NLP Course 097215 - Winter 2022-23 - HW1-Dry

1 Hidden Markov Models (HMMs) and Maximum Entropy Markov Models (MEMMs) for Sequential Tagging

In this section of the assignment we are going to consider feature representation with the HMM and MEMM models for sequential tagging, and the resulting impact on parameter estimation and inference (prediction) in these models.

We will first focus on the simple first order HMM:

$$p(x, y) = \prod_{i=1:n} p(x_i|y_i) \cdot p(y_i|y_{i-1}) \tag{1}$$

One way to solve the out-of-vocabulary (OOV) words problem, is to represent every combination of (x_i, y_i) as a feature vector where $p(x_i|y_i) = f(\vec{\varphi}(x_i, y_i))$, and $p(y_i|y_{i-1})$ is unchanged.

1. Which features would solve the OOV problem ? That is, which features you need to include in the feature representation of (x, y) so that the tagger will be able to correctly tag unknown words ? Try to discuss at least three different feature sets.
2. Now suppose that we represent every word with the following features: the two preceding words, the current words and their labels. Suppose that our vocabulary consists of 100,000 words and the label set consists of 25 labels.
   (a) How many different feature combinations are possible under this model?
   (b) Could the parameters for $p(x_i|y_i)$ be *effectively* estimated based on a reasonable-size corpus with the maximum likelihood principle?

As an alternative to maximum likelihood estimation we can write $p(x_i|y_i)$ in a parametric form ($\tilde{X}$ is the set of all possible feature vectors over input examples):

$$p(x_i = x|y_i = y) = \frac{p(x_i = x, y_i = y)}{p(y_i = y)} = \frac{p(x_i = x, y_i = y)}{\sum_{\hat{x}\in\tilde{X}} p(x_i = \hat{x}, y_i = y)} = \frac{exp(w \cdot f(x, y))}{\sum_{\hat{x}\in\tilde{X}} exp(w \cdot f(\hat{x}, y))} \tag{2}$$

3. Can you explain why the *exp* function is used in this formulation? Name 3 reasons
4. Given that the features we use correspond to words, POS tags, sub-word characters and so on. Would you use binary or integer features? For example, suppose that you would like to encode the previous word as a feature and you have 100,000 words in the lexicon. Would you use one feature with 100,000 values (the first word is encoded as 1, the second as 2 and so on) or 100,000 binary features? Explain.
5. The parameters of this model are the coordinates of w so the model has the same number of parameters as in question 2. The advantage of this formulation is that we do not have unobserved cases anymore - even if we get a sentence with word and/or tag combination that did not appear in the training data, we can represent it with its features and multiply by w. Still there is a problem that will prevent us from using this formulation, can you say what this problem is?

Due to the above obstacles, we turn to a different model - MEMM - which you have seen in class (the below formulation is not exactly the one presented in class):

$$p(x, y) = \prod_{i=1:n} p(y_{i+1}|y_i, x) =\sim \prod_{i=1:n} p(y_{i+1}|y_i, x_i) = p(y|x) \tag{3}$$

$$p(y_{i+1} = y|x_i = x, y_i = y') = \frac{exp(w \cdot f(x, y', y))}{\sum_{\hat{y} \in Y} exp(w \cdot f(x, y', \hat{y}))} \tag{4}$$

A model for $p(y|x)$ is called a *discriminative model* (because it discriminates between different y assignments for x) as opposed to a model for $p(x, y)$ which is called a *generative model* (because it generates x and y together).

6. Do we still have the problem of the model as formulated in equation 2 (question 5)? Please explain.

7. MEMM now looks like the perfect model. Does HMM have any advantages over it?