DALHOUSIE UNIVERSITY
Inspiring Minds

Fall 2021 (Sep7-Dec7)
Faculty of Computer Science

# CSCI 4152/6509 — Natural Language Processing

## Assignment 2

---

**Due**: *Friday, Oct 22, 2021 by midnight*
**Worth**: 128 marks (= 30 + 22 + 26 + 25 + 25)
**Instructor**: Vlado Keselj, CS bldg 432, 902.494.2893, vlado@cs.dal.ca vlado@dnlp.ca

---

**Assignment Instructions**:

The submission process for Assignment 2 is mostly based on the `submit-nlp` command on `timberlea` as discussed in the lab, or in the equvalent way by using the course web site, where you need to follow 'Login' and then the 'File Submission' menu option.

**Important:** You must make sure that your course files on timberlea are **not** readable by other users. For example, if you keep your files in the directory `csci6509` or `csci4152` you can check its permission using the command:

```
ls -ld csci6509
```

or

```
ls -ld csci4152
```

and the output must start with `drwx------`. If it does not, for example if it starts with `drwxr-xr-x` or similar, then the permissions should fixed using the command:

```
chmod 700 csci6509
```

or

```
chmod 700 csci4152
```

**1)** (30 marks) Complete the Lab 2 as instructed. In particular, you will need to properly:

a) (5 marks) Submit the file '`lab2-matching.pl`' as instructed,

b) (5 marks) Submit the file '`lab2-matching-data.pl`' as instructed,

c) (5 marks) Submit the file '`lab2-word-counter.pl`' as instructed,

d) (5 marks) Submit the file '`lab2-replace.pl`' as instructed, and

e) (5 marks) Submit the file '`lab2-ngram-output.txt.gz`' as instructed.

f) (5 marks) Submit the file '`lab2-line-count.pl`' as instructed.

Note that any program that you submit needs to compile. Even if complete source code is given in the lab, you need to type it instead of using cut-and-paste, and you need to make sure not to introduce any errors into the program. This follows from the lab instructions that programs must be tested before submitting.

**2)** (22 marks) Complete Lab 3 as instructed. In particular, you will need to properly:

a) (3 marks) Submit the example file '`lab3-array-examples.pl`' as instructed.

b) (3 marks) Submit the example file '`lab3-test-hash.pl`' as instructed.

c) (5 marks) Submit the program file '`lab3-letter_counter_blanks.pl`' and the output file '`lab3-out_letters.txt`' (or `lab3-out_letters.txt.gz`) as instructed.

d) (6 marks) Submit the program '`lab3-word_counter.pl`' and the output file '`lab3-out_word_counter.txt`' as instructed.

e) (5 marks) Submit the program '`lab3-word_counter2.pl`' as instructed (Step 6).

Note that any program that you submit needs to compile. Even if a complete source code is given in the lab, you need to type it instead of using cut-and-paste, and you need to make sure not to introduce any errors into the program. This follows from the lab instructions that programs must be tested before submitting.

**3)** (26 marks) Complete Lab 4 (GitLab and Git) as instructed. In particular, you will need to properly:

a) (5 marks) Prepare and submit via Git the file `README.md` as instructed.

b) (5 marks) Prepare and submit via Git the directory `lab5g` and your public key `id_rsa.pub` as instructed.

c) (6 marks) Prepare and submit via Git the files `explore.pl` (commits for version 1.0 and 1.1) and the Hamlet file as instructed.

d) (5 marks) Create the branch `ada-main-program` with required commits and merged later into `master` as instructed.

e) (5 marks) Create the branch `bob-function-explore` with required commits and merged later into `master` as instructed.

**4)** (25 marks) Your solution should be in a file named `a2q4.pdf`, `a2q4.jpg`, or `a2q4.txt` and it should be submitted the `submit-nlp` command. If you need to submit more JPG files, then you should name them: `a2q4-1.jpg`, `a2q4-2.jpg`, `a2q4-3.jpg`, and so on.

Your solution can be submitted as a plain-text, PDF, or image JPG file. The parts b) and c) require drawing of the curves, which you can create in any software that you like, or even draw it by hand using a ruler and take picture of it. It should be clear to read and verify that it is correct.

