HW 1 - Language Models and HMM

Andrei A Simion

September 16, 2023

Problem 1

Suppose we have 3 sentences:

$$x^{(1)} = (\text{START}, a, b, c, \text{STOP})$$
$$x^{(2)} = (\text{START}, a, c, d, \text{STOP})$$
$$x^{(3)} = (\text{START}, b, d, e, a, \text{STOP})$$

What is $\mathcal{V}$? What are the Bigram Model estimates that we get by maximum likelihood if we also use Unigram smoothing? Specifically, for this problem, set the parameters to be smooth: $\theta(v|u) = \frac{count(u,v)+k\theta^\star(v)}{count(u)+k}$ where $\theta^\star(u)$ are the maximum likelihood Unigram estimates and $k > 0$. Fill in the small Python notebook and make sure all the asserts pass in the version you pass back to us. Also assuming we tried to fit the standard Bigram model, if we write out the likelihood as $\mathcal{L} = \sum_{i=1}^{3} \log p(x^{(i)})$ rewrite this as $\mathcal{L} = \sum_{(u,v)} count(u,v) \log \theta(v|u)$ and identify all the relevant terms and counts for $(START, a)$, (b, c) and (a, e).

Problem 2

Suppose we have $N - Gram$ Language Models with maximum likelihood estimates $\theta^\star(w|v,u)$, $\theta^\star(v|u)$ and $\theta^\star(v)$. Assume we have the START and STOP tokens as usual and assume the vocabulary is $\mathcal{V}$ with size $|\mathcal{V}|$. Suppose $count(u,v,w), count(u,v)$ and $count(u)$ are the Trigram, Bigram, and Unigram counts as in Lecture.

Show the following:

- Suppose we use $\theta(v|u) = \frac{count(u,v)+1}{count(u)+|\mathcal{V}|}$ to smooth the probabilities. Show $\theta(v|u) > 0$ and this is a probability distribution for any u.

- Suppose we use $\theta(v|u) = \frac{count(u,v)+k}{count(u)+k|\mathcal{V}|}$ with $k > 1$ to smooth the probabilities. Show $\theta(v|u) > 0$ and this is a probability distribution for any u.

- Suppose we use $\theta(v|u) = \frac{count(u,v)+k\theta^\star(v)}{count(u)+k}$ with $k > 1$ to smooth the probabilities. Show $\theta(v|u) > 0$ and this is a probability distribution for any u.

- Suppose we use $\theta(w|v,u) = \lambda_1\theta^\star(w|v,u) + \lambda_2\theta^\star(v|u) + \lambda_3\theta^\star(v)$ where $\lambda_i \geq 0$ and $\sum_{i=1}^3 \lambda_i = 1$. Show $\theta(w|v,u)$ is a probability distribution for any (u,v). How can we ensure the Language Model with this θ does not have infinite Perplexity?

- As in lecture, suppose we use a map which assigns a bucket to $count(u,v)$ for any (u,v); for example suppose if $count(u,v) \leq 2$ we get 1 and if $2 \leq count(u,v) \leq 5$ we get 3. Suppose $\Pi(u,v) = b$ is the mapping for each (u,v). Let $\theta(w|v,u) = \lambda_1^{\Pi(u,v)}\theta^\star(w|v,u) + \lambda_2^{\Pi(u,v)}\theta^\star(v|u) + \lambda_3^{\Pi(u,v)}\theta^\star(v)$ where $\lambda_i^b \geq 0$ and $\sum_{i=1}^3 \lambda_i^b = 1$ (assume there is a finite set of buckets which cover the entire range of $count(u,v)$). Show $\theta(w|v,u)$ is a probability distribution for any (u,v).

- Suppose we have a Bigram language model and we define the set $\mathcal{B}(u) = \{v : count(u,v) = 0\}$ and $\mathcal{A}(u) = \{v : count(u,v) > 0\}$. Suppose we use the discounting technique where $count^\beta(u,v) = count(u,v) - \beta$ for $v \in \mathcal{A}(u)$ and $0 < \beta < 1$. Assume $a(u) = 1 - \sum_{v\in\mathcal{A}(u)} \frac{count^\beta(u,v)}{count(u)} > 0$ and we set

$$\theta^\beta(v|u) = \begin{cases} \frac{count^\beta(u,v)}{count(u)}, & \text{if } v \in \mathcal{A}(u) \\ a(u)\frac{\theta^\star(v)}{\sum_{w\in\mathcal{B}(u)}\theta^\star(w)}, & \text{if } v \in \mathcal{B}(u) \end{cases}$$

  Show that $\theta^\beta(v|u) > 0$ and that this is a probability distribution for each u.

Problem 3

Suppose we have a Language Model where we set $\theta(w|v,u) = \lambda_1\theta^\star(w|v,u) + \lambda_2\theta^\star(v|u) + \lambda_3\theta^\star(v)$ where $\lambda_i \geq 0$ and $\sum_{i=1}^{3}\lambda_i = 1$. Here, the $\theta^\star$ parameter estimates were computed on some other Training set. Suppose we have a Development corpus $\{x^{(i)}\}_{i=1}^{N}$ and we formulate the likelihood under this as $\mathcal{L}(\lambda_1,\lambda_2,\lambda_3) = \frac{\sum_{i=1}^{N}\log p(x^{(i)})}{N} = \frac{\mathit{count}(u,v,w)\log\theta(w|v,u)}{N}$. Assume we maximize this likelihood with respect to λ_i. Show that doing this is the same as minimizing the Perplexity of this Language Model on the Development set.

