每个标签中，寻找一个子标签 DOCNO，将预处理后的文本对应放入该 DOCNO 子标签的文本中。

在每个标签中，寻找所有 TEXT 子标签，如果对应的预处理后的文本为空，则将一个空格字符放入该子标签的文本中，并跳出循环。

如果对应的预处理后的文本不为空，根据标记 i 判断当前 TEXT 子标签是不是寻找到的第一个 TEXT 子标签，如果是，就将对应的预处理后的文本放入该子标签的文本中，并将 i 标记为 1；如果不是，说明对应的预处理后的文本已经放好了，因此删除该子元素。

最后再将文件写回到对应的 document 文件中。

```
131         j = 0
132         for file in root.findall('DOC'):
133             elem = file.find('DOCNO')
134             elem.text = stem_DOCNO[j]
135             i = 0
136             for elem in file.findall('TEXT'):
137                 if stem_TEXT[j] == '':
138                     elem.text = ' '
139                     break
140                 if i != 0:
141                     file.remove(elem)
142                 else:
143                     elem.text = stem_TEXT[j]
144                     i = 1
145             j += 1
146         root.write('./xml_form/' + type + '.xml')
```

跳转到 document_depart 函数，打开并读入预处理后的 document 文件，对于每行，先去掉换行符，得到 temp1 字符串，然后进行 if 判断。

如果 temp1 是'<DOC>'，说明是一个子文件的开始，标记 start 为 1，根据 folder_name 和 title_list 列表中的标题打开并创建一个 xml 文件。

如果 start 为 1，说明 xml 文件已经创建并打开，可以开始写入读取的行。

如果 temp1 是'</DOC>'，说明到达一个子文件的结尾（此时元素的结尾标志也已经写入了），关闭文件，print 对应的成功标志，标记 start 为 0，表示文件写入停止。

最后删除已经无用处的 xml 格式的 document 文件，节约内存。

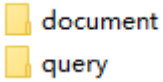
```
148     def document_depart(type, title_list, folder_name):
149         file = open('./xml_form/' + type + '.xml', 'r')
150         Lines = file.readlines()
151         file.close()
152         k = 0
153         start = 0
154         for line in Lines:
155             temp1 = line.strip('\n')
156             if temp1 == '<DOC>':
157                 start = 1
158                 file = open('./Answer/document/' + folder_name + '/' + title_list[k] + '.xml', 'w')
159             if start == 1:
160                 file.write(line)
161             if temp1 == '</DOC>':
162                 file.close()
163                 print(title_list[k] + ' succeed')
164                 start = 0
165                 k += 1
166         os.remove('./xml_form/' + type + '.xml')
```

跳转回主函数的 for 循环外，此时所有 query 和 document 文件都已经预处理并切分完毕，删除已经无用的 xml_form 文件夹，完成 lab1 的预处理。

