3 Semi-Supervised Text Classification Using EM

Kamal Nigam
Andrew McCallum
Tom Mitchell

For several decades, statisticians have advocated using a combination of labeled and unlabeled data to train classifiers by estimating parameters of a generative model through iterative Expectation-Maximization (EM) techniques. This chapter explores the effectiveness of this approach when applied to the domain of text classification. Text documents are represented here with a bag-of-words model, which leads to a generative classification model based on a mixture of multinomials. This model is an extremely simplistic representation of the complexities of written text. This chapter explains and illustrates three key points about semi-supervised learning for text classification with generative models. First, despite the simplistic representation, some text domains have a high positive correlation between generative model probability and classification accuracy. In these domains, a straightforward application of EM with the naive Bayes text model works well. Second, some text domains do not have this correlation. Here we can adopt a more expressive and appropriate generative model that does have a positive correlation. In these domains, semi-supervised learning again improves classification accuracy. Finally, EM suffers from the problem of local maxima, especially in high dimension domains such as text classification. We demonstrate that deterministic annealing, a variant of EM, can help overcome the problem of local maxima and increase classification accuracy further when the generative model is appropriate.

3.1 Introduction

The idea of learning classifiers from a combination of labeled and unlabeled data is an old one in the statistics community. At least as early as 1968, it was suggested that labeled and unlabeled data could be combined to build classifiers with likelihood maximization by testing all possible class assignments [Hartley and Rao, 1968]. The seminal paper by Day [1969] presents an iterative EM-like approach for parameters of a mixture of two nor-

mals with known covariances from unlabeled data alone. Similar iterative algorithms for building maximum likelihood classifiers from labeled and unlabeled data with an explicit generative model followed, primarily for mixtures of normal distributions [McLachlan, 1975, Titterington, 1976].

Dempster et al. [1977] presented the theory of the EM framework, bringing together and formalizing many of the commonalities of previously suggested iterative techniques for likelihood maximization with missing data. Its applicability to estimating maximum likelihood (or maximum a posteriori) parameters for mixture models from labeled and unlabeled data [Murray and Titterington, 1978] and then using this for classification [Little, 1977] was recognized immediately. Since then, this approach continues to be used and studied [McLachlan and Ganesalingam, 1982, Ganesalingam, 1989, Shahshahani and Landgrebe, 1994]. Using likelihood maximization of mixture models for combining labeled and unlabeled data for classification has more recently made its way to the machine learning community [Miller and Uyar, 1996, Nigam et al., 1998, Baluja, 1999].

The theoretical basis for Expectation-Maximization shows that with sufficiently large amounts of unlabeled data generated by the model class in question, a more probable model can be found than if using just the labeled data alone. If the classification task is to predict the latent variable of the generative model, then with sufficient data a more probable model will also result in a more accurate classifier.

This approach rests on the assumption that the generative model is correct. When the classification task is one of classifying human-authored texts (as we consider here) the true generative model is impossible to parameterize, and instead practitioners tend to use very simple representations. For example, the commonly-used naive Bayes classifier represents each authored document as a bag of words, discarding all word ordering information. The generative model for this classifier asserts that documents are created by a draw from a class-conditional multinomial. As this is an extreme simplification of the authoring process, it is interesting to ask whether such a generative modeling approach to semi-supervised learning is appropriate or beneficial in the domain of text classification.

This chapter demonstrates that generative approaches are appropriate for semi-supervised text classification when the selected generative model probabilities are well-correlated with classification accuracy, and when suboptimal local maxima can be mostly avoided. In some cases, the naive Bayes generative model, despite its simplicity, is sufficient. We find that model probability is strongly correlated with classification accuracy, and Expectation-Maximization techniques yield classifiers with unlabeled data that are significantly more accurate then those built with labeled data alone. In other cases, the naive Bayes generative model is not well-correlated with classification accuracy. By adopting a more expressive generative model, accuracy and model probability correlations are restored, and again EM yields good results.

One of the pitfalls of EM is that it only guarantees the discovery of a local maxima and not a global maxima in model probability space. In domains like text classification, with a very large number of parameters, this effect can be very significant. We show that when model probability and classification are well-correlated, the use of deterministic annealing, an alternate modeling estimation process, finds more probable and thus more accurate classifiers.

Non-generative approaches have also been used for semi-supervised text classification. Joachims [1999] uses transductive support vector machines to build discriminative classifiers for several text classification tasks. Blum and Mitchell [1998] use the co-training setting to build naive Bayes classifiers for web pages, using anchor text and the page itself as two different sources of information about an instance. Zelikovitz and Hirsh [2000] use unlabeled data as background knowledge to augment a nearest-neighbor classifier. Instead of matching a test example directly to its closest labeled example, they instead match a test example to a labeled example by measuring their similarity to a common set of unlabeled examples.

This chapter proceeds as follows. Section 3.2 presents the generative model used for text classification and shows how to perform semi-supervised learning with EM. Section 3.3 shows an example where this approach works well. Section 3.4 presents a more expressive generative model that works when the naive Bayes assumption is not sufficient, and experimental results from a domain that needs it. Section 3.5 presents deterministic annealing and shows that this finds model parameterizations that are much more probable than those found by EM, especially when labeled data are sparse.

3.2 A Generative Model for Text

This section presents a framework for characterizing text documents and shows how to use this to train a classifier from labeled and unlabeled data. The framework defines a probabilistic generative model, and embodies three assumptions about the generative process: (1) the data are produced by a mixture model, (2) there is a one-to-one correspondence between mixture components and classes, and (3) the mixture components are multinomial distributions of individual words. These are the assumptions used by the naive Bayes classifier, a commonly-used tool for standard supervised text categorization [Lewis, 1998, McCallum and Nigam, 1998a].

We assume documents are generated by a *mixture of multinomials* model, where each mixture component corresponds to a class. Let there be M classes and a vocabulary of size $|\mathcal{X}|$; each document x_i has $|x_i|$ words in it. How do we create a document using this model? First, we roll a biased M-sided die to determine the class of our document. Then, we pick up the biased $|\mathcal{X}|$-sided die that corresponds to the chosen class. We roll this die $|x_i|$ times, and count how many times each word occurs. These word counts form the generated document.

Formally, every document is generated according to a probability distribution defined by the parameters for the mixture model, denoted θ. The probability distribution consists of a mixture of components $c_j \in [M]$.[1] A document, x_i, is created by first selecting a mixture component according to the mixture weights (or class probabilities), $\mathrm{P}(c_j|\theta)$, then using this selected mixture component to generate a document according to its own parameters, with distribution $\mathrm{P}(x_i|c_j;\theta)$. Thus, the likelihood of seeing document x_i is a sum of total

1. We use the notation $[M]$ to refer to the set $\{1, \ldots, M\}$.

probability over all mixture components:

$$\mathrm{P}(x_i|\theta) = \sum_{j\in[M]} \mathrm{P}(c_j|\theta)\mathrm{P}(x_i|c_j;\theta). \tag{3.1}$$

Each document has a class label. We assume a one-to-one correspondence between mixture model components and classes, and thus use c_j to indicate the jth mixture component, as well as the jth class. The class label for a particular document x_i is written y_i. If document x_i was generated by mixture component c_j we say $y_i = c_j$.

