ADS HW1 Report

404261476 資工四甲 楊培澤

Problem : Compound Words

Solution :

Using Trie to solve this problem. First, create Trie nodes for every character in a single word. And then for each word, split it in every index and search the split results in Trie. If the two split results are found in Trie, then the word is a compound word in this case.

Discussion :

Simple question but it is not specific that the capital character and non-capital character should be the same word or not. (it doesn't change the result in this case.)

Code :

```python
class TrieNode:
    def __init__(self):
        self.child = [None]*1000
        self.isEndOfWord = False

class Trie:
    def __init__(self):
        self.root = self.getNode()

    def getNode(self):
        return TrieNode()

    def insert(self, key):
        leaf = self.root
        for i in range(len(key)):
            idx = ord(key[i])
            if not leaf.child[idx]:
                leaf.child[idx] = self.getNode()
            leaf = leaf.child[idx]
        leaf.isEndOfWord = True

    def search(self, key):
        leaf = self.root
        for i in range(len(key)):
            idx = ord(key[i])
            if not leaf.child[idx]:
                return False
            leaf = leaf.child[idx]
        return leaf != None and leaf.isEndOfWord
```

```python
if __name__ == '__main__':
    # dir
    input_dir = './data/Compound Words.in'
    output_dir = './data/Compound Words.out'

    # create Trie
    dictionary = Trie()

    # insert words
    for line in open(input_dir, 'r').readlines():
        key = line.strip()
        dictionary.insert(key)

    # doc
    result = []

    # search Compound Words
    for line in open(input_dir, 'r').readlines():
        word = line.strip()
        for i in range(1, len(word)):
            sub_first = word[0:i]
            sub_second = word[i:len(word)]
            if dictionary.search(sub_first) and dictionary.search(sub_second):
                result.append(word)
                break
```

```python
# check accuracy
ground_truth = open(output_dir, 'r').readlines()
if len(result) != len(ground_truth):
    print('Result count inconsist')
else:
    count = 0
    for i in range(len(result)):
        word = ground_truth[i].strip()
        if result[i] == word:
            count += 1
    if count == len(result):
        print('Match output file')
    else:
        print('Some error in result')

# make output file
filename = ''.join(['./', 'output.txt'])
with open(filename, 'w') as f:
    for i in range(len(result)-1):
        f.write("%s\n" % result[i])
    f.write("%s" % result[len(result)-1])
print('Output predict file...')
```