Information Retrieval

NUS SoC, AY 2019/20, Semester II, Fridays 12:00-14:00 @ LT15

Last updated: Thursday, January 24, 2019 - Information updated for AY19/20 Semester 2.

Deadline for submission: **9 Feb 2020, 10pm SGT**

Homework #1 » Language Detection

In this homework, you will be implementing a language detection module using the ngram knowledge you learned in Week 1. Given a string representing some natural language utterance, your program should be able to predict whether the text is Indonesian, Malaysian or (phonetically transcribed into English) Tamil. So given the following three strings:

Semua manusia dilahirkan bebas dan samarata dari segi kemuliaan dan hak-hak.
Semua orang dilahirkan merdeka dan mempunyai martabat dan hak-hak yang sama.
Maitap piiviyiar cakalarum cutantiramkav piakkiaar

... an ideal program should output/predict the following labels for the strings:

malaysian	*Semua manusia dilahirkan bebas ...*
indonesian	*Semua orang dilahirkan merdeka ...*
tamil	*Maitap piiviyiar cakalarum cutantiramkav piakkiaar ...*

Build and test your language models (LMs)

You should run your program to build and test your LMs in this format:

```
$ python3 build_test_LM.py -b input-file-for-building-LM -t input-
```

where `input-file-for-building-LM` is a file given to you that contains a list of strings with their labels for you to build your ngram language models, `input-file-for-testing-LM` is a file containing a list of strings for you to test your language models, and `output-file` is a file where you store your predictions.

Evaluate your predictions

To evaluate the accuracy of your predictions, you run the evaluation file `eval.py` which is given to you:

```
$ python3 eval.py file-containing-your-results file-containing-cor
```

where `file-containing-your-results` is the output-file from the build and test step, and `file-containing-correct-results` is a file containing the correct string labels.

For example, in the homework package, you are given several files, including a skeleton `build_test_LM.py`, `eval.py`, `input.train.txt`, `input.test.txt`, and `input.correct.txt`. To build and test your LMs, run:

```
$ python3 build_test_LM.py -b input.train.txt -t input.test.txt -o
```

which will store your predictions in `input.predict.txt`. To evaluate your predictions, run:

```
$ python3 eval.py input.predict.txt input.correct.txt
```

which prints the accuracy of your predictions.

What's expected in `build_test_LM.py`?

The python program `build_test_LM.py` is given to you as a skeleton script. You are required to complete this script by implementing the `build_LM()` and `test_LM()` functions.

You need collect the 4-grams from the string where the gram units are characters. For example, for the string *Semua manusia dilahirkan bebas ...*, the below character 4-grams would be collected (Note that you can choose to pad the beginning and end of the string as shown in lecture 1; it's up to you, but you'll want to document your choice).

[('S', 'e', 'm', 'u'), ('e', 'm', 'u', 'a'), ('m', 'u', 'a', ' '), ('u', 'a', ' ', 'm'), ('a', ' ', 'm', 'a'), (' ', 'm', 'a', 'n'), ('m', 'a', 'n', 'u), ('a', 'n', 'u', 's'), ...]
For each of the malaysian, indonesian and tamil labels, you then build a language model with add one smoothing, similar to the ones shown in lecture 1, which smooths out all observed ngrams. The differences are that you are required to use probabilities instead of counts and 4-grams instead of unigrams. Your language models for the three labels should look in a way similar to the following table, where rows 3 to 4 are the language models for malaysian, indonesian, and tamil, respectively. Note that each row should sum up to 1, and there are other entries in the table that have been omitted for clarity.

Labels	4-grams

	...	('e','m','u','a')	('m','u','a',' ')	('u','a',' ','m')	...
malaysian		3.11e-07	1.03e-07	2.07e-07	
indonesian		5.17e-07	2.07e-07	1.52e-07	
tamil		2.89e-06	5.17e-07	9.65e-07	

To test a new string, you should multiply the probabilities of the 4-grams for this string, and return the label (i.e., malaysian, indonesian, and tamil) that gives the highest product. Ignore the four-gram if it is not found in the LMs.