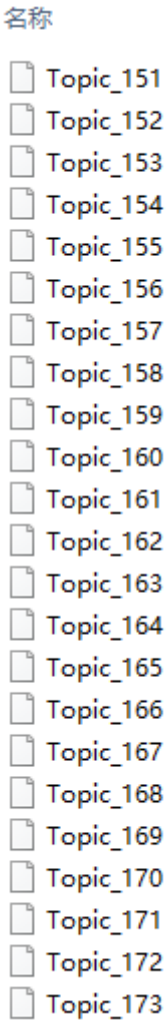
```
255    os.rmdir('./xml_form')
```

Result：

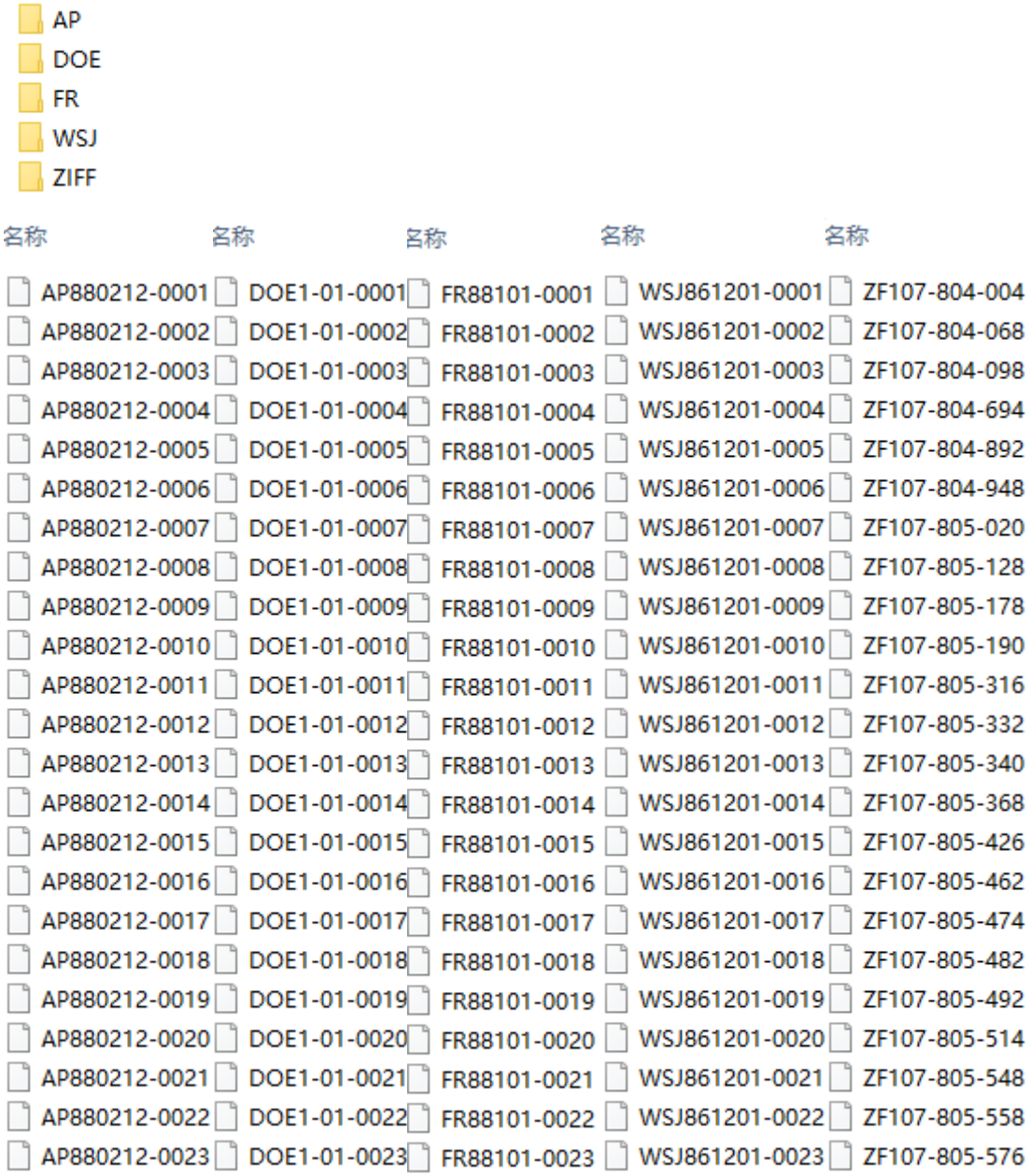
最后得到的 Answer 文件夹中，含有 query 和 document 两个子文件夹。



