Information Retrieval, 8, 181–196, 2005
© 2005 Springer Science + Business Media, Inc. Manufactured in The Netherlands.

Test Data Likelihood for PLSA Models*

THORSTEN BRANTS brants@google.com
Google, Inc., 1600 Amphitheatre Parkway, Bldg. 42, Mountain View, CA 94043, USA

Abstract. Probabilistic Latent Semantic Analysis (PLSA) is a statistical latent class model that has recently received considerable attention. In its usual formulation it cannot assign likelihoods to unseen documents. Furthermore, it assigns a probability of zero to unseen documents during training. We point out that one of the two existing alternative formulations of the Expectation-Maximization algorithms for PLSA does not require this assumption. However, even that formulation does not allow calculation of the actual likelihood values. We therefore derive a new test-data likelihood substitute for PLSA and compare it to three existing likelihood substitutes. An empirical evaluation shows that our new likelihood substitute produces the best predictions about accuracies in two different IR tasks and is therefore best suited to determine the number of EM steps when training PLSA models. The new likelihood measure and its evaluation also suggest that PLSA is not very sensitive to overfitting for the two tasks considered. This renders additions like tempered EM that especially address overfitting unnecessary.

Keywords: Probabilistic Latent Semantic Analysis, PLSA, likelihood

1. Introduction

Probabilistic Latent Semantic Analysis is a statistical latent class model (or aspect model) that recently has shown excellent results in several IR related tasks (Gildea and Hofmann 1999, Hofmann and Puzicha 1998, Hofmann 2000). It has the positive feature of assigning probability distributions over classes to documents and words, unlike other clustering techniques that do a hard assignment to one class. Aspect models are also successfully used in other areas of natural language processing, like language modeling (Saul and Pereira 1997) or parsing (Rooth et al. 1999).

However, PLSA does not provide a way of assigning likelihoods to instances of the observed variables that do not occur in the training corpus. In the case of retrieval tasks, it cannot assign likelihoods to unseen test documents. Even less desirable, its usual formulation proposes zero probabilities for unseen documents during training and then ignores this fact during testing when calculating probabilities conditioned on the test document.

As we will show, one of the two alternative formulations of the Expectation-Maximization algorithm for PLSA does not assume zero-probabilities for test documents, which favors this version. But still there is no intrinsic way of doing likelihood calculations. We therefore derive a new test data likelihood substitute and compare it empirically with three existing and previously used likelihood substitutes. Our new likelihood substitute has two advantages over previously used measures: first, we show that its maximum is achieved by the same

*The work reported here was carried out while the author was at the Palo Alto Research Center (PARC).

parameter setting as the true but unknown likelihood, and second, the empirical comparison reveals a good qualitative match between the new likelihood substitute and task results, which makes it best suited to determine the number of EM steps for training PLSA in an unsupervised setting.

As an additional result of our comparison we find evidence that PLSA is very robust against overfitting for the two tasks considered. On the one hand, aspect models have previously been reported to be very sensitive to overfitting when applied to information retrieval tasks (Blei et al. 2001, Gildea and Hofmann 1999, Hofmann and Puzicha 1998, Hofmann 2000) even with small numbers of classes. Therefore, the studies usually proposed using tempered EM in order to avoid overtraining. On the other hand, studies using aspect models for language modeling and parsing report these models to be relatively robust against overtraining when the number of classes is much smaller than the size of the vocabulary (Saul and Pereira 1997, Rooth et al. 1999). Our experiment plotting the number of EM steps vs. retrieval and segmentation task results show that PLSA is not sensitive to overfitting in these two tasks and therefore render techniques like tempered EM for PLSA unneccary.

The rest of the paper is structured as follows. Section 2 shortly introduces the PLSA model. Section 3 presents an alternative formulation and points out its advantage of avoiding zero probabilities for test documents. Section 4 introduces our new method of calculating substitutes for PLSA test documents likelihoods and also presents three existing methods. Section 5 empirically compares the four likelihood measures against each other and against results in document retrieval and text segmentation. Section 6 contains conclusions.

2. The PLSA model

Probabilistic Latent Semantic Analysis (PLSA) is a statistical latent class model or aspect model (Hofmann 2000, Hofmann 1999). The model is fitted to a training corpus by the Expectation Maximization (EM) algorithm (Dempster et al. 1977). It assigns probability distributions over classes to words and documents and thereby allows them to belong to more than one class, and not to only one class as is true of most other classification methods. PLSA represents the joint probability of a document d and a word w based on a latent class variable z[1]:

$$P(d, w) = P(d) \sum_{z} P(w \mid z) P(z \mid d) \tag{1}$$

PLSA has the following view of how a document is generated: first a document $d \in \mathcal{D}$ (i.e., its dummy label) is chosen with probability $P(d)$. For each word in document d, a latent topic $z \in \mathcal{Z}$ is chosen with probability $P(z \mid d)$, which in turn is used to choose the word $w \in \mathcal{W}$ with probability $P(w \mid z)$.

