CS674 Natural Language Processing

- Today
  - The EM algorithm

Some of the slides are created based on notes by Thorsten Joachims, Lillian Lee, and Roni Rosenfeld

The Statistical Modeling Framework

- Task
  - Given data x and a model parameterized by θ, find the θ that maximizes the likelihood of x.

$$\theta^* = \arg\max_\theta P_\theta(x) = \arg\max_\theta \log(P_\theta(x))$$

- Why using EM for ML estimation?
  - Suppose $P_\theta(x)$ hard to maximize, but there exists hidden data h such that $\arg\max_\theta P_\theta(x, h)$ is easy
  - EM (Dempster, Laird, Rubin, 1977) enables us to make MLE even under the presence of hidden data

Combining Language Models

- Given two language models M_A and M_B, create a hybrid model M_H that, every time it is consulted, stochastically chooses which of the two models to use, with probability λ of choosing M_A.
- Now, given a sample $D = \{s_1, s_2, \ldots, s_n\}$ generated by M_H, find an ML estimate for λ.

Combining Language Models

MLE: $\arg\max_\lambda \log(P(D \mid \lambda))$

$$\begin{aligned}
&\log(P(D \mid \lambda)) \\
&= L(s_1, s_2, \ldots, s_n \mid \lambda) \\
&= \log \prod_i P_H(s_i) \\
&= \sum_i \log P_H(s_i) \\
&= \sum_i \log(\lambda P_A(s_i) + (1-\lambda) P_B(s_i))
\end{aligned}$$

Combining Language Models

- Observation
  - If we knew which of the two models was used to generate each string s_i, we could estimate λ as follows:

$$\lambda = \frac{\text{number of times } M_A \text{ was chosen}}{n}$$

  - But the choice of the two models is a *hidden event*!
  - But we do *not* need to know which model was used
  - We only need to know # times M_A was chosen
  - Still we do not know this quantity
  - But we can calculate its expected value!

Combining Language Models

- Idea: guess the value of λ and iteratively improve the estimate (with respect to the log likelihood)

Combining Language Models

- Initialize λ to some arbitrarily non-zero value
- At step k
  - Let λ^k be the current estimate for λ
  - Compute the expected # times M_A was used from data

$$\left.\begin{aligned} & E_{\lambda^k}(M = A \mid s_1, s_2, \ldots, s_n) \\ &= \sum_i P_{\lambda^k}(M = A \mid s_i) \\ &= \sum_i \frac{P(M_A, s_i)}{P(s_i)} \\ &= \sum_i \frac{\lambda^k * P_A(s_i)}{\lambda^k * P_A(s_i) + (1-\lambda^k) * P_B(s_i)} \end{aligned}\right\} \text{E-step}$$

Combining Language Models

- At step k
  - Let λ^k be the current estimate for λ
  - Compute the expected # times M_A was used from data
  - Improve the estimate of λ using the statistics obtained from the E-step

$$\left.\lambda^{k+1} = \frac{E_{\lambda^k}(M = A \mid s_1, s_2, \ldots, s_n)}{n}\right\} \text{M-step}$$

- Terminate if $L(s_1, s_2, \ldots, s_n \mid \lambda^{k+1}) \approx L(s_1, \ldots, s_n \mid \lambda^k)$

Combining Language Models

- The log likelihood function is bounded above and always increases after each iteration
  - EM always converges (in terms of log likelihood)
- Resulting λ is the ML solution given the data

Sufficient Statistics

- If we knew which of the two models was used to generate each string s_i, we could estimate λ as follows:

$$\lambda = \frac{number\ of\ times\ M_A\ was\ chosen}{n}$$

- But the choice of the two models is a *hidden event*!
- Again, we do *not* need to know which model was used
- We only need to know # times M_A was chosen
- The number of times M_A was chosen is a sufficient statistic of the distribution of the hidden event
- Sufficient statistics convey all the information that we need to estimate the parameters

