6.864, Fall 2007: Problem Set 3

Total points: 190 regular points
Due date: 5pm, 8th November 2007
Submit to Igor Malioutov either by email to `igorm@csail.mit.edu`, or by hand to Stata G360
Late policy: 5 points off for every day late, 0 points if handed in after 5pm on 12th November

Question 1 (15 points)

In this question we will develop an algorithm, based on the EM algorithm, for modeling of topics underlying documents. In this model, the training sample $x^1, x^2, \ldots x^m$ is a sequence of m documents. We will take each document x^i to consist of n words, $x^i_1, x^i_2, \ldots x^i_n$. The hidden variables y in the EM approach can take one of K values, $1, 2, \ldots K$. The model is defined as follows:

$$P(x, y|\Theta) = P(y) \prod_{j=1}^{n} P(x_j|y)$$

Thus if $\mathcal{V}$ is the vocabulary—the set of possible words in any document—the parameters in the model are:

- $P(y)$ for $y = 1 \ldots K$
- $P(w|y)$ for $y = 1 \ldots K$ and $w \in \mathcal{V}$

Our aim in this question will be to derive EM updates which optimize the log-likelihood of the data:

$$L(\Theta) = \sum_{i=1}^{m} \log P(x^i|\Theta) = \sum_{i=1}^{m} \log \sum_{y} P(x^i, y|\Theta)$$

Give pseudo-code showing how to derive an updated parameter vector Θ^t from a previous parameter vector Θ^{t-1}. I.e., show pseudo-code that takes as input parameter estimates $P^{t-1}(y)$ for all y and $P^{t-1}(w|y)$ for all w, y, and as output provides updated parameter estimates $P^t(y)$ and $P^t(w|y)$ using EM. Use the notation $C(w, x)$ to denote the number of times word w is seen in document x.

Question 2 (15 points)

In this question we'll derive an EM approach to word clustering. In this model, the training sample $x^1, x^2, \ldots x^m$ is a sequence of m bigrams of the following form: each x^i is of the form w^i_1, w^i_2 where w^i_1, w^i_2 are words, and w^i_2 is seen following w^i_1 in the corpus. The hidden variables y can take one of K values, $1, 2, \ldots K$. The model is defined as follows:

$$P(w_2, y|w_1, \Theta) = P(y|w_1)P(w_2|y)$$

Thus if $\mathcal{V}$ is the vocabulary—the set of possible words in any document—the parameters in the model are:

- $P(y|w)$ for $y = 1 \ldots K$, for $w \in \mathcal{V}$
- $P(w|y)$ for $y = 1 \ldots K$ and $w \in \mathcal{V}$

Our aim in this question will be to derive EM updates which optimize the log-likelihood of the data:

$$L(\Theta) = \sum_{i=1}^{m} \log P(w_2^i | w_1^i, \Theta) = \sum_{i=1}^{m} \log \sum_{y} P(w_2^i | y) P(y | w_1^i)$$

Give pseudo-code showing how to derive an updated parameter vector Θ^t from a previous parameter vector Θ^{t-1}. I.e., show pseudo-code that takes as input parameter estimates $P^{t-1}(y|w)$ for all y, w and $P^{t-1}(w|y)$ for all w, y, and as output provides updated parameter estimates $P^t(y|w)$ and $P^t(w|y)$ using EM.

Question 3 (15 points)

In lecture (see also the accompanying note on EM) we saw how the forward-backward algorithm could be used to efficiently calculate probabilities of the following form for an HMM:

$$P(y_j = p | x, \Theta) = \sum_{y : y_j = p} P(y | x, \Theta)$$

and

$$P(y_j = p, y_{j+1} = q | x, \Theta) = \sum_{y : y_j = p, y_{j+1} = q} P(y | x, \Theta)$$

where x is some sequence of output symbols, and Θ are the parameters of the model (i.e., parameters of the form π_i, $a_{j,k}$ and $b_j(o)$ as defined in the lecture). Here y_j is the j'th state in a state sequence y, and p, q are integers in the range $1 \ldots N - 1$ assuming an N state HMM.