A model is fitted to a document collection $\mathcal{D}$ by maximizing the log-likelihood function $\mathcal{L}$:

$$\mathcal{L} = \sum_{d \in \mathcal{D}} \sum_{w \in d} f(d, w) \log P(d, w). \tag{2}$$

The E-step in the EM-algorithm is

$$P(z \mid d, w) = \frac{P(z)P(d \mid z)P(w \mid z)}{\sum_{z'} P(z')P(d \mid z')P(w \mid z')} \tag{3}$$

and the M-step consists of

$$P(w \mid z) = \frac{\sum_d f(d, w)P(z \mid d, w)}{\sum_{d,w'} f(d, w')P(z \mid d, w')} \tag{4}$$

$$P(d \mid z) = \frac{\sum_w f(d, w)P(z \mid d, w)}{\sum_{d',w} f(d', w)P(z \mid d', w)} \tag{5}$$

$$P(z) = \frac{\sum_{d,w} f(d, w)P(z \mid d, w)}{\sum_{d,w} f(d, w)}. \tag{6}$$

The parameters are either randomly initialized or according to some prior knowledge.

The parameters $P(w \mid z)$ obtained in the training process are used to calculate $P(z \mid q)$ for new documents q (queries) with the folding-in process. Folding-in uses Expectation-Maximization as in the training process; the E-step is identical, the M-step keeps all the $P(w \mid z)$ constant and re-calculates $P(z \mid q)$. Usually, a very small number of iterations is sufficient for folding-in.

3. Alternative PLSA formulation

3.1. *Zero-probability documents*

Note that the EM algorithm in (3)–(6) is based on the probabilities $P(d \mid z)$ while the probabilities used for folding-in of query documents are $P(z \mid q)$. Bayes' formula allows the reverse conditioning

$$P(z \mid q) = \frac{P(z)P(q \mid z)}{P(q)}. \tag{7}$$

A subtle but crucial point of PLSA is that we do not know $P(q)$. Even worse, the model as it is usually formulated (Hofmann 2000) (and as it is shortly presented in the previous section) postulates $P(q \mid z) = 0$ for all $z \in \mathcal{Z}$, and hence $P(q) = 0$, for all $q \notin \mathcal{D}$, where $\mathcal{D}$ is the training collection. This can be derived from Eq. (5). For any given $z \in \mathcal{Z}$, we have

$$\sum_{d\in\mathcal{D}} P(d \mid z) = \sum_{d\in\mathcal{D}} \frac{\sum_{w\in\mathcal{W}} f(d, w)P(z \mid d, w)}{\sum_{d'\in\mathcal{D}} \sum_{w\in\mathcal{W}} f(d', w)P(z \mid d', w)} = 1 \tag{8}$$

Since all the probability mass is on the $d \in \mathcal{D}$, it follows that $P(q \mid z) = 0$ for any $q \notin \mathcal{D}$.

The PLSA formulation based on $P(d \mid z)$ simply ignores this zero-probability problem and proceeds with the folding-in process. It calculates probabilities for $P(z \mid q)$ during folding-in and avoids looking at $P(q)$, hence it ignores the division of 0 by 0. The devision by zero is not made explicitly during folding-in, but it is there implicitly because of Eq. (7).

3.2. *PLSA reformulation*

We now switch to a formulation of EM for PLSA that was used in Gildea and Hofmann (1999) and Hofmann (2001). These publications presented the reformulation without giving a motivation for preferring one version or the other, and without addressing implications.

We now use probabilities $P(z \mid d)$ instead of $P(d \mid z)$. The E-Step is

$$P(z \mid d, w) = \frac{P(z \mid d)P(w \mid z)}{\sum_{z'} P(z' \mid d)P(w \mid z')} \tag{9}$$

and the M-step consists of

$$P(w \mid z) = \frac{\sum_d f(d, w)P(z \mid d, w)}{\sum_{d,w'} f(d, w')P(z \mid d, w')} \tag{10}$$

$$P(z \mid d) = \frac{\sum_w f(d, w)P(z \mid d, w)}{\sum_{w,z'} f(d, w)P(z' \mid d, w)} \tag{11}$$

This section investigates the differences between the two formulations. The log-likelihood function (2) can be written as

$$\mathcal{L} = \sum_{d\in\mathcal{D}} \sum_{w\in d} f(d, w) \log P(d) + \sum_{d\in\mathcal{D}} \sum_{w\in d} f(d, w) \log P(w \mid d) \tag{12}$$

because of

$$\log P(d, w) = \log P(d)P(w \mid d) = \log P(d) + \log P(w \mid d). \tag{13}$$

Both parts of Eq. (12) can be optimized independently of each other (cf. Hofmann and Puzicha (1998)). The PLSA parameters $P(w \mid z)$ and $P(z \mid d)$ only occur in the right hand part (in $P(w \mid d)$). Therefore, it is sufficient to maximize that part, which is done by Eqs. (9)–(11).

EM for PLSA based on $P(z \mid d)$ does not use probabilities $P(d)$, and hence $P(q)$, at all. They are not part of Eqs. (9)–(11). The formulas only implicitly require $P(d) > 0$ for all $d \in \mathcal{D}$ in order to obtain valid probabilities conditioned on d. And they require

$$\sum_{d\in\mathcal{D}} P(d) < 1 \tag{14}$$

in order to reserve probability mass to unseen documents q (i.e., having $P(q) > 0$ for test documents q). We still do not have a way of calculating the probability value for unseen documents, but the model now accounts for them and no longer postulates zero probabilities during training. Knowledge of the exact value of $P(d)$ is not necessary during training because it does not show up in the EM steps. We only need probabilities conditioned on d, so we assume $P(d) > 0$ and $P(q) > 0$.

With this formulation, folding-in uses the same EM formulas as training. For a new document q to be folded in, the algorithm alternates between the E-step for $P(z \mid q, w)$ (9) and the M-step for $P(z \mid q)$ (11). The values for $P(w \mid z)$ are given from the training process and held constant. Note that both $P(z \mid q, w)$ and $P(z \mid q)$ are independent of any $q' \neq q$, i.e., folding-in of q proceeds without information about other test documents q'.

