Ling 575: Alignment

John Cadigan
github: https://github.com/johncadigan/WordAlignment.git
run: compile2.10.sh then run.sh

Motivation

I explored some multi-threaded and combined alignment approaches to improve the model, but I did not have much success. I did find ways of formulating the calculations which will resemble those I will use on Spark for my project.

Description

IBM Model 1

I used the standard IBM Model 1. To help it converge faster, I set every target word e and is set to to be equally probable from each of the n words from f it is found with $P(e|f) = \frac{1}{n+1}$

With each iteration of the EM algorithm, two maps are used to store the expected probability counts are created: $count(e|f)$ and $count(f)$. Then for each sentence the deltas are calculated for every target word; the delta is the sum of all the translation probabilities which apply to target word

$delta(e_i) = \sum_{f \epsilon F} P(e_i|f)$

For each possible combination of e and f, the counts are both incremented by the probability divided by delta

$count(e|f) += \frac{P(e|f)}{delta(e)}$ $count(f) += \frac{P(e|f)}{delta(e)}$

After all sentences, these counts are normalized to produce newer, more accurate estimations of $P(e|f)$ and the process continues until all iterations are complete

$P(e|f) = \frac{count(e|f)}{count(f)}$

Alignment and symmetrization

I then go through every sentence and output the highest probable alignment for each word based only on $P(e|f)$. I then unify the output from models trained in each direction to recover more translations and only report translations found by both.

Results

Iterations	Precision	Recall	AER
5	0.870	0.659	.235
10	0.866	0.671	0.229
20	0.863	0.674	0.229