The Parameter Estimation Problem

- Given some incomplete data and a parametric model, use the data to compute the ML estimate of the parameters $\{\theta_i\}$ of the model

$$\theta^* = \arg\max_\theta L(data \mid \theta)$$

The EM Algorithm

- Identify the sufficient statistics for estimating the θ's
- Initialize the θ's to some arbitrary (non-zero) values θ_i^0
- Iterate the E-step and the M-step. During step k,
  - compute the sufficient statistics based on the data and the current parameter estimates θ_i^k (E-step)
  - derive θ_i^{k+1} as an ML estimate using the values of the sufficient statistics computed in the E-step (M-step)
- Terminate when $L(data \mid \theta_i^{k+1}) \approx L(data \mid \theta_i^k)$

Properties of EM

- Log likelihood is guaranteed to converge
  - EM is guaranteed to converge
  - But convergence may be to a *local* maximum

NLP Applications using EM

- Estimating the values of hidden variables
  - HMM training: forward-backward/Baum-Welch (1972)
  - PCFG training: inside-outside (Baker, 1979)
  - Word alignment in a parallel corpus (Brown et al., 1993)
- Unsupervised learning of clusters
  - Distributional clustering of nouns (Periera et al., 1993)
  - Learning subcategorization frames (Rooth et al., 1999)
- Improving parameter estimates of finite mixtures
  - Semi-supervised text classification (Nigam et al., 2000)

Text Classification

- Assign pieces of text to predefined categories based on content
- Types of text
  - Documents, paragraphs, sentences
- Different types of categories
  - Topic, author, style

Naïve Bayes Classifiers for Text

- Assumption: choosing the best class *c** for a text *d* amounts to choosing the most probable class for the text

$$
\begin{aligned}
c^* &= \arg\max_{c \in C} P(c \mid d) \\
&= \arg\max_{c \in C} \frac{P(c)P(d \mid c)}{P(d)} \qquad \text{(Bayes rule)} \\
&= \arg\max_{c \in C} P(c) \prod_{i=1}^{\#\text{words in } d} P(w_i \mid c)
\end{aligned}
$$

Parameter Estimation from Training Data

- Define
  - *D*: training data (a set of labeled texts)
  - *V*: vocabulary
  - *N(w,d):* number of times word *w* occurs in text *d*
- Estimate *P(w | c):*

$$P(w_t \mid c_j) = \frac{\sum_{i=1}^{|D|} N(w_t, d_i) P(c_j \mid d_i)}{\sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N(w_s, d_i) P(c_j \mid d_i)}$$

- P(*c*): prior probabilities

$$P(c_j) = \frac{\sum_{i=1}^{|D|} P(c_j \mid d_i)}{|D|}$$

What do we want to do now?

- We know how to build a naïve Bayes classifier for text from training data
  - Naïve Bayes maximizes the probability of the model θ given the training data *D*

$$\theta^* = \arg\max_{\theta} P(\theta \mid D)$$

  - So θ is a *maximum a posteriori* (MAP) hypothesis
- What if we have very little training data?
  - The model parameters may not be estimated accurately

What do we want to do now?

- Goal
  - To train a naïve Bayes classifier θ* on both the labeled data D and the unlabeled data *U*

$$\theta^* = \arg\max_{\theta} P(\theta \mid (D \cup U))$$

What do we want to do now?

- Goal
  - To train a naïve Bayes classifier θ* on both the labeled data D and the unlabeled data *U*

$$\theta^* = \arg\max_{\theta} P(\theta \mid (D \cup U))$$

- EM allows us to do that!

The EM Algorithm

- Identify the sufficient statistics for estimating the θ's
- Initialize the θ's to some arbitrary (non-zero) values $\theta_i^{\,0}$
- Iterate the E-step and the M-step. During step k,
  - compute the sufficient statistics based on the data and the current parameter estimates θ_i^k (E-step)
  - derive θ_i^{k+1} as an ML/MAP estimate using the values of the sufficient statistics computed in the E-step (M-step)
- Terminate when $L(data \mid \theta_i^{k+1}) \approx L(data \mid \theta_i^{k})$

Identifying the Sufficient Statistics

- θ = < *P(w | c), P(c)* >

$$P(w_t \mid c_j) = \frac{\sum_{i=1}^{|D \cup U|} N(w_t, d_i) P(c_j \mid d_i)}{\sum_{s=1}^{|V|} \sum_{i=1}^{|D \cup U|} N(w_s, d_i) P(c_j \mid d_i)}$$

$$P(c_j) = \frac{\sum_{i=1}^{|D \cup U|} P(c_j \mid d_i)}{|D \cup U|}$$

- The sufficient statistics are $|D \cup U|$, $|V|$, $N(w_t, d_i)$, and $P(c_j \mid d_i)$