To sum up, the reformulation maximizes the conditional part of the likelihood function in Eq. (12). Therefore, it does not need to refer to $P(d)$ and, unlike the original formulation, can assume $P(q) > 0$ for documents q not in the training set. To our knowledge, this motivation for the particular PLSA model formulation has not been published before.

This is only half of the solution to the PLSA likelihood problem. We know that $P(q) > 0$, but we still do not know the actual value of $P(q)$. However, knowledge of non-training data likelihoods is very useful when determining the number of EM-steps on a held-out data set during unsupervised learning.

4. Likelihood calculation

The EM formulation for PLSA in Section 3.2 eliminates the weakness of assigning zero probabilities to unseen documents. However, it does so by avoiding to assign a particular likelihood to any of the documents. Neither the usual formulation nor the reformulation can assign a likelihood to an unseen query document. Previously, three different test data likelihood substitutes have been suggested, two of which have been used in published experiments. In this section, we first present a new method to calculate likelihood substitutes for unseen documents based on folding-in. This method is derived from the PLSA likelihood maximization definition. The method does not give us the true likelihood values, but we show that it is maximized at the same points in the parameter space as the true likelihood. We then present the three existing likelihood substitutes.

For the sake of brevity, we make two simplifying assumptions without losing generality. First, we do not use logarithms here (for an implementation, however, logarithmic values need to be used). Second, we treat documents as multi-sets; if a term occurs several times, it is assumed to be repeated in q. This saves us from keeping track of frequency counts $f(q, w)$ and lets us concentrate on the differences of the four alternatives.

4.1. Likelihood based on folding-in

The overall goal is to calculate a likelihood $p_{\text{test}}(\mathcal{Q} \mid \theta)$ for a test collection $\mathcal{Q}$ conditioned on the model parameters θ, which is in analogy to (2):

$$p_{\text{test}}(\mathcal{Q} \mid \theta) = \prod_{q \in \mathcal{Q}} \prod_{w \in q} P(q, w \mid \theta). \tag{15}$$

In the following, we will omit θ from the notation. (15) can be re-written as

$$p_{\text{test}}(\mathcal{Q}) = \prod_{q \in \mathcal{Q}} \prod_{w \in q} P(q)P(w \mid q) \tag{16}$$

$$= \prod_{q \in \mathcal{Q}} \left[P(q)^{|q|} \prod_{w \in q} P(w \mid q) \right] \tag{17}$$

where $|q|$ is the number of words in test document q. The difficult part is coming up with a value for $P(q)$. However, the following re-formulation allows us to eliminate $P(q)$ from the calculation.

Further separating the elements in (17) yields

$$p_{\text{test}}(\mathcal{Q}) = \left[\prod_{q \in \mathcal{Q}} P(q)^{|q|} \right] \prod_{q \in \mathcal{Q}} \prod_{w \in q} P(w \mid q) \tag{18}$$

$$= \left[\prod_{q \in \mathcal{Q}} P(q)^{|q|} \right] \prod_{q \in \mathcal{Q}} \prod_{w \in q} \left[\sum_{z} P(w \mid z)P(z \mid q) \right] \tag{19}$$

The left part in square brackets is not dependent on the model parameters $P(w \mid z)$ and $P(z \mid d)$, and also not on $P(z \mid q)$. Therefore, the left part and the rest can be maximized independently in order to find the product's maximum. In case we are interested in finding the model parameters that yield the maximization it is sufficient to calculate the right (conditional) part

$$p_{\text{test}}^{\text{cond}}(\mathcal{Q}) = \prod_{q \in \mathcal{Q}} \prod_{w \in q} \left[\sum_{z} P(w \mid z)P(z \mid q) \right] \tag{20}$$

Probabilities $P(w \mid z)$ are the model parameters, $P(z \mid q)$ for test documents q can be derived by folding-in. Our proposed method of generating a likelihood substitute $p_{\text{test}}^{fi}(\mathcal{Q})$ for an unseen test collection $\mathcal{Q}$ uses (20):

$$p_{\text{test}}^{fi}(\mathcal{Q}) = \prod_{q \in \mathcal{Q}} \prod_{w \in q} \sum_{z} P(w \mid z)P_{fi}(z \mid q) \tag{21}$$

$$= \prod_{q \in \mathcal{Q}} \prod_{w \in q} P_{fi}(w \mid q) \tag{22}$$

where P_{fi} marks the probabilities obtained by folding-in, the others are obtained from the training process. p_{test}^{fi} does not represent the true likelihood value, but if our goal is likelihood maximazition we find exactly the same parameter setting as if we had maximized the true likelihood, which follows from the decomposition in (19).

4.2. *Likelihood based on marginalizing over the training documents*

Another method estimates $P(q)$ by marginalizing over all training documents d, yielding:

$$p^{d}_{\text{test}}(\mathcal{Q}) = \prod_{q\in\mathcal{Q}} \sum_{d\in\mathcal{D}} P(d) \prod_{w\in q} \sum_{z} P(w \mid z) P(z \mid d) \tag{23}$$

$$= \prod_{q\in\mathcal{Q}} \sum_{d\in\mathcal{D}} P(d) \prod_{w\in q} P(w \mid d) \tag{24}$$

This method was used in Blei et al. (2001) for the comparison of different models. Its view is that a test document q is an "average training document", i.e., it calculates the probability that a particular training document d generates all words w of the test document and then averages over all training documents. Likelihood based on marginalizing over the training documents conditions word probabilities on each training document separately and then averages. Unfortunately, this does not match very well the view of PLSA that a document is generated by its own (and possibly unique) mixture of topics z. We therefore expect this likelihood estimate in general to be too low.

