CS1003 W05 Practical 170025298 22 Feb 2018

Abstract:
This report about week 5 practical that aiming at show the similarity of input arguments. The similarity with be shown as jaccard index and will shows as format:
“jaccard index” + “Result”

There are 2 classes that provided in practicals, I added W05-Practical and changes mainly available at ScoredResult.

Design:
W05Practical class is the main method, this method throws IO Exception and its sub-Exception File not found exception. I used if command to make sure that there are 2 arguments available. If not, the program will return Usage.
Try and catch is used to prevent error that might happens when there is no such file or java issue. The error will be shown to tell users where was wrong.

```
    if(args.length != 2){
        System.out.println("Usage: W05Practical <File> <Query>");
        System.exit( status: 1);
    }
    try {
        SentenceReader sentenceReader = new SentenceReader(args[0],args[1]);
    }
    catch (FileNotFoundException e){
        System.out.println("File not found: " + e.getMessage());
    }
    catch (IOException e){
        System.out.println("IO Exception: " + e.getMessage());
    }
}
```

Scored result class haven’t changed.

Main changed happened in Sentence reader class: One constructor has been add and 4 methods have been added. Parameter provided by practical has deleted in readAllSentence method and all methods have been set to private.

-list: readAllSentence():

```
private List<String> readAllSentences() throws IOException {

    String file = "";
    String line = "";
    List<String> list = new ArrayList<>();

    BufferedReader reader = new BufferedReader(new FileReader(filePath));
    while ((line = reader.readLine()) != null) {
        file += " " + line;
    }
      file = sanitiseSentence(file);
    String[] fields = file.split( regex: "\\.");


    for (String s : fields) {
        list.add(sanitiseSentence(s));
          System.out.println(s);
    }
    //System.out.println(list.get(11));
      sc.close();

    return list;
```

This method uses buffered reader to read all sentence from file. I was eager to use scanner but I found that it's a little bit hard for me. The similar condition happens to me while using buffered reader method: to split ".", I am required to use "\\.", this gave me a lot troubles but have been dealt by teachers' help.
The design of this method is to put all the text in a String and split it as arrays. All elements in the array will be added into the list and will be called by scoreCalculate method.

-List: Biagram():
This method uses substring to divide the word by using for loop.

```
    private List<String> biagram(String sectence) throws IOException {
        List<String> biagram = new ArrayList<>();
        for (int i = 0; i < sectence.length() - 1; i++) {
            biagram.add(sectence.substring(i, i + 2));
//              System.out.println(biagram.get(i));
        }
        return biagram;
    }
```

The biagram will be returned as lists.

-List: Score Calculate:
This method aims at calculate the score by using biagram method. I putted the design in java doc as following

```
There are orders in list but there is no order in set.
* While comparing the biagram, I put all the sentence in
the list but put the biagram into sets
* Then I compare words between two sets
* The result is like the following
* One thing puzzeled me a lot is sanitise and I have been
stuck in it for a long time
* I ignored to sanitise the input word and caused a lot of
mistakes
```

```
    for (String s : sentences) {
        score = 0;
        sentenceBiagram.addAll(biagram(s));
        totalBiagram.addAll(sentenceBiagram);
        totalBiagram.addAll(queryList);

        for(String s1 : queryList){
            if(sentenceBiagram.contains(s1)){
                score += 1;
            }
        }
        //System.out.println(totalBiagram);
        //System.out.println();
        results.add(new ScoredResult<>(s, score: score / totalBiagram.size()));

        sentenceBiagram.clear();
        totalBiagram.clear();
    }
    return results;
```

For loop is used to calculate the jaccard index.

-void:showResult():
Because of the decimal is required in 4 decimal number, I use Decimal Format to do so. To prevent there might be text have less than 50 “.”, result().size and if is used to prevent it
To make the accuracy, I set the rounding mode as halp up to make the decimal upper 5 will be up to 10

```
private void showResults() throws IOException{

    DecimalFormat df = new DecimalFormat( pattern: "0.0000");
    df.setRoundingMode(RoundingMode.HALF_UP);

    List<ScoredResult<String>> results;
    results = scoreCalculate();

    int num = 50;

    Collections.sort(results);
    if(results.size() < 50){
        num = results.size();
    }

    for (int i = 0; i < num; i++) {
        System.out.println(df.format(results.get(i).getScore()) + " " + results.get(i).getResult());
    }

}
//calculate the score
```

At the end of the method, System.out.println() is used to print the message.

