

Solutions to Quiz 1

Question 1

- The first definition,

$$\pi(k, u, v) = \max_{w \in \mathcal{S}_{k-2}} (\pi(k-1, w, u) \times q(v|u) \times e(x_k|v))$$

is **correct**. To see this, note the similarity to the definition of the Viterbi algorithm for trigram HMMs, which is as follows:

$$\pi(k, u, v) = \max_{w \in \mathcal{S}_{k-2}} (\pi(k-1, w, u) \times q(v|w, u) \times e(x_k|v))$$

The new definition is identical, except that we have replaced $q(v|w, u)$ with $q(v|u)$. In some sense, we are again decoding with a trigram HMM, but whenever we look up $q(v|w, u)$ we are getting a value $q(v|u)$ that does not actually depend on w . From this interpretation, it is clear that the algorithm is correct.

- The second definition, $\pi(0, *) = 1$ and

$$\pi(k, v) = \max_{u \in \mathcal{S}_{k-1}} (\pi(k-1, u) \times q(v|u) \times e(x_k|v))$$

is also **correct**. In this algorithm, $\pi(k, v)$ is the highest probability for any tag sequence ending in tag v in position k . The justification for the algorithm is then very similar to that for the Viterbi algorithm for trigram HMMs. We search over all tags u at position $k-1$, and multiply together two terms: 1) $\pi(k-1, u)$, which is the highest probability for any tag sequence ending in u at position $k-1$; 2) $q(v|u) \times e(x_k|v)$, the probability of tag v following tag u , combined with the probability of generating word x_k from tag v .

Note that this definition is actually preferable to the first definition, because it leads to an algorithm that is more efficient (although both are correct). The first definition give an algorithm with a runtime of $O(n|\mathcal{S}|^3)$, whereas the second definition gives a runtime of $O(n|\mathcal{S}|^2)$. (Here we use $|\mathcal{S}|$ to refer to the number of possible tags.)

- The third definition, $\pi(0, *) = 1$ and

$$\pi(k, v) = \max_{u \in \mathcal{S}_{k-1}} (\pi(k-2, u) \times q(v|u) \times e(x_k|v))$$

is **incorrect**. In this case we have mistakenly multiplied in $\pi(k-2, u)$. There is no coherent explanation for this algorithm (and in fact for $k=1$, $\pi(k-2, u)$ is undefined, so the algorithm fails at the very first position).

Question 2

The first statement, “*There may be some bigrams u, v such that $q_{\text{BO}}(v|u) < 0$* ” is **true**. In particular, for any bigram u, v such that $\text{Count}(u, v) = 1$, we have $\text{Count}^*(u, v) = -0.5$ and $q_{\text{BO}}(v|u) = \frac{-0.5}{\text{Count}(u)}$, which is negative.

The second statement, “*There may be some words u such that $\sum_{v \in \mathcal{V} \cup \{\text{STOP}\}} q_{\text{BO}}(v|u) \neq 1$* ” is **false**. Intuitively, the fact that some values of Count^* are negative does nothing to break the method we have used to ensure that the distribution sums to 1. More formally, for any u we have

$$\sum_{v \in \mathcal{V} \cup \{\text{STOP}\}} q_{\text{BO}}(v|u) = \sum_{v \in \mathcal{A}(u)} q_{\text{BO}}(v|u) + \sum_{v \in \mathcal{B}(u)} q_{\text{BO}}(v|u)$$

which follows because $\mathcal{A}(u) \cup \mathcal{B}(u) = \mathcal{V} \cup \{\text{STOP}\}$, and $\mathcal{A}(u) \cap \mathcal{B}(u) = \emptyset$. It can then be verified that

$$\sum_{v \in \mathcal{A}(u)} q_{\text{BO}}(v|u) = \sum_{v \in \mathcal{A}(u)} \frac{\text{Count}^*(u, v)}{\text{Count}(u)} = 1 - \alpha(u)$$

and

$$\sum_{v \in \mathcal{B}(u)} q_{\text{BO}}(v|u) = \sum_{v \in \mathcal{B}(u)} \left(\alpha(u) \times \frac{q_{\text{ML}}(v)}{\sum_{v \in \mathcal{B}(u)} q_{\text{ML}}(v)} \right) = \alpha(u)$$

giving

$$\sum_{v \in \mathcal{A}(u)} q_{\text{BO}}(v|u) + \sum_{v \in \mathcal{B}(u)} q_{\text{BO}}(v|u) = 1 - \alpha(u) + \alpha(u) = 1$$

