Ling 473 Project 2 Unigrams Counting
Daniel Campos 08/14/2018

Results table

tgl Kasalukuyang nakararanas ng pag-unlad ang bansa sa mga remittances na ipinapadala pauwi ng mga OFW. Isa sa mga pinakaumuunlad na sektor ang teknolohiyang pang-impormasyon .

swh	-52.8260565482
fra	-49.8171184097
eng	-57.9833349681
dut	-50.2549063072
dan	-54.6937460156
por	-48.2280049678
deu	-57.9833349681
nob	-57.9833349681
ita	-51.4756900907
swe	-55.0488365169
pol	-48.0569686265
gla	-50.0597336897
spa	-57.9833349681
fin	-57.9833349681
tgl	-22.7613478411

result tgl

Approach

For my approach I tried to keep things simple and then scale. First I read through the corpus and when I encountered a new word I set the occourence of that word in all langauges to 0. As I came across actual numbers I updated. Once I was done reading the entire corpus I used additive smoothing(see smoothing approach) to update the counts in the corpus. Once this was done I started reading in the target file(train.text and test.txt) and remove all unwanted punctuation(using the translate library) and then proceeding to caclulate the log probability of the string given the langauge. If a word was not in my corpus lexicon than I assumed it was equaly likley to come from any langauge and applied addative smoothing(since it was unfound in any language total count would be 15 so odds are 1/15) and calculated an end weight. Then I loop though all the langauge probabilites and choose the langauge that has the highest number. This allowed me to get 14/15 examples in the train file.

Smoothing Approach

Before choosing a smoothing approach I researched various smoothing approaches in popular literature and found out that the most common in NLP style tasks with unknown words is Addative smoothing. The logic is we add 1 occourence to all different categories. This makes langauges that have count 0 be 1/total(very rare). This allows calculations to happen with ease with minimal core stat disruption.

Special Features

The Extra Credit

Extra credit.

For my extra credit approach I looked at both the extra-credit.train and regular train and looked at the difference between log probability of the predicted language vs the average log probability of all the langauges. In general I found that when the classifier was accurate, the difference was usually > 25. Based on this, when the diff is < 25 I output unk. If I wanted to be even more sure I would likley use a different smoothing function since my additive smoothing probabily was not the best for unknown langauges. Additionatlly when I had an unkown word I just treated it as the word being as likley from coming from any of the 15 target languagues, this biased my classifer toward predicting to one of these langauges since it didnt penalize out of vocabulary strings that much.

Missing Features

Not robust to errors