Testing:

```
pc5-025-l:~/Documents/cs1003/W05-Practical/src hl74$ javac *.java
pc5-025-l:~/Documents/cs1003/W05-Practical/src hl74$ stacscheck /cs/studres/CS1003/Practicals/W05/Tests/
Testing CS1003 Week 5 Practical
- Looking for submission in a directory called 'src': Already in it!
* BUILD TEST - public/build : pass
* TEST - public/alice/01/test : pass
* TEST - public/alice/02/test : pass
* TEST - public/alice/03/test : pass
* TEST - public/alice/04/test : pass
* TEST - public/alice/05/test : pass
* TEST - public/great/01/test : pass
* TEST - public/great/02/test : pass
* TEST - public/great/03/test : pass
* TEST - public/great/04/test : pass
* TEST - public/great/05/test : pass
* TEST - public/top10/good-morning/test : pass
* TEST - public/top10/success/test : pass
13 out of 13 tests passed
```

Stacscheck have passed and I have also used argument as alice to test “alice.txt”

Evaluation:
I think I have programmed a neat and tidy code. Methods are used closely and codes is much lower than I imagined. However, scanner might be a better way to scan the “.” in text and buffered reader might increased the working hardness and made the time of reading file longer.
Some new classes might be added to lower the method in only one class.
I am unable to understand the compareTo method in scored result method.

Conclusion:
During the process of programming the program, many difficulties have been met such as the lower-case and upper-case of query and I should be more familiar to programming. However, but check the program once and once again and compare it with expect output in practical in studres, All drawbacks have been solved and I have finally done well.
If I could gain more time, I will try to add some functions such as scan the file automatically or make the result outputted into file.
In conclusion, I think this is a success program.

Extensions will be made and will be reported at the following page:

Extension 1: n_gram choice:

This is the first extension that make people can choose the number of n in n-gram.

Input stream reader is used to achieve the goal:

```
InputStreamReader isr = new InputStreamReader(System.in);
BufferedReader br = new BufferedReader(isr);
public SentenceReader(String filepath, String query) throws IOException {
    this.filePath = filepath;
    this.query = query;
    System.out.println("Please enter the number of gram: ");
    n_gram = Integer.parseInt(br.readLine());
```

The value of n is given in constructor and is called in biagram:

```
private List<String> biagram(String sectence) throws IOException {
    List<String> biagram = new ArrayList<>();
    for (int i = 0; i < sectence.length() - n_gram - 1; i++) {
        biagram.add(sectence.substring(i, i + n_gram));
          System.out.println(biagram.get(i));
    }
    return biagram;
}
```

Extension 2: word based choice:

This Extension uses scanner to achieve the goal.

Because the word is divided by space, I use scanner to divide all the word with space.

```
private List<String> biagram(String sectence) throws IOException {
    List<String> biagram = new ArrayList<>();
      for (int i = 0; i < sectence.length() - n_gram - 1; i++) {
          biagram.add(sectence.substring(i, i + n_gram));
          System.out.println(biagram.get(i));
      }
    Scanner sc = new Scanner(sectence);
    while (sc.hasNext()) biagram.add(sc.next());
    return biagram;
}
```

The result is as follow:

```
/usr/lib/jvm/java-1.8.0-openjdk/bin/java ...
0.3333 alice sighed wearily
0.3333 alice was silent
0.3333 yes shouted alice
0.3333 nothing said alice
0.2500 exactly so said alice
0.2500 alice felt dreadfully puzzled
0.2500 what for said alice
0.2500 alice was thoroughly puzzled
0.2500 nothing whatever said alice
0.2500 everybody looked at alice
```

which uses alice.txt in practical and the second argument for “alice”

Extension 3: search multiple file with one one query

```
InputStreamReader isr = new InputStreamReader(System.in);
BufferedReader br = new BufferedReader(isr);
int i = 0;
while (i == 0) {
    SentenceReader sentenceReader = new SentenceReader(args[0], args[1]);
    System.out.println("Please select the next file (exit for 1) :");
    String nextFile = br.readLine();
    switch (nextFile){
        case "1" : {
            i += 1;
        }
        default:{
            args[0] = nextFile;
        }
    }
}
br.close();
isr.close();
```

As you can see from the picture above, there is a switch that was created and to replace the argument.

```
0.1111 then you keep moving round i suppose said alice
0.1111 alice waited till the eyes appeared and then nodded
0.1111 i never saw one or heard of one said alice
0.1111 what fun said the gryphon half to itself half to alice
0.1111 i never heard of uglification alice ventured to say
Please select the next file (exit for 1) :
1
Thank you for using this software!
```

While input 1 , the program will exit with polite System.out.println();

Thanks for looking at my extensions!