Question 3

Language Model 1 gives lower perplexity on this corpus. It can be verified that under language model 1, $p(\text{the dog STOP}) = 1$, whereas under language model 2, $p(\text{the dog STOP}) = 0.5$. Thus language model 1 gives higher probability for the test corpus, and it follows through the definition of perplexity that it gives a lower perplexity on the test corpus.

Question 4

The following definition is correct:

$$\pi(0, *, *) = 1$$

$$\pi(k, u, v) = \max_{w \in S_{k-2}} (\pi(k-1, w, u) \times q(v|w, u) \times m(v))$$

where $m(v) = \max_{x \in \mathcal{V}} e(x|v)$.

Under this definition, $\pi(k, u, v)$ is the highest probability for any tag sequence $y_1 \dots y_k$ paired with any word sequence $x_1 \dots x_k$. More formally, the justification for this algorithm is very similar to that for the Viterbi algorithm for trigram taggers, but where we have a new definition for r :

$$r(y_{-1}, y_0, y_1, \dots, y_k) = \max_{x_1 \dots x_k} \left(\prod_{i=1}^k q(y_i|y_{i-2}, y_{i-1}) \prod_{i=1}^k e(x_i|y_i) \right)$$

where again we have

$$\pi(k, u, v) = \max_{\langle y_{-1}, y_0, y_1, \dots, y_k \rangle \in S(k, u, v)} r(y_{-1}, y_0, y_1, \dots, y_k)$$

Note that under this definition of r ,

$$r(y_{-1}, y_0, y_1, \dots, y_k) = \left(\prod_{i=1}^k q(y_i|y_{i-2}, y_{i-1}) \prod_{i=1}^k m(y_i) \right)$$

because when taking the max over $x_1 \dots x_k$ each $e(x_i|y_i)$ term can be maximized separately.

Question 5

We need to ensure that

$$\sum_{x_1 \dots x_n \in \mathcal{V}^\dagger} p(x_1 \dots x_n) = 1$$

where $\mathcal{V}^\dagger$ is again the set of all sentences ending in STOP. First, note that for any $n > 0$ there are 3^{n-1} possible sentences. For example for $n = 1$ we have the single sentence, *STOP*; for $n = 2$ we have three possible sentences, *the STOP*, *a STOP*, *dog STOP*; for $n = 3$ we have nine possible sentences, because the first word can be any one of *the*, *a* or *dog*, the second word can be any one of these three words, and the third word must be *STOP*; and so on.

From this it follows that if we set $\gamma = 1/3^{n-1}$, we have

$$\sum_{x_1 \in \mathcal{V}^\dagger} p(x_1 \dots x_n) = 0.5$$

(here we have a sum over all sentences of length 1: there is only one sentence of length 1, namely *STOP*). We have

$$\sum_{x_1, x_2 \in \mathcal{V}^\dagger} p(x_1 \dots x_n) = 3 \times \frac{1}{3} \times 0.5^2 = 0.5^2$$

(because there are 3 sentences of length 2, each with probability $\frac{1}{3} \times 0.5^2$). We have

$$\sum_{x_1, x_2, x_3 \in \mathcal{V}^\dagger} p(x_1 \dots x_n) = 3^2 \times \frac{1}{3^2} \times 0.5^3 = 0.5^3$$

(because there are 3^2 sentences of length 3, each with probability $\frac{1}{3^2} \times 0.5^3$)

$$\sum_{x_1, x_2, x_3, x_4 \in \mathcal{V}^\dagger} p(x_1 \dots x_n) = 3^3 \times \frac{1}{3^3} \times 0.5^4 = 0.5^4$$

and so on, hence

$$\sum_{x_1 \dots x_n \in \mathcal{V}^\dagger} p(x_1 \dots x_n) = \sum_{n=0}^{\infty} 0.5^n = 1$$

