W4705 HW2 (wj2227)

For fast run of all scripts, please use *run.sh*

Question 4

Part1: how to run your code step by step

To run the code, please follow the following commands:

python question4.py > q4_align_20

After running, the alignment of the first 20 sentences will be output to file "*q4_align_20*". Besides, the values of t parameters that are greater than 0 for German-English word pairs after running EM for IBM Model 1 are saved in file "*q4_t_ibm1*", and for each word in "*devwords.txt*", the list of the 10 foreign words with the highest t(f|e) parameter (and the parameter itself) are stored in file "*q4_devwords_top10*".

Part2: performance for your algorithm

In this part, EM algorithm has been run for 5 iterations in order to calculate t values of German-English word pairs based on IBM 1 model.

Part3: observations and comments about your experimental results.

From the result of devwords, we could see that some of the English words has corresponding German words converge with high probability, for example, English "i" and "dimension" matches German words "ich" and "dimension" with probability 0.70 and 0.67 respectively. However for part of the words, the probabilities are still low for all possible translation and spreads across a large number of words. Although it may be affected by the fact that a word can have multiple meanings and correspond to multiple words in English, it is likely that after IBM model 1 part of the t's have not been well converged around the true values.

The alignment has been compared to be the same as the sample output of this part in HW3 packet.

Question 5

Part1: how to run your code step by step

To run the code, please follow the following commands:

python question5.py > q5_align_20

After running, the alignment of the first 20 sentences will be output to file "*q5_align_20*". Besides, the values of t and q parameters that are greater than 0 after running EM for IBM Model 2 are saved in file "*q5_t_ibm2*" and "*q5_q_ibm2*" respectively.

Part2: performance for your algorithm

In this part, t values from the IBM 1 model in Question 4 are used as initial values. EM algorithm has been run for 5 iterations in order to calculate t and q values for IBM 2 model.

Part3: observations and comments about your experimental results.

Comparing to IBM 1 model, IBM 2 model considers in addition the probability of alignment variable a_i taking the value *j* conditioned on the lengths *l* and *m* of the English and foreign sentences. So it encode more information into the model, which lead to more concrete alignment generation.

Question 6

Part1: how to run your code step by step

To run the code, please follow the following commands:

python question6.py > unscrambled.en
python eval_scramble.py unscrambled.en original.en

After running, the unscrambled English sentences based on the IBM 2 model are output to file "*unscrambled.en*".

Part2: performance for your algorithm

The accuracy of the recovered English sentences order is 92% as shown below.

```
I:\W4705-NLP\hw3\hw3>python eval_scramble.py unscrambled.en original.en
Right   Total   Acc
92      100     0.920
```

Part3: observations and comments about your experimental results.

We could notice that for a portion of the test German sentences, either unseen words in training corpus exist (t parameters are missing) or q parameters are missing. In this question for such situations, a large negative number

-50 was added to the sum of log probability instead of negative infinity (which will lose comparison measurement for comparing alignments). It works fine in this way since each of the missing t or q cases are penalized but not too extreme to compare the alignments.