ASSIGNMENT 1: WRITTEN TO SPOKEN TEXT CONVERSION

Motivation: The motivation of this assignment is to get an insight into writing rule-based NLP engines. It will also, hopefully, showcase the sheer number of corner cases that arise in NLP, and, thus, the subsequent importance of data-driven machine learning systems.

Problem Statement: The goal of the assignment is to write a written to spoken text converter. The input of the code will be a set of tokenized sentences, and the output will be conversions where date, time, and numerical quantities are rewritten in the way they will be spoken.

At a high level, the guidelines for text conversion are as follows:

1. All abbreviations are separated as they are spoken. Example, “U.S.” or “US” is converted to “u s”.

2. All dates are converted into words. Example, “29 March 2012” will be converted to “the twenty ninth of march twenty twelve”. “2011-01-25” will be converted to “the twenty fifth of january twenty eleven”.

3. All times are converted into words. Example, “04:40 PM” is converted to “four forty p m”. “21:30:12” is converted to “twenty one hours thirty minutes and twelve seconds”

4. All numeric quantities are converted to words using Western numbering system. Example, “10345” is converted to “ten thousand three hundred forty five”. “24349943” is converted to “twenty four million three hundred forty nine thousand nine hundred forty three”. “184.33” is converted to “one hundred eighty four point three three”. “-20” is converted to “minus twenty”.

5. All units are spelled out as spoken. “53.77 mm” is converted to “fifty three point seven seven millimeters”, “3 mA” is converted to “three milliamperes”, and “14 sq m” is converted to “fourteen square meters”.

6. Currency is also spelled out. “$15.24” is converted to “fifteen dollars and twenty four cents”. “£11” is converted to “eleven pounds”.

Please make all conversions in lowercase only.

Note that this is not an exhaustive list of patterns that occur in the provided data. You are encouraged to figure out additional cases by iteratively performing an error analysis of data.

Input and Output formats:

Write a rule-based program which takes a json file as input and outputs the conversions. The input file is in the json format. Each entry in the input file has the following structure.

__sid__: A unique identifier for each sentence

__input_tokens__: A list of tokens in the input sentence

For each entry in the input file, your program should convert the input_tokens into their spoken form. For tokens which do not need conversion, your program should output **<self>** token. For punctuations your program should output a **sil** token. Final output of your program should be a new json file. Each entry in the file should have the following structure.

__sid__: A unique identifier corresponding to an input sentence.

__output_tokens__: A list of converted tokens corresponding to input_tokens from the input sentence

The input and output files in the assignment data follow the above structure. We are sharing code for parsing and dumping json files here https://colab.research.google.com/drive/1bJn4SUzwD18uPgfPCTg7RDaAdsFwG6OL?usp=sharing

Evaluation Metric:

Your code will be evaluated on f-measure. Consider the following categories:

1. An input token (that is not a sil or a self token) is correctly converted. For this a 1 will be added to both the numerator and denominator of both precision and recall.

2. An input token (that is a sil or a self token) is correctly converted. This will not affect the precision/recall computation.

3. An input token (that is not a sil or a self token) should have been converted but the program missed it (i.e., converted it to sil or self). For this a 1 will be added to the denominator of recall.

4. An input token should not have been converted (i.e., it should have been sil or self), but the program erroneously converted it. For this a 1 will be added to the denominator of precision.

5. An input token (non-sil, non-self) should have been converted but the program converted it incorrectly (to a non-sil, non-self). For this, a 1 will be added to the denominator of both the precision and recall.

F-score is the Harmonic mean of precision and recall.

Example:

Consider the following entry from the input json file.

{

```
   "sid": 0,
   "input_tokens": ["Joseph", "F.", "Baugher", "(", "1 July 2000", ")", "."]
}
```

The corresponding entry in output json file is

```
{
   "sid": 0,
   "output_tokens": ["<self>", "f", "<self>", "sil", "the first of july two thousand", "sil", "sil"]
}
```

Observe that input tokens ‘Joseph’ and ‘Baugher’ are converted to **<self>** and punctuation are coveted to **sil**.

What to submit?

Submissions will happen over moodle. We will release the detailed submission guidelines soon. Please make sure that you have VPN and HPC access. At this stage we provide in the data and i/o format. Detailed submission instructions will be disclosed over piazza. All doubts/clarifications will be discussed on Piazza.

What is allowed? What is not?

1. The assignment is to be done individually.
2. You should use Python 3.6 for this assignment. We only allow json, regex and in-built Python packages. Any additional packages can be requested over Piazza and will be allowed only after verification.
3. You must not discuss this assignment with anyone outside the class. **Make sure you mention the names in your write-up in case you discuss with anyone from within the class.** Please read academic integrity guidelines on the course home page and follow them carefully.
4. You must not use the data provided for training a machine learning system. You are only allowed to use the data to get insights on how well your rule-based system works.
5. You must not search the Web for papers or other websites on how to build such a system. However, you are welcome to search the Web for “lists'' of concepts, like a list of common abbreviations, etc. You must cite the appropriate references in your writeup.txt, which will be submitted along with the code.
6. We will run plagiarism detection software. Any person found guilty will be awarded a suitable penalty as per IIT rules.
7. Your code will be automatically evaluated. You will get a 20% flat penalty if it does not conform to output guidelines.