query 文件夹中存放 query 的所有预处理并切分的结果。



document 文件夹中含有 AP、DOE、FR、WSJ、ZIFF 五个子文件夹，分别存放不同类型 document 的所有预处理并切分的结果。



提交的文件示例：

在提交文件时，为了注明这是哪个文件切分后的结果，命名的文件夹格式与上述得到的结果有些不同，增添了子文件夹，注明这是哪个磁盘哪个文件切分后的结果。

以下只取了 disk1 的 FR891013 作处理前和处理后的对比，其他提交的文件请在 Answer 文件夹中查看。

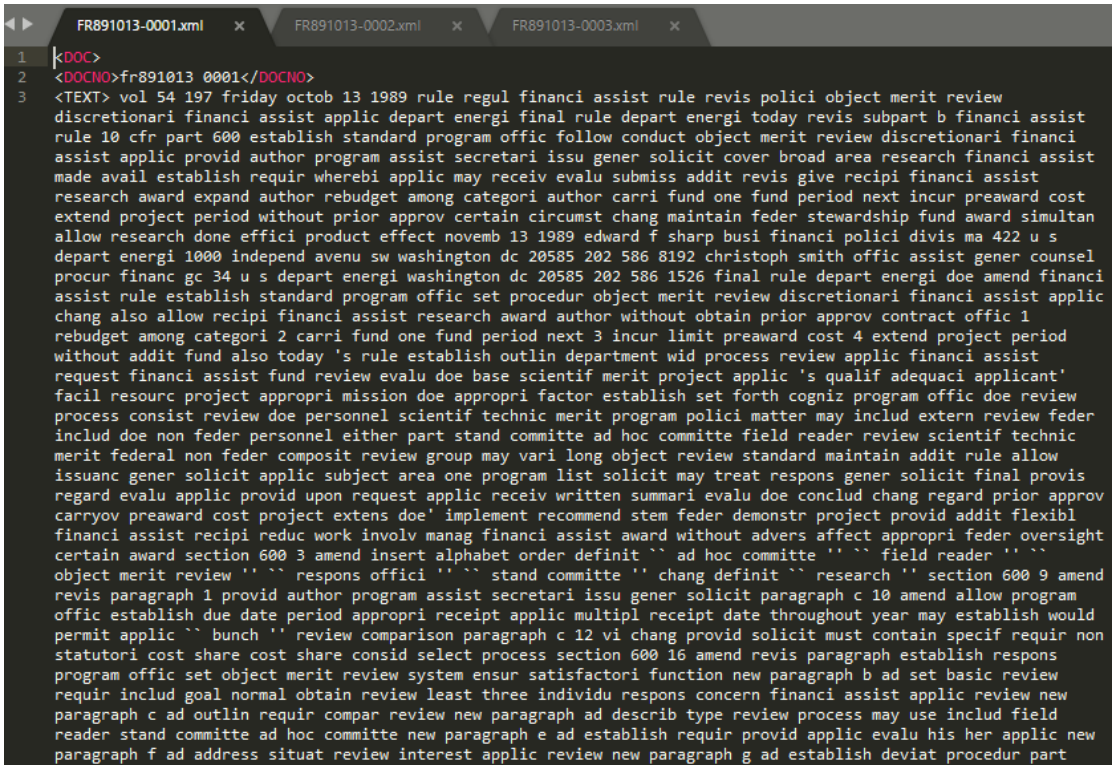
处理前：

FR891013

```
<DOC>
<DOCNO> FR891013-0001 </DOCNO>
<DOCID>fr.10-13-89.f2.A1000</DOCID>
<TEXT>
<FTAG tagnum=4700></FTAG>
<ITAG tagnum=90>
<T4>Federal Register</T4> / Vol. 54, No. 197 / Friday, October 13, 1989
/ Rules and Regulations
<ITAG tagnum=1>Vol. 54, No. 197 </ITAG>

<ITAG tagnum=2>Friday, October 13, 1989 </ITAG>

<ITAG tagnum=94>
<ITAG tagnum=69>
<ITAG tagnum=50>DEPARTMENT OF ENERGY</ITAG>

<ITAG tagnum=18>Office of the Secretary</ITAG>

<ITAG tagnum=52>10 CFR Part 600 </ITAG>

<ITAG tagnum=52>Financial Assistance Rules; Revised Policy on Objective Merit Review
of Discretionary Financial Assistance Applications </ITAG>

<ITAG tagnum=10>
<T2>AGENCY: </T2>Department of Energy.
</ITAG>

<ITAG tagnum=10>
<T2>ACTION: </T2>Final rule.
 </ITAG>

<ITAG tagnum=10>
<T2>SUMMARY: </T2>The Department of Energy today is revising subparts A
and B of the Financial Assistance Rules, 10 CFR part 600, to establish
standards for program offices to follow in conducting the objective merit
```

