CMPE 561 NATURAL LANGUAGE PROCESSING 09.11.2021
PROJECT #1

In this project, you will design and implement a preprocessing toolkit for Turkish. The system will take a document as input and will provide the following basic text processing functionalities:

- Tokenization
  The tokenizer should take into account special entities like URLs, hashtags, e-mails, MWEs (multi-word expressions), etc.
- Sentence splitting
  The sentence splitter should be able to detect different types of end-of-sentence markers. It should also handle complex cases such as the occurrence of end-of-sentence markers within quotations.
- Normalization
  The normalizer should be able to work on general domain (e.g. news domain), rather than being specific to some particular domains.
- Stemming
  The stemmer should take into account all inflectional suffixes and a wide range of derivational suffixes.
- Stopword elimination
  This operation should be for the general domain; not for the stopwords of a particular domain.

For the tokenization and sentence splitting processes, you need to build both a rule-based model and a machine learning-based model. That is, you will have two tokenizers and two splitters. You can combine them into a hybrid model if you wish, but they should also be able to operate independently. For the machine learning models, you will use Naïve Bayes and logistic regression (Naïve Bayes in one of the tasks (tokenization or splitting) as your choice, logistic regression in the other task) – other classical machine learning methods or deep learning-based methods are not allowed. You will design the feature sets for both of these tasks manually.

For the stopword elimination process, you need to employ both a static method (using a predefined list of stopwords) and a dynamic approach (building the list of stopwords dynamically from a corpus using some metrics).

During the design of these preprocessing operations, you will need to research on several topics such as effective feature sets for these tasks, inflectional and derivational suffixes of the language, dynamic stopword approaches, etc. You will also need to compile several resources like normalization lexicon, MWE lexicon, suffix lexicon, etc.

For Naïve Bayes models, you need to implement the classifier yourself. For logistic regression models, it is not necessary to implement a logistic regression classifier. Use a tool such as scikit-learn (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html), liblinear (https://www.csie.ntu.edu.tw/~cjlin/liblinear/), or another one. Test your machine learning models (both Naïve Bayes and logistic regression) with different feature sets for both tokenization and sentence splitting, and compare their success rates.

Documentation:

A progress report will be submitted on 23.11.2021 23:59 via e-mail. A progress report is an about 5 pages long documentation in free style stating the progress in the project. The purpose of this submission is to ensure that you have started working on the project and it will briefly explain what have been done up to that time. At the time of the report, nearly half of the project should have been completed.

Prepare a system paper as the final documentation of the project that describes your system. It will be in the form of a 5-6 pages long conference paper. A template is available on the course web page. Please follow the style in this example paper. The overleaf template for the system paper is also available on the course web page. You can copy it to your project.

Submit the program and the system paper via Moodle. The deadline for the submission is 07.12.2021 23:59.

Notes:

- As in the case of the paper presentations, the project will be done on a group basis.
- The deadlines are strict. There will be a grade reduction in the case of being late.
- The deadline of the project is four weeks past the announcement, which is sufficient to build a sophisticated system. Therefore, you are expected to build such a system.
- There will be a demo for each group. In addition to your test scenarios, we will also test your system on some other texts from general domain.
- You can use any programming language you wish.
- ***All the source codes must completely be written by you. You must explicitly cite all the resources collected from other sources.***