Natural Language Processing:

Assignment 8: Machine Translation

Jordan Boyd-Graber

Out: **10. November 2014**
Due: **21. November 2014**

Introduction

As always, check out the Github repository with the course homework templates:

`git://github.com/ezubaric/cl1-hw.git`

The code for this homework is in the `hw8` directory. The goal of this homework is for you to build a machine translation scoring function based on IBM Model 1.

Data

The data are gzipped samples from Europarl. And can be found in the GitHub directory with the code.

Also in the directory are two text files with lists of words. These can help you monitor the progress of the algorithm to see if the lexical translation probabilities look like what they should.

What to Do

You have been given a partial EM implementation of IBM Model 1 translating foreign (f) from English (e).[1] The maximization function is complete, but the expectation is not, nor is the function to score a complete translation pair. You need to fill in two functions.

[1] This is **different** from what we initially discussed in class; we're applying the noisy-channel model here: $p(f|e)p(e)$, which translates into translating from English to foreign mathematically, even if the system is to translate from foreign to English.

Generating Counts (20 points)

The first function you need to fill in is `sentence_counts`, which should return an iterator over english and foreign words pairs along with their expected count. The expected count is the expected number of times that word pair was translated in the sentence, given by the equation

$$c(f|e;\vec{e},\vec{f}) = \sum_a p(a|\vec{e},\vec{f}) \sum_{j=1}^{l_f} \delta(f,f_j)(e,e_{a(j)}). \tag{1}$$

Scoring (10 points)

The second function you need to fill in is the noisy channel scoring method `translate_score`, which is the translation probability (given by Model 1) times the probability of the English output under the language model.

Running the Model (10 points)

Run the model and produce the lexical translation table for the development words. Don't leave this until the last moment, because this can take a while.

How to solve the Problem

Don't start using the big data immediately. Start with the small data. If you run a correct implementation on the toy data, you should get something like the following:

What to turn in

You will turn in your complete `ibm_trans.py` file and the word translations for the supplied test file (devwords.txt) after three (3) iterations of EM.

Extra Credit (5 points)

If you would like extra credit, add an additional function that computes the best alignment between a pair of sentences.

Listing 1: Successful output from running on toy data

```
python ibm_trans.py 5 toy toy_test.txt
Corpus <__main__.ToyCorpus instance at 0x27d08c0>, 5 m1 iters
Model1 Iteration 0 ...
Sentence 0

blind:blind:0.250000 an:0.138889 hier:0.138889 von:0.138889
bist:0.111111 du:0.111111 nicht:0.111111 irish:du:0.166667
irisch:0.166667 kind:0.166667 mein:0.166667 wilest:0.166667
wo:0.166667 is:ist:0.181818 der:0.090909 fahrrad:0.090909
himmel:0.090909 mein:0.090909 nicht:0.090909 noch:0.090909
rosa:0.090909 siebte:0.090909 weit:0.090909 cries:auch:0.200000
der:0.200000 himmel:0.200000 weint:0.200000 wo:0.200000
are:du:0.206897 bist:0.120690 blind:0.120690 nicht:0.120690
irisch:0.086207 kind:0.086207 mein:0.086207 wilest:0.086207
wo:0.086207

done

...

========COMPUTING WORD TRANSLATIONS=======
balloons 99 0.203438
not blind 0.028217
blind blind 0.693760
jets 99 0.341654

LM Sentence 0
========COMPUTING SENTENCE TRANSLATIONS=======
ENGLISH: my bike cries 99 fighter jets
FOREIGN: mein fahrrad weint 99 dusenjaeger
SCORE: -30.100041

ENGLISH: the heaven is pink about now
FOREIGN: von hier an bleibe der himmel rosa
SCORE: -44.224041
```

Questions

1. *My code is really slow; is it okay if I don't run my code on the entire dataset? (by using the limit argument)*

   Yes, but if your numbers look too far off from what is expected, you may lose points.

2. *I'm getting probabilities greater than 1.0!*

   The logprob function of `nltk`'s language model returns the negative log probability.

3. *I can't get my translation probabilities to match.*

   - One little thing that may cause a slight difference in computing the translation probability: I add 0.001 to each of the inputs (to guard against zeros) while summing up across the foreign string.
   - You will need to add a `None` to the English sentence in the `translate_score` function. The assert is there to make sure it hasn't happened outside of the function.
   - Make sure you compute the `lm` probabilities conditioning the first word on the empty string (`''`).