Overview

I implemented a translation model in Translation.java. I am trying to find e that maximizes p(e|f). This is equivalent to finding f that maximizes p(f|e)p(e). The p(f|e) term was mostly taken care of by the translation model from the first part of the assignment. For p(e), I used a bigram language model.

Translation Model

The value p(f|e) was computed by the translation model for all values of f and e. However, I needed to do some pre-processing to create a map that contained the 10 values of e that maximized p(f|e) for each f (so the key is f). This is done in the makeLikelyWordsFile() method.

Language Model

I reused a bigram language model from a previous assignment. I used the discount model, and chose the discount parameter by checking various possible values in the tuneParameters() method. The training data was separate from the spanish corpus provided for this assignment.

Putting it Together

I used a beam search to find the most likely translation. At each step, I determined the most likely english words that would explain the current foreign word. I tried adding this word to all current options for partial translations, and updated the probability of that translation using the language model. After each step, I would only keep the most likely BEAM_SIZE possibilities.

Results

The translations were very poor. We consider the example "la casa verde es grande", which should translate to "the green house is large". This was actually translated as "la herzogenrath grey-green faulted moulding".

This is clearly not right. Let's examine what went wrong. Here is one step in the algorithm, when we are considering the second word to add - the word corresponding to "casa". We see that "home" is indeed on the list, but there are many other obsure words on the list as well. The probability in parentheses is the probability of the word under tje translation model, and the next probability is the computed probability using the bigram model.

```
la -- -0.1773619275583292
  objectively-speaking (0.1005454008868519)  0.375
  herzogenrath (0.2051585500198051)  0.375
  cara (0.12187983337331303)  7.573217870370746E-8
  chambers (0.1047390287244682)  4.543930722222447E-7
  flashy (0.15942881814531612)  0.375
  home-produced (0.192964141448603)  7.573217870370746E-8
  gremlins (0.12505330524849947)  0.375
  burns (0.11016193673372791)  1.5146435740741492E-7
  home (0.11874007018251743)  2.9308353158334785E-5
  'wage' (0.14250147233032792)  0.375
```

It appears that by chance, both p(f|e) and p(e) were large for obscure english words. This led to incorrect translations.

Possible Improvements

- Training on more data would remove the noise from obscure words
- Training on data similar to the spanish-english corpus might help
- Lowering the discount could punish uncommon words even more
- We could try swapping word orderings. In the example above, the correct translation includes a word swap, which is not allowed under my current model.