Assignment 2
CSE 517: Natural Language Processing

University of Washington

Spring 2018
Due: April 27, 2018

1 Maximum *a Posteriori* Estimation

In the noisy channel paradigm, we consider two random variables, X and Y, and imagine that the observable X is a noisy or corrupted version of Y.

$$\boxed{\text{source}} \longrightarrow Y \longrightarrow \boxed{\text{channel}} \longrightarrow X$$

For example, if we observe a Chinese sentence (a value for X) but believe it was "originally" an English sentence (a value for Y), we can build source and channel models to capture, respectively, the distribution over English sentences and the translation of English into Chinese. Then, we use Bayes' rule to "decode":

$$\hat{y} = \operatorname*{argmax}_y p(y \mid x) = \operatorname*{argmax}_y \frac{p(y) \cdot p(x \mid y)}{p(x)} = \operatorname*{argmax}_y p(y) \cdot p(x \mid y)$$

Question 1: Why can we drop the $p(x)$ factor in the denominator?

Interestingly, this same "noisy channel" idea can also be applied when we are estimating the parameters of a probabilistic model, though it goes by a different name: maximum *a posteriori* (MAP) estimation. The story looks like this:

$$\boxed{\text{source}} \longrightarrow \Theta \longrightarrow \boxed{\text{channel}} \longrightarrow X$$

Instead of Y, we use the random variable Θ for the unknown parameters of our model. X corresponds to the training data.

Applying Bayes' rule gives us:

$$\hat{\theta} = \operatorname*{argmax}_\theta p(\theta) \cdot p(x \mid \theta)$$

Here, we're using the notation $p(\boldsymbol{x} \mid \boldsymbol{\theta})$ instead of $p_{\boldsymbol{\theta}}(\boldsymbol{x})$, since we're thinking of $\boldsymbol{\theta}$ as a random variable with its own probability distribution, but both are usually fine and they mean the same thing.

It's helpful to think about this for a specific model, so let's take the simplest one we know: the unigram model. The parameters for the unigram model are a vector $\boldsymbol{\theta} \in \triangle^V$.[1] The evidence is a sequence of words

[1] The probability simplex in V dimensions, denoted $\triangle^V$, is defined as the set of all V-length vectors that are (i) nonnegative and (ii) sum to one.

in a text, denoted $\boldsymbol{x}$, though—as discussed in class—we really only need to know the count of each word, not the order in which they occur.

Let's consider the "channel" model first; it defines the probability of $\boldsymbol{x}$ given the parameters. Recall this model from class:

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \prod_{i=1}^{n} \theta_{x_i} = \prod_{v \in \mathcal{V}} \theta_v^{c_{\boldsymbol{x}}(v)} \tag{1}$$

where $c_{\boldsymbol{x}}(v)$ is the count of word v in the data $\boldsymbol{x}$.

We discussed maximum likelihood estimation (MLE), in which we select $\hat{\boldsymbol{\theta}}$ to maximize the above quantity, and proved that relative frequency estimation accomplishes MLE. You should think of MAP as an *alternative* to MLE.

The "source" model is a bit harder to think about at first. We need a probability distribution over $\triangle^V$. An attractive candidate for $p(\Theta)$ is the **Dirichlet** distribution.[2]

Here is the form of the Dirichlet distribution:

$$p_{\boldsymbol{\alpha}}(\boldsymbol{\theta}) = \frac{\Gamma\left(\sum_{v \in \mathcal{V}} \alpha_v\right)}{\prod_{v \in \mathcal{V}} \Gamma(\alpha_v)} \prod_{v \in \mathcal{V}} \theta_v^{(\alpha_v - 1)} \tag{2}$$

This is somewhat daunting, so we'll break it down:

- The parameters $\boldsymbol{\alpha} \in \mathbb{R}^V$ are a vector of (strictly) positive values. If we normalize them (divide each by their sum), we get the mean value of the Dirichlet; if we sampled many different values of Θ, their average would be close to this normalized version of $\boldsymbol{\alpha}$. As these values get larger, the tendency of draws from the Dirichlet to be close to the mean increases. When they are smaller, draws from the Dirichlet will be more diffuse.
- The Γ function is an extension of the factorial function to the nonnegative reals. For natural numbers x, $\Gamma(x) = (x-1)!$. The more general form is not important for this assignment; if you are curious, read more at `http://mathworld.wolfram.com/GammaFunction.html`.
- Fortunately for us, the Γ factors are constant with respect to $\boldsymbol{\theta}$, so they don't affect our estimation procedure.

