NLP - Fall 2014 - Midterm Exam

(1) TrueCasing is the process of taking text with missing or unreliable case information and producing the proper case for each word, e.g. if the input looks like:

```
as previously reported , target letters were issued last month to
michael milken , drexel 's chief of junk-bond operations ; mr. milken 's
brother lowell ; cary maultasch , a drexel trader ; james dahl , a
drexel bond salesman ; and bruce newberg , a former drexel trader .
```

Then the output of the TrueCasing program should be:

```
As previously reported , target letters were issued last month to
Michael Milken , Drexel 's chief of junk-bond operations ; Mr. Milken 's
brother Lowell ; Cary Maultasch , a Drexel trader ; James Dahl , a
Drexel bond salesman ; and Bruce Newberg , a former Drexel trader .
```

Assume **we can only use** the following two probability distributions:

- A *translation probability* $P(w \mid W)$ where w is the lowercase variant of the TrueCase word W (note that the TrueCase word might still be lowercase). The function `lower` can be used to lowercase a word, e.g. "*HAL9001*".`lower`() = "*hal9001*"
- A *bigram probability* $P(W \mid W')$. A language model $P(W_1, \ldots, W_n)$ is used to provide the probability of a sentence. A bigram language model approximates the probability of a sentence as follows:

$$\Pr(W_1, \ldots, W_n) \approx \prod_{i=1}^{n} P(W_i \mid W_{i-1})$$

We assume that $W_{-1} = w_{-1} = \textit{none}$ is a dummy word that begins each sentence.

a. (8pts) Complete the following formula to provide a model of the TrueCasing task by using only the translation probability $P(w \mid W)$ and the bigram probability $P(W \mid W')$:

$$\begin{aligned} W_1^*, \ldots, W_n^* &= \arg \max_{W_1, \ldots, W_n} \Pr(W_1, \ldots, W_n \mid w_1, \ldots, w_n) \\ &= \textit{provide this formula} \end{aligned}$$

Answer:

$$\begin{aligned} W_1^*, \ldots, W_n^* &= \arg \max_{W_1, \ldots, W_n} P(W_1, \ldots, W_n \mid w_1, \ldots, w_n) \\ &= \frac{P(W_1, \ldots, W_n) \cdot P(w_1, \ldots, w_n \mid W_1, \ldots, W_n)}{P(w_1, \ldots, w_n)} \\ &\approx P(W_1, \ldots, W_n) \cdot P(w_1, \ldots, w_n \mid W_1, \ldots, W_n) \\ &= \prod_{i=1}^{n} \underbrace{P(w_i \mid W_i)}_{\textit{translation probability}} \cdot \underbrace{P(W_i \mid W_{i-1})}_{\textit{bigram language model}} \end{aligned}$$

b. (4pts) Using maximum likelihood, provide a formula to estimate the the translation probability parameters $P(w \mid W)$ for lowercase words w and TrueCase words W. Assume you **only** have access to a sufficient amount of TrueCase text.

Answer: For each TrueCase word W convert it to lowercase w using $w = W.\texttt{lower}()$ and count $f(w, W)$ and $f(W)$. Then,

$$P(w \mid W) = \frac{f(w, W)}{\sum_{w'} f(w', W)}$$

c. (8pts) Provide the pseudo code or recurrence relation that will find the score for $W_1^*, \ldots, W_n^*$, which is the most likely TrueCase output according to our model.

Provide the $O(\cdot)$ time complexity of the algorithm in terms of input length, which is the number of lowercase words $|w|$ and the number of TrueCase words $|W|$.

Your algorithm has to be a polynomial time algorithm (i.e. not the obvious brute force algorithm which will be exponential in the input length).

Answer: We can use the Viterbi algorithm. For input $w_1, \ldots, w_n$ we can efficiently compute the best score upto TrueCase word candidate W_{i+1} recursively as follows:

$$\text{Viterbi}[i+1, W_{i+1}] = \max_{W_i} \{\text{Viterbi}[i, W_i] \times P(w_{i+1} \mid W_{i+1}) \times P(W_{i+1} \mid W_i)\}$$

We initialize the recursion with the value for $\text{Viterbi}[1, W_1]$.

If the number of lowercase words is $|w|$ and the number of TrueCase words is $|W|$ then the time complexity is $O(|w| \cdot |W|^2)$.

(2) Consider the sentence pair:

e and the program has been implemented

f le programme a été mis en application

Assuming **f** has I words and **e** has J words the probability of each alignment vector **a** according to IBM Model 1 is:

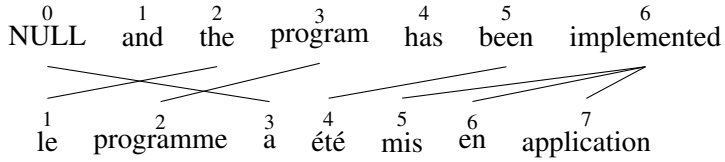
$$\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) = \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\Pr(\mathbf{f} \mid \mathbf{e})} = \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})} = \frac{\prod_{i=1}^{I} t(f_i \mid e_{a_i})}{\sum_{\mathbf{a}} \prod_{i=1}^{I} t(f_i \mid e_{a_i})}$$

a. (2pts) Provide the number of possible alignments for the above example sentence pair. Assume there is an extra English word e_0 which is the *NULL* word. Write the answer as x^y where x and y are integers.

Answer:

$$(J+1)^I = (6+1)^7 = 7^7 = 823543$$

b. (3pts) Given the alignment:



Provide the alignment vector **a**.

Answer:

$$\mathbf{a} = (2, 3, 0, 5, 6, 6, 6)$$

c. (5pts) If the alignment is given, we can use relative frequency to compute the maximum likelihood (ML) estimate for the parameters $t(f \mid e)$. For the alignment given above, provide the ML estimate for the following parameters:

1. $t(le \mid the)$
2. $t(le \mid and)$
3. $t(mis \mid implemented)$
4. $t(en \mid implemented)$
5. $t(application \mid \text{NULL})$

The answer can be written as a fraction.

Answer:

$$t(f \mid e) = \frac{\text{count}(f, e)}{\sum_f \text{count}(f, e)}$$

(1) $t(le \mid the) = 1$, (2) $t(le \mid and) = 0$, (3) $t(mis \mid implemented) = \frac{1}{3}$, (4) $t(en \mid implemented) = \frac{1}{3}$, (5) $t(application \mid \text{NULL}) = 0$

d. (10pts) Use add one smoothing to estimate the values of the same five parameters above. Assume that the entire corpus only has the one sentence pair given above. The answer can be written as a fraction.

Answer:

$$t(f \mid e) = \frac{1 + \text{count}(f, e)}{\mathcal{V}_f + \sum_f \text{count}(f, e)}$$

Vocabulary size of French is: $\mathcal{V}_f = 7$ (1) $t(le \mid the) = \frac{1+1}{7+1} = \frac{1}{4}$, (2) $t(le \mid and) = \frac{1+0}{7+0} = \frac{1}{7}$, (3) $t(mis \mid implemented) = \frac{1+1}{7+3} = \frac{1}{5}$, (4) $t(en \mid implemented) = \frac{1+1}{7+3} = \frac{1}{5}$, (5) $t(application \mid \text{NULL}) = \frac{1+0}{7+1} = \frac{1}{8}$