CS 445 Natural Language Processing

Project 1: Lexicon and Rule-based Named Entity Recognition

Due Date: November 15, 23:55

In this assignment, you are expected to develop a lexicon and rule-based named entity recognition system (NER) for Turkish. Your NER system should be able to identify the followings over a given Turkish text:

- Person
- Location
- Organization
- Date and Time

For an input file which consists of only the following two lines:

> Sabancı Üniversitesi 1999 yılında Prof. Dr. Tosun Terzioğlu kurucu rektörlüğünde İstanbul, Tuzla ilçesinde kurulmuştur.
> Şimdiki rektörü Prof. Dr. Yusuf Leblebici'dir.

Your NER system should output the following:

```
Line 1: PERSON Tosun Terzioğlu
Line 1: LOCATION İstanbul
Line 1: LOCATION Tuzla
Line 1: ORGANIZATION Sabancı Üniversitesi
Line 1: TIME 1999
Line 2: PERSON Yusuf Leblebici
```

For each line, you will initially output the line count (starting from 1 and incremented for each line). This will be followed by the type of the entity (either PERSON, LOCATION, ORGANIZATION or TIME). Finally, the entity itself will be outputted. All these will be space separated.

As you can see from the output, the order of entities within the same line is not important. It is based on your implementation, which is totally fine. Of course, in the output you should output the lines in the same order as the input file. For example, all the entities in the first line should be printed out before any entities in the second or third line.

Your NER system will consist of only rules and lexicons (a.k.a lists or gazeetters). You are not allowed to use machine learning approaches in this project.

- **Rules:** Your rules are going to be in the form of regular expressions. You will use Python's RE library to implement these rules. Several regular expressions are provided to you as examples. These are very simple and incomplete rules. You should modify these as well.

  We have provided a list of entities that you can use as guideline at the last page. Of course, these are not all possibilities. Based on the examples in this list you should write regular expressions which cover similar and additional ones as well. You should write at least 20 regular expressions to cover different cases and different types of entities. Writing similar regular expressions with only small variations is not allowed. In such cases, you will get partial grade. Remember the goal of this project is for you to practice regular expressions. So please practice.

- **Lexicons:** These are lists of entities. Especially retrieved from databases or online resources. These are used to cover person, location or organization entities. For example an example list for cities in Turkey is available here: https://tr.wikipedia.org/wiki/T%C3%BCrkiye%27nin_illeri

  There are different ways to create these lists. You can create them either manually or by writing some scripts, even crawlers. Of course, you can download the lists from online resources as well. Whatever you do, you should explain it clearly by citing the resource. If you use any script to create those lists, you should also submit that.

  As a lower boundary, you should have at least 25000 entries as locations, 2000 as organizations with at least 5 different categories (Bank, University, Company, etc) and 2000 person name entities. All these lists should be saved in files based on the type or subtype, and submitted.

In order to give you more examples, a small dataset will be shared with you. In this dataset PERSON, ORGANIZATION and LOCATIONs are tagged. A special tag is used to tag the entities. You don't have to worry about that special tag. This dataset is just an example where you can see how named entities are labeled. Please keep in mind that there can be some mistakes in this data. You can use this data but also please use your common sense as well. When you are not sure about how certain cases should be handled, you can look at this data. If you are still confused, you can ask to the instructor Dr. Yeniterzi.

While grading this assignment we will run your code over a test dataset. Therefore, you need to make sure to submit all necessary files (scripts, lists etc). Overall, you should submit the followings:

- **ner.py**: Your main script. We will use this to check your system.
- **lexicons (aka. lists, gazetteers):** Lists that you have created. There should be at least three lists: one for person, one for location and at least one for organization (you may have more depending on the type of organization). You ner.py script will read these files and put the entities in a data structure. Hard coding these entities in your script is not allowed.
- **scripts to create lexicons (optional):** Any script that you used to create these lists. Creating much larger lists will be useful for test cases.