The fascinating thing about Dirichlet distributions and categorical distributions like the unigram model's $\boldsymbol{\theta}$ is that the form of the posterior distribution $p(\Theta \mid \boldsymbol{x})$ is also a Dirichlet distribution. When this happens, Bayesian statisticians get very excited and use the term "conjugate prior" to denote the special relationship between a family of priors (here, Dirichlet distributions) and likelihood functions (here, categorical distributions). In fact, this property is why the Dirichlet was chosen as the prior in latent Dirichlet allocation in the first place.

[2] We mentioned this family briefly when we talked about latent Dirichlet allocation, but did not go into details.

Consider the logarithm of this quantity as a function of $\boldsymbol{\theta}$:

$$\log p(\boldsymbol{\theta} \mid \boldsymbol{x}) = \log p_{\boldsymbol{\alpha}}(\boldsymbol{\theta}) + \log p(\boldsymbol{x} \mid \boldsymbol{\theta}) - \log p(\boldsymbol{x}) \tag{3}$$

$$= \log p_{\boldsymbol{\alpha}}(\boldsymbol{\theta}) + \log p(\boldsymbol{x} \mid \boldsymbol{\theta}) + \text{constant} \tag{4}$$

$$= \log \frac{\Gamma\left(\sum_{v \in \mathcal{V}} \alpha_v\right)}{\prod_{v \in \mathcal{V}} \Gamma(\alpha_v)} \prod_{v \in \mathcal{V}} \theta_v^{(\alpha_v - 1)} + \log \prod_{v \in \mathcal{V}} \theta_v^{c_{\boldsymbol{x}}(v)} + \text{constant} \tag{5}$$

$$= \log \prod_{v \in \mathcal{V}} \theta_v^{(\alpha_v - 1)} + \log \prod_{v \in \mathcal{V}} \theta_v^{c_{\boldsymbol{x}}(v)} + \text{constant} \tag{6}$$

where we've just plugged in the expressions from (1) and (2). Starting on line (4), we fold all terms that do not depend on $\boldsymbol{\theta}$ into "constant."[3]

Question 2: Conjugacy means that the above posterior can be expressed as a Dirichlet distribution with some parameters; call them $\boldsymbol{\alpha'}$. Write α'_v as a function of $\boldsymbol{\alpha}$ and the counts $c_{\boldsymbol{x}}(*)$.

MAP estimation requires that we find the most probable value of $\boldsymbol{\theta}$ under this posterior—in other words, its mode. The mode has a closed form whenever all $\alpha_v > 1$ for all $v \in \mathcal{V}$. The closed form is:

$$\left[\operatorname*{argmax}_{\boldsymbol{\theta}} p_{\boldsymbol{\alpha}}(\boldsymbol{\theta}) \cdot p(\boldsymbol{x} \mid \boldsymbol{\theta})\right]_v = \frac{\alpha'_v - 1}{\sum_{v' \in \mathcal{V}} \alpha'_{v'} - V} \tag{7}$$

for each $v \in \mathcal{V}$.

Question 3: For an appropriate choice of $\boldsymbol{\alpha}$, the Dirichlet can be "uninformative," meaning that it assigns equal probability to all possible $\boldsymbol{\theta}$. In this case, MAP is the same as MLE! Derive the values of $\boldsymbol{\alpha}$ that will make this happen.

