COMS 4705, Fall 2014: Problem Set 3

Total points: 140

Analytic Problems (due November 10th at 5pm)

Question 1 (25 points)

Say that we have used IBM model 2 to estimate a model of the form

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}, m) = \prod_{j=1}^{m} t(f_j|e_{a_j})q(a_j|j, l, m)$$

where $\mathbf{f}$ is a French sequence of words $f_1, f_2, \dots f_m$, $\mathbf{a}$ is a sequence of alignment variables $a_1, a_2, \dots a_m$, and $\mathbf{e}$ is an English sequence of words $e_1, e_2, \dots e_l$. (Note that the probability p is conditioned on the identity of the English sentence, $\mathbf{e}$, as well as the length of the French sentence, m.)

Question 1(a) (10 points) Give pseudo-code for an efficient algorithm that takes an input an English string $\mathbf{e}$, and an integer m, and returns

$$\arg\max_{\mathbf{f},\mathbf{a}} p(\mathbf{f}, \mathbf{a}|\mathbf{e}, m)$$

where the $\arg\max$ is taken over all $\mathbf{f}, \mathbf{a}$ pairs whose length is m.

Question 1(b) (10 points) Give pseudo-code for an efficient algorithm that takes an input an English string $\mathbf{e}$, and an integer m, and returns

$$\arg\max_{\mathbf{f}} p(\mathbf{f}|\mathbf{e}, m)$$

where the $\arg\max$ is taken over all $\mathbf{f}$ strings whose length is m. Note that

$$p(\mathbf{f}|\mathbf{e}, m) = \sum_{\mathbf{a}:|\mathbf{a}|=\mathbf{m}} \prod_{j=1}^{m} t(f_j|e_{a_j})q(a_j|j, l, m)$$

Question 1(c) (5 points) Given that it is possible to efficiently find

$$\arg\max_{\mathbf{f}} p(\mathbf{f}|\mathbf{e})$$

when p takes the above form, why is it preferable to search for

$$\arg\max_{\mathbf{e}} p(\mathbf{f}|\mathbf{e})p_{LM}(\mathbf{e})$$

rather than

$$\arg\max_{\mathbf{e}} p(\mathbf{e}|\mathbf{f})$$

when translating from French to English? (Note: p_{LM} is a language model, for example a trigram language model)

Question 2 (20 points)

IBM model 2 for statistical machine translation defines a model of the form

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}, m) = \prod_{j=1}^{m} t(f_j|e_{a_j})q(a_j|j, l, m)$$

where $\mathbf{f}$ is a French sequence of words $f_1, f_2, \ldots f_m$, $\mathbf{a}$ is a sequence of alignment variables $a_1, a_2, \ldots a_m$, and $\mathbf{e}$ is an English sequence of words $e_1, e_2, \ldots e_l$. (Note that the probability p is conditioned on the identity of the English sentence, $\mathbf{e}$, as well as the length of the French sentence, m.) The parameters of the model are translation parameters of the form $t(f|e)$ and alignment parameters of the form $q(a_j|j, l, m)$.

Now say we modify the model to be

$$p_{M3}(\mathbf{f}, \mathbf{a}|\mathbf{e}, m) = \prod_{j=1}^{m} t(f_j|e_{a_j})q(a_j|a_{j-1}, j, l, m)$$

where a_0 is defined to be 0. Hence the alignment parameters are now modified to be conditioned in addition upon the previous alignment variable.

Give pseudo-code for an efficient algorithm that takes as input an English string $\mathbf{e}$ of length l, a French string of length m, and returns

$$\arg\max_{\mathbf{a}} p_{M3}(\mathbf{f}, \mathbf{a}|\mathbf{e}, m)$$

where the $\arg\max$ is taken over all values for $\mathbf{a}$ whose length is m.

Question 3 (20 points)

Consider a phrase-based translation model with a distortion limit $d = 0$. That is, the set of valid derivations consists of sequences of phrases $p_1 p_2 \ldots p_L$ such that each word in the source language is translated exactly once, such that $s(p_1) = 1$, and such that $s(p_j) = t(p_{j-1}) + 1$ for $j \in \{2 \ldots L\}$. (Recall that each phrase p_k is a triple (s, t, e) where s is the start point of the phrase, t is the end point, and e is a sequence of one or more English words. We use $s(p)$ and $t(p)$ to denote the start and end of a phrase p respectively.)

Define the score for any sequence of phrases $y = p_1 \ldots p_L$ to be

$$f(y) = \sum_{j=1}^{L} g(p_j)$$

where $g(p_j)$ is the score for the phrase p_j. Thus the score is a sum of scores for the different phrases in the translation. Note that there is no language model.

Write a dynamic programming algorithm that takes a source language sentence $x = x_1 x_2 \ldots x_N$ as its input, and returns

$$\max_{y \in \mathcal{Y}(x)} f(y)$$

as its output, where $\mathcal{Y}(x)$ is the set of valid derivations for the input x.

You may use $\mathcal{P}$ to denote set of possible phrases for the input sentence x: each phrase $p \in \mathcal{P}$ is a triple (s, t, e) where s is the start-point of the phrase, t is the end point, and e is a sequence of English words.