4.3. *Likelihood based on a word unigram model*

The third method, mentioned in Blei et al. (2001), creates a word unigram model by marginalizing out d and z from the PLSA model and then calculates test document likelihood by multiplying the word probabilities:

$$p^{w}_{\text{test}}(\mathcal{Q}) = \prod_{q\in\mathcal{Q}} \prod_{w\in q} \sum_{d\in\mathcal{D}} P(d) \sum_{z} P(w \mid z) P(z \mid d) \tag{25}$$

$$= \prod_{q\in\mathcal{Q}} \prod_{w\in q} P(w) \tag{26}$$

As with the previous methods, this way of calculating a likelihood ignores the PLSA view of how a document is created: select a document, select a mixture of topics for the document, select the words based on the mixture of topics.

4.4. *Likelihood based on partial prediction*

The fourth method is based on a split of the test data and using one part to predict the words of the other part. Each document q in the test collection $\mathcal{Q}$ is split into two disjoint parts $\pi_1(q)$ and $\pi_2(q)$, with $\pi_1(q) \cup \pi_2(q) = q$. The first part $\pi_1(q)$ is used for folding-in, generating $P_{fi}(z \mid \pi_1(q))$. The second part $\pi_2(q)$ is used to calculate $P_{fi}(w \mid \pi_1(q))$ for all terms w in the second part $\pi_2(q)$ based on $P(z \mid \pi_1(q))$:

$$P_{fi}(w \mid \pi_1(q)) = \sum_{z} P(w \mid z) P_{fi}(z \mid \pi_1(q)) \tag{27}$$

In other words, this method does not calculate the probability of a complete new document, but instead it calculates the probability of additional words given that it already knows some of the words in the document. This yields the following likelihood based on partial prediction:

$$p_{\text{test}}^{\text{part}}(\mathcal{Q}) = \prod_{q\in\mathcal{Q}} \prod_{w\in\pi_2(q)} \sum_z P(w \mid z) P_{fi}(z \mid \pi_1(q)) \tag{28}$$

$$= \prod_{q\in\mathcal{Q}} \prod_{w\in\pi_2(q)} P_{fi}(w \mid \pi_1(q)) \tag{29}$$

This method of calculating a likelihood substitute was used in the work by Hofmann e.g. (Hofmann 2001, Hofmann and Puzicha 1998).[2] Their plots of the perplexity (which is directly related to the log likelihood) are equivalent to those that we will present in Section 5, e.g. those presented in figures 5 and 6 of Hofmann and Puzicha (1998), or those presented in figure 7 of Hofmann (2001).

4.5. *Concluding remarks on the four methods*

PLSA does not provide a way of calculating test data likelihoods. Our newly presented method, calculating the likelihood based on folding-in, circumvents the problem by calculating only a part of the likelihood, but, as we showed, this part is maximized by the same arguments that maximize the true likelihood. The other three methods try to circumvent the problem by marginalizing over one or two parameters, or by predicting one part of a test document based on another part. Likelihood based on folding-in directly follows the view of PLSA of how a document is generated ($d \to z \to w$). The other three methods deviate substantially from the PLSA generative view and we therefore expect the folding-in likelihood to be best suited.

Sections 4.1 to 4.4 presented the likelihood methods using notational simplifications. Practical applications would perform the calculations using logarithms, and they would not simply repeat multiple occurrences of the same word but directly model the frequency $f(d, w)$. We therefore repeat the four likelihood formulas, but now are using logarithms and $f(d, w)$.

Folding-in log-likelihood (cf. Eq. (21)):

$$\mathcal{L}_{\text{test}}^{fi}(\mathcal{Q}) = \sum_{q\in\mathcal{Q}} \sum_{w\in q} f(q, w) \log \sum_z P(w \mid z) P_{fi}(z \mid q) \tag{30}$$

Document-marginal log-likelihood (cf. Eq. (23)):

$$\mathcal{L}_{\text{test}}^{d}(\mathcal{Q}) = \sum_{q\in\mathcal{Q}} \log \sum_{d\in\mathcal{D}} P(d) \prod_{w\in q} \left[\sum_z P(w \mid z) P(z \mid d) \right]^{f(q,w)} \tag{31}$$

Document-marginal log-likelihood is the most difficult to calculate of the four methods. Because of the sum within the logarithm the product over all words in q needs to be actually calculated, which means one has to take care of floating point underflows for long test documents.

Word unigram log-likelihood (cf. Eq. (25)):

$$\mathcal{L}_{\text{test}}^{w}(\mathcal{Q}) = \sum_{q\in\mathcal{Q}} \sum_{w\in q} f(q,w) \log \sum_{d\in\mathcal{D}} P(d) \sum_{z} P(w \mid z) P(z \mid d) \tag{32}$$

Likelihood based on partial prediction (cf. Eq. (28)):

$$\mathcal{L}_{\text{test}}^{\text{part}}(\mathcal{Q}) = \sum_{q\in\mathcal{Q}} \sum_{w\in\pi_2(q)} f(q,w) \log \sum_{z} P(w \mid z) P_{fi}(z \mid \pi_1(q)) \tag{33}$$

5. Experiments

The experiments reported in this section aim at aligning the four likelihood measures presented in the previous section with error rates obtained in information retrieval tasks. The first task is a standard retrieval task, the second task evaluates on text segmentation.