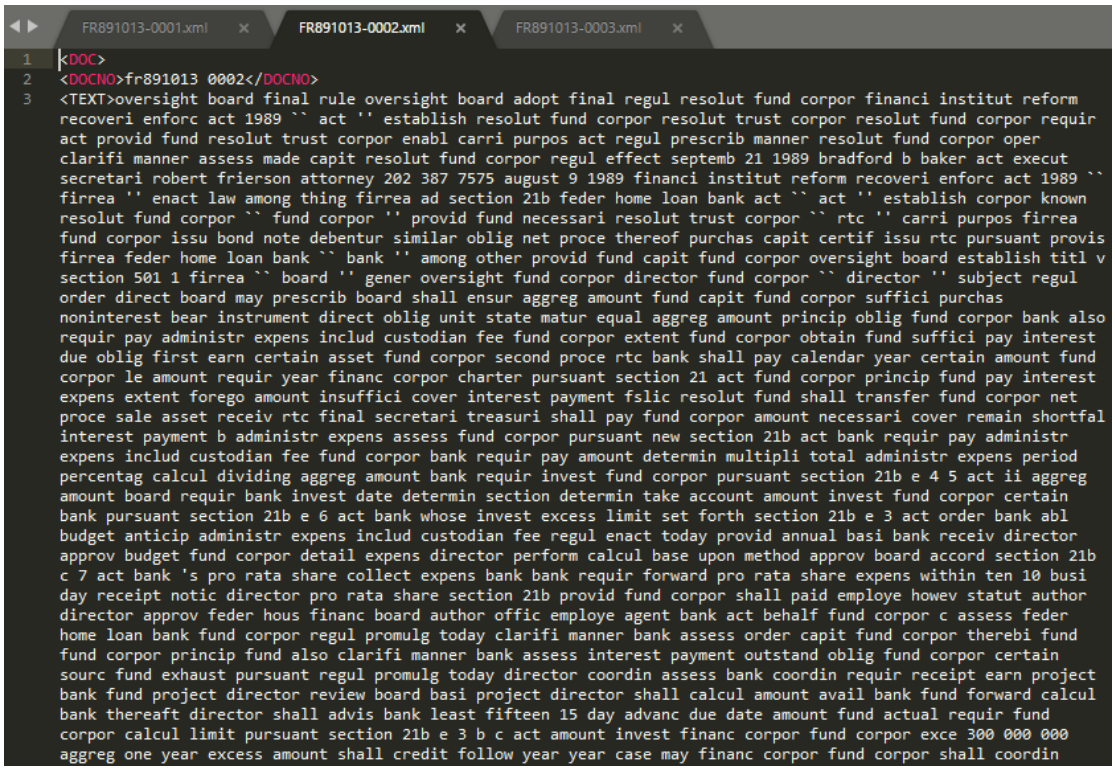
处理后部分文件：

FR891013-0001.xml



```
FR891013-0001.xml  FR891013-0002.xml  FR891013-0003.xml
<DOC>
<DOCNO>fr891013 0001</DOCNO>
<TEXT> vol 54 197 friday octob 13 1989 rule regul financi assist rule revis polici object merit review
discretionari financi assist applic depart energi final rule depart energi today revis subpart b financi assist
rule 10 cfr part 600 establish standard program offic follow conduct object merit review discretionari financi
assist applic provid author program assist secretari issu gener solicit cover broad area research financi assist
made avail establish requir wherebi applic may receiv evalu submiss addit revis give recipi financi assist
research award expand author rebudget among categori author carri fund one fund period next incur preaward cost
extend project period without prior approv certain circumst chang maintain feder stewardship fund award simultan
allow research done effici product effect novemb 13 1989 edward f sharp busi financi polici divis ma 422 u s
depart energi 1000 independ avenu sw washington dc 20585 202 586 8192 christoph smith offic assist gener counsel
procur financ gc 34 u s depart energi washington dc 20585 202 586 1526 final rule depart energi doe amend financi
assist rule establish standard program offic set procedur object merit review discretionari financi assist applic
chang also allow recipi financi assist research award author without obtain prior approv contract offic 1
rebudget among categori 2 carri fund one fund period next 3 incur limit preaward cost 4 extend project period
without addit fund also today 's rule establish outlin department wid process review applic financi assist
request financi assist fund review evalu doe base scientif merit project applic 's qualif adequaci applicant'
facil resourc project appropri mission doe appropri factor establish set forth cogniz program offic doe review
process consist review doe personnel scientif technic merit program polici matter may includ extern review feder
includ doe non feder personnel either part stand committe ad hoc committe field reader review scientif technic
merit federal non feder composit review group may vari long object review standard maintain addit rule allow
issuanc gener solicit applic subject area one program list solicit may treat respons gener solicit final provis
regard evalu applic provid upon request applic receiv written summari evalu doe conclud chang regard prior approv
carryov preaward cost project extens doe' implement recommend stem feder demonstr project provid addit flexibl
financi assist recipi reduc work involv manag financi assist award without advers affect appropri feder oversight
certain award section 600 3 amend insert alphabet order definit `` ad hoc committe '' `` field reader '' ``
object merit review '' `` respons offici '' `` stand committe '' chang definit `` research '' section 600 9 amend
revis paragraph 1 provid author program assist secretari issu gener solicit paragraph c 10 amend allow program
offic establish due date period appropri receipt applic multipl receipt date throughout year may establish would
permit applic `` bunch '' review comparison paragraph c 12 vi chang provid solicit must contain specif requir non
statutori cost share cost share consid select process section 600 16 amend revis paragraph establish respons
program offic set object merit review system ensur satisfactori function new paragraph b ad set basic review
requir includ goal normal obtain review least three individu respons concern financi assist applic review new
paragraph c ad outlin requir compar review new paragraph ad describ type review process may use includ field
reader stand committe ad hoc committe new paragraph e ad establish requir provid applic evalu his her applic new
paragraph f ad address situat review interest applic review new paragraph g ad establish deviat procedur part
```

