CS B551, Fall 2016, In-class individual activity #2

Write your name and IU user id: ______________________________

A certain little-known human language consists of only nouns and verbs. A sentence begins with a noun with probability 75% and with a verb with probability 25%. If a given word in a sentence is a noun, there is a 75% probability that the next word is a verb and a 25% probability that the next word is a noun. If a given word is a verb, there is a 75% probability that the next word in the sentence is a noun, and a 25% probability that is a verb.

We can model this language using a Markov chain with two states: noun and verb.

1. Draw the Markov chain described above.

2. What is the probability that the first three words of a sentence are *all* nouns?

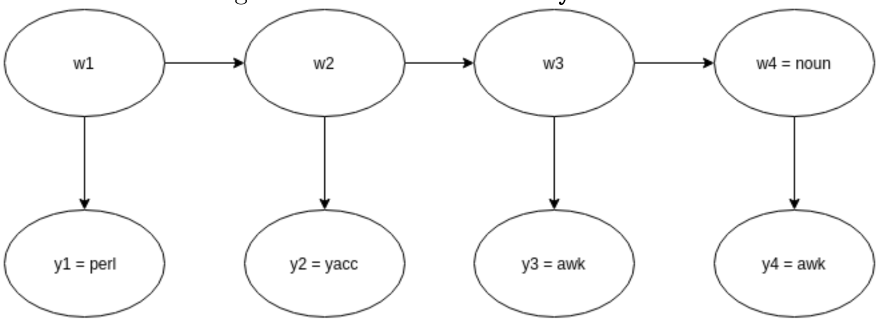
$$(0.75)(0.25)(0.25) \approx 0.0469$$

3. What is the probability that the third word of a sentence is a noun?
   For the third word to be a noun, the first three words must be NNN, NVN, VNN, or VVN. Thus the probability that the third word is a noun is:

$$(0.75)(0.25)(0.25) + (0.75)(0.75)(0.75) + (0.25)(0.75)(0.25) + (0.25)(0.25)(0.75) = .5625$$

4. There are only four words in the language: `awk`, `yacc`, `grep`, and `perl`. Each of these words can be either a noun or a verb. Of all noun occurrences, 10% are the word `awk`, 20% are the word `yacc`, 40% are the word `grep`, and 30% are the word `perl`. Of all verb occurrences, 20% are the word `awk`, 30% are the word `yacc`, 45% are the word `grep`, and 5% are the word `perl`.

   (a) Using this model and the Variable Elimination algorithm, find the probability that the fourth word of the following sentence is a noun: `Perl yacc awk awk.`



We'd like to calculate $p(s_4 = noun|y_1 = perl,\ y_2 = yacc,\ y_3 = awk,\ y_4 = awk)$, which is equal to:

$$\frac{p(s_4 = noun, y_1 = perl,\ y_2 = yacc,\ y_3 = awk,\ y_4 = awk)}{p(y_1 = perl,\ y_2 = yacc,\ y_3 = awk,\ y_4 = awk)}$$

First we compute the numerator using Variable elimination. Let's use an elimination ordering of s_1, s_2, s_3. The initial set of factors F involved in our problem is: $\{P(s_4 = noun|s_3), P(s_3|s_2), P(s_2|s_1), P(s_1), P(y_4 = awk|s_4 = noun), P(y_3 = awk|s_3), P(y_2 = yacc|s_2), P(y_1 = perl|s_1)\}$.