A document, x_i, is a vector of word counts. We write x_{it} to be the number of times word w_t occurs in document x_i. When a document is to be generated by a particular mixture component a document length, $|x_i| = \sum_{t=1}^{|\mathcal{X}|} x_{it}$, is first chosen independently of the component[2]. Then, the selected mixture component is used to generate a document of the specified length, by drawing from its multinomial distribution.

From this we can expand the second term from Equation 3.1, and express the probability of a document given a mixture component in terms of its constituent features: the document length and the words in the document[3].

$$\mathrm{P}(x_i|c_j;\theta) \propto \mathrm{P}(|x_i|) \prod_{w_t\in\mathcal{X}} \mathrm{P}(w_t|c_j;\theta)^{x_{it}}. \tag{3.2}$$

This formulation embodies the standard naive Bayes assumption: that the words of a document are conditionally independent of the other words in the same document, given the class label.

Thus the parameters of an individual mixture component define a multinomial distribution over words, *i.e.* the collection of word probabilities, each written $\theta_{w_t|c_j}$, such that $\theta_{w_t|c_j} \equiv \mathrm{P}(w_t|c_j;\theta)$, where $t \in [|\mathcal{X}|]$ and $\sum_t \mathrm{P}(w_t|c_j;\theta) = 1$. Since we assume that for all classes, document length is identically distributed, it does not need to be parameterized for classification. The only other parameters of the model are the mixture weights (class probabilities),$\theta_{c_j} \equiv \mathrm{P}(c_j|\theta)$, which indicate the probabilities of selecting the different mixture components. Thus the complete collection of model parameters, θ, defines a set of multinomials and class probabilities: $\theta = \{\theta_{w_t|c_j} : w_t \in \mathcal{X}, c_j \in [M]\ ;\ \theta_{c_j} : c_j \in [M]\}$.

To summarize, the full generative model, given by combining equation (3.1) and (3.2), assigns probability $P(x_i|\theta)$ to generating document x_i as follows:

$$\mathrm{P}(x_i|\theta) \propto \mathrm{P}(|x_i|) \sum_{j\in[M]} \mathrm{P}(c_j|\theta) \prod_{w_t\in\mathcal{X}} \mathrm{P}(w_t|c_j;\theta)^{x_{it}} \tag{3.3}$$

where the set of word counts x_{it} is a sufficient statistic for the parameter vector θ in this

2. This assumes that document length is independent of class, though length could also be modeled and parameterized on a class-by-class basis.
3. We omit here the multinomial coefficients for notational simplicity. For classification purposes, these coefficients cancel out.

generative model.

3.2.1 Supervised Text Classification with Generative Models

Learning a naive Bayes text classifier from a set of labeled documents consists of estimating the parameters of the generative model. The estimate of the parameters θ is written $\hat{\theta}$. Naive Bayes uses the maximum a posteriori (MAP) estimate, thus finding $\arg\max_\theta \mathrm{P}(\theta|X,Y)$. This is the value of θ that is most probable given the evidence of the training data and a prior.

Our prior distribution is formed with the product of Dirichlet distributions—one for each class multinomial and one for the overall class probabilities. The Dirichlet is the commonly-used conjugate prior distribution for multinomial distributions. The form of the Dirichlet is:

$$\mathrm{P}(\theta_{w_t|c_j}|\alpha) \propto \prod_{w_t \in \mathcal{X}} \mathrm{P}(w_t|c_j)^{\alpha_t - 1} \tag{3.4}$$

where the α_t are constants greater than zero. We set all $\alpha_t = 2$, which corresponds to a prior that favors the uniform distribution. This is identical to Laplace and m-estimate smoothing. A well-presented introduction to Dirichlet distributions is given by Stolcke and Omohundro [1994].

The parameter estimation formulae that result from maximization with the data and our prior are the familiar smoothed ratios of empirical counts. The word probability estimates $\hat{\theta}_{w_t|c_j}$ are:

$$\hat{\theta}_{w_t|c_j} \equiv \mathrm{P}(w_t|c_j;\hat{\theta}) = \frac{1 + \sum_{x_i \in X} \delta_{ij} x_{it}}{|\mathcal{X}| + \sum_{s=1}^{|\mathcal{X}|} \sum_{x_i \in X} \delta_{ij} x_{is}}, \tag{3.5}$$

where δ_{ij} is given by the class label: 1 when $y_i = c_j$ and 0 otherwise.

The class probabilities, $\hat{\theta}_{c_j}$, are estimated in the same manner, and also involve a ratio of counts with smoothing:

$$\hat{\theta}_{c_j} \equiv \mathrm{P}(c_j|\hat{\theta}) = \frac{1 + \sum_{i=1}^{|X|} \delta_{ij}}{M + |X|}. \tag{3.6}$$

The derivation of these ratios-of-counts formulae comes directly from maximum a posteriori parameter estimation. Finding the θ that maximizes $\mathrm{P}(\theta|X,Y)$ is accomplished by first breaking this expression into two terms by Bayes' rule: $\mathrm{P}(\theta|X,Y) \propto \mathrm{P}(X,Y|\theta)\mathrm{P}(\theta)$. The first term is calculated by the product of all the document likelihoods (from Equation 3.1). The second term, the prior distribution over parameters, is the product of Dirichlets. The whole expression is maximized by solving the system of partial derivatives of $\log(\mathrm{P}(\theta|X,Y))$, using Lagrange multipliers to enforce the constraint that the word probabilities in a class must sum to one. This maximization yields the ratio of counts seen above.

Given estimates of these parameters calculated from labeled training documents, it is possible to turn the generative model backwards and calculate the probability that a particular mixture component generated a given document to perform classification. This follows from an application of Bayes' rule:

$$\begin{aligned} \mathrm{P}(y_i = c_j | x_i; \hat{\theta}) &= \frac{\mathrm{P}(c_j|\hat{\theta})\mathrm{P}(x_i|c_j;\hat{\theta})}{\mathrm{P}(x_i|\hat{\theta})} \\ &= \frac{\mathrm{P}(c_j|\hat{\theta}) \prod_{w_t \in \mathcal{X}} \mathrm{P}(w_t|c_j;\hat{\theta})^{x_{it}}}{\sum_{k=1}^{M} \mathrm{P}(c_k|\hat{\theta}) \prod_{w_t \in \mathcal{X}} \mathrm{P}(w_t|c_k;\hat{\theta})^{x_{it}}} \end{aligned} \tag{3.7}$$

If the task is to classify a test document x_i into a single class, then the class with the highest posterior probability, $\arg\max_j \mathrm{P}(y_i = c_j|x_i;\hat{\theta})$, is selected.

3.2.2 Semi-Supervised Text Classification with EM

In the semi-supervised setting with labeled and unlabeled data, we would still like to find MAP parameter estimates, as in the supervised setting above. Because there are no labels for the unlabeled data, the closed-form equations from the previous section are not applicable. However, using the Expectation-Maximization (EM) technique, we can find locally MAP parameter estimates for the generative model.

