SNLP assignment 1: creating a Twitter corpus

Deadline: May 09, 2018

Almost all natural language processing tools and applications rely on corpora. As a result, the quality and usability of NLP applications depend on the availability and quality of corpora. From choice of the linguistic material to linguistic processing, building the right corpus is a difficult task. You will be practicing with a small part of this task, by crawling Twitter and building a small corpus in Python.

Each class participant is required to collect a small corpus for a different language. Collectively, we will create a small multi-lingual corpus.

Note that the collection process may take some time – maybe day(s) – depending on the method and the language. As a result, you may fail to submit your exercise on time if you do not start early.

Before proceeding, please read the general instructions in the course syllabus.

Why am I doing this?

- Improve your python knowledge/experience
- Collect linguistic data on the Internet
- Deal with storage and distribution of the corpora

Exercise 1. *Choose a language*

Choose a language and register your choice by editing the table at `https://github.com/snlp2018/a1-common`.[1]

The rules are:

- Each participant or group is *required* to choose a different language from the rest of the participants.[2]
- First come, first served. If the language you are interested in is already taken, pick another one.
- Prefer a *low-resource* language (with a reasonable presence in Twitter). Although we do not have strict rules for choosing the language, you are discouraged to choose a well-studied (European) language.
- The knowledge of the language(s) chosen is not necessary, but it may help with some of the tasks below.

[1] Read the whole exercise sheet before you make your choice. If you do not have a preference, or your favorite languages are all taken, you can have a look at `https://en.wikipedia.org/wiki/List_of_ISO_639-2_codes` for inspiration.

[2] You are welcome to choose two languages, especially you are doing this exercises with another participant. However, this is not a requirement.

Exercise 2. *Download tweets*

Write a Python program using Twitter streaming API with tweepy library, to obtain 10 000 tweets in the language you have chosen.

Rules/Tips:

- Do not store the tweets shorter than 50 (Unicode) characters.[3]
- You should try to be fairly sure that the tweets you store are in the language you are working with. For this purpose, you can use a language-detection software.[4]
- You will need a Twitter account, and you need to create access keys for this purpose.[5] There are many tutorials on the Internet on how create an application connected to a Twitter account, and how to get the necessary access keys.

[3] You can use a lower threshold for languages with logogram-based writing systems (e.g., Chinese).

[4] You can use Python langdetect library. The default installation (e.g., with `pip`) comes with a number of languages. You can also add support for new languages using the data from, for example, Wikipedia (see the `langdetect` documentation).

[5] You should *never* store passwords, or secret keys in git repositories.

- Using Twitter API, there are at least the following three methods for obtaining tweets in a particular language.
  - The API allows you to specify coordinates of a rectangular area. If you are fairly sure about the location(s) that the language is spoken, you can set the location, and receive the stream from Twitter.
  - You can use geosearch to search tweets in a geopolitical region, or nearby a particular location.
  - You can also filter the incoming stream by a set of keywords ('seed words'). Keywords can be obtained from publicly available word-frequency lists,[6] Note that you want seed words that are frequent in the language of interest but also rare in other languages.
- Store the tweets in your private repository as a JSON file with the name ⟨`lang-id`⟩`.json`, where ⟨`lang-id`⟩ is the three-letter ISO code of the language.

[6] For example, `https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists`, or you can create one from Wikipedia.

Exercise 3. *Share your corpus – the right way*

- Write another program that reads the corpus, and outputs only the Tweet IDs.[7] Your output should be a *text file with a single tweet ID per line*.
- Store the twitter IDs in the collaborative repository `https://github.com/snlp2018/a1-common` with the name ⟨`lang-id`⟩`.id`, where, as before, ⟨`lang-id`⟩ is the three-letter ISO language code.

[7] We will use the IDs later. Twitter policy does not allow tweets to be distributed. A common trick in collaborative work on twitter corpora is to distribute the IDs, which can be fetched by the third parties later.

Exercise 4. *Document your approach*

Describe the approach you used for collectiong the data briefly in the `README.md` file in your repository. At a minimum, document the way you selected the correct language tweets, the resources you used for the task, and the 'yield rate' (the ratio of long-enough tweets in the target language to the total number of tweets your stream received).[8]

[8] Good programmers document their code, good Python programmer use docstrings. So, this exercise is not much more than copying the docstring from your main Python file to `README.md`.