The EM Algorithm

- Identify the sufficient statistics for estimating the θ's
- Initialize the θ's to some arbitrary (non-zero) values $\theta_i^{\,0}$
- Iterate the E-step and the M-step. During step k,
  - compute the sufficient statistics based on the data and the current parameter estimates θ_i^k (E-step)
  - derive θ_i^{k+1} as an ML/MAP estimate using the values of the sufficient statistics computed in the E-step (M-step)
- Terminate when $L(data \mid \theta_i^{k+1}) \approx L(data \mid \theta_i^{k})$

Initializing the Parameters

- Arbitrarily initialize θ?
- We have labeled training data!
- Initialize θ based on the labeled data

$$P(w_t \mid c_j) = \frac{\sum_{i=1}^{|D|} N(w_t, d_i) P(c_j \mid d_i)}{\sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N(w_s, d_i) P(c_j \mid d_i)}$$

$$P(c_j) = \frac{\sum_{i=1}^{|D|} P(c_j \mid d_i)}{|D|}$$

The EM Algorithm

- Identify the sufficient statistics for estimating the θ's
- Initialize the θ's to some arbitrary (non-zero) values $\theta_i^{\,0}$
- Iterate the E-step and the M-step. During step k,
  - compute the sufficient statistics based on the data and the current parameter estimates θ_i^k (E-step)
  - derive θ_i^{k+1} as an ML/MAP estimate using the values of the sufficient statistics computed in the E-step (M-step)
- Terminate when $L(data \mid \theta_i^{k+1}) \approx L(data \mid \theta_i^{k})$

Computing the Sufficient Statistics

- Sufficient statistics
  - *$|D \cup U|$, $|V|$, $N(w_t, d_i)$, and $P(c_j \mid d_i)$*
  - The first three can be computed without knowledge of the label of a text
- Use Bayes rule to compute *$P(c_j \mid d_i)$* based on the current estimates of the parameters of θ

$$P(c_j \mid d_i) \propto P(c_j)P(d_i \mid c_j)$$
$$= P(c_j)\prod_{k=1}^{|d_i|} P(w_{d_{i,k}} \mid c_j)$$

The EM Algorithm

- Identify the sufficient statistics for estimating the θ's
- Initialize the θ's to some arbitrary (non-zero) values $\theta_i^{\,0}$
- Iterate the E-step and the M-step. During step k,
  - compute the sufficient statistics based on the data and the current parameter estimates θ_i^k (E-step)
  - derive θ_i^{k+1} as an ML/MAP estimate using the values of the sufficient statistics computed in the E-step (M-step)
- Terminate when $L(data \mid \theta_i^{k+1}) \approx L(data \mid \theta_i^{k})$

Re-estimating the Parameters

- Improve the estimate θ of based on the current values of the sufficient statistics

$$P(w_t \mid c_j) = \frac{\sum_{i=1}^{|D \cup U|} N(w_t, d_i) P(c_j \mid d_i)}{\sum_{s=1}^{|V|} \sum_{i=1}^{|D \cup U|} N(w_s, d_i) P(c_j \mid d_i)}$$

$$P(c_j) = \frac{\sum_{i=1}^{|D \cup U|} P(c_j \mid d_i)}{|D \cup U|}$$

The EM Algorithm

- Identify the sufficient statistics for estimating the θ's
- Initialize the θ's to some arbitrary (non-zero) values θ_i^0
- Iterate the E-step and the M-step. During step k,
  – compute the sufficient statistics based on the data and the current parameter estimates θ_i^k (E-step)
  – derive θ_i^{k+1} as an ML/MAP estimate using the values of the sufficient statistics computed in the E-step (M-step)
- Terminate when $L(data \mid \theta_i^{k+1}) \approx L(data \mid \theta_i^k)$

Can EM really improve the model?

- It depends on whether
  – the data is generated by a mixture
  – there is a 1-to-1 mapping between the mixture components and classes
  – the mixture components are multinomial distributions of words