First we eliminate s_1. The subset of factors in F involving s_1 is $\{P(s_2|s_1), P(s_1), P(y1 = perl|s_1)\}$. We want to sum over all possible values of s_1 of the product of these three factors to eliminate s_1; the resulting expression will still involve s_2, so we'll get a new factor $\tau_1(s_2)$:

$$\begin{aligned}\tau_1(s_2) &= \sum_{s_1 \in \{noun, verb\}} P(s_2|s_1)P(s_1)P(y_1 = perl|s_1) \\ &= P(s_2|s_1 = noun)P(s_1 = noun)P(y_1 = perl|s_1 = noun) \\ &\quad + P(s_2|s_1 = verb)P(s_1 = verb)P(y_1 = perl|s_1 = verb)\end{aligned}$$

Plugging in numbers, we get:

$$\begin{aligned}\tau_1(s_2 = noun) &= (0.25)(0.75)(0.3) + (0.75)(0.25)(0.05) \approx 0.065625 \\ \tau_1(s_2 = verb) &= (0.75)(0.75)(0.3) + (0.25)(0.25)(0.05) \approx 0.171875\end{aligned}$$

Our new set F of factors is now $\{P(y_4 = awk|s_4 = noun), P(s_4 = noun|s_3), P(s_3|s_2), P(y_3 = awk|s_3), P(y_2 = yacc|s_2), \tau_1(s_2)\}$. We've eliminated s_1!

Now we eliminate s_2:

$$\tau_2(s_3) = \sum_{s_2 \in \{noun, verb\}} P(s_3|s_2)\tau(s_2)P(y_2 = yacc|s_2)$$

so:

$$\begin{aligned}\tau_2(s_3 = noun) &= (0.25)(0.065625)(0.2) + (0.75)(0.171875)(0.3) \approx 0.041953 \\ \tau_2(s_3 = verb) &= (0.75)(0.065625)(0.2) + (0.25)(0.171875)(0.3) \approx 0.022734\end{aligned}$$

Our new set F of factors is now $\{P(y_4 = awk|s_4 = noun), P(s_4 = noun|s_3), P(y_3 = awk|s_3), \tau_2(s_3)\}$. We've eliminated s_2!

Now eliminate s_3:

$$\tau_2(s_4) \quad = \sum_{s_3 \in \{noun, verb\}} P(y_4 = awk|s_4 = noun)P(s_4 = noun|s_3)\tau(s_3)P(y_3 = awk|s_3)$$

so:

$$\begin{aligned}
\tau_3(s_4 = noun) &= (0.1)\{(0.25)(0.041953)(0.1) + (0.75)(0.022734)(0.2)\} \approx 0.0004458925 \\
\tau_3(s_4 = verb) &= (0.2)\{(0.75)(0.041953)(0.1) + (0.25)(0.022734)(0.2)\} \approx 0.000856635
\end{aligned}$$

Now τ_3 is the numerator we wanted to calculate ($p(s_4, y_1 = perl,\ y_2 = yacc,\ y_3 = awk,\ y_4 = awk)$). To compute the denominator, we can just marginalize out over the two possible values of s_4 (noun or verb):

$$p(s_4 = noun|y_1 = perl,\ y_2 = yacc,\ y_3 = awk,\ y_4 = awk) = \frac{0.0004458925}{0.0004458925 + 0.000856635} \approx 0.3423286$$

(b) Using this model and the Viterbi algorithm, find the most likely part-of-speech state sequence for the following sentence: `Perl yacc awk awk.`
To simplify the notation, let p_{ij} be the transition probabilities, $e_i(y)$ be the emission probability of outputing y given state i, and w_i be the initial probability for state i. As we saw in class, we'll define $v_i(t)$ to be the probability of the most possible path ending in state i at position t, i.e.:

$$v_i(t) = \max_{q_1, q_2, q_{t-1}} = P(q_1, q_2, \ldots, q_{t-1}, q_t = i|y_1, y_2, \ldots, y_t)$$

As we saw, Viterbi can be implemented via dynamic programming. In other words, we can calculate $v_i(t)$ in an iterative way:

$$\begin{aligned}
\text{for } t = 1: \quad v_j(1) &= \quad w_j e_j(y_t) \\
\text{for } t > 1: \quad v_j(t) &= \max_{i \in \{n, v\}} [v_i(t-1)p_{ij}]e_j(y_t)
\end{aligned}$$

