6.864, Fall 2007: Problem Set 4

Total points: 160 points

Due date: 5 pm, 30th of 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 4th December

Question 1 (15 points)

In class, we introduced several measures of similarity between probability distributions. The table below summarizes three of these measures: KL divergence and information radius (IRad). In the following questions, you will analyze properties of these similarity measures.

Similarity Measure	**Definition**
KL divergence	$D(p\|\|q) = \sum_i p_i \log \frac{p_i}{q_i}$
IRad	$D(p\|\|\frac{p+q}{2}) + D(q\|\|\frac{p+q}{2})$

1. Show that IRad is bounded by $2 \log 2$
2. Show that the KL divergence is not symmetric by finding an example of two distributions p and q for which $D(p||q) \neq D(q||p)$

Question 2 (15 points)

The min-cut segmentation algorithm presented in class computes text segmentation based on changes in lexical distribution. We would like to refine this algorithm by taking into account the length of computed segments. The function $cost(l)$ captures the penalty associated with a segment of length l — the assumption is that some values of segment length are more likely than others.

- There are several ways to specify $cost(l)$. Provide a definition for $cost(l)$, and explain your choice.
- Explain how to modify the definition of the K-way partitioning cost to take the length penalty $cost(l)$ into account. Similarly to the original definition, the modified partitioning cost has to be defined recursively.

Question 3 (25 points)

Figure 1 shows the perceptron algorithm, as described in lecture. Now say we alter the parameter update step to be the following:

```
If $(z_i \neq y_i)$
    $\mathcal{A} = \{z : z \in \texttt{GEN}(x_i), z \neq y_i, \mathbf{w} \cdot \mathbf{f}(x_i, z) \geq \mathbf{w} \cdot \mathbf{f}(x_i, y_i)\}$
    $m = |\mathcal{A}|$
    $\mathbf{w} = \mathbf{w} + \mathbf{f}(x_i, y_i) - \frac{1}{m} \sum_{z \in \mathcal{A}} \mathbf{f}(x_i, z)$
```

Show that the modified algorithm makes at most $\frac{R^2}{\delta^2}$ updates before convergence, where R and δ are as defined in the lecture (i.e., show that the convergence theorem that we described in lecture also holds for this algorithm). Hint: the proof is quite similar to the proof of convergence given in lecture.

```
Inputs:            Training set $(x_i, y_i)$ for $i = 1 \ldots n$

Initialization:    $\mathbf{w} = 0$

Define:            $F(x) = \text{argmax}_{y \in \texttt{GEN}(x)} \mathbf{f}(x, y) \cdot \mathbf{w}$

Algorithm:         For $t = 1 \ldots T$, $i = 1 \ldots n$
                       $z_i = F(x_i)$
                       If $(z_i \neq y_i)$    $\mathbf{w} = \mathbf{w} + \mathbf{f}(x_i, y_i) - \mathbf{f}(x_i, z_i)$

Output:            Parameters $\mathbf{w}$
```

Figure 1: The perceptron algorithm, as introduced in lecture.

Question 4 (30 points)

In the lecture on Word Sense Disambiguation, we saw decision lists as a learning method. Consider a method for disambiguating the word *plant* between the "vegetation" and "industrial" senses. We will look at two features when building the decision list: the identity of w_{+1}, the word to the immediate right of the instance of *plant* that is being classified; and w_{-1}, the word to the immediate left of the instance. For each possible word w seen to the right of *plant* in the training data, we can gather the following counts:

$$Count(sense = vegetation, w_{+1} = w)$$

$$Count(sense = industrial, w_{+1} = w)$$

$$Count(w_{+1} = w)$$

For example, we might have

$$Count(sense = vegetation, w_{+1} = life) = 100$$

$$Count(sense = industrial, w_{+1} = life) = 1$$

$$Count(w_{+1} = life) = 101$$

We can gather similar counts for the w_{-1} context.

Decision lists require an estimate of $P(sense|feature)$. In method (a), which is the same as that defined in lecture, we define

$$P(sense|feature) = \frac{Count(sense, feature) + \alpha}{Count(feature) + 2\alpha}$$

where $\alpha > 0$ is a small constant (e.g., $\alpha = 0.1$). For example, we would estimate

$$P(sense = vegetation|w_{+1} = life) = \frac{100 + \alpha}{101 + 2\alpha}$$

Remember that the decision list method then creates an ordered list of rules of the form *feature* $\rightarrow$ *sense* (e.g., w_{+1} = *lift* $\rightarrow$ sense=vegetation). The rules are ordered according to the probabilities $P(sense|feature)$.