5.1. Corpora and pre-processing

For the retrieval task, we used the MED collection, consisting of 1,033 document abstracts from the National Library of Medicine, 15 queries, and manually assigned relevance judgements for each of the 15 queries.

For the segmentation task, we used the ModApte split of the Reuters-21578 Corpus. The training set consists of 7,769 news articles with approx. 1.1 million tokens. The test set consists of 3,019 articles with approx. 400,000 tokens. The corpus was prepared according to the description in Li and Yamanishi (2000). Exactly two original documents from the test part are concatenated to form one test document, one from one of the 10 larger categories, one from a randomly chosen other category. In total, 500 test documents are generated. The task is to find the boundary of the two original documents. More details about the segmentation task can be found in Hearst (1997) and Choi et al. (2001). The error metric is described in Beeferman et al. (1999). The use of PLSA for the segmentation task is described in Brants et al. (2002).

In both cases, the corpora were stemmed, stopwords and words with frequency 1 were removed, the rest was mapped to lower case.

5.2. Likelihood calculation

We performed all experiments with each of the two corpora using numbers of classes ranging from 16 to 8,192 in powers of 2, and repeated all experiments using several different random EM initializations. Results using the different numbers of classes and initializations where very similar. We therefore present as representatives details of the experiments with

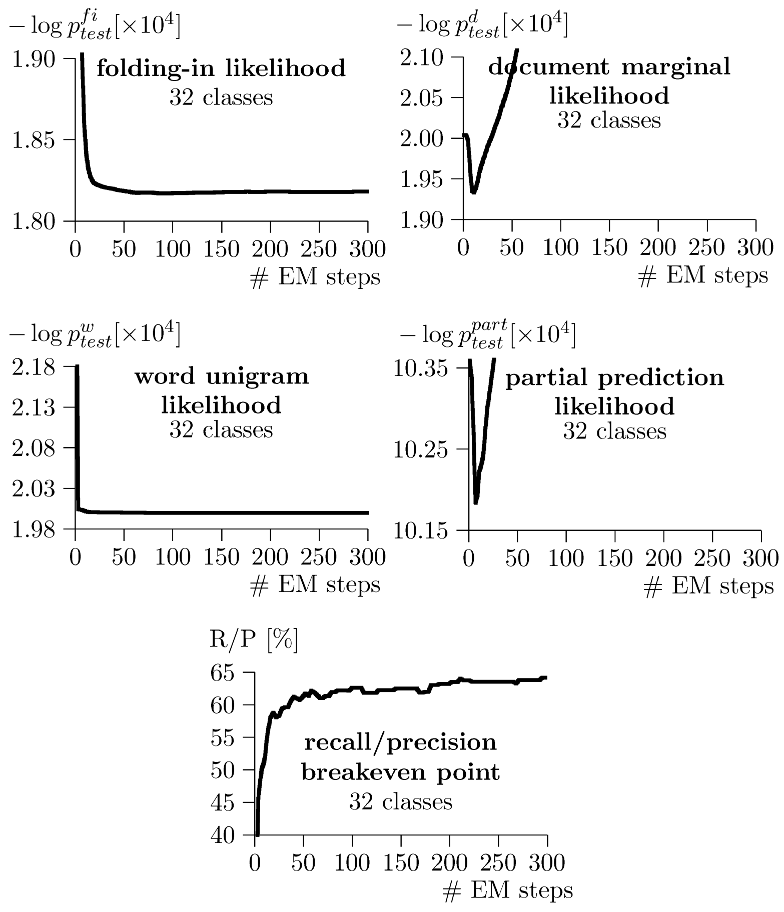

Figure 1. MED corpus likelihood and retrieval performances, 32 classes.

32 classes and 128 classes. The PLSA models were randomly initialized. After each EM step, we recorded the four likelihood measures for the test set as presented in Section 4. Figures 1 to 4 show the negative log likelihoods for steps 1 to step 300 (please note that lower values mean higher likelihood and vice versa since we are plotting the negative log likelihood).

The folding-in log-likelihood $\mathcal{L}_{\text{test}}^{fi}$ sharply decreases for the first 20 to 25 iterations in all four plots, then flattens out and changes only very slighly afterwards.

The document marginal log-likelihood $\mathcal{L}_{\text{test}}^{d}$ sharply increases for 10 to 20 initial iterations, and then rapidly decreases towards 0 (with the exception of the graph for the Reuters Corpus with 32 classes is slightly different; we further increased the number of EM steps and found that likelihood in this case will finally also decrease below random level). In the other cases, the likelihood is down to random level after 20–80 EM iterations, i.e. at the same level where it was after initialization. The steep decrease in document marginal likelihood can be explained by Eq. 23 and that with larger numbers of classes PLSA tends to move most

