UMass CS690N: Advanced Natural Language Processing, Spring 2018

Assignment 3: Semi-Markov models

Model

Consider a semi-Markov CRF name tagging (segmentation) model with two types of segments: O and NAME, and the probability of a segmentation is, as in the Sarawagi and Cohen (2004):[1]

$$P(s \mid x) \propto \prod_j \exp\left(\theta^\mathsf{T} f(y_j, y_{j-1}, t_j, u_j, x)\right)$$

where j indexes a segment from segmentation s, y_j and y_{j-1} are the labels for segments j and $j-1$, and span for segment j is $[t_j, u_j]$. Assume that any segment could have length from 1 to L (where max length L is fixed in advance). The length of a segment is $u - t + 1$.

1 Question (3 points)

What is a potential advantage of a semi-Markov segmentation CRF compared to a BIO sequence tagger CRF?

2 Question (2 points)

What is a potential downside?

3 Question (5 points)

Imagine the following rule-based name tagger: tag every consecutive sequence of capitalized words as a NAME; all other tokens should be length-1 O segments. The tagger should identify the longest possible NAME segments.
Design a feature function and parameter weights for a semi-Markov CRF such that its Viterbi prediction implements that rule. Explain why it works.

4 Question (5 points)

Show that your model correctly segments "i went to Blue Heron" as "[i] [went] [to] [Blue Heron]$_{\text{NAME}}$" and not "[i] [went] [to] [Blue]$_{\text{NAME}}$ [Heron]$_{\text{NAME}}$", and explain why.

[1] http://www.cs.cmu.edu/~wcohen/postscript/semiCRF.pdf

5 Question: Implementation (35 points)

Implement the semi-Markov Viterbi algorithm to predict NAME segments for a sentence with a particular model (feature function and parameter weights), with the capability for arbitrary features where you have a function implementing $f(y_j, y_{j-1}, t_j, u_j)$ that returns a sparse vector—in Python, return a dict with string keys (feature names) and values (typically 1).
Test it with the capitalization-based model you designed. Show its output in the writeup for some given examples. Submit the code on Moodle, but please ensure the inputs/outputs illustration can be read just from your main writeup.

Model questions

We abbreviate a local potential function as

$$\psi_j(y, y', t, u) \equiv \exp\left(\theta^\mathsf{T} f(y, y', t, u, x)\right) \tag{1}$$

This will typically include observation features for the segment y over $[t, u]$ and transition features from y' to y. It does not include observation features for y' (it can't since it doesn't know the span for y').

6 Question (5 points)

The joint probability of an entire segmentation can be written in terms of the local potentials:

$$P(s \mid x) = \prod_{j}^{J(s)} \psi(y_j, y_{j-1}, t_j, u_j) \tag{2}$$

Where, implicitly, the (y_j, t_j, u_j) variables are part of the structured variable s.
The number of terms in the product can be variable, even for a single sentence x. Why?

Marginal probabilities

In this section you'll develop an algorithm for marginal inference. The goal is to compute, for a given label y and span (t, u),

$$p(y_{t:u} = y \mid x) \tag{3}$$

where $y_{t:u}$ denotes the label for a segment with span t through u. (To carefully define it, if there is no segment over that span, let $y_{t:u}$ have special value "NA".) The probability above does not mention the segmentation for the rest of the sentence; those segmentations are marginalized out.

7 Question (5 points)

If we assume the segment over (t, u) is the jth segment, we could write this as

$$p(y_{t:u} = y \mid x) = p(y_j = y \mid (t_j, u_j) = (t, u), x) \tag{4}$$

but note that for a particular (t, u) span, its index j could be different in different segmentations.
Why? Explain with an example.

8 Question (5 points)

Say there are two labels, O and NAME. Is it the case that for a span (t, u), that their marginal probabilities sum to one? If so, why? If not, explain why not, and give a counterexample.

Marginal inference

Note that the probability of a particular segmentation can be written in part

$$P(s \mid x) = \ldots \ \psi(y_{j-1}, y_{j-2}, t_{j-1}, u_{j-1}) \ \psi(y_j, y_{j-1}, t_j, u_j) \ \psi(y_{j+1}, y_j, t_{j+1}, u_{j+1}) \ \ldots \tag{5}$$

(At least for when j is somewhere in the middle of the sentence.) Thus the marginal probability could be written as

$$p(y_{t:u} = y \mid x) = \sum_{stuff} \ldots \ \psi(y_{j-1}, y_{j-2}, t_{j-1}, u_{t-1}) \ \psi(y, y_{j-1}, t, u) \ \psi(y_{j+1}, y_{j-2}, t_{j+1}, u_{j+1}) \ \ldots \tag{6}$$

where $stuff$ signifies summing over all $(y_{j'}, t_{j'}, u_{j'})$ variables (for all j' excluding the segment over (t, u)) over all possible segmentations.
Instead of using S+C's implicit marginal inference approach, we'll develop a direct analogue of forward-backward. Consider the following definitions, where V and α (forward probabilities) are from the Sarawagi and Cohen paper, while we define a new variable β: backward probabilities.

$$V(i, y) = \max_{d, y'} V(i - d, y') + \log \psi(y, y', i - d, i) \tag{7}$$

$$\alpha(i, y) = \sum_{d, y'} \alpha(i - d, y') \ \psi(y, y', i - d, i) \tag{8}$$

$$\beta(i, y) = \sum_{d, y''} \beta(i + d, y'') \ \psi(y'', y, i, i + d) \tag{9}$$

9 Question (30 points)

Assume α and β have been calculated, and you'd like to use them to calculate the marginal for (y, t, u):

$$p(y_{t:u} = y \mid x)$$

Please write out the correct, efficient formula in terms of forward and backward probabilities and show why it correctly calculates it.
HINT: you may find it convenient to assume that for the n tokens in the data, we also model a special one-length STOP segment at position $n + 1$. There are other ways to do it too. However you add assumptions, be very clear and explain what you're doing.

10 Question (5 points)

Numerical underflow can be a major issue when calculating forward or backward probabilities; once a double-precision floating point number goes below 10^{-300} or so, it underflows to 0. One approach to solve

the problem is to use a log probability representation; a disadvantage is that it can be slow. A different approach is to maintain the α values in the original probability space (in $[0, \infty)$), but to apply normalization for individual α messages: for example, for one t, for each y rescale $\alpha(t, y)$ by a constant.
Develop a variant of the forward and backward calculations that applies this type of rescaling, and can still be used to perform marginal inference. Write it out as pseudocode. Explain any limitations of the approach. (The Rabiner 1989 tutorial on HMMs presents a version of this for the HMM setting.)