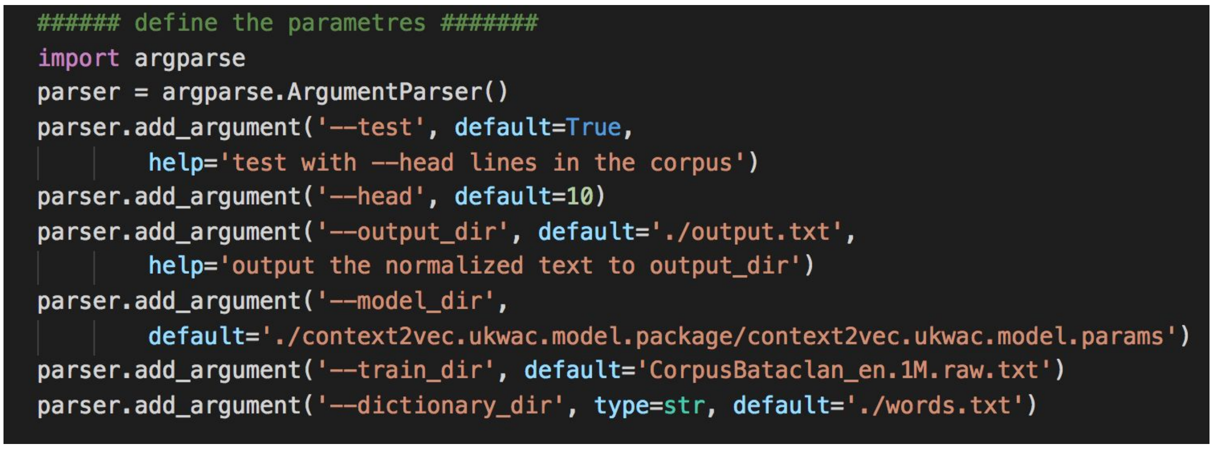
TP3 Normalization of raw twitter text

Xiangnan YUE
xiangnanyue@gmail.com

- How this system works:

This system integrates the context2vec program with a simple Levenshtein Distance algorithm, a clean text function and a lexicon dictionary. System parametres is defined as follows:

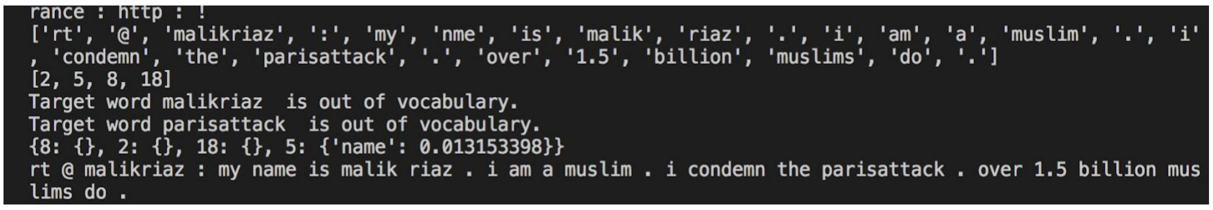


```
###### define the parametres #######
import argparse
parser = argparse.ArgumentParser()
parser.add_argument('--test', default=True,
        help='test with --head lines in the corpus')
parser.add_argument('--head', default=10)
parser.add_argument('--output_dir', default='./output.txt',
        help='output the normalized text to output_dir')
parser.add_argument('--model_dir',
        default='./context2vec.ukwac.model.package/context2vec.ukwac.model.params')
parser.add_argument('--train_dir', default='CorpusBataclan_en.1M.raw.txt')
parser.add_argument('--dictionary_dir', type=str, default='./words.txt')
```

where we defined :
--test : whether to test or not, if args.test is True, only first args.head lines will be tested and output the normalized lines in the output file
--head : how many lines of the train file to test if test is True.
--output_dir : path of the output file.
--model_dir : path of the context information model.
--train_dir : the path of the train file.
--dictionary_dir : the path of the lexicon dictionary words.txt

We first use the data online words.txt to construct our lexicon dictionary rather than the nltk.corpus.words library (it was tested yet badly performed because it identifies different forms of verbs and noms like "cuts" as non-words.). The txt file was input and stored as a dictionary in the words.pkl file to avoid time waste.

Codes in context2vec.eval.explore_context2vec.py were integrated to function similarity_matrix(sent, targets). This function will calculate the similarity degree for each non-word (the same word should be non-punctuation, non-https, non-number) in the input sentence:
An example for sentence : "RT @MalikRiaz: My nme is Malik Riaz. I am a Muslim. I condemn the #ParisAttack. Over 1.5 billion Muslims do. "



```
rance : http : !
['rt', '@', 'malikriaz', ':', 'my', 'nme', 'is', 'malik', 'riaz', '.', 'i', 'am', 'a', 'muslim', '.', 'i'
, 'condemn', 'the', 'parisattack', '.', 'over', '1.5', 'billion', 'muslims', 'do', '.']
[2, 5, 8, 18]
Target word malikriaz  is out of vocabulary.
Target word parisattack  is out of vocabulary.
{8: {}, 2: {}, 18: {}, 5: {'name': 0.013153398}}
rt @ malikriaz : my name is malik riaz . i am a muslim . i condemn the parisattack . over 1.5 billion mus
lims do .
```

In the picture, you can see that indexes {2,5,8,18}, which are {‘malikriaz’, ‘nme’, ‘riaz’, ‘parisattack’} are identified as non-word, and the function output the possible word for replacement. If the non-word doesn’t exist in the vocabulary dictionary of the model indicated by the path model_dir, we will ignore it and leave it unchanged, yet if the non-word exists in the model, a word list for replacement will be output and the most possible word is selected.

Function similarity_matrix() takes the input of cleaned word list “sent” and the non-word index “targets”. Original program context2vec will output the similarity matrix and 10 most possible choice for replacements. Here we input the 10 most possible words into the Levenshtein Distance dynamic program to filter out the words with a distance value > 2.

- Things working and things not working

In the example above, you can see that the typo “nme” was rightly corrected to “name”. However the word “parisattack” doesn’t exist in the model’s data base and thus is left unchanged.

However, this problem may be solved by parsing the word using upper case letter ParisAttack. This can be seen as a case in the One-to-many or many to one problem.

Another possible improvement is we didn’t calculate the frequency table as a prior distribution for this twitter data set. This should be easy to implement, and was left for future development.

- Use of the system

Put the corpus data “CorpusBataclan_en.1M.raw.txt” and “context2vec.ukwac.model.package/context2vec.ukwac.model” under the “system”.

Set in the bash the global name to indicate the path of file, eg :
MODEL_NAME="context2vec.ukwac.model.package/context2vec.ukwac.model"
TRAIN_DATA='CorpusBataclan_en.1M.raw.txt'

You can simply “cd” to “system” repository and run “python2.7 normalize.py --head=30 --output_dir=./output.txt --model_dir=$MODEL_NAME.params --train_dir=$TRAIN_DATA” to test 30 lines in the Corpus data. The normalized 10 sentences will be stored in the “output.txt”.

Otherwise, try the “./run_system.sh $MODEL_NAME $TRAIN_DATA ” directly.

Contact me by mail : xiangnanyue@gmail.com if anything got wrong.