4a) (10 points)

Now say we define our estimate to be

$$P(sense|feature) = \lambda \frac{Count(sense, feature)}{Count(feature)} + (1 - \lambda) * 0.5$$

where $\lambda = 0.5$. We assume that $Count(feature)$ is always greater than 0. Is this estimation method better, worse, or about as good as method (a) described above? Give justification for your answer.

4b) (10 points)

Now say we define our estimate to be

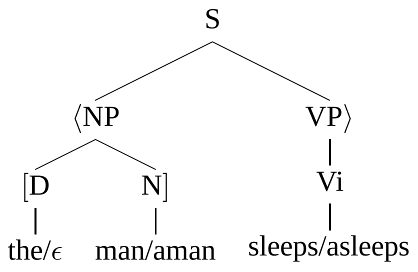
$$P(sense|feature) = \frac{Count(sense, feature) + \alpha}{Count(feature) + \alpha}$$

Describe a significant problem with this estimation method.

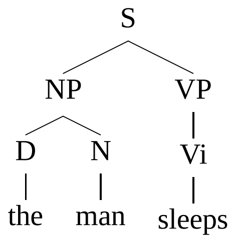
4c) (10 points)

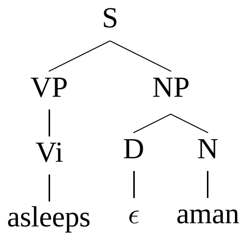
Finally, your task is to design a supervised word sense disambiguation algorithm that disambiguates every word in a document. Assume that a sense of the word depends on the word on the left, the word on the right, and on the sense of the previous word. In 200 words or less sketch an approach for word sense disambiguation that makes use of log-linear models.

Question 5 (25 points) In the lecture on 10/25 we defined transduction PCFGs. For example, a transduction PCFG would assign a probability to a structure such as the following (see the lecture notes for more details):



The above structure can be considered to represent an English string **e**, an English parse tree **E**, a French string **f**, and a French parse tree **F**, in this case:





Say $P(\mathbf{e}, \mathbf{E}, \mathbf{f}, \mathbf{F})$ is the probability assigned to an $\mathbf{e}, \mathbf{E}, \mathbf{f}, \mathbf{F}$ tuple by the transduction PCFG. Give pseudo-code for an algorithm that takes an $\mathbf{e}, \mathbf{E}$ pair as input, and returns

$$\arg\max_{\mathbf{f},\mathbf{F}} P(\mathbf{e}, \mathbf{E}, \mathbf{f}, \mathbf{F})$$

Question 6 (25 points) In the lecture on 10/18 we covered phrase-based models, and decoding methods for phrase-based models (for the latter, see the slides from Philipp Koehn that we went over in class, and are posted on the class webpage).

A "state" in the decoding process contains an English string, a summary of which words in the foreign language have been translated, and a probability. For example, one state might be:

E: Mary
F: * - - - - - -
P: 0.534

This indicates that the English string is "Mary" and that the foreign string consists of seven words, of which only the first has been translated (this is the meaning of * - - - - - -).

Assume that the foreign string being translated is "Maria no dia una bofetada a Maria". In addition, assume that the phrase lexicon contains the following entries:

Maria	⇔	Mary
a Maria	⇔	to Mary
no	⇔	not
dia	⇔	give
no dia	⇔	did not give
una	⇔	a
una bofetada	⇔	a slap

Show the full set of states that are reachable in one step of decoding from the state:

E: Mary
F: * - - - - - -
P: 0.534

For each of the next states, show E and F, but do not show the value for P.

Question 7 (25 points)

In the lecture on 10/30 we introduced the synchronous CFG formalism for machine translation from David Chiang. Slides 11 and 14 specify the rules in a synchronous CFG. In lecture we traced one possible translation for:

Aozhou shi yu Beihan you bangjiao de shaoshy guojia zhiyi

Slide 18 in the notes shows a parse tree for this sentence under the Chinese side of the grammar; slide 24 shows the final translation for this parse tree.

Question: show one other parse tree for the Chinese sentence above, and the corresponding translation into English. Your solution can make use of any rules in the grammar, with the restrictions:

1. You can't use the following rule:

   $X \rightarrow \langle$ yu $X_{\boxed{1}}$ you $X_{\boxed{2}}$, have $X_{\boxed{2}}$ with $X_{\boxed{1}}$ $\rangle$
2. You must use the following two rules:

   $X \rightarrow \langle$ $X_{\boxed{1}}$ de $X_{\boxed{2}}$, the $X_{\boxed{2}}$ that $X_{\boxed{1}}$ $\rangle$

   $X \rightarrow \langle$ $X_{\boxed{1}}$ zhiyi, one of $X_{\boxed{1}}$ $\rangle$
3. You must use the additional rules:

   $X \rightarrow \langle$ yu, with $\rangle$

   $X \rightarrow \langle$ you, have $\rangle$