The EM technique as applied to the case of labeled and unlabeled data with naive Bayes yields a straightforward and appealing algorithm. First, a naive Bayes classifier is built in the standard supervised fashion from the limited amount of labeled training data. Then, we perform classification of the unlabeled data with the naive Bayes model, noting not the most likely class but the probabilities associated with each class. Then, we rebuild a new naive Bayes classifier using all the data—labeled and unlabeled—using the estimated class probabilities as true class labels. This means that the unlabeled documents are treated as several fractional documents according to these estimated class probabilities. We iterate this process of classifying the unlabeled data and rebuilding the naive Bayes model until it converges to a stable classifier and set of labels for the data. This algorithm is summarized in Table 3.1.

More formally, learning a classifier is approached as calculating a maximum a posteriori estimate of θ, *i.e.* $\arg\max_\theta \mathrm{P}(\theta)\mathrm{P}(X, Y|\theta)$, which is equivalent to maximizing the log of the same. Consider the second term of the maximization, the probability of all the observable data. The probability of an individual unlabeled document is a sum of total probability over all the classes, as in Equation 3.1. For the labeled data, the generating component is already given by label y_i and we do not need to refer to all mixture components—just the one corresponding to the class. Using X_u to refer to the unlabeled examples, and X_l to refer to the examples for which labels are given, the expected log probability of the full data is:

- **Inputs:** Collections X_l of labeled documents and X_u of unlabeled documents.
- Build an initial naive Bayes classifier, $\hat{\theta}$, from the labeled documents, X_l, only. Use maximum a posteriori parameter estimation to find $\hat{\theta} = \arg\max_\theta \mathrm{P}(X_l|\theta)\mathrm{P}(\theta)$ (see Equations 3.5 and 3.6).
- Loop while classifier parameters improve, as measured by the change in $l(\theta|X, Y)$ (the log probability of the labeled and unlabeled data, and the prior) (see Equation 3.8):
  - **(E-step)** Use the current classifier, $\hat{\theta}$, to estimate component membership of each unlabeled document, *i.e.*, the probability that each mixture component (and class) generated each document, $\mathrm{P}(c_j|x_i;\hat{\theta})$ (see Equation 3.7).
  - **(M-step)** Re-estimate the classifier, $\hat{\theta}$, given the estimated component membership of each document. Use maximum a posteriori parameter estimation to find $\hat{\theta} = \arg\max_\theta \mathrm{P}(X, Y|\theta)\mathrm{P}(\theta)$ (see Equations 3.5 and 3.6).
- **Output:** A classifier, $\hat{\theta}$, that takes an unlabeled document and predicts a class label.

Table 3.1 The basic EM algorithm for semi-supervised learning of a text classifier.

$$\begin{aligned} l(\theta|X,Y) &= \log(\mathrm{P}(\theta)) + \sum_{x_i \in X_u} \log \sum_{j \in [M]} \mathrm{P}(c_j|\theta)\mathrm{P}(x_i|c_j;\theta) \\ &\quad + \sum_{x_i \in X_l} \log\left(\mathrm{P}(y_i = c_j|\theta)\mathrm{P}(x_i|y_i = c_j;\theta)\right). \end{aligned} \tag{3.8}$$

(We have dropped the constant terms for convenience.) Notice that this equation contains a log of sums for the unlabeled data, which makes a maximization by partial derivatives computationally intractable. The formalism of Expectation-Maximization (EM) [Dempster et al., 1977] provides an iterative hill-climbing approach to finding a local maxima of model probability in parameter space. The E-step of the algorithm estimates the expectations of the missing values (i.e. unlabeled class information) given the latest iteration of the model parameters. The M-step maximizes the likelihood of the model parameters using the previously-computed expectations of the missing values as if they were the true ones.

In practice, the E-step corresponds to performing classification of each unlabeled document using Equation 3.7. The M-step corresponds to calculating a new maximum a posteriori estimate for the parameters, $\hat{\theta}$, using Equations 3.5 and 3.6 with the current estimates for $\mathrm{P}(c_j|x_i;\hat{\theta})$.

Essentially all initializations of the parameters lead to some local maxima with EM. Many instantiations of EM begin by choosing a starting model parameterization randomly. In our case, we can be more selective about the starting point since we have not only unlabeled data, but also some labeled data. Our iteration process is initialized with a priming M-step, in which only the labeled documents are used to estimate the classifier parameters, $\hat{\theta}$, as in Equations 3.5 and 3.6. Then the cycle begins with an E-step that uses this classifier to probabilistically label the unlabeled documents for the first time.

The algorithm iterates until it converges to a point where $\hat{\theta}$ does not change from one iteration to the next. Algorithmically, we determine that convergence has occurred by observing a below-threshold change in the log-probability of the parameters (Equation 3.8),

which is the height of the surface on which EM is hill-climbing.

3.2.3 Discussion

The justifications for this approach depend on the assumptions stated in Section 3.2, namely, that the data are produced by a mixture model, and that there is a one-to-one correspondence between mixture components and classes. If the generative modeling assumptions were correct, then maximizing model probability would be a good criteria indeed for training a classifier. In this case the Bayes optimal classifier, when the number of training examples approaches infinity, corresponds to the MAP parameter estimates of the model. When these assumptions do not hold—as certainly is the case in real-world textual data—the benefits of unlabeled data are less clear. With only labeled data, the Naive Bayes classifier does a good job of classifying text documents [Lewis and Ringuette, 1994, Craven et al., 2000, Yang and Pedersen, 1997, Joachims, 1997, McCallum et al., 1998]. This observation is explained in part by the fact that classification estimation is only a function of the sign (in binary classification) of the function estimation [Domingos and Pazzani, 1997, Friedman, 1997]. The faulty word independence assumption exacerbates the tendency of naive Bayes to produce extreme (almost 0 or 1) class probability estimates. However, classification accuracy can be quite high even when these estimates are inappropriately extreme.

Semi-supervised learning leans more heavily on the correctness of the modeling assumptions than supervised learning. The next section will show empirically that this method can indeed dramatically improve the accuracy of a document classifier, especially when there are only a few labeled documents.

3.3 Experimental Results with Basic EM

In this section we demonstrate that semi-supervised learning with labeled and unlabeled data provides text classifiers that are more accurate than those provided by supervised learning using only the labeled data. This is an interesting result as the mixture of multinomials generative model is a dramatic simplification of the true authoring process. However, we demonstrate that for some domains, the optimization criteria of model probability is strongly correlated with classification accuracy.