After finishing calculating all the $v_j(t)$, then the state sequence that outputs the largest probability is what we want. Specifically,

$$\begin{aligned}
v_{noun}(1) &= w_{noun}e_{noun}(y_1) \\
&= w_{noun}e_{noun}(perl) \\
&= 0.75 \times 0.3 \\
&= 0.225
\end{aligned}$$

$$\begin{aligned}
v_{verb}(1) &= w_{verb}e_{verb}(y_1) \\
&= w_{verb}e_{verb}(perl) \\
&= 0.25 \times 0.05 \\
&= 0.0125
\end{aligned}$$

$$\begin{aligned}
v_{noun}(2) &= \max\{v_{noun}(1)P_{noun,noun}, v_{verb}(1)P_{verb,noun}\}e_{noun}(yacc) \\
&= \max\{0.225 \times 0.25, \cancel{0.125} \times 0.75\} \times 0.2 \\
&= 0.01125 \qquad 0{,}0125
\end{aligned}$$

$$
\begin{aligned}
v_{verb}(2) &= \max\{v_{noun}(1)P_{noun,verb}, v_{verb}(1)P_{verb,verb}\}e_{verb}(yacc) \\
&= \max\{0.225 \times 0.75, 0.125 \times 0.25\} \times 0.3 \\
&= 0.16875 \times 0.3 = 0.050625
\end{aligned}
$$

$q_2^* = verb$

$$
\begin{aligned}
v_{noun}(3) &= \max\{v_{noun}(2)P_{noun,noun}, v_{verb}(2)P_{verb,noun}\}e_{noun}(awk) \\
&= \max\{0.01125 \times 0.25, \cancel{0.16875}\ 0.050625 \times 0.75\} \times 0.1 \\
&= \cancel{0.01265625} \\
& \quad 0.003796875
\end{aligned}
$$

$q_3^* = noun$

$$
\begin{aligned}
v_{verb}(3) &= \max\{v_{noun}(2)P_{noun,verb}, v_{verb}(2)P_{verb,verb}\}e_{verb}(awk) \\
&= \max\{0.01125 \times 0.75, \cancel{0.16875}\ 0.050625 \times 0.25\} \times 0.2 \\
&= \cancel{0.0084375} \\
& \quad 0.002531250
\end{aligned}
$$

$$
\begin{aligned}
v_{noun}(4) &= \max\{v_{noun}(3)P_{noun,noun}, v_{verb}(3)P_{verb,noun}\}e_{noun}(awk) \\
&= \max\{\cancel{0.01265625} \times 0.25, \cancel{0.0084375} \times 0.75\} \times 0.1 \\
& \quad 0.003796875 \qquad 0.002531250 \\
&= \cancel{0.0006328125} \\
& \quad 0.0001898
\end{aligned}
$$

$$
\begin{aligned}
v_{verb}(4) &= \max\{v_{noun}(3)P_{noun,verb}, v_{verb}(3)P_{verb,verb}\}e_{verb}(awk) \\
&= \max\{\cancel{0.01265625} \times 0.75, \cancel{0.0084375} \times 0.25\} \times 0.2 \\
&= \cancel{0.0018984375} \\
& \quad 0.0005695
\end{aligned}
$$

$q_4^* = verb$ since $0.00057 > 0.00019$

Now we know that the optimal state sequence ends with verb, i.e. $q_4^* = verb$, since $v_{verb}(4) > v_{noun}(4)$. ✓ We now "trace backwards" from there. In the maximization inside $v_{verb}(4)$, we chose the first of the two possibilities (that the prior state was a noun), so $q_3^* = noun$. ✓ In the maximization where we computed $v_{noun}(3)$, we chose the second possibility (that prior state was a verb), so $q_2^* = verb$, and so on. So the best state sequence is noun, verb, noun verb. ✓ ✓