- **README file:** This file is going to be a text file, which contains all the details regarding your project. You will describe your regular expressions in this file. You will also describe how you create the lexicons. If you use any online resource, you should cite it here. This file is going to be like your report for this project. So please be specific while writing this readme file.

All files should be under the same directory (named as your student ID, only the 5 digit numbers), which you will zip and submit.

We should be able to run your code with the following command:

```
python ner.py input_file_path > output_file_path
```

Therefore, your main script should be named as ner.py script. This script will do everything like reading the lists, searching for entities and outputting. This script should also contain the regular expressions. Other scripts are only there for creating the lists.

Please do not submit ipynb files. Make sure that your ner.py is human readable. Do not export from ipynb. You can do your developments in there but at the end make sure to copy paste them to a python file for submission. You need to make sure that your python script runs with the above command without any problems

You will use Python re library. Remember regular expressions can show variety based on the platform or programming language. So you are strongly recommended to look at re library's manual (https://docs.python.org/3/library/re.html). This module contains some other useful operators or functions as well. Make sure to check them out. You can use the popular/standard python packages. In case you are not sure of a particular library, please ask to the instructor or TA.

The output file should be in the exact same format described above. We will use automatic scripts to check your outputs. All your submissions will be automatically unzipped, tested and graded. So make sure to abide the project instructions carefully at each step. Otherwise, penalties will be applied.

An example starter script is shared with you. This is just a very simple example code with simple and incomplete regular expressions. It does not read from a file or output in the requested format. Please be aware of these differences.

You are expected to implement this project at your own. Your scripts, regular expressions, lexicons will be analyzed by using state-of-the-art tools for any type of plagiarism.

Your grade will depend on your scripts, regular expressions, lexicons and of course your performance on the test set. We have provided some minimum requirements to you. Anyone who satisfies all these will get a big percentage of the grade (assuming that they submitted everything as described here). But, keep in mind that this is an open-ended project. Therefore, if you do more you will have higher chances to get higher score from test cases. We won't be cruel when testing. However, we will try to be fair so that the ones who have done more and better work will get slightly better grades for their efforts. We will only use correctly written text for the testing purposes. Test cases won't include any misspellings.

Not all the projects are going to be open-ended. There will be two projects, which are going to have very specific set of instructions. You should use open-ended projects, like this one, as a way to collect more points. Instead of giving a bonus to ones who did exceptional work, that part will be integrated to the original grade over 100.

Example entity types that you should handle:

Please keep in mind that these are some example cases. When you see … it means there can be variations. Your regular expressions should handle multiple token entities and Turkish morphology. We will test some cases of morphology handling but don't worry we won't be cruel. Of course, when we say Apple, you may consider adding Microsoft etc. to your lexicon as well. Remember having more entities in your lexicon will be something useful for you. Take this list as a big hint of what we are expecting from you.

- Türkiye
- ABD
- TBMM
- Mustafa Kemal ATATÜRK
- Cumhurbaşkanı/Vali/Belediye Başkanı/… …
- … Futbol Takımı
- Prof. Dr. …
- … Bey/Hanım/Abi/Hoca/…
- Sayın/Sevgili/… …
- … Köyü/Dağı/Mahallesi/Sokak/….
- … Köprüsü/Sarayı/Mezarlığı/…
- Ankara'ya
- … Üniversitesi/Holding/Vakfı/Federasyonu/Enstitüsü/Kurumu/Bankası/…
- .. A.Ş/LTD/….
- Apple
- Türk Hava Yolları (THY)
- New York
- 01/01/2000 (Different type of seperators)
- 1 Ocak 2000
- Ocak 2000
- 2000 yılı
- Ocak ayı
- 2000'de

Submission Instructions:

- You will submit this homework via SUCourse.
- Please check the slides for the late submission policy.
- You can resubmit your homework (until the deadline) if you need to.
- Please read this document again before submitting.

- After submitting, you should download your submission to a different path to double check whether everything is in order.
- Please do your assignment individually, do not copy from a friend or the Internet. Plagiarized assignments will receive -100.