Question 4: For a (different) appropriate choice of $\boldsymbol{\alpha}$, we make MAP estimation equivalent to Laplace smoothing.[4] Derive the values of $\boldsymbol{\alpha}$ that will make this happen.

Question 5: A *symmetric* Dirichlet distribution is one where all α_v take the same value (say, a). Show that, for any a that is sufficiently large, we can write the MAP estimate as an interpolation between the MLE (weighted by some λ) and a uniform distribution over $\mathcal{V}$ (weighted by $1 - \lambda$). Derive λ as a function of a.

[3] What we mean here is only that they are constant *with respect to* $\boldsymbol{\theta}$ and therefore do not affect our objective of finding $\boldsymbol{\theta}$ to maximize the (log-)likelihood.

[4] In lecture, Laplace smoothing was covered casually but not in detail and without the name "Laplace smoothing." It's what the instructor described as "the simplest smoothing approach you can think of." Before normalizing counts into probabilities (as you do in MLE), take the count of every $v \in \mathcal{V}$ (including the v that have a count of zero) and add one to it. So now, everything you didn't see, you're pretending you saw once. Everything you *did* see, you're pretending you saw it one more time than you did. If, instead of adding one, you add a (positive) value λ, then this method is called "add-λ" smoothing.

2 Document Clustering

A model we did not talk about in class, but which is related to probabilistic latent semantic analysis (PLSA) and other probabilistic topic models, is the **mixture of unigrams** model. Given a corpus $\mathcal{C}$ of C documents where the cth document is $\boldsymbol{x}_c = \langle x_{c,1}, x_{c,2}, \ldots, x_{c,\ell_c} \rangle$ (each $x_{*,*} \in \mathcal{V}$ and ℓ_c is the length of this document), this model assigns each document c to a (hard) cluster z_c, which takes a value in $\mathcal{T} = \{1, \ldots, k\}$. You can think of $\mathcal{T}$ as a set of topics.

The probability of the corpus is given by:

$$p(\boldsymbol{x}) = \prod_{c \in \mathcal{C}} p(\boldsymbol{x}_c) \tag{8}$$

$$= \prod_{c \in \mathcal{C}} \sum_{z \in \mathcal{T}} p(z) \cdot \prod_{i=1}^{\ell_c} p(x_{c,i} \mid z) \tag{9}$$

$$= \prod_{c \in \mathcal{C}} \sum_{z \in \mathcal{T}} \gamma_z \cdot \prod_{i=1}^{\ell_c} \theta_{x_{c,i}|z} \tag{10}$$

The model has the following parameters:

- $\boldsymbol{\gamma}$, a discrete distribution over $\mathcal{T}$; and
- for each topic $z \in \mathcal{T}$, $\boldsymbol{\theta}_{*|z}$, a distribution over the vocabulary $\mathcal{V}$.

Question 6: The relationship between documents and topics assumed by this model is fundamentally different from that in PLSA. Explain the difference, in one sentence.

Question 7: We noted in the lecture that PLSA had a problem: it could not assign probability to any document not in its training corpus. Does the mixture-of-unigrams model have that same problem? Note: we are not asking about the values of the model parameters, but rather about the set of things the model can generate, in principle. Another way to think about this question is this: if we consider a new document not in the training corpus, the parameters do not give us a way to estimate its likelihood; we are asking whether the same is true here.

When you estimate this model, there are C latent variables: $Z_1, \ldots, Z_C$, each ranging over topics. Suppose you were going to use EM to estimate the parameters of this model.

Question 8: For the M step, assume we have the "soft count" of every word $v \in \mathcal{V}$ occurring in a document in $\boldsymbol{x}$ with every topic $z \in \mathcal{T}$: $\tilde{c}_{\boldsymbol{x}}(z, v)$. Give the formula for relative frequency estimation of $\theta_{v|z}$, then explain why we might prefer to do something slightly different from relative frequency estimation.