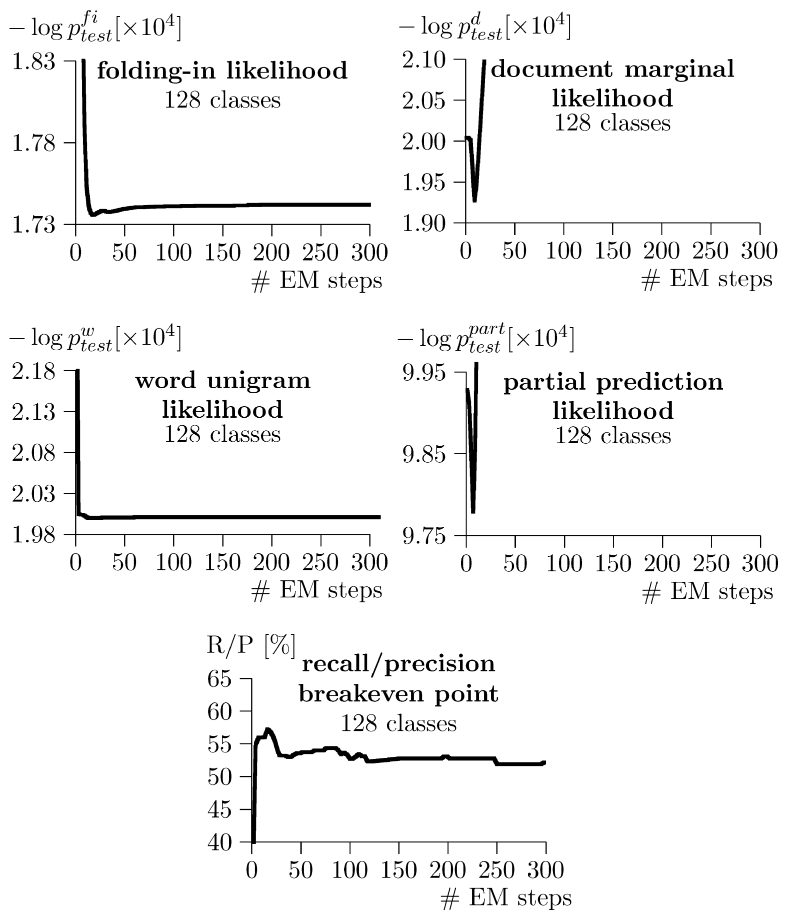

Figure 2. MED corpus likelihoods and retrieval performance, 128 classes.

of the probabilities $P(w \mid z)$ and $P(z \mid d)$ very close to zero. This will drop a large number of $P(w \mid d)$ to zero and finally assign a probability of 0 to all test documents unless they are (almost) identical to one of the training documents. The flaw is that this likelihood uses the probability that any training document has generated the vector of words of the test document $P(w \mid d)$, but what we are really interested in is the probability that the test document has generated them, $P(w \mid q)$.

Word unigram likelihood $\mathcal{L}^{w}_{\text{test}}$ increases sharply from random level for only very few EM iterations (in some cases for one iteration only), and stays almost constant thereafter. The Reuters Corpus graph shows a small decrease around 10 iterations, the MED graph levels out immediately. Using this as a model performance measure would mean that the model is overtrained already after just one step in the Reuters case, and after very few steps in the MED case. This way of calculating likelihood suffers from its very restricted underlying model, word unigrams, which is not the way PLSA assigns probabilities.

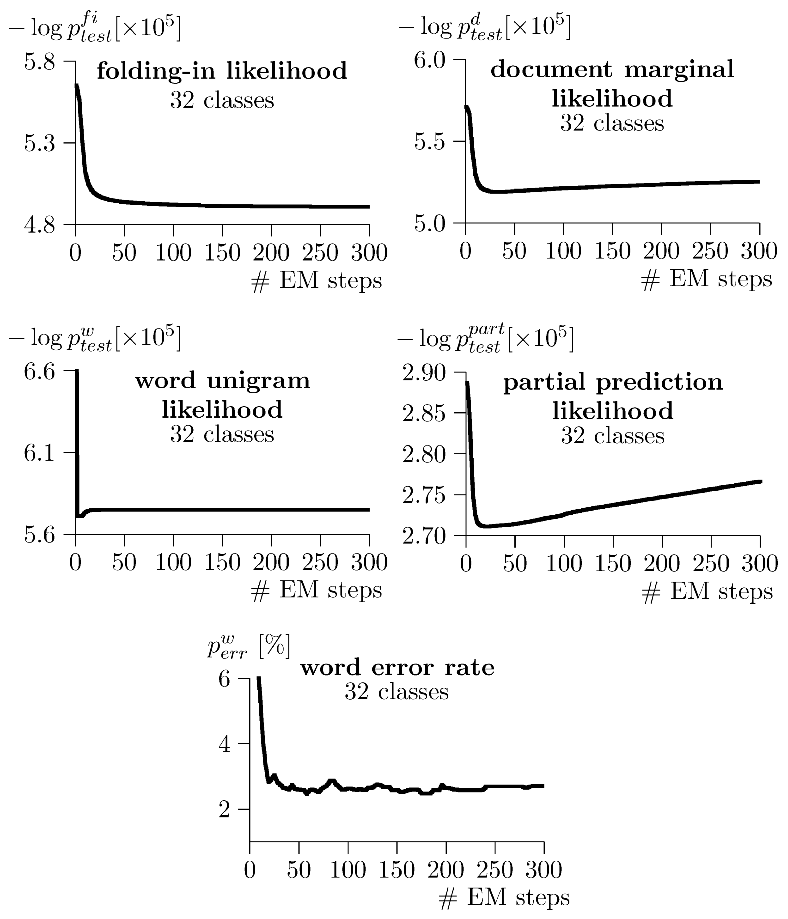

Figure 3. Reuters-21578 corpus likelihoods and word error rate, 32 classes.

Partial prediction likelihoods show similar characteristics as document marginal likelihoods, with a slightly earlier maximum (after the initial 5 to 15 iterations) and a sharper increase thereafter.

5.3. Recall/precision and error rate

We now present results on IR tasks for PLSA models that are trained with varying numbers of EM iterations. The goal is a qualitative comparison with the four methods of likelihood calculation. Therefore, we do not use additional methods like tempered EM or interpolation with other models to improve results but report plain PLSA results.