Question 3(a) (5 points) State how the following quantity can be calculated in terms of the forward-backward probabilities, and some of the parameters in the model:

$$P(y_2 = 1, y_3 = 2, y_4 = 1 | x, \Theta)$$

(we assume that the sequence x is of length at least 4)

Question 3(b) (5 points) State how the following quantity can be calculated in terms of the forward-backward probabilities, and some of the parameters in the model:

$$P(y_2 = 1, y_5 = 1 | x, \Theta)$$

(we assume that the sequence x is of length at least 5. Don't worry too much about the efficiency of your solution: we **do** expect you to use forward and backward terms, but we **don't** expect you to calculate any other quantities using dynamic programming.)

Question 3(c) (5 points) Say that we now wanted to calculate probabilities for an HMM such as the following:

$$\max_{y : y_j = p} P(y | x, \Theta)$$

so this is the maximum probability of any state sequence underlying x, with the constraint that the j'th label y_j is equal to p.

How would you modify the definition of the forward and backward terms—i.e., the recursive method for calculating them—to support this kind of calculation? How would you then calculate

$$\max_{y:y_3=1} P(y|x,\Theta)$$

assuming that the input sequence x is of length at least 3?

Question 4 (25 points)

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

$$P_{M2}(\mathbf{f},\mathbf{a}|\mathbf{e},m) = \prod_{j=1}^{m} T(f_j|e_{a_j})D(a_j|j,l,m)$$

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_{M2} is conditioned on the identity of the English sentence, $\mathbf{e}$, as well as the length of the French sentence, m.)

Question 4a (10 points) Give pseudo-code for an efficient algorithm that takes an input an English string $\mathbf{e}$, and an integer m, and returns

$$\arg\max_{\mathbf{f},\mathbf{a}} P_{M2}(\mathbf{f},\mathbf{a}|\mathbf{e},m)$$

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

Question 4b (10 points) Give pseudo-code for an efficient algorithm that takes an input an English string $\mathbf{e}$, and an integer m, and returns

$$\arg\max_{\mathbf{f}} P_{M2}(\mathbf{f}|\mathbf{e},m)$$

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

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

Question 4c (5 points) Given that it is possible to efficiently find

$$\arg\max_{\mathbf{f}} P_{M2}(\mathbf{f}|\mathbf{e})$$

when P_{M2} takes the above form, why is it preferable to search for

$$\arg\max_{\mathbf{e}} P_{M2}(\mathbf{f}|\mathbf{e})P_{LM}(\mathbf{e})$$

rather than

$$\arg\max_{\mathbf{e}} P_{M2}(\mathbf{e}|\mathbf{f})$$

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

Question 5 (30 points)

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

$$P_{M2}(\mathbf{f}, \mathbf{a}|\mathbf{e}, m) = \prod_{j=1}^{m} T(f_j|e_{a_j})D(a_j|j, l, m)$$

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_{M2} 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 $D(a_j|j, l, m)$.

Now say we modify the model to be

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

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 6 (90 points)

In this question you will implement code for IBM translation model 1. The files `corpus.en` and `corpus.de` have English and German sentences respectively, where the i'th sentence in the English file is a translation of the i'th sentence in the German file.

Implement a version of IBM model 1, which takes `corpus.en` and `corpus.de` as input. Your implementation should have the following features:

- The parameters of the model are $T(f|e)$, where f is a German word, and e is an English word or the special symbol NULL. You should only store parameters of the form $T(f|e)$ for (f, e) pairs which are seen somewhere in aligned sentences in the corpus.
- In the initializiation step, you should set $T(f|e) = \frac{1}{n(e)}$ where $n(e)$ is the number of different German words seen in German sentences aligned to English sentences that contain the word e.
- Your code should run 10 iterations of the EM algorithm to re-estimate the $T(f|e)$ parameters.

Note: your code should have the following functionality. It should be able to read in a file, line by line, where each line has an English word, for example

dog
eats
man
...

For each line it should return a list of German words, together with probabilities $T(f|e)$. The list of German words should contain all words for which $T(f|e) > 0$.