

```

1  # CIS 511 NLP - Assignment 4.1 - Natural Language Understanding for Dialog Systems
2
3  """
4  Created on Sat April 11 20:30:18 2020
5
6  @author: Siyu Yang
7  @unique name: siyuya
8  @UMID:76998080
9  """
10
11  from collections import defaultdict
12  import pandas
13  from sklearn.tree import DecisionTreeClassifier
14  from sklearn.metrics import accuracy_score
15  from sklearn.model_selection import train_test_split
16  import re
17  import sys
18
19
20  #=== Functions ===#
21
22  # 1) find the middle content between begin word and end word
23  def middle_content(begin, end, content):
24      mid_content = ''
25
26      if content.find(begin): # find the line start with begin words
27          beginword = content[content.find(begin):content.rfind(end)]
28          mid_content = beginword[len(begin):]
29          return mid_content
30
31
32  # 2) convert I/O/B into number 1/0/2
33  def IOB_to_Num(x):
34      IOB_Char = x[-1]
35      if IOB_Char=='I':
36          return 1
37      elif IOB_Char == 'B':
38          return 2
39      else :
40          return 0
41
42
43
44  # 3) Process Train File to get train data
45  def Process_TrainFile(trainfile):
46
47      trainList = open(trainfile,'r').read() #read the file and remove the empty lines.
48      name_list = []
49      train_fea_dic = defaultdict(list) # use default dictionary as train file feature
50      dictionary
51
52      for line in trainList.split("\n\n"):
53
54          if("<class" in line):
55              text = line.split("\n")
56              text1 = text[0]
57              text2 = re.sub('([.,!?( )])', ' ', text1)
58              text3 = re.sub('\s{2,}', ' ', text2)
59              value= text3.split(" ") #get the value from each line
60
61              while "" in value: #remove empty lines
62                  value.remove("")
63
64              for i in range(0, len(value)):
65                  # Feature 1: the value of the token
66                  train_fea_dic['Value'].append(value[i])
67
68                  # Feature 2: is token all uppercase?
69                  if(value[i].isupper):
70                      train_fea_dic['UpperCase'].append(1)
71                  else:

```

```

72         train_fea_dic['UpperCase'].append(0)
73
74     # Feature 3: does token start with capital?
75     if(value[i].istitle()):
76         train_fea_dic['Capital'].append(1)
77     else:
78         train_fea_dic['Capital'].append(0)
79
80     # Feature 4: length of token
81     train_fea_dic['Length'].append(len(value[i]))
82
83     # Feature 5: does the token consist only of numbers?
84     if(value[i].isnumeric()):
85         train_fea_dic['Numeric'].append(1)
86     else:
87         train_fea_dic['Numeric'].append(0)
88
89     # Feature 6: does token start with vowel?
90     if value[i] in 'aeiou':
91         train_fea_dic['Vowel'].append(1)
92     else:
93         train_fea_dic['Vowel'].append(0)
94
95     # Feature 7: length of left word < 4
96     if(len(value[i])<4):
97         train_fea_dic['Length<Four'].append(1)
98     else:
99         train_fea_dic['Length<Four'].append(0)
100
101     # Feature 8: if Right word contains "."
102     if('.') in value[i]):
103         train_fea_dic['Period_in_token'].append(1)
104     else:
105         train_fea_dic['Period_in_token'].append(0)
106
107
108
109
110     if text[1].startswith("<class"):
111         mid_content = middle_content("<class", ">", line)
112         content = mid_content.split("\n")
113
114         for num in range(0,len(content)):
115             if("id=" in content[num]):
116                 ID_value = content[num].split("=")[1]
117             if("name=" in content[num]):
118                 name_list = content[num].split("=")[1].split(" ")
119
120
121         for i in range(0, len(value)):
122             for j in range(0,len(name_list)):
123                 if(value[i] == name_list[j]):
124                     if(j==0):
125                         value[i]=name_list[j]+("/B")
126                     elif(j>0):
127                         value[i]=name_list[j]+("/I")
128             if(value[i] == ID_value):
129                 value[i]=value[i]+("/B")
130
131         else:
132             value[i]=value[i]+("/O")
133         train_fea_dic['IOB'].append(value[i])
134
135
136 train_data = pandas.DataFrame.from_dict(train_fea_dic)
137 train_data['Value'] = train_data.index
138 train_data['IOB'] = train_data['IOB'].map(IOB_to_Num)
139
140 return train_data
141
142
143 # 4) Process Test File to get test data