Experiments in this section use the well-known 20 Newsgroups text classification dataset [Mitchell, 1997], consisting of about 20,000 Usenet articles evenly distributed across twenty newsgroups. The task is to classify an article into the newsgroup to which it was posted. For preprocessing, stopwords are removed and word counts of each document are scaled such that each document has constant length, with potentially fractional word counts. As the data have timestamps, a test set is formed from the last 20% of articles from each newsgroup. An unlabeled set is formed by randomly selecting 10,000 articles from those remaining. Labeled training sets are formed by partitioning the remaining documents into non-overlapping sets. We create up to ten training sets per size of the set, as data are available. When posterior model probability is reported and shown on graphs, some

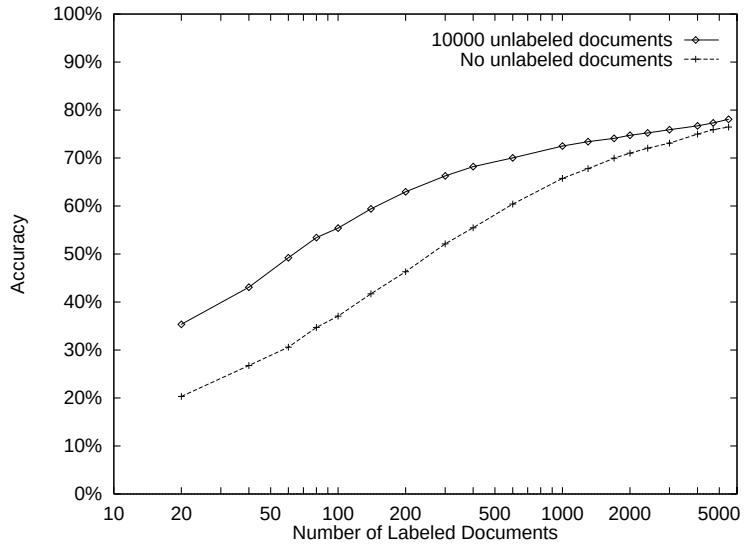

Figure 3.1 Classification accuracy on the 20 Newsgroups data set, both with and without 10,000 unlabeled documents. With small amounts of training data, using EM yields more accurate classifiers. With large amounts of labeled training data, accurate parameter estimates can be obtained without the use of unlabeled data, and classification accuracies of the two methods begin to converge.

additive and multiplicative constants are dropped, but the relative values are maintained.

Figure 3.1 shows the effect of using EM with unlabeled data on this data set. The vertical axis indicates average classifier accuracy on test sets, and the horizontal axis indicates the amount of labeled training data on a log scale. We vary the amount of labeled training data, and compare the classification accuracy of traditional naive Bayes (no unlabeled documents) with an EM learner that has access to 10000 unlabeled documents.

EM performs significantly better than traditional naive Bayes. For example, with 300 labeled documents (15 documents per class), naive Bayes reaches 52% accuracy while EM achieves 66%. This represents a 30% reduction in classification error. Note that EM also performs well even with a very small number of labeled documents; with only 20 documents (a single labeled document per class), naive Bayes obtains 20%, EM 35%. As expected, when there is a lot of labeled data, and the naive Bayes learning curve is close to a plateau, having unlabeled data does not help nearly as much, because there is already enough labeled data to accurately estimate the classifier parameters. With 5500 labeled documents (275 per class), classification accuracy increases from 76% to 78%. Each of these results is statistically significant ($p < 0.05$).[4]

How does EM find more accurate classifiers? It does so by optimizing on posterior model probability, not classification accuracy directly. If our generative model were perfect then we would expect model probability and accuracy to be correlated and EM to be helpful.

4. When the number of labeled examples is small, we have multiple trials, and use paired t-tests. When the number of labeled examples is large, we have a single trial, and report results instead with a McNemar test. These tests are discussed further by Dietterich [1998].

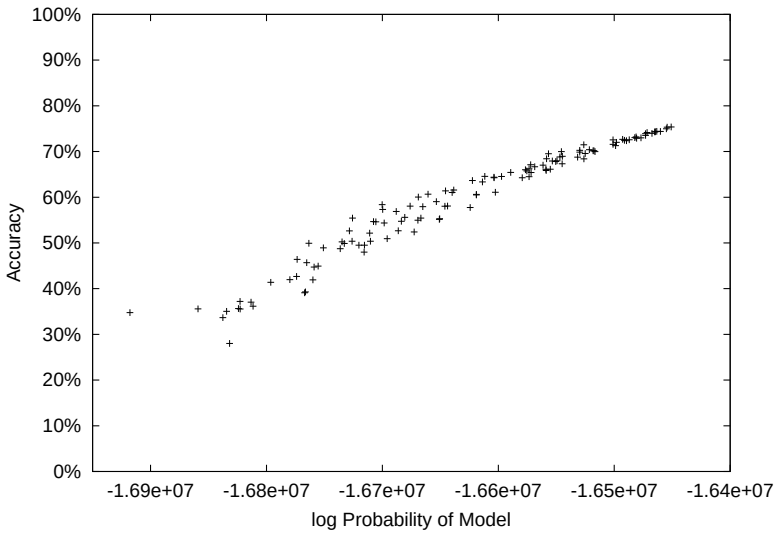

Figure 3.2 A scatterplot showing the correlation between the posterior model probability and the accuracy of a model trained with labeled and unlabeled data. The strong correlation implies that model probability is a good optimization criteria for the 20 Newsgroups dataset.

But, we know that our simple generative model does not capture many of the properties contained in the text. Our 20 Newsgroups results show that we do not need a perfect model for EM to help text classification. Generative models are representative enough for the purposes of text classification if model probability and accuracy are correlated, allowing EM to indirectly optimize accuracy.

To illustrate this more definitively, let us look again at the 20 Newsgroups experiments, and empirically measure this correlation. Figure 3.2 demonstrates the correlation—each point in the scatterplot is one of the labeled and unlabeled splits from Figure 3.1. The labeled data here are used only for setting the EM initialization and are not used during iterations. We plot classification performance as accuracy on the test data and show the posterior model probability.

For this dataset, classification accuracy and model probability are in good correspondence. The correlation coefficient between accuracy and model probability is 0.9798, a very strong correlation indeed. We can take this as a post-hoc verification that this dataset is amenable to using unlabeled data via a generative model approach. The optimization criteria of model probability is applicable here because it is in tandem with accuracy.

3.4 Using a More Expressive Generative Model

The second assumption of the generative model of Section 3.2 states that there is a one-to-one correspondence between classes and components in the mixture model. In some text domains, it is clear that such an assumption is a dangerous one. Consider the task of text filtering, where we want to identify a small well-defined class of documents from a very

large pool or stream of documents. One example might be a system that watches a network administrator's incoming emails to identify the rare emergency situation that would require paging her on vacation. Modeling the non-emergency emails as the negative class with only one multinomial distribution will result in an unrepresentative model. The negative class contains emails with a variety of sub-topics: personal emails, non-emergency requests, spam, and many more.

What would be a more representative model? Instead of modeling a sea of negative examples with a single mixture component, it might be better to model it with many components. In this way, each negative component could, after maximization, capture one clump of the sea of examples. This section takes exactly the approach suggested by this example for text data, and relaxes the assumption of a one-to-one correspondence between mixture components and classes. We replace it with a less restrictive assumption: a *many*-to-one correspondence between mixture components and classes. This allows us to model the sub-topic structure of a class.