Problem 4

Suppose we use a map which assigns a bucket to $\mathit{count}(u,v,w)$ for any (u,v,w); for example suppose if $\mathit{count}(u,v,w) \leq 2$ we get 1 and if $2 \leq \mathit{count}(u,v,w) \leq 5$ we get 3. Suppose $\Pi(u,v,w)$ is the mapping for each (u,v,w). Let $\theta(w|v,u) = \lambda_1^{\Pi(u,v,w)}\theta^\star(w|v,u) + \lambda_2^{\Pi(u,v,w)}\theta^\star(v|u) + \lambda_3^{\Pi(u,v,w)}\theta^\star(v)$ where $\lambda_i^b \geq 0$ and $\sum_{i=1}^{3}\lambda_i^b = 1$ (assume there is a finite set of buckets which cover the entire range of $\mathit{count}(u,v,w)$). What is wrong if we do this?

Problem 5

Suppose we have a Bigram Language Model with nonzero estimates $\theta^\star(u,v)$. Suppose we want to find the maximum probability sequence $(x_0,x_1,x_2,\ldots,x_T)$ where $x_0 =$ START and $x_T =$ STOP. Design an $O(T|\mathcal{V}|^2)$ algorithm to do this.

Problem 6

Suppose we have training data where we have 100 examples with $x_1 = a, x_2 = b$ and $y_1 = A, y_2 = B$, 100 examples with $x_1 = a, x_2 = c$ and $y_1 = A, y_2 = C$ and 800 examples with $x_1 = c, x_2 = d$ and $y_1 = B, y_2 = D$. Suppose we first a Bigram HMM Tagger on this data. What are the estimated parameters? Also, what are the highest probability sequences y we get when we tag (a,c) and (c,d).

Problem 7

Assume that we have a Bigram HMM Tagger and we estimate all the parameters $\theta(v|u)$ and $\vartheta(x|u)$. Assume we want to find

$$\max_{x_1,\dots,x_T,y_0,\dots,y_T} p(x_1,\dots,x_T,y_0,\dots,y_T)$$

In the above as usual we have $y_0 = \text{START} = *$ and $y_T = \text{STOP}$. Note that this problem is different from lecture, we want to find not just the highest sequence y but we jointly want (x, y). Devise an efficient method to do this and write its computational complexity in Big O notation.

Problem 8

For this problem, you will implement a Trigram HMM Tagger on the CONLL dataset. The tags here are for Named Entity Recognition (NER) and there are 5 tag types: person names (PER), organizations (ORG), locations (LOC), miscellaneous names (MISC) and (O) for other. Each of the first 4 tags can be either of the type I-TAG or B-TAG. A new entiry starts with I-TAG but if the same type of entity is next to itself (but is a different one) we start this with B-TAG. For example, "O I-PER O" means there is one person but "O I-MISC B-MISC O" means there are two names right next to each other for different entities; for example this might be "the French Mediterranean is". Ste by step, do the parts below and fill in the attached notebook.

- You are given Training data *ner_train.dat* and Test data $ner_dev.dat$ where each line of the training set is of the form (x, u) where x is a word and u is a tag. Between different sentences, there is an empty line. The Development has just words, you need to get the tags. The goal of this problem is to code up the HMM tagger and compare it with the another tagger via some utilities we give you.

- Run the given $count\_freqs.py$ function on *ner_train.dat* to get a new file *ner.counts*; you can do this: *python* $count\_freqs.py$ *ner_train.dat* $>$ *ner.counts*. The output of this command is several lines where for example we get $count(u, x)$ on a line that has WORDTAG in it. Similarly, we have other lines with 2-GRAM or 3-GRAM, these have the counts of $count(u, v)$ and $count(u, v, w)$ respectively. Basically this utility gives us the counts we need to get the emission $\vartheta(u|x)$ and transition $\theta(w|v, w)$ probabilities.

Part A Fill in the function *get_q_e_counts* which gives is the counts for θ and ϑ. Here, we have $q(w|v, w) = \theta(w|v, w)$ and $e(x|u) = \vartheta(x|u)$.

Part B Fill in the function *transform_data* which takes *e_counts* and writes to a new file *ner_train_rare.dat* the same training data $ner_train.dat$ but all words x which have a count < 5 are replaced with _RARE_.

Part C Fill in the *get_emission* function for $\vartheta(x|y)$.

Part D Re-run the count utility given on the *ner_train_rare.dat* dataset created to get *ner_rare.counts*.

Part E Fill in the *baseline_ner_tagger*. This baseline decodes via

$$\max_{y_1,\cdots,y_T} \vartheta(x_1|y_1)\ldots\vartheta(x_T|y_T)$$

Get the predictions for the Development data by creating

ner_dev.baseline_predictions

Part F Use the eval utility to get the performance of this tagger. You should see about a 31% overall F-Measure.

```
python eval_ne_tagger.py ner_dev.key ner_dev.baseline_predictions
```

Part G Fill in the *hmm_ner_tagger*. Get the predictions for the Development data by creating *ner_dev.hmm_predictions*.

Part H Use the eval utility to get the performance of this tagger. You should see about a 68% overall F-Measure.

```
python eval_ne_tagger.py ner_dev.key ner_dev.hmm_predictions
```