```

```

144 def Process_TestFile(testfile):
145     testList = open(testfile, 'r').read() #read the file and remove the empty lines.
146     name_list = []
147     test_fea_dic = defaultdict(list) # use default dictionary as test file feature
        dictionary
148
149     for line in testList.split("\n\n"):
150
151         if("<class" in line):
152             testtext = line.split("\n")
153             testtext1 = testtext[0]
154             testtext2 = re.sub('([.,!?(())]', ' ', testtext1)
155             testtext3 = re.sub('\s{2,}', ' ', testtext2)
156             testvalue = testtext3.split(" ") #get the value from each line
157
158
159             while "" in testvalue: #remove empty lines
160                 testvalue.remove("")
161
162             for i in range(0, len(testvalue)):
163                 # Feature 1: the value of the token
164                 test_fea_dic['Value'].append(testvalue[i])
165
166                 # Feature 2: is token all uppercase?
167                 if(testvalue[i].isupper):
168                     test_fea_dic['UpperCase'].append(1)
169                 else:
170                     test_fea_dic['UpperCase'].append(0)
171
172                 # Feature 3: does token start with capital?
173                 if(testvalue[i].istitle()):
174                     test_fea_dic['Capital'].append(1)
175                 else:
176                     test_fea_dic['Capital'].append(0)
177
178                 # Feature 4: length of token
179                 test_fea_dic['Length'].append(len(testvalue[i]))
180
181                 # Feature 5: does the token consist only of numbers?
182                 if(testvalue[i].isnumeric()):
183                     test_fea_dic['Numeric'].append(1)
184                 else:
185                     test_fea_dic['Numeric'].append(0)
186
187                 # Feature 6: does token start with vowel?
188                 if testvalue[i] in 'aeiou':
189                     test_fea_dic['Vowel'].append(1)
190                 else:
191                     test_fea_dic['Vowel'].append(0)
192
193
194                 # Feature 7: length of left word < 4
195                 if(len(testvalue[i])<4):
196                     test_fea_dic['Length<Four'].append(1)
197                 else:
198                     test_fea_dic['Length<Four'].append(0)
199
200
201                 # Feature 8: if Right word contains "."
202                 if('.') in testvalue[i]:
203                     test_fea_dic['Period_in_token'].append(1)
204                 else:
205                     test_fea_dic['Period_in_token'].append(0)
206
207
208             if testtext[1].startswith("<class"):
209                 mid_content = middle_content("<class", ">", line)
210                 content = mid_content.split("\n")
211
212
213                 for num in range(0, len(content)):
214                     if("name=" in content[num]):

```

```

215         name_list = content[num].split("=")[1].split(" ")
216
217         if("id=" in content[num]):
218             ID_value = content[num].split("=")[1]
219
220         for i in range(0, len(testvalue)):
221             for j in range(0, len(name_list)):
222                 if(testvalue[i] == name_list[j]):
223                     if(j==0):
224                         testvalue[i]=name_list[j]+("/B")
225                     elif(j>0):
226                         testvalue[i]=name_list[j]+("/I")
227                 if(testvalue[i] == ID_value):
228                     testvalue[i]=testvalue[i]+("/O")
229
230             else:
231                 testvalue[i]=testvalue[i]+("/O")
232             test_fea_dic['IOB'].append(testvalue[i])
233
234     test_data = pandas.DataFrame.from_dict(test_fea_dic)
235     test_data['Value'] = test_data.index
236     test_data['IOB'] = test_data['IOB'].map(IOB_to_Num )
237
238     return test_data, test_fea_dic
239
240 # 5) Calculate The Accuracy and Generate Output
241 def accuracy(train_data, test_data, test_fea_dic, output):
242
243     features = ['Value', 'UpperCase', 'Capital', 'Length', 'Numeric', 'Vowel',
244                 'Length<Four', 'Period_in_token']
245     X = train_data[features]
246     Y = train_data['IOB']
247     test_X = test_data[features]
248     test_Y = test_data['IOB']
249     X_train, X_test, Y_train, Y_test = train_test_split(X, Y)
250
251     # Create Decision Tree classifier object and get the predict value
252     predict = DecisionTreeClassifier().fit(X_train, Y_train).predict(test_X)
253
254     # Calculate and print the accuracy score
255     accuracy = accuracy_score(test_Y, predict)
256     print("Accuracy:", accuracy)
257
258     for i in range(0, len(predict)):
259         output.write('\n')
260         if(predict[i] == 0):
261             output.write(str(test_fea_dic['Value'][i]))
262             output.write('/O')
263         elif(predict[i] == 1):
264             output.write(str(test_fea_dic['Value'][i]))
265             output.write('/I')
266         else:
267             output.write(str(test_fea_dic['Value'][i]))
268             output.write('/B')
269
270     output.close()
271     return accuracy
272
273
274
275
276 if __name__ == "__main__":
277
278     # 1) load file
279     trainfile = sys.argv[1]
280     testfile = sys.argv[2]
281
282     # 2) create outputfile
283     outputName = testfile + ".out"
284     output = open(outputName, "w")

```

```
285
286 # 2) Process Train File to get train data
287 train_data= Process_TrainFile(trainfile)
288
289 # 3) Process Test File to get test data
290 test_data,test_fea_dic = Process_TestFile(testfile)
291
292 # 4) generate output and calculate accuracy
293 accuracy(train_data, test_data,test_fea_dic, output)
294
```