3.4.1 Multiple Mixture Components per Class

The new generative model must account for a many-to-one correspondence between mixture components and classes. As in the old model, we first pick a class with a biased die roll. Each class has several sub-topics; we next pick one of these sub-topics, again with a biased die roll. Now that the sub-topic is determined, the document's words are generated. We do this by first picking a length (independently of sub-topic and class) and then draw the words from the sub-topic's multinomial distribution.

Unlike previously, there are now two missing values for each unlabeled document—its class and its sub-topic. Even for the labeled data there are missing values; although the class is known, its sub-topic is not. Since we do not have access to these missing class and sub-topic labels, we must use a technique such as EM to estimate local MAP generative parameters. As in Section 3.2.2, EM is instantiated as an iterative algorithm that alternates between estimating the values of missing class and sub-topic labels, and calculating the MAP parameters using the estimated labels. After EM converges to high-probability parameter estimates the generative model can be used for text classification by turning it around with Bayes' rule.

The new generative model specifies a separation between mixture components and classes. Instead of using c_j to denote both of these, $c_j \in [N]$ now denotes only the jth mixture component (sub-topic). We write $t_a \in [M]$ for the ath class; when component c_j belongs to class t_a, then $q_{aj} = 1$, and otherwise 0. This represents the pre-determined, deterministic, many-to-one mapping between mixture components and classes. We indicate the class label and sub-topic label of a document by y_i and z_i, respectively. Thus if document x_i was generated by mixture component c_j we say $z_i = c_j$, and if the document belongs to class t_a then we say $y_i = t_a$.

If all the class and sub-topic labels were known for our dataset, finding MAP estimates for the generative parameters would be a straightforward application of closed-form equations similar to those for naive Bayes seen in Section 3.2.1. The formula for the word probability parameters is identical to Equation 3.5 for naive Bayes:

$$\hat{\theta}_{w_t|c_j} \equiv \mathrm{P}(w_t|c_j;\hat{\theta}) = \frac{1 + \sum_{x_i \in X} \delta_{ij} x_{it}}{|\mathcal{X}| + \sum_{s=1}^{|\mathcal{X}|} \sum_{x_i \in X} \delta_{ij} x_{is}}. \tag{3.9}$$

The class probabilities are analogous to Equation 3.6, but using the new notation for classes instead of components:

$$\hat{\theta}_{t_a} \equiv \mathrm{P}(t_a|\hat{\theta}) = \frac{1 + \sum_{i=1}^{|X|} \delta_{ia}}{M + |X|}. \tag{3.10}$$

The sub-topic probabilities are similar, except they are estimated only with reference to other documents in that component's class:

$$\hat{\theta}_{c_j|t_a} \equiv \mathrm{P}(c_j|t_a;\hat{\theta}) = \frac{1 + \sum_{i=1}^{|X|} \delta_{ij}\delta_{ia}}{\sum_{j=1}^{N} q_{aj} + \sum_{i=1}^{|X|} \delta_{ia}}. \tag{3.11}$$

At classification time, we must estimate class membership probabilities for an unlabeled document. This is done by first calculating sub-topic membership and then summing over sub-topics to get overall class probabilities. Sub-topic membership is calculated analogously to mixture component membership for naive Bayes, with a small adjustment to account for the presence of two priors (class and sub-topic) instead of just one:

$$\mathrm{P}(z_i = c_j|x_i;\hat{\theta}) = \frac{\sum_{a \in [M]} q_{aj} \mathrm{P}(t_a|\hat{\theta}) \mathrm{P}(c_j|t_a;\hat{\theta}) \prod_{w_t \in \mathcal{X}} \mathrm{P}(w_t|c_j;\hat{\theta})^{x_{it}}}{\sum_{r \in [N]} \sum_{b \in [M]} q_{br} \mathrm{P}(t_b|\hat{\theta}) \mathrm{P}(c_r|t_b;\hat{\theta}) \prod_{w_t \in \mathcal{X}} \mathrm{P}(w_t|c_r;\hat{\theta})^{x_{it}}}. \tag{3.12}$$

Overall class membership is calculated with a sum of probability over all of the class's sub-topics:

$$\mathrm{P}(y_i = t_a|x_i;\hat{\theta}) = \sum_{j \in [N]} q_{aj} \mathrm{P}(z_i = c_j|x_i;\hat{\theta}) \tag{3.13}$$

These equations for supervised learning are applicable only when all the training documents have both class and sub-topic labels. Without these we use EM. The M-step, as with basic EM, builds maximum a posteriori parameter estimates for the multinomials and priors. This is done with Equations 3.9, 3.10, and 3.11, using the probabilistic class and sub-topic memberships estimated in the previous E-step. In the E-step, for the unlabeled documents we calculate probabilistically-weighted sub-topic and class memberships (Equations 3.12 and 3.13). For labeled documents, we must estimate sub-topic membership. But, we know from its given class label that many of the sub-topic memberships must be zero—those sub-topics that belong to other classes. Thus we calculate sub-topic memberships as for the unlabeled data, but setting the appropriate ones to zero, and normalizing the non-zero ones over only those topics that belong to its class.

If we are given a set of class-labeled data, and a set of unlabeled data, we can now apply

EM if there is some specification of the number of sub-topics for each class. However, this information is not typically available. As a result we must resort to some techniques for model selection. There are many commonly-used approaches to model selection such as cross-validation, AIC, BIC and others. Since we do have the availability of a limited number of labeled documents, we use cross-validation to select the number of sub-topics for classification performance.

3.4.2 Experimental Results

Here, we provide empirical evidence that to use unlabeled data with a generative modeling approach, more expressive generative models are sometime necessary. With the original generative model, classification accuracy and model probability can be negatively correlated, leading to lower classification accuracy when unlabeled data are used. With a more expressive generative model, a moderate positive correlation is achieved, leading to improved classification accuracies.

The Reuters 21578 Distribution 1.0 dataset consists of about 13,000 news articles from the Reuters newswire labeled with 90 topic categories. Documents in this data set have multiple class labels, and each category is traditionally evaluated with a binary classifier. Following several other studies [Joachims, 1998, Liere and Tadepalli, 1997] we build binary classifiers for each of the ten most populous classes to identify the topic. We use a stoplist, but do not stem. The vocabulary size for each Reuters trial is selected by optimizing accuracy as measured by leave-one-out cross-validation on the labeled training set. The standard ModApte train/test split is used, which is time-sensitive. Seven thousand of the 9603 documents available for training are left unlabeled. From the remaining, we randomly select up to ten non-overlapping training sets of just ten positively labeled documents and 40 negatively labeled documents.

The first two columns of results in Table 3.2 repeat the experiments of Section 3.3 with basic EM on the Reuters dataset. Here we see that for most categories, classification accuracy decreases with the introduction of unlabeled data. For each of the Reuters categories EM finds a significantly more probable model, given the evidence of the labeled and unlabeled data. But frequently this more probable model corresponds to a lower-accuracy classifier—not what we would hope for.

The first graph in Figure 3.3 provides insight into why unlabeled data hurts. With one mixture component per class, the correlation between classification accuracy and model probability is very strong ($r = -0.9906$), but in the wrong direction! Models with higher probability have significantly lower classification accuracy. By examining the solutions found by EM, we find that the most probable clustering of the data has one component with the majority of negative documents and the second with most of the positive documents, but significantly more negative documents. Thus, the classes do not separate with high-probability models.