FR891013-0002.xml



```
FR891013-0001.xml  FR891013-0002.xml  FR891013-0003.xml
<DOC>
<DOCNO>fr891013 0002</DOCNO>
<TEXT>oversight board final rule oversight board adopt final regul resolut fund corpor financi institut reform
recoveri enforc act 1989 `` act '' establish resolut fund corpor resolut trust corpor resolut fund corpor requir
act provid fund resolut trust corpor enabl carri purpos act regul prescrib manner resolut fund corpor oper
clarifi manner assess made capit resolut fund corpor regul effect septemb 21 1989 bradford b baker act execut
secretari robert frierson attorney 202 387 7575 august 9 1989 financi institut reform recoveri enforc act 1989 ``
firrea '' enact law among thing firrea ad section 21b feder home loan bank act `` act '' establish corpor known
resolut fund corpor `` fund corpor '' provid fund necessari resolut trust corpor `` rtc '' carri purpos firrea
fund corpor issu bond note debentur similar oblig net proce thereof purchas capit certif issu rtc pursuant provis
firrea feder home loan bank `` bank '' among other provid fund capit fund corpor oversight board establish titl v
section 501 1 firrea `` board '' gener oversight fund corpor director fund corpor `` director '' subject regul
order direct board may prescrib board shall ensur aggreg amount fund capit fund corpor suffici purchas
noninterest bear instrument direct oblig unit state matur equal aggreg amount princip oblig fund corpor bank also
requir pay administr expens includ custodian fee fund corpor extent fund corpor obtain fund suffici pay interest
due oblig first earn certain asset fund corpor second proce rtc bank shall pay calendar year certain amount fund
corpor le amount requir year financ corpor charter pursuant section 21 act fund corpor princip fund pay interest
expens extent forego amount insuffici cover interest payment fslic resolut fund shall transfer fund corpor net
proce sale asset receiv rtc final secretari treasuri shall pay fund corpor amount necessari cover remain shortfal
interest payment b administr expens assess fund corpor pursuant new section 21b act bank requir pay administr
expens includ custodian fee fund corpor bank requir pay amount determin multipli total administr expens period
percentag calcul dividing aggreg amount bank requir invest fund corpor pursuant section 21b e 4 5 act ii aggreg
amount board requir bank invest date determin section determin take account amount invest fund corpor certain
bank pursuant section 21b e 6 act bank whose invest excess limit set forth section 21b e 3 act order bank abl
budget anticip administr expens includ custodian fee regul enact today provid annual basi bank receiv director
approv budget fund corpor detail expens director perform calcul base upon method approv board accord section 21b
c 7 act bank 's pro rata share collect expens bank bank requir forward pro rata share expens within ten 10 busi
day receipt notic director pro rata share section 21b provid fund corpor shall paid employe howev statut author
director approv feder hous financ board author offic employe agent bank act behalf fund corpor c assess feder
home loan bank fund corpor regul promulg today clarifi manner bank assess order capit fund corpor therebi fund
fund corpor princip fund also clarifi manner bank assess interest payment outstand oblig fund corpor certain
sourc fund exhaust pursuant regul promulg today director coordin assess bank coordin requir receipt earn project
bank fund project director review board basi project director shall calcul amount avail bank fund forward calcul
bank thereaft director shall advis bank least fifteen 15 day advanc due date amount fund actual requir fund
corpor calcul limit pursuant section 21b e 3 b c act amount invest financ corpor fund corpor exce 300 000 000
aggreg one year excess amount shall credit follow year year case may financ corpor fund corpor shall coordin
```

FR891013-0003.xml