SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

01.112 Machine Learning, Fall 2019
Lecture Notes for Week 9

14. Hidden Markov Models (I)

Last update: Tuesday 5th November, 2019 15:41

Motivation

In many practical problems, we would like to model pairs of sequences. Consider, for instance, the task of part-of-speech (POS) tagging. Given a sentence, we would like to compute the corresponding tag sequence:

- Input:"Faith is a fine invention"
- Output:"Faith/N is/V a/D fine/A invention/N"

More generally, a *sequence labeling problem* involves mapping a sequence of observations $x_1, x_2, ..., x_n$ into a sequence of tags $y_1, y_2, ..., y_n$. In the example above, every word x is tagged by a single label y. One possible approach for solving this problem would be to label each word independently. For instance, a classifier could predict a part-of-speech tag based on the word, its suffix, its position in the sentence, etc. In other words, we could construct a feature vector on the basis of the observed "context" for the tag, and use the feature vector in a linear classifier. However, tags in a sequence are dependent on each other and this classifier would make each tagging decision independently of other tags. We would like our model to directly incorporate these dependencies. For instance, in our example sentence, the word "*fine*" can be either noun (N), verb (V) or adjective (A). The label V is not suitable since a tag sequence "D V" is very unlikely. Today, we will look at a model – a hidden Markov model (HMM) – that allows us to capture some of these correlations.

Generative Tagging Model

Assume a finite set of words/observations $\mathcal{O}$ and a finite set of tags/states $\mathcal{T}$. Define $\mathcal{S}$ as the set of all possible observation-state sequence pairs $(x_1, \ldots, x_n, y_1, \ldots, y_n)$, $x_i \in \mathcal{O}$ and $y_i \in \mathcal{T}$ for $i = 1 \ldots n$. $\mathcal{S}$ here contains sequences of different lengths as well, *i.e.*, n varies as well. A generative tagging model is a probability distribution p over pairs of sequences:

- $p(x_1, \ldots, x_n, y_1, \ldots, y_n) \geq 0 \;\; \forall (x_1, \ldots, x_n, y_1, \ldots, y_n) \in \mathcal{S}$
- $\sum_{(x_1,\ldots,x_n,y_1,\ldots,y_n)\in\mathcal{S}} \; p(x_1, \ldots, x_n, y_1, \ldots, y_n) = 1$

If we have such a distribution, then we can use it to predict the most likely sequence of tags $y_1^*, \ldots, y_n^*$ for any observed sequence of words $x_1, \ldots, x_n$, as follows: [1]

$$\begin{aligned} y_1^*, \ldots, y_n^* &= \arg\max_{y_1,\ldots,y_n} p(x_1, \ldots, x_n, y_1, \ldots, y_n) \quad (1) \\ &= f(x_1, \ldots, x_n) \quad (2) \end{aligned}$$

where we view f as a function/projection/mapping from one structured space (consisting of word or observation sequences) to another (consisting of tag or state sequences).

Some Key Questions

- (Modeling) How to specify $p(x_1, \ldots, x_n, y_1, \ldots, y_n)$ with a few number of parameters (degrees of freedom)
- (Learning) How to estimate the parameters in this model based on observed sequences of words (and possibly tags).
- (Inference) How to infer, *i.e.*, how to find the conditional marginal probabilities (the joint probability of certain variables given some other variables as evidences) such as: $p(y_2|x_1, x_2, \ldots, x_n)$
- (Decoding) How to predict, *i.e.*, how to find the most likely sequence of tags for any observed sequence of words: evaluate $\arg\max_{y_1,\ldots,y_n} p(y_1, \ldots, y_n|x_1, \ldots, x_n)$

1 Modeling

Let $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be sequences of random variables of length n. We wish to specify a joint probability distribution

$$P(X_1 = x_1, \ldots, X_n = x_n, Y_1 = y_1, \ldots, Y_n = y_n) \quad (3)$$

where $x_i \in \mathcal{O}, y_i \in \mathcal{T}$. For brevity, we will write it as $p(x_1, \ldots, x_n, y_1, \ldots, y_n)$, *i.e.*, treat it as a function of values of the random variables without explicating the variables themselves. We will define two additional random variables Y_0 and Y_{n+1}, which always takes the value `START` and `STOP`. We assume before the generative process starts there is always a start symbol `START`. Since our model is over variable length sequences, we will use the end symbol to model when to stop. In other words, if we observe $x_1, \ldots, x_n$, then clearly the symbol after $y_1, \ldots, y_n$, *i.e.*, y_{n+1}, had to be `STOP` (otherwise we would have continued generating more symbols). Similarly, y_0 has to be `START`.