The documents in this dataset often have multiple class labels. With the basic generative model, the negative class covers up to 89 distinct categories. Thus, it is unreasonable to expect to capture such a broad base of text with a single mixture component. For this reason, we relax the generative model and model the positive class with a single mixture

Category	NB1	EM1	NB*	EM*
acq	86.9	81.3	88.0 (4)	**93.1** (10)
corn	94.6	93.2	96.0 (10)	**97.2** (40)
crude	94.3	94.9	95.7 (13)	**96.3** (10)
earn	94.9	95.2	**95.9** (5)	95.7 (10)
grain	94.1	93.6	96.2 (3)	**96.9** (20)
interest	91.8	87.6	95.3 (5)	**95.8** (10)
money-fx	93.0	90.4	94.1 (5)	**95.0** (15)
ship	94.9	94.1	**96.3** (3)	95.9 (3)
trade	91.8	90.2	94.3 (5)	**95.0** (20)
wheat	94.0	94.5	96.2 (4)	**97.8** (40)

Table 3.2 Classification accuracy of binary classifiers on Reuters with traditional naive Bayes (NB1), basic EM (EM1) with labeled and unlabeled data, multiple mixture components using just labeled data (NB*), and multiple mixture components EM with labeled and unlabeled data (EM*). For NB* and EM*, the number of components is selected optimally for each trial, and the median number of components across the trials used for the negative class is shown in parentheses. Note that the multi-component model is more natural for Reuters, where the negative class consists of many topics. Using both unlabeled data and multiple mixture components per class increases performance over either alone, and over naive Bayes.

component and the negative class with between one and forty mixture components, both with and without unlabeled data.

The second half of Table 3.2 shows results of using a multiple mixtures per class generative model. Note two different results. First, with labeled data alone (NB*) classification accuracy improves over the single component per class case (NB1). Second, with unlabeled data, the new generative model results (EM*) are generally better than the other results. This increase with unlabeled data, measured over all trials of Reuters, is statistically significant ($p < 0.05$).

With ten mixture components the correlation between accuracy and model probability is quite different. Figure 3.3 on the right shows the correlation between accuracy and model probability when using ten mixture components to model the negative class. Here, there is a moderate correlation between model probability and classification accuracy in the right direction ($r = 0.5474$). For these solutions, one component covers nearly all the positive documents and some, but not many, negatives. The other ten components are distributed through the remaining negative documents. This model is more representative of the data for our classification task because classification accuracy and model probability are correlated. This allows the beneficial use of unlabeled data through the generative model approach.

One obvious question is how to automatically select the best number of mixture components without having access to the test set labels. We use leave-one-out cross-validation. Results from this technique (EM*CV), compared to naive Bayes (NB1) and the best EM (EM*), are shown in Table 3.3. Note that cross-validation does not perfectly select the number of components that perform best on the test set. The results consistently show that selection by cross-validation chooses a smaller number of components than is best.

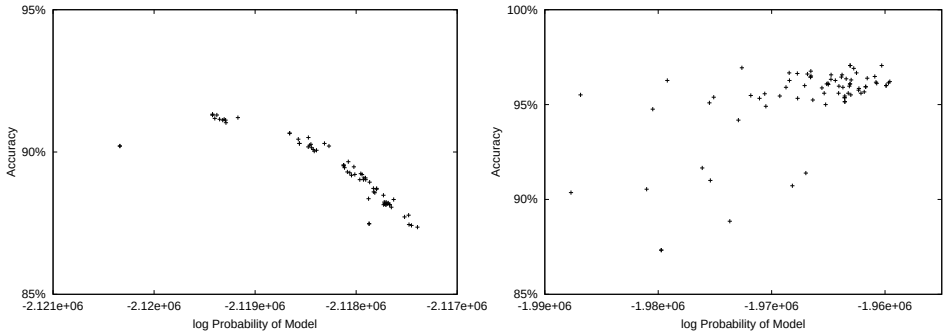

Figure 3.3 Scatterplots showing the relationship between model probability and classification accuracy for the Reuters acq task. On the left, with only one mixture component for the negative class, probability and accuracy are inversely proportional, exactly what we would not want. On the right, with ten mixture components for negative, there is a moderate positive correlation between model probability and classification accuracy.

Category	NB1	EM*	EM*CV	EM*CV vs NB1
acq	86.9	93.1 (10)	91.1 (5)	+4.2
corn	94.6	97.2 (40)	93.2 (3)	-1.4
crude	94.3	96.3 (10)	95.4 (3)	+1.1
earn	94.9	95.7 (10)	95.2 (1)	+0.3
grain	94.1	96.9 (20)	94.7 (3)	+0.6
interest	91.8	95.8 (10)	92.0 (3)	+0.2
money-fx	93.0	95.0 (15)	92.3 (3)	-0.7
ship	94.9	95.9 (3)	94.4 (3)	-0.5
trade	91.8	95.0 (20)	90.7 (3)	-1.1
wheat	94.0	97.8 (40)	96.3 (6)	+2.3

Table 3.3 Performance of using multiple mixture components when the number of components is selection via cross-validation (EM*CV) compared to the optimal selection (EM*) and straight naive Bayes (NB1). Note that cross-validation usually selects too few components.

3.4.3 Discussion

There is tension in this model selection process between complexity of the model and data sparsity. With as many sub-topics as there are documents, we can perfectly model the training data—each sub-topic covers one training document. With still a large number of sub-topics, we can accurately model existing data, but generalization performance will be poor. This is because each multinomial will have its many parameters estimated from only a few documents and will suffer from sparse data. With very few sub-topics, the opposite problem will arise. We will very accurately estimate the multinomials, but the model will be overly restrictive, and not representative of the true document distribution. Cross-validation should help in selecting a good compromise between these tensions with specific regard to classification performance.

Note that our use of multiple mixture components per class allows us to capture some dependencies between words on the class-level. For example, consider a sports class consisting of documents about both hockey and baseball. In these documents, the words *ice* and *puck* are likely to co-occur, and the words *bat* and *base* are likely to co-occur. However, these dependencies cannot be captured by a single multinomial distribution over words in the sports class. With multiple mixture components per class, one multinomial can cover the hockey sub-topic, and another the baseball sub-topic. In the hockey sub-topic, the word probability for *ice* and *puck* will be significantly higher than they would be for the whole class. This makes their co-occurrence more likely in hockey documents than it would be under a single multinomial assumption.

3.5 Overcoming the Challenges of Local Maxima

In cases where the likelihood in parameter space is well-correlated with classification accuracy, our optimization yields good classifiers. However, local maxima significantly hinder our progress. For example, the local maxima we discover with just a few labeled examples in Section 3.3 are more than 40 percentage points below the classification accuracy provided when labeled data are plentiful. Thus it is important to consider alternate approaches that can help bridge this gap, especially when labeled data are sparse.