Suppose that a search engine returned 16 ranked results to our query. We checked those results and the following are out judgements on their relevance:
1. relevant
2. relevant

3. not relevant
4. relevant
5. relevant
6. not relevant
7. relevant
8. relevant
9. relevant
10. not relevant
11. not relevant
12. relevant
13. not relevant
14. not relevant
15. relevant
16. not relevant

Assuming that the total number of relevant documents in the collection is 25, do the following tasks:

a) (5 marks) Calculate precision, recall, and F-measure for the returned results.

b) (10 marks) Draw the Precision-Recall curve for these results. Show how the appropriate coordinates were calculated.

c) (10 marks) Draw the interpolated precision curve. Show how the appropriate coordinates were calculated.

**5)** (25 marks, programming) Write and submit a program written in Perl, Python, C, C++, or Java, named `a2q5.pl`, `a2q5.py`, `a2q5.c`, `a2q5.cc`, or `a2q5.java`, which calculates Precision, Recall, and F-measure per class, and also macro-averaged Precision, Recall, and F-measure.

The program must read the standard input, and it must print to the standard output. Each line of input contains one result of a classification experiment, and the last line contains the word '`end`'. Each line containing an experiment result has the format:
`true label:`$C_1$ `result:`$C_2$
where $C_1$ and $C_2$ are non-empty strings of non-whitespace characters, and $C_1$ is the true label of a document, and $C_2$ is the classification result of the same document.

The following is an example of such input:

```
true label:A result:A
true label:A result:A
true label:A result:A
true label:B result:B
true label:B result:B
true label:C result:C
```

```
true label:A result:B
true label:B result:C
true label:A result:B
end
```

If we construct a confusion matrix of these results, we would get:

| | Gold standard | | | |
|---|---|---|---|---|
| | $A$ | $B$ | $C$ | |
| $A$ | 3 | 0 | 0 | 3 |
| $B$ | 2 | 2 | 0 | 4 |
| $C$ | 0 | 1 | 1 | 2 |
| | 5 | 3 | 1 | 9 |

and we can see that per-class Precision, Recall, and F1-measure are:
$P(A) = 3/3 = 1, \quad R(A) = 3/5 = 0.6, \quad F_1(A) = 0.75$
$P(B) = 2/4 = 0.5, \quad R(B) = 2/3 \approx 0.66667, \quad F_1(A) = 0.57143$
$P(C) = 1/2 = 0.5, \quad R(C) = 1/1 = 1, \quad F_1(A) = 0.66667$

Finally, we can calculate the macro-averaged Precision and Recall, and F-measure based on them as follows:

$$P = \frac{P(A) + P(B) + P(C)}{3} \approx 0.66667,$$

$$R = \frac{R(A) + R(B) + R(C)}{3} = \frac{3/5 + 2/3 + 1}{3} \approx 0.75556$$

$$F_1 = \frac{2 \cdot P \cdot R}{P + R} \approx 0.70833$$

you can notice that we rounded all values to up to five decimals. Your program must calculate and print out these values rounded to five decimals following the following format:

```
P(A)=1.00000 R(A)=0.60000 F1(A)=0.75000
P(B)=0.50000 R(B)=0.66667 F1(B)=0.57143
P(C)=0.50000 R(C)=1.00000 F1(C)=0.66667
P=0.66667 R=0.75556 F1=0.70833
```

A few additional notes to have in mind when implementing your solution are:

- Your program must collect classes based on the input, and in the output precision, recall and F1 measure per class must be printed in alphabetical order of the classes. The number of classes will always be at least 2.
- It is guaranteed that the input will be in a valid format, and there will be at least one 'result' line in input, before ending with the 'end' line.

- Follow the output format preciselly; for example, round the results with exactly 5 decimals, using trailing zeros if needed. Use no spaces around equal signs in the output, and use exactly one space between printed results as shown.
- It is possible that during computation, there appears division by 0. Since this is not defined, the program would stop because of an error by default, but your program must treat division by zero as having the result zero, and continue execution. This is not mathematically correct, but it makes sense to treat is as zero if this appears in precision calculation, for example, so we will adopt this rule.

The above sample input and output will be made available in the assignment directory on the course web site, and on `timberlea` in the directory: `~prof6509/public/a2`