[1] We will know why we can use a joint probability instead of the conditional probability $p(y_1, \ldots, y_n|x_1, \ldots, x_n)$ later when we discuss decoding.

Now, let's start by rewriting the distribution a bit according to general rules that apply to any distribution. The goal is to put the distribution in a form where we can easily explicate our assumptions. First,

$$p(x_1, \ldots, x_n, y_0, y_1, \ldots, y_{n+1}) = p(y_0, y_1, \ldots, y_{n+1}) p(x_1, \ldots, x_n | y_0, y_1, \ldots, y_{n+1}) \text{ (chain rule)} \quad (4)$$

Then we will use the chain rule repeatedly along the sequence of tags

$$p(y_0, \ldots, y_{n+1}) = p(y_0) p(y_1|y_0) p(y_2|y_0, y_1) \ldots p(y_{n+1}|y_0, y_1, \ldots, y_n) \text{ (chain rule)} \quad (5)$$

So far, we have made no assumptions about the distribution at all. Since we don't expect tags to have very long dependences along the sequence, we will simply say that the next tag only depends on the current tag. In other words, we will "drop" the dependence on tags further back

$$\begin{aligned} p(y_0, y_1, \ldots, y_{n+1}) &\approx p(y_0) p(y_1|y_0) p(y_2|y_1) \ldots p(y_{n+1}|y_n) \text{ (independence assumption)} & (6) \\ &= p(y_1|y_0) p(y_2|y_1) \ldots p(y_{n+1}|y_n) & (7) \\ &= \prod_{i=1}^{n+1} p(y_i|y_{i-1}) & (8) \end{aligned}$$

Put another way, we assume that the tags form a Markov sequence (future tags are independent of the past tags given the current one). Let's now make additional assumptions about the observations as well

$$\begin{aligned} & p(x_1, \ldots, x_n | y_0, y_1, \ldots, y_{n+1}) \\ &= p(x_1|y_0, \ldots, y_{n+1}) p(x_2|x_1, y_0, \ldots, y_{n+1}) \ldots p(x_n|x_1, \ldots, x_{n-1}, y_0, \ldots, y_{n+1}) \text{ (chain rule)} & (9) \\ &\approx p(x_1|y_1) p(x_2|y_2) \ldots p(x_n|y_n) \text{ (independence assumption)} & (10) \\ &= \prod_{i=1}^{n} p(x_i|y_i) & (11) \end{aligned}$$

In other words, we say that the identity of each word only depends on the corresponding tags. This is a drastic assumption but still (often) leads to a reasonable tagging model, and simplifies our calculations. A more formal statement here is that the random variable X_i is conditionally independent of all the other variables in the model given Y_i (see more on conditional independence in the Bayesian networks lectures).

Now, we have a much simpler tagging model

$$p(x_1, \ldots, x_n, y_0, y_1, \ldots, y_{n+1}) = \prod_{i=1}^{n+1} p(y_i|y_{i-1}) \prod_{i=1}^{n} p(x_i|y_i) \quad (12)$$

We call $p(y_i|y_{i-1})$ the *transition parameters*, and $p(x_i|y_i)$ the *emission parameters*. Let's understand this model a bit more carefully by looking at how the pairs of sequences could be generated from the model. Here's the recipe:

The Generative Process

1. Set $y_0 = \texttt{START}$ (we always start from this START symbol) and let $i = 1$.
2. Generate tag y_i from the conditional distribution $p(y_i|y_{i-1})$ where y_{i-1} already has a value (*e.g.*, $y_{i-1} = \texttt{START}$ when $i = 1$)
3. If $y_i = \texttt{STOP}$, we terminate the process and return $y_0, y_1, \ldots, y_i, x_1, \ldots, x_{i-1}$. Otherwise we generate x_i from the emission distribution $p(x_i|y_i)$.
4. Set $i = i + 1$, and return to step 2.

HMM Formal Definition

The model we have defined is a hidden Markov model or HMM for short. An HMM is defined by a tuple $\langle \mathcal{T}, \mathcal{O}, \theta \rangle$, where

- $\mathcal{T}$ is set of states including the START state and the STOP state: $0, 1, \ldots, |\mathcal{T}| - 1$ (by convention, we always assume 0 is the start state START, and $|\mathcal{T}| - 1$ is the STOP state).
- $\mathcal{O}$ is the set of observation symbols. For example, $\mathcal{O}$= {"the", "dog"}.
- $\theta = \langle a, b \rangle$ consists of two sets of parameters
  - Parameter $a_{u,v} \equiv p(y_{next} = v|y_{curr} = u)$ for $u = 0, 1, \ldots, |\mathcal{T}|-2$ and $v = 1, \ldots, |\mathcal{T}|-1$ is the probability of transitioning from state u to state v: $\sum_{v'=1}^{|\mathcal{T}|-1} a_{u,v'} = 1$ for any u.
  - Parameter $b_u(o) \equiv p(x = o|y = u)$ for $u = 1, \ldots, |\mathcal{T}| - 2$ and $o \in \mathcal{O}$ is the probability of emitting symbol o from state u: $\sum_{o \in \mathcal{O}} b_u(o) = 1$ for any u.