Format of the input/output files

- `input.train.txt:` the input file for you to build your LMs, which contains about 900 lines, where each line is a label/string pair separated by a space. The label will either read malaysian, indonesian, and tamil.
- `input.test.txt:` the input file for you to test your LMs, which contains 20 strings, each in a line. A small number of strings have been inserted by extraterrestrial aliens, and do not belong to any of the three languages your program is asked to detect. This is just to keep it interesting; you can tell us what language you think they are, and see whether you can tune your program to tell us that they are other languages.
- `input.correct.txt:` contains the correct labels for the input in `input.test.txt`, in the same format as `input.train.txt` (i.e., each label/string pair is separated by a space).
- Your output file from `build_test_LM.py` (e.g., `input.predict.txt`): contains your predictions in the same format as `input.train.txt`.

Essay questions:

You are also encouraged to think about the following essay questions as you work on the assignment. These questions are meant to enhance your understanding of the lecture materials and the assignment. You are NOT required to submit your answers, but you are welcome to do so and discuss your answers with us. Note that these are open-ended questions and do not have gold standard answers. A paragraph or two are usually sufficient for each question. It would be even better if you can support your answers with experimental results.

1. In the homework assignment, we are using character-based ngrams, i.e., the gram units are characters. Do you expect token-based ngram models to perform better?
2. What do you think will happen if we provided more data for each category for you to build the language models? What if we only provided more data for Indonesian?
3. What do you think will happen if you strip out punctuations and/or numbers? What about converting upper case characters to lower case?
4. We use 4-gram models in this homework assignment. What do you think will happen if we varied the ngram size, such as using unigrams, bigrams and trigrams?

Submission Formatting

For us to grade this assignment in a timely manner, we need you to adhere strictly to the following submission guidelines. They will help me grade the assignment in an appropriate manner. You will be penalized if you do not follow these instructions. Your student number in all of the following statements should not have any spaces and all letters should be in CAPITALS (inclusive of the ending check letter). You are to turn in the following files:

- A plain text documentation file `README.txt` (e.g., in UPPERCASE): this is a text only file that describes any information you want me to know

about your submission. You should give an overview of your program and describe the important algorithms/steps in your program. However, You should not include any identifiable information about your assignment (your name, phone number, etc.) except your student number and email (we need the email to contact you about your grade, please use your *[u|a|g]*******@u.nus.edu* address, not your email alias). This is to help you get an objective grade in your assignment, as we won't associate student numbers with student names.
- Any source code (in particular, `build_test_LM.py` for this homework assignment): We will be reading your code, so please do us a favor and format it nicely.
- (Optional) A plain text file `ESSAY.txt` that contains your answers to the essay questions (if you choose to attempt the questions). Again, do not disclose any identifiable information in your essay answers.

These files will need to be suitably zipped in a single file called `<student number>.zip`. Please use a zip archive and not tar.gz, bzip, rar or cab files. Make sure when the archive unzips that all of the necessary files are found in a directory called `<student number>`.

A package of skeleton files have been released in LumiNUS to help you prepare the submission. You may use the homework checker script in the same package to check whether your zip archive fulfills the criteria above. Upload the resulting zip file to LumiNUS by the due date: 9 Feb 2020, 10pm SGT. There absolutely will be no extensions to the deadline of this assignment. Read the late policy if you're not sure about grade penalties for lateness.

Grading Guidelines

The grading criteria for the assignment is tentatively:

- 50% Correctness of your code

- 20% Documentation
  - 5% For following the submission instructions and formatting your documentation accordingly.
  - 5% For code level documentation.
  - 10% For your high level documentation, in your README document.
- 30% Evaluation results: Your program will be tested against 200 new strings. The strings will be in the same format as the provided `input.test.txt` (may have some alien strings too).
- 0% Essay questions
- Note: Nominal bonus marks (+1) might be given to submissions that excel in the above-mentioned components.

Miscellaneous

- Language Identification (from Wikipedia)
- It's well known that Indonesian and Malaysian are closer to dialects of the same language (Malay), rather than independent languages. So many of the strings you'll get in training and testing will be likely be confused by your program, since they may be acceptable in both languages. After all,

  > A language is a dialect with an army and navy Max Weinreich -- see (Wikipedia)

  So don't expect your program to distinguish these two languages well.