The task performed on the MED corpus is document retrieval. The test set consists of 15 queries, each consisting of one line of text. The goal is to find relevant documents in the collection. Automatically retrieved documents are compared against manual judgements

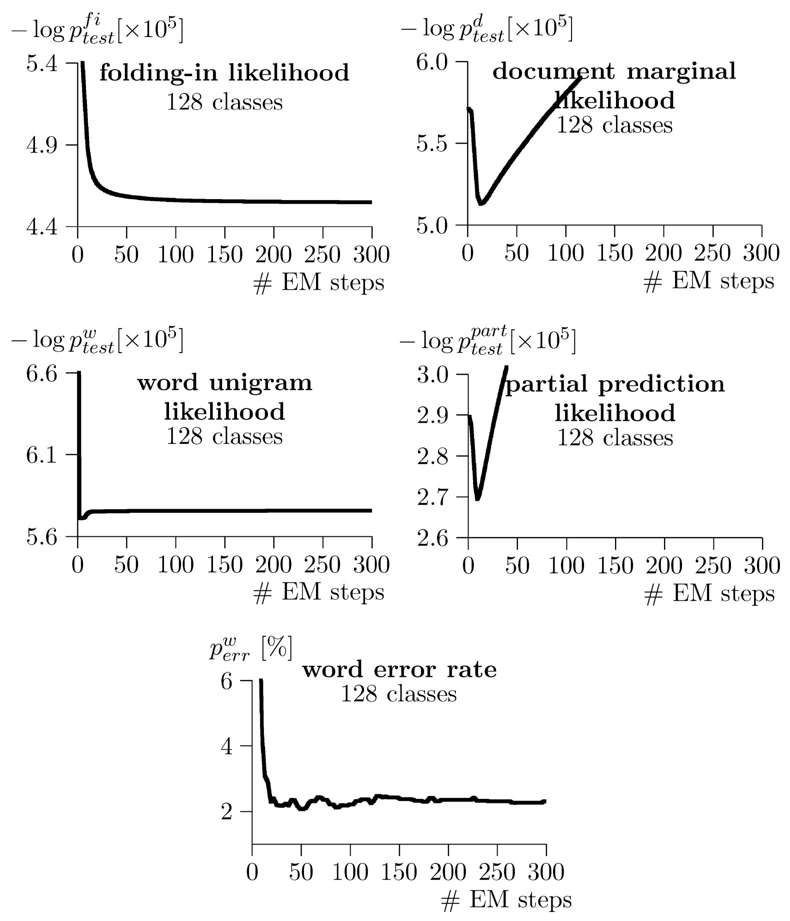

Figure 4. Reuters-21578 corpus likelihoods and word error rate, 128 classes.

on the documents. We report the recall/precision breakeven point, i.e., we take as many documents from the top of the ranked retrieved list of documents until recall and precision have the same value. The recall/precision breakeven point is plotted against the number of EM used for training the PLSA model. Results are shown at the bottom of figures 1 and 2.

We do not find any overfitting during the first 300 iterations for the model with 32 classes. For 128 classes, the model improves until around 20 iterations, then slightly drops and stays relatively constant after 50 iterations. This result is a surprise since PLSA was previously reported to require tempered EM to avoid overfitting (e.g., Hofmann (2000) and Blei et al. (2001)).

The second task is text segmentation, performed on the Reuters corpus. Results (error rates) are shown in the bottom part of figures 3 and 4. Again, we find improving results until around 20 iterations. After this point word error rate stays relative constant. There is no sign of overtraining within 300 iterations.

Table 1. Recall/precision rates for the Med Corpus and word error rates for the Reuters corpus when stopping early based on the four likelihoods and after the optimum number of iterations (the number of iterations are shown in brackets).

Corpus	#classes	$\mathcal{L}^{fi}_{\text{test}}$	$\mathcal{L}^{d}_{\text{test}}$	$\mathcal{L}^{w}_{\text{test}}$	$\mathcal{L}^{\text{part}}_{\text{test}}$	Optimum
Med	32	58.26 (17)	50.93 (9)	40.68 (3)	49.93 (7)	64.21 (298)
Med	128	57.10 (17)	54.12 (9)	49.58 (3)	55.99 (7)	57.23 (16)
Reuters	32	2.69 (33)	2.82 (19)	−(2)	5.16 (11)	2.46 (58)
Reuters	128	2.17 (31)	3.07 (13)	12.96 (4)	5.75 (9)	2.07 (49)

Qualitatively comparing the graphs of the four likelihood measures with the retrieval and segmentation results, the curves of folding-in likelihood and task performance match best in the overall shape. Document marginal likelihood and word unigram likelihood have very different curves. Folding-in likelihood even predicts the slight decrease in recall/precision around iteration 20 for the retrieval task with 128 classes (figure 2). The newly presented folding-in likelihood was derived from the PLSA likelihood definiton as it is used for the Expectation-Maximization algorithm and empirically matches recall/precision and error rates very well, while the other three are ad-hoc substitutes that do not share the generative view of PLSA. Determining the task performance at the early stopping points for the four likelihoods (i.e., the maxima or at the point where the curve flattens and its change is less than 0.1%) also shows the good fit of folding-in likelihoods (cf. Table 1). In all four experiments, folding-in likelihood comes closest to the optimum.

