NLP Course 097215 - Winter 2019-20 - HW1-Dry

April 20, 2020

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 and their labels and the current word and its label. 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 ?

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 ?

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 the Coursera lecture videos (the below formulation is not exactly the one presented in the Coursera lecture):

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

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

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

2 Word2Vec

In this section of the assignment we derive the famous Word2Vec model [MSC+13]. This model is simply a language model, but here we shall describe it as follows:

- Every word w is associated with a vector $v_w \in R^d$. The coordinates of these vectors are the parameters of the model.
- The probabilities of the language model are given as a function of the vector representations of the words in the corpus.
- And, finally, our goal is to estimate the model parameters (vector representations for the participating words) that optimize the language model objective. Optimization is done with stochastic gradient descent (gradient descent where optimization is done over one example in each iteration).

Recall the language model objective (in this assignment we assume a bigram language model):

$$p(D) = \prod_{i=1}^{N} p(w_{i+1}|w_i) \tag{5}$$

For ease of presentation we will write this slightly differently:

$$p(D) = \prod_{(x,y) \in D} p(y|x) \tag{6}$$

where $D = (x, y)$ corresponds to all pairs of consecutive words in their order of appearance (y is the word which directly follows x) in the training corpus.

In the Word2Vec model we write:

$$p((x, y) \in D) = \frac{e^{v_x \cdot v_y}}{\sum_{z \in V \setminus \{x\}} e^{v_x \cdot v_z}} = p(y|x) \tag{7}$$

The goal of the model is the to find the optimal set of parameters θ^* such that:

$$\theta^* = \arg\max_{\theta} \sum_{(x,y) \in D} log(p(y|x)) \tag{8}$$

Let's start with some questions:

1. Write down the full term for θ^* as a function of the word vectors (i.e. integrate equation 7 into equation 8).

2. Using the above, write down the partial derivation of the objective according to its parameters by v_x.

3. Suppose that every word is represented with a 500 dimensional vector and that the vocabulary consists of 500K words.

   (a) How many parameters does the model have?

   (b) How much data is required for the training of the model (this is a qualitative question)?

   (c) Is the model computationally feasible?

   Please support your answers with logical explanations.

As you have seen in the last question, the above model is not computationally tractable. [MSC+13] hence proposes a similar model where for every pair of words (x, y) we ask whether it appears in the training corpus (and hence $(x, y) \in D$) or not. Under this model for a pair of words that appear in the corpus they write, $D = 1$, and the probabilistic model is given utilizing the non-linear Sigmoid function:

$$p(D = 1|x, y; \theta) = \frac{1}{1 + e^{-v_x \cdot v_y}} = \sigma(v_x \cdot v_y) \tag{9}$$

For a pair (x, y) that does not appear in the corpus they write:

$$p(D = 0|x, y; \theta) = 1 - p(D = 1|x, y; \theta) \tag{10}$$

The derivative of the Sigmoid function is:

$$\sigma'(a) = \sigma(a) \cdot (1 - \sigma(a)) \tag{11}$$

Now let's proceed with some more questions:

4. Write down the full objective: $\theta^* = \arg\max_\theta \sum_{(x,y)\in V^2} log(p(D|x, y; \theta))$.

5. Write down its partial derivation by v_x.

6. Is there a computational problem with the computation of this objective or of its partial derivations ?

7. Explain what is the problem with the optimization under this model.

The above problem is solved by introducing a set of randomly sampled word pairs that do not appear in the training corpus: $D' = (x, y)$. This procedure is called **Negative Sampling**. The resulting objective is then:

$$\theta^* = \arg\max_\theta \sum_{(x,y)\in D} log(p(D = 1|x, y; \theta)) + \sum_{(x,y)\in D'} log(p(D = 0|x, y; \theta)) \tag{12}$$

This equation has a rather simple form and its gradients can be easily computed with the chain rule.

The final two questions are qualitative:

8. Can you give another explanation as to why Negative Sampling should be helpful for a model?

9. The word representations induced by the model are used to enhance models that take into account word meaning. In lexical semantics we distinguish between the relation of **similarity** (for example the pairs *(car, vehicle) and (tiger, lion)* consist of similar words), and **association** (for example the pairs *(Maradona, football) and (star, sky)* consist of associated words). Would you expect words with similar Word2Vec vectors to be highly associated or highly similar ?

References

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.