For the E step, we require soft counts of topic/word pairs in the corpus. These will take the form:

$$\tilde{c}_{\boldsymbol{x}}(z, v) = \sum_{c \in \mathcal{C}} \tilde{c}_{\boldsymbol{x}_c}(z, v) \tag{11}$$

$$= \sum_{c \in \mathcal{C}} c_{\boldsymbol{x}_c}(v) \cdot p(Z_c = z \mid \boldsymbol{x}_c) \tag{12}$$

Question 9: Give the formula for the posterior $p(Z_c = z \mid \boldsymbol{x}_c)$, in terms of the model parameters. Hint: $p(A = a \mid B = b) = \frac{p(A=a,B=b)}{\sum_{a'} p(A=a',B=b)}$.

Question 10: What is the asymptotic runtime of the E step? Express it as a function of V, C, and k.

Question 11: This model is an "unsupervised" version of a text categorization model we saw in class, with a particular choice of features. This means that it makes the same assumptions as the text categorization model, but makes the document categories latent, trying to learn to distinguish among them without ever observing one. Which model, and which features?

3 Alternative E Steps

In a variant of the EM algorithm sometimes called "hard EM," we calculate "hard" counts instead of soft ones. What this means is that, for each latent variable, we assign it to its most likely value, under the current model. In the case of the model in the last problem, for example, this sets:

$$p(Z_c = z \mid \boldsymbol{x}_c) = \begin{cases} 1 & \text{if } z = \text{argmax}_z \, p_{\boldsymbol{\gamma},\boldsymbol{\theta}}(z, \boldsymbol{x}_c) \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

Hard EM is closely related to a general-purpose clustering algorithm called k-means. k-means is extremely easy to describe and understand. Let the data instances to be clustered be represented by $\mathbf{x}_1, \ldots, \mathbf{x}_C$, each in $\mathbb{R}^d$.

Here's the procedure:

1. Pick k points $\mathbf{m}_1, \ldots, \mathbf{m}_k$ each in $\mathbb{R}^d$. These are the initial "means" of our clusters.
2. For each $\mathbf{x}_c$ ($c \in \{1, \ldots, C\}$), let:

$$y_c \leftarrow \underset{i \in \{1,\ldots,k\}}{\text{argmin}} \, \|\mathbf{m}_i - \mathbf{x}_c\|_2 \tag{14}$$

   That is, assign each data instance to the nearest cluster (in Euclidean distance).

3. For each cluster i, from 1 to k, update its mean:

$$\mathbf{m}_i \leftarrow \text{average}\left(\{\mathbf{x}_c : y_c = i\}\right) = \frac{\sum_{c:y_c=i} \mathbf{x}_c}{|\{c : y_c = i\}|} \tag{15}$$

   That is, move the mean of cluster i to actually be the mean of the data points assigned to that cluster (instances c such that $y_c = i$).

4. Go to step 2.

It is not hard to show that k-means will eventually converge to a stable solution, that this solution depends on the initial means, and that the procedure chooses a collection of means that (locally) optimize a likelihood function.

Now is a good time to review the multivariate normal distribution (e.g., https://en.wikipedia.org/wiki/Multivariate_normal_distribution).

Question 12: Show that k-means is equivalent to hard EM where the probability of the data is given by:

$$p(\mathbf{x}) = \sum_{i=1}^{k} \frac{1}{k} \cdot \text{Normal}(\mathbf{x}; \mathbf{m}_i, \mathbf{I}) \tag{16}$$

Hint: start from the probabilistic model, and work out the hard E step (taking a logarithm will help you) and then the M step. Your goal is to arrive at k-means.

Question 13: Describe modifications to the k-means algorithm that will arrive at hard EM for the document clustering algorithm in section 2. Hint 1: the "bag of words" assumption inherent to the document clustering model implies a vector representation $\mathbf{x}_c$ for each document $c \in \mathcal{C}$. Hint 2: you will need to replace the Euclidean distance in step 2 with something else.

Question 14: Another alternative to the E step is to approximate the exact posterior with random samples from that posterior; sometimes it's computationally easier to draw samples than it is to represent the distribution exactly. Can you think of another advantage to this idea?