Our empirical results might also hint towards an explanation why Hofmann et al. found better results for tempering than for early stopping in their experiments on the MED corpus. For likelihoods based on partial prediction, maximum likelihood on the test or held-out data was achieved too early. As an example, partial-prediction log-likelihood in figure 2 reaches the maximum of −97,792 after 7 iterations. At this point, the recall-precision break-even point is at 55.99%. Recall/precision maximum is at a later point: it is 57.23% after 16 iterations. When using the partial-prediction likelihood, tempering seems necessary in order to go beyond the too early stopping point. However, folding-in likelihood does not need this heuristic addition of tempering. It achieves the maximum of −17,358 after 17 iterations and therefore comes very close to the optimal performance.

Similarly, document marginal likelihoods reach the maximum too early: in figure 2, document marginal log-likelihood reaches the maximum of −19,270 after 9 iterations. Results also favor folding-in likelihood in the other cases: partial-prediction likelihoods and document marginal likelihoods have their maximum too early. Tempering might still be useful in order to escape unfavorable local maxima, but it is not necessary to avoid overfitting in the two presented tasks.

6. Conclusions

Investigating the two alternative formulations of EM for PLSA we observed that the use of $P(z \mid d)$ instead of $P(d \mid z)$ removes the weakness of assuming zero probabilities for unseen documents. However, both formulations do not assign a particular likelhood value to test documents which makes it necessary to resort to substitutes.

We presented a new test data likelihood substitute that we derived from the PLSA likelihood definition as used for the Expectation-Maximization algorithm. This was named the folding-in likelihood.

We plotted graphs of the folding-in likelihood as well as three other likelihood substitutes: document-marginal likelihood, word unigram likelihood, and partial-prediction likelihood. Folding-in likelihood suggests that there is very little overfitting for the two corpora considered, document marginal likelihood and partial-prediction likelihood both suggest that there is heavy overfitting, word unigram likelihood seems unusable because it drops for very few (sometimes only one) iterations and then stays almost constant.

Claims in the literature that PLSA models in retrieval are very likely to overfit when using (untempered) EM cannot be confirmed; more to the contrary, PLSA models seem very robust. To our knowledge, this is the first investigation that systematically increases the number of EM steps for a PLSA model and directly reports recall/precision/error rate after each step. Previous publications used the test set perplexity instead (which is directly related to the likelihood), but, as we showed, three of the four presented likelihood measures are poor performance predictors. Earlier claims that early stopping yields poor results where based on the use of partial-prediction likelihood or document-marginal likelihood which we showed to predict the maximum too early.

Comparision to recall/precision in a retrieval task and error rate in a segmentation task showed that the best predictions are made by folding-in likelihood both from the overall match of the curves as well as the point at which the best likelihood/best accuracy is reached. This suggest that folding-in likelihood is well suited to determine the number of EM steps in an unsupervised learning setting.

Acknowledgments

I am very grateful to Francine Chen, Thomas Hofmann, Jay Ponte, Kris Popat, Stefan Riezler, and Prateek Sarkar for valuable comments and discussions on the topic of this paper.

Notes

1. Unless otherwise noted, we use the following notational conventions: training documents $d, d' \in \mathcal{D}$, test documents $q, q' \in \mathcal{Q}$, words $w, w' \in \mathcal{W}$, and classes $z, z' \in \mathcal{Z}$.
2. Personal communication with Thomas Hofmann; the referenced papers do not mention these formulas.

References

Beeferman D, Berger A and Lafferty J (1999) Statistical models for text segmentation. Machine Learning 34(1–3):177–210.

Blei D, Ng A and Jordan M (2001) Latent dirichlet allocation. In: Proceedings of NIPS-2001, Vancouver, BC, Canada, pp. 601–608.

Brants T, Chen F and Tsochantaridis I (2002) Topic-based document segmentation with probabilistic latent semantic analysis. In: International Conference on Information and Knowledge Management (CIKM), McLean, VA, pp. 211–218.

Choi FYY, Wiemer-Hastings P and Moore J (2001) Latent semantic analysis for text segmentation. In: Lee L and Harman D, Eds., In Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing, pp. 109–117.

Dempster AP, Laird NM and Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, 39(1):1–21.

Gildea D and Hofmann T (1999) Topic based language models using EM. In: Proceedings of 6th European Conference On Speech Communication and Technology (Eurospeech'99), Budapest, Hungary, pp. 2167–2170.

Hearst MA (1997) TextTiling: Segmenting text into multi-paragraph subtopic passages. Computational Linguistics, 23(1):33–64.

Hofmann T (1999) Probabilistic latent semantic analysis. In: Proceedings of Uncertainty in Artificial Intelligence, Stockholm, Sweden, pp. 289–296.

Hofmann T (2000) Probabilistic latent semantic indexing. In: Proceedings of SIGIR-99, Berkeley, CA, pp. 35–44.

Hofmann T (2001) Unsupervised learning by probabilistic latent semantic analysis. Machine Learning, 42:177–196.

Hofmann T and Puzicha J (1998) Unsupervised learning from dyadic data. Technical Report TR-98-042, ICSI, Berkeley, CA.

Li H and Yamanishi K (2000) Topic analysis using a finite mixture model. In: Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, pp. 35–44.

Rooth M, Riezler S, Prescher D, Carroll G, and Beil F (1999) Inducing a semantically annotated lexicon via EM-based clustering. In: Proceedings of ACL-99, College Park, MD, USA, pp. 104–111.

Saul L and Pereira F (1997) Aggregate and mixed-order markov models for statistical language processing. In: Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP), San Francisco, CA, Association for Computational Linguistics, pp. 81–89.