Typically variants of, or alternatives to, EM are created for the purpose of speeding up the rate of convergence [McLachlan and Krishnan, 1997]. In the domain of text classification however, we have seen that convergence is very fast. Thus, we can easily consider alternatives to EM that improve the local maxima situation at the expense of slower convergence. Deterministic annealing makes exactly this tradeoff.

3.5.1 Deterministic Annealing

The intuition behind deterministic annealing is that it begins by maximizing on a very smooth, convex surface that is only remotely related to our true probability surface of interest. Initially we can find the global maximum of this simple surface. Ever-so-slowly, we change the surface to become both more bumpy, and more close to the true probability surface. If we follow the original maximum as the surface gets more complex, then when the original surface is given, we'll still have a highly probable maximum. In this way, it avoids many of the local maxima that EM would otherwise get caught in.

One can think of our application of EM in the previous sections as an optimization problem where the loss function is the negation of the likelihood function (Equation 3.8). The iterations of EM are a hill-climbing algorithm in parameter space that locally minimizes this loss.

Consider the closely related set of loss functions:

$$l(\theta|X,Y) = \sum_{x_i \in X_u} \log \sum_{c_j \in [M]} [\mathrm{P}(c_j|\theta)\mathrm{P}(x_i|c_j;\theta)]^\beta + \sum_{x_i \in X_l} \log([\mathrm{P}(y_i = c_j|\theta)\mathrm{P}(x_i|y_i = c_j;\theta)]^\beta), \quad (3.14)$$

where β varies between zero and one. When $\beta = 1$ we have our familiar probability surface of the previous sections, with good correlation to classification accuracy, but with many harmful local maxima. In the limit as β approaches zero, the surface value of the loss function in parameter space becomes convex with just a single global maxima. But, at this extreme, the provided data have no effect on the loss function, so the correlation with classification accuracy is poor. Values in between zero and one represent various points in the tradeoff between smoothness of the parameter space and the similarity to the well-correlated probability surface provided by the data.

This insight is the one that drives the approach called deterministic annealing [Rose et al., 1992], first used as a way to construct a hierarchy during unsupervised clustering. It has also been used to estimate the parameters of a mixture of Gaussians from unlabeled data [Ueda and Nakano, 1995] and to construct a text hierarchy from unlabeled data [Hofmann and Puzicha, 1998].

For a fixed value of β, we can find a local maxima given the loss function by iterating the following steps:

- E-step: Calculate the expected value of the class assignments,

$$\hat{z}_{ij}^{(k+1)} = E[y_i = c_j|x_i;\hat{\theta}^k] = \frac{[\mathrm{P}(c_j|\hat{\theta}^k)\mathrm{P}(x_i|c_j;\hat{\theta}^k)]^\beta}{\sum_{c_r \in [M]} [\mathrm{P}(c_r|\hat{\theta}^k)\mathrm{P}(x_i|c_r;\hat{\theta}^k)]^\beta}. \quad (3.15)$$

- M-step: Find the most likely model using the expected class assignments,

$$\hat{\theta}^{(k+1)} = \arg\max_\theta \mathrm{P}(\theta|X;Y;\hat{\mathbf{z}}^{(k+1)}). \quad (3.16)$$

The M-step is identical to that of Section 3.2.2, while the E-step includes reference to the loss constraint through β.

Formally, β is a Lagrange multiplier when solving for a fixed loss in the likelihood space subject to an optimization criteria of maximum entropy (or minimum relative entropy to the prior distribution). A β near zero corresponds to finding the maximum entropy parameterization for a model with a very large allowable loss.

Consider how model likelihood (Equation 3.14) is affected by different target losses. When the target loss is very large, β will be very close to zero; the probability of each model will very nearly be its prior probability as the influence of the data will be negligible. In the limit as β goes to zero, the probability surface will be convex with a single global maximum. For a somewhat smaller loss target, β will be small but not negligible. Here, the probability of the data will have a stronger influence. There will no longer be a single

global maximum, but several. When $\beta = 1$ we have our familiar probability surface of the previous chapters, with many local maxima.

These observations suggest an annealing-like process for finding a low-loss model. If we initialize β to be very small, we can easily find the global maximum a posteriori solution with EM, as the surface is convex. When we raise β the probability surface will get slightly more bumpy and complex, as the data likelihood will have a larger impact on the probability of the model. Although more complex, the new maximum will be very close to the old maximum if we have lowered the temperature ($1/\beta$) only slightly. Thus, when searching for the maximum with EM, we can initialize it with the old maximum and will converge to a good maximum for the new probability surface. In this way, we can gradually raise β, while tracking a highly probable solution. Eventually, when β becomes 1, we will have a good local maximum for our generative model assumptions. Thus, we will have found a high-probability local maximum from labeled and unlabeled data that we can then use for classification.

Note that the computational cost of deterministic annealing is significantly higher than EM. While each iteration takes the same computation, there are many more iterations with deterministic annealing, as the temperature is reduced very slowly. For example, in our experiments, we performed 390 iterations for deterministic annealing, and only seven for EM. When this extra computation can be afforded, the benefit may be more accurate classifiers.

3.5.2 Experimental Results

In this section we see empirically that deterministic annealing finds more probable parameters and more accurate classifiers than EM when labeled training data are sparse.

For the experimental results, we use the News5 dataset, a subset of 20 Newsgroups containing the five confusable **comp.*** classes. We fix a single vocabulary for all experiments as the top 4000 words as measured by mutual information over the entire labeled dataset. For running the deterministic annealing, we initialize β to 0.02, and at each iteration we increase β by a multiplicative factor of 1.01 until $\beta = 1$. We made little effort to tune these parameters. Since each time we increase β the probability surface changes only slightly, we run only one iteration of EM at each temperature setting. Six hundred random documents per class (3000 total) are treated as unlabeled. A fixed number of labeled examples per class are also randomly selected. The remaining documents are used as a test set.

Figure 3.4 compares classification accuracy achieved with deterministic annealing to that achieved by regular EM. The initial results indicate that the two methods perform essentially the same when labeled data are plentiful, but deterministic annealing actually performs worse when labeled data are sparse. For example with two labeled examples per class (10 total) EM gives 58% accuracy where deterministic annealing gives only 51%. A close investigation of the confusion matrices shows that there is a significant detrimental effect of incorrect class-to-component correspondence with deterministic annealing when labeled data are sparse. This occurs because, when the temperature is very high, the global maximum will have each multinomial mixture component very close to its prior,