Programming Problems (due November 17th at 5pm)

Please read the submission instructions, policy and hints on the course webpage.

In this programming problem you will implement IBM translation models 1 and 2 and use your implementation to learn word alignments in an English/German parallel corpus.

The two files *corpus.en.gz* and *corpus.de.gz* contain 20,000 English and German sentences respectively. The i-th sentence in the English file is a translation of the i-th sentence in the German file. The files are in one-sentence-per-line format (but compressed using *gzip*). Words in each line are separated by single spaces.

Question 4 (25 points) - IBM Model 1

Recall that IBM model 1 only has word translation parameters $t(f|e)$, which can be interpreted as the conditional probability of generating a foreign word f from an English word e (or from NULL).

We can estimate $t(f|e)$ using the EM algorithm (see handout).

Implement a version of IBM model 1, which takes *corpus.en.gz* and *corpus.de.gz* as input.

- Your implementation should only store t parameters for possible pairs of foreign and English words (i.e. words that occur together in a parallel translation) and the special English word NULL.

  In the initialization step set $t(f|e)$ to be the uniform distribution over all foreign words that could be aligned to e in the corpus.

  More specifically
  $$t(f|e) = \frac{1}{n(e)},$$
  where $n(e)$ is the number of different words that occur in any translation of a sentence containing e. Note that the special English word NULL can be aligned to any foreign word in the corpus.

- Starting from the initial $t(f|e)$ parameters, run 5 iterations of the EM algorithm for IBM model 1 (this may take a while). Then, for each English word e in *devwords.txt*, print the list of the 10 foreign words with the highest $t(f|e)$ parameter (and the parameter itself). Submit your code and the result.

- Finally, use your model to find alignments for the first 20 sentence pairs in the training data. For each sentence, align each foreign word f_i to the English word with the highest $t(f|e)$ score, i.e.
  $$a_i = \arg\max_{j \in \{0 \cdots l\}} t(f_i|e_j).$$

  Print the alignments as a list of m integers containing the a_i values.

Question 5 (25 pts) - IBM Model 2

We will now extend our alignment model to IBM model 2 by adding alignment parameters $q(j|i, l, m)$.

- Initialize the q parameters to the uniform distribution over all j for each i, l, and m, i.e.

$$q(j|i,l,m) = \frac{1}{l+1}$$

  You only need to store parameters for pairs of sentence lengths l and m that occur in the corpus. To initialize the $t(f|e)$ parameters, use the last set of parameters (after 5 iterations) produced by your implementation of IBM model 1.

- Extend your implementation of the EM algorithm to IBM model 2. Adapt the delta function to include $q(j|i,l,m)$ parameters

$$\delta(k,i,j) = \frac{q(j|i,l_k,m_k)t(f_i^{(k)}|e_j^{(k)})}{\sum_{j=0}^{l_k} q(j|i,l_k,m_k)t(f_i^{(k)}|e_j^{(k)})}.$$

  and compute expected counts $c(j|i,l,m)$ and $c(i,l,m)$.

  After each iteration through the corpus re-estimate the $t(f_i, e_j)$ parameters as before and the q parameters:

$$q(j|i,l,m) = \frac{c(j|i,l,m)}{c(i,l,m)}.$$

  Then, run 5 iterations of EM for IBM model 2.

- As before, use the model to compute alignments for the sentence pairs in the corpus. For each foreign word f_i, the best alignment is

$$a_i = \arg\max_{j \in \{0 \cdots l\}} q(j|i,l,m)t(f_i|e_j).$$

  Print the alignments for the first 20 sentence pairs as before and comment on the difference in model performance. Submit your code.

Question 6 (25 pts) - Finding translations

Claudia Clumsy dropped her German/English parallel transcripts of a European Parliament debate, so that the sentences are no longer aligned. English sentences are in the file *scrambled.en*, German sentences are in *original.de*. Use your trained IBM Model 2 from the last question to recover the original English sentence by computing

$$\arg\max_e \max_a P(f,a|e)$$

for each German sentence, where

$$P(f,a|e) = \prod_{i=1}^{m} q(a_i|i,l,m)t(f_i|e_{a_i})$$

That is for each German sentence find the English sentence that produces the highest-scoring alignment.

Note: You may run into underflow issues when aligning. To avoid this problem we recommend using log-probs, i.e. solve the following problem

$$\arg\max_e \max_a P(f,a|e) = \arg\max_e \max_a \log P(f,a|e) = \arg\max_e \max_a \sum_{i=1}^{m} \log(q(a_i|i,l,m)t(f_i|e_{a_i}))$$

When the inner product $q(a_i|i,l,m)t(f_i|e_{a_i})$ is zero, you can substitute a large negative constant.

- Print out a new file `unscrambled.en` with the best sentence match from `scrambled.en` in the order of the original German sentences. You should run `python eval_scramble.py unscrambled.en original.en` to check the accuracy of your solution. Comment on the efficiency of your method and any issues you saw while unscrambling.