Discussions Here θ is a vector of parameters. How many parameters are there? How many free parameters are there?

Example

- States are $\mathcal{T} = \{\texttt{START}, \text{A}, \text{B}, \texttt{STOP}\}$.
- $\mathcal{O}$ = {"the", "dog"}
- Transition parameters $a_{u,v}$ are

$u \setminus v$	A	B	STOP
START	1.0	0.0	0.0
A	0.5	0.5	0.0
B	0.0	0.8	0.2

- Emission parameters $b_u(o)$ are

$u \setminus o$	"the"	"dog"
A	0.9	0.1
B	0.1	0.9

An HMM specifies a probability for each possible $(\mathbf{x}, \mathbf{y})$ pair, where $\mathbf{x} = (x_1, \ldots, x_n)$ is a sequence of symbols drawn from $\mathcal{O}$ and $\mathbf{y} = (y_1, \ldots, y_n)$ is a sequence of states drawn from the integers $1, \ldots, (|\mathcal{T}| - 1)$.

$$p_\theta(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}, \mathbf{y}; \theta) = \quad a_{\texttt{START},y_1} \cdot \tag{13}$$

$$a_{y_n,\texttt{STOP}} \cdot \tag{14}$$

$$\textstyle\prod_{i=2}^{n} a_{y_{i-1},y_i} \cdot \tag{15}$$

$$\textstyle\prod_{i=1}^{n} b_{y_i}(x_i) \tag{16}$$

Consider the example: "the/A, dog/B, the/A". The probability of such a sequence is:

$$a_{\texttt{START},\text{A}} \cdot b_{\text{A}}(the) \cdot a_{\text{A,B}} \cdot b_{\text{B}}(dog) \cdot a_{\text{B,A}} \cdot b_{\text{A}}(the) \cdot a_{\text{A},\texttt{STOP}} = 0 \tag{17}$$

What does this mean? It means under such a hidden Markov model with the specific parameterization, there is no probability of generating the given example sequence pair. Or, in other words, if the given observation sequence is "the dog the", then the underlying tag sequence cannot be "A B A".

Now you may wonder, how do we find the *most probable tag sequence* $\mathbf{y}$ for a given observation sequence $\mathbf{x}$? This is the decoding problem! We'll learn the decoding algorithms later. Before we talk about decoding, let us take a quick look at supervised learning.

2 Supervised Learning

We will first look at the fully observed case (complete data case), where our training data contains both $\mathbf{x}$ and $\mathbf{y}$. We will do maximum likelihood estimation. Consider the examples: $\mathcal{O} = \{e, f, g, h\}$, $\mathcal{T} = \{\texttt{START}, \text{A}, \text{B}, \texttt{STOP}\}$. Observations: $(e/\text{A}, g/\text{B})$, $(e/\text{A}, h/\text{B})$, $(f/\text{A}, h/\text{B})$, and $(f/\text{A}, g/\text{B})$. To find the MLE, we will simply look at the counts of events, *i.e.*, the number of transitions between tags, the number of times we saw an output (observation) symbol together with a specific tag (state). After normalizing the counts to yield valid probability estimates, we get

$$a_{u,v} = \frac{\text{Count}(u; v)}{\text{Count}(u)} \tag{18}$$

$$a_{\text{A,B}} = \frac{\text{Count(A; B)}}{\text{Count(A)}} = \frac{4}{4} = 1, a_{\text{B,B}} = \frac{\text{Count(B; B)}}{\text{Count(B)}} = \frac{0}{4} = 0, \ldots \tag{19}$$

where Count$(u; v)$ is the number of times we have (u, v) as two successive states and Count(u) is the number of times state u appears in the sequence. Similarly,

$$b_u(o) = \frac{\text{Count}(u \to o)}{\text{Count}(u)} \tag{20}$$

$$b_{\text{A}}(e) = \frac{\text{Count(A} \to e)}{\text{Count(A)}} = \frac{2}{4} = 0.5, \ldots \tag{21}$$

where Count$(u \to o)$ is the number of times we see that state u emits symbol o. Count(u) was already defined above.

Learning Objectives

You need to know:

1. What is a hidden Markov model.
2. What are the parameters associated with a hidden Markov model.
3. How to compute the joint likelihood of a hidden Markov model.
4. How to perform supervised learning of a hidden Markov model.