The key idea is that for all sentences of length n , we split the probability evenly between the 3^{n-1} possible sentences of length n , by setting

$$p(x_1 \dots x_n) = \frac{1}{3^{n-1}} \times 0.5^n$$

Question 6

Under maximum-likelihood estimation, we get the following parameter values: $q(D|*, *) = 1$, $q(N|*, D) = 1$, $q(V|D, N) = q(STOP|D, N) = 0.5$, $q(D|N, V) = 1$, $q(N|V, D) = 1$, with all other q values being 0. In addition, we get $e(the|D) = 1$, $e(dog|N) = e(saw|N) = 0.25$, $e(cat|N) = 0.5$, $e(saw|V) = 1$.

For the sentence *the cat saw the saw*, under these parameter values, there is only one tag sequence whose probability is greater than 0, namely $D N V D N STOP$. This is easily verified because $q(D|*, *) = 1$, hence D must be at position 1; $q(N|*, D) = 1$, so N must be at position 2; $q(STOP|D, N) = q(V|D, N) = 0.5$, hence V must be at position 3, because the sentence is of length 5 so $STOP$ cannot be at position 3; and so on.

This tag sequence has probability

$$\begin{aligned} & q(D|*, *) \times q(N|*, D) \times q(V|D, N) \times q(D|N, V) \times q(N|V, D) \times q(STOP|D, N) \\ & \times e(the|D) \times e(cat|N) \times e(saw|V) \times e(the|D) \times e(saw|N) \\ = & 0.5 \times 0.5 \times 0.5 \times 0.25 = 0.03125 \end{aligned}$$

Question 7

The key idea is to intuitively allow state 1 in the HMM to play the role of the word *a*, and state 2 to play the role of the word *the*. We set $e(a|1) = e(the|2) = 1$, and all other emission parameters set to 0. We set $q'(1|*) = q(a|*) = 0.6$, $q'(2|*) = q(the|*) = 0.4$, $q'(1|1) = q(a|a) = 0.9$, $q'(STOP|1) = q(STOP|a) = 0.1$, $q'(2|2) = q(the|the) = 0.8$, and so on.

We illustrate how these parameters work on a simple example, the sentence *a a*. Under the language model, this has probability

$$q(a|*) \times q(a|a) \times q(a|a) \times q(STOP|a)$$

If we consider the HMM run on this sequence, there is only one tag sequence with probability greater than 0, namely 1 1 1 STOP (any other tag sequence will have *a* paired with state 2 at one or more positions, which has probability $e(a|2) = 0$). Thus if we sum over all tag sequences of length 3, we get

$$\sum_{y_1 \dots y_{n+1}} p'(a \ a \ a, y_1 \dots y_{n+1}) = p'(a \ a \ a, 1 \ 1 \ 1 \ STOP)$$

However it is easily verified that

$$\begin{aligned} p'(a \ a \ a, 1 \ 1 \ 1 \ STOP) &= q'(1|*) \times q'(1|1) \times q'(1|1) \times q'(STOP|1) \\ &= q(a|*) \times q(a|a) \times q(a|a) \times q(STOP|a) \end{aligned}$$

hence the desired condition is satisfied for this sentence. A similar argument can be made for any sequence of words.

The key insight here is to leverage the fact that there are as many HMM states as there are words, to define a one-to-one mapping between states and words (1 corresponds to *a*, and 2 corresponds to *the*), to define emission parameters to be 0 or 1, enforcing this mapping, and finally to define the q' parameters appropriately. Note that this method can be used for *any* language model/HMM pair where the HMM has as many states as there are words.

There is a second solution to this problem, where we assign state 2 to *a* and state 1 to *the*, giving $e(a|2) = e(the|1) = 1$, $q'(2|*) = q(a|*) = 0.6$, $q'(1|*) = q(the|*) = 0.4$, $q'(2|2) = q(a|a) = 0.9$, $q'(STOP|2) = q(STOP|a) = 0.1$, $q'(1|1) = q(the|the) = 0.8$, and so on. This was ruled out though because the question specified that $q'(1|*) = 0.6$.