Aashish Yadavally
Fall 2019

Homework 3

CSCI 6900
September 5, 2019

By running the following Python code snippet on a machine with *pynini* installed, which also contains the attached script *quantity_normalizer.py*, the following outputs are achieved for the corresponding quantites:

```
>>> from quantity_normalizer import normalize
>>> normalize("3 c.")
three cups
>>> normalize("40 mi.")
forty miles
>>> normalize("1 lb.")
one pound
>>> normalize("1000 lbs.")
one thousand pounds
>>> normalize("9904 ft.")
nine thousand nine hundred four feet
>>> normalize("1 doz.")
one dozen
>>> normalize("1024 GB")
one thousand twenty four gigabytes
>>> normalize("407 rpm")
four hundred seven revolutions per minute
```

A variety of mappings were implemented in order to build a word-normalizer using *pynini*. They include:

- zero_del: It is a transducer which take ’O’ to a blank ””, thus, helping in eliminating 0’s from the string.
- ones_map: This represents the mapping of the digits in the unit’s place. It is a union of transducers mapping digits between 1 to 9 to their corresponding word-forms, i.e, ’one’ to ’nine’.
- teens_map: This represents the mapping of numbers between 10 and 19, to their corresponding word-forms, i.e, ’ten’ to ’nineteen’.
- tens_map: This represents the mapping of digits in the ten’s place between ’2’ and ’9’, corresponding to ’twenty’ and ’ninety’ respectively.
- hundreds_map: This represents the mapping of digits in the ten’s place between ’1’ and ’9’, corresponding to ’one hundred’ and ’nine hundred’ respectively.
- thousands_map: This represents the mapping of digits in the ten’s place between ’1’ and ’9’, corresponding to ’one thousand’ and ’nine thousand’ respectively.

- units_map: Mapping of abbreviations of units to their corresponding plural noun forms.
- singularize_map: Mapping of plural noun forms to their corresponding singular noun forms.

Furthermore, a composition of multiple *pynini.cdrewrite()* are used to rewrite rules using finite-state transducers. Firstly, given the string, the quantity abbreviation is converted into it's corresponding plural form using the units_map transducer in the cdrewrite composition. Next, the strings which have a left context of "1 " are replaced by the singular noun forms, mapped according to the singularize_map. All digits in the thousandth place are defined such that their right context is a concatenation of [1-9] three times; all digits in the hundredth place are defined such that their right context is a concatenation of [1-9] two times; all digits in tenth place are defined such that their right context is [1-9], wherein, the leading digit is not '1'; all numbers between [10-19] are replaced by their corresponding word-forms, as mapped in teens_map. When a digit's left and right contexts are empty, then it is replaced by digits between [1-9], each mapped according to ones_map. Finally, all the extra zeroes are deleted to make the word-normalized string. A series of these compositions on the input string in the normalize() function returns the desired outputs.

By following the technique as described in the previous paragraph, the results as mentioned in the code-snippet were obtained.