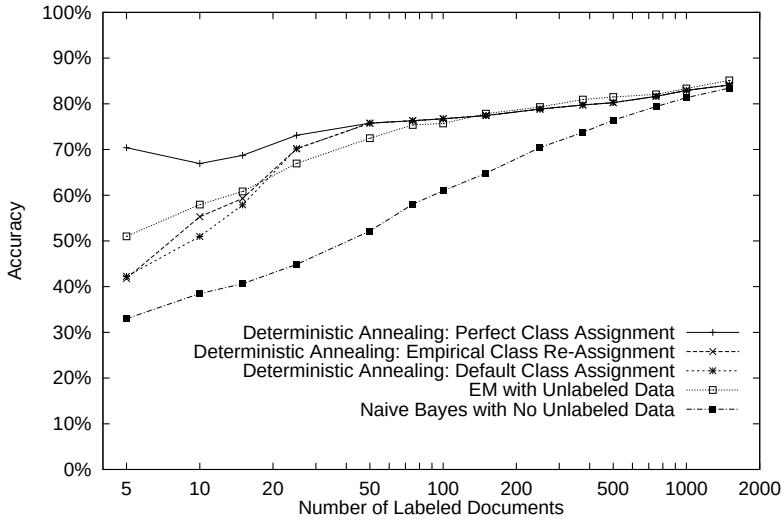

Figure 3.4 The performance of deterministic annealing compared to EM. If class-to-component assignment was done perfectly deterministic annealing would be considerably more accurate than EM when labeled data are sparse. Although the default correspondence is poor, this can be corrected with a small amount of domain knowledge.

and the influence of the labeled data is minimal. Since the priors are the same, each mixture component will be essentially identical. As the temperature lowers and the mixture components become more distinct, one component can easily track the cluster associated with the wrong class, when there is insufficient labeled data to pull it toward the correct class.

In an attempt to remedy this, we alter the class-to-cluster correspondence based on the classification of each labeled example after deterministic annealing is complete. Figure 3.4 shows both the accuracy obtained by empirically selected correspondence, and also the optimal accuracy achieved by perfect correspondence. We see that by empirically setting the correspondence, deterministic annealing improves accuracy only marginally. Where before it got 51%, by changing the correspondence we increase this to 55%, still not better than EM at 58%. However if we could perform perfect class correspondence, accuracy with deterministic annealing would be 67%, considerably higher than EM.

To verify that the higher accuracy of deterministic annealing comes from finding more probable models, Figure 3.5 shows a scatterplot of model probability versus accuracy for deterministic annealing (with optimal class assignment) and EM. Two results of note stand out. The first is that indeed deterministic annealing finds much more probable models, even with a small amount of labeled data. This accounts for the added accuracy of deterministic annealing. A second note of interest is that models found by deterministic annealing still lie along the same probability-accuracy correlation line. This provides further evidence that model probability and accuracy are strongly correlated for this dataset, and that the correlation is not just an artifact of EM.

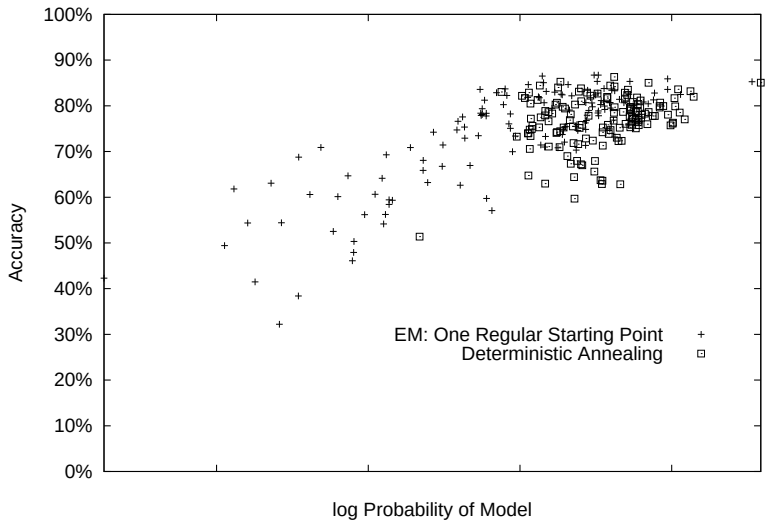

Figure 3.5 A scatterplot comparing the model probabilities and accuracies of EM and deterministic annealing. The results show that deterministic annealing succeeds because it finds models with significantly higher probability.

graphics	os.ms-windows.misc	sys.ibm.pc.hardware	sys.mac.hardware	windows.x
jpeg	windows	scsi	apple	window
image	ei	ide	mac	widget
graphics	win	drive	lc	motif
images	um	controller	duo	xterm
gif	dos	bus	nubus	server
format	ms	dx	fpu	lib
pub	ini	bios	centris	entry
ray	microsoft	drives	quadra	openwindows
tiff	nt	mb	iisi	usr
siggraph	el	card	powerbook	sun

Table 3.4 The top ten words per class of the News5 dataset, Usenet groups in the comp hierarchy. The words are sorted by the weighted log-likelihood ratio. Note that from just these ten top words, any person with domain knowledge could correctly correspond clusters and classes.

3.5.3 Discussion

The experimental results show that deterministic annealing indeed could help classification considerably if class-to-component correspondence were solved. Deterministic annealing successfully avoids getting trapped in some poor local maxima and instead finds more probable models. Since these high-probability models are correlated with high-accuracy classifiers, deterministic annealing makes good use of unlabeled data for text classification.

The class-correspondence problem is most severe when there are only limited labeled data. This is because with fewer labeled examples, it is more likely that small perturbations

can lead the correspondence astray. However, with just a little bit of human knowledge, the class-correspondence problem can typically be solved trivially. In all but the largest and most confusing classification tasks, it is straightforward to identify a class given its most indicative words, as measured by a metric such as the weighted log-likelihood ratio. For example, the top ten words per class of our dataset by this metric are shown in Table 3.4. From just these ten words, any person with even the slightest bit of domain knowledge would have no problem perfectly assigning classes to components. Thus, it is not unreasonable to require a small amount of human effort to correct the class correspondence after deterministic annealing has finished. This effort can be positioned within the active learning framework. Thus, when labeled training data are sparsest, and a modest investment by a trainer is available to map class labels to cluster components, deterministic annealing will successfully find more probable and more accurate models than traditional EM.

Even when this limited domain knowledge or human effort is not available, it should be possible to estimate the class correspondence automatically. One could perform both EM and deterministic annealing on the data. Since EM solutions generally have the correct class correspondence, this model could be used to fix the correspondence of the deterministic annealing model. That is, one could measure the distance between each EM class multinomial and each deterministic annealing class multinomial (with KL-divergence, for example). Then, this matrix of distances could be used to assign the class labels of the EM multinomials to their closest match to a multinomial in the deterministic annealing model.

3.6 Conclusions and Summary

This chapter has explored the use of generative models for semi-supervised learning with labeled and unlabeled data in domains of text classification. The widely-used naive Bayes classifier for supervised learning defines a mixture of multinomials mixture model. In some domains, model likelihood and classification accuracy are strongly correlated, despite the overly-simplified generative model. Here, Expectation-Maximization finds more likely models and improved classification accuracy. In other domains, likelihood and accuracy are not well correlated with the naive Bayes model. Here, we can use a more expressive generative model that allows for multiple mixture components per class. This helps restore a moderate correlation between model likelihood and classification accuracy, and again, EM finds more accurate models. Finally, even with a well-correlated generative model, local maxima are a significant hindrance with EM. Here, the approach of deterministic annealing does provide much higher likelihood models, but often loses the correspondence with the class labels. When class label correspondence is easily corrected, high accuracy models result.