Deterministic Latent Variable Models and their Pitfalls

Max Welling[*] Chaitanya Chemudugunta Nathan Sutter
Bren School of Information and Computer Science
University of California, Irvine
{welling,chandra,nsutter}@ics.uci.edu

Abstract

We derive a number of well known deterministic latent variable models such as PCA, ICA, EPCA, NMF and PLSA as variational EM approximations with point posteriors. We show that the often practiced heuristic of "folding-in" can lead to overly optimistic estimates of the test-set log-likelihood and we verify this result experimentally. We trace this problem back to an infinitely negative entropy term that is ignored in the variational approximation.

1 Introduction

Deterministic latent variable models (DLVM) such as K-means, PCA, NMF [10], PLSA [1] and EPCA [2] are popular data-analysis tools in the machine learning and data-mining communities. The latent variables in these models become deterministic because they are set to their MAP (maximum a posteriori) values[1]. In the context of PLSA, it has been observed in the literature (see e.g. [5]) that this seriously complicates its interpretation as a valid probabilistic model over the space of all allowable input configurations. In fact, one could argue that the resulting model assigns zero probability to all input configurations that are not in the training set. To enable PLSA to assign non-zero probability to test-set data configurations, researchers proposed the heuristic of "folding-in"[1]. The folding-in procedure first assigns the latent variables of the *test-data* to their MAP values before computing the test-set perplexity. In [5], it is mentioned that this procedure would give an unfair advantage to PLSA because "parameters are fit on the test-data". One of the conclusions of our analysis is that this characterization does not clarify the real problem.

We approach this issue through the variational-EM (VEM) framework [7]. This framework has the distinct advantage of shifting the approximation from the model to the learning and testing of *algorithms*. Moreover, PLSA can now be seen to fall in line with a long tradition of similar approximations underlying algorithms such as K-means, PCA, NMF, and sparse ICA. In the VEM framework, we bound the training (or testing) log-likelihood by approximating the posterior distribution of the latent variables by a parameterized family of tractable distributions over the same input space. We show that DLVMs with *discrete* latent variables can be obtained by approximating the true posterior distribution by a simplistic variational substitute, namely a delta-peak at its MAP value. In this case, learning DLVMs becomes an instance of variational EM. However, this approach also reveals the pitfall for deterministic *continuous* latent variables: in the continuous case the entropy term of the variational distribution becomes $-\infty$ and ignoring it violates the bounding property of the variational objective. Using this perspective, we show that the standard learning algorithms for PLSA and EPCA optimize an objective function that is numerically unrelated to the log-likelihood or a bound thereof.

Perhaps the most important conclusion is that the folding-in heuristic for continuous DLVMs, as it is routinely used in the research community, can lead to hugely optimistic estimates of test-set perplexity. We argue that this effect worsens in high dimensional spaces and in cases where the posterior is highly concentrated. This conclusion, however, is *not* valid for discrete latent variables (where the entropy term is actually 0) or for mean field approaches (where the entropy term is finite and taken into account), in fact in those cases we can show that folding-in will lead to pessimistic estimates of test-set perplexity (i.e. an upper bound). Our explanation is, therefore, intrinsically different to the one offered in [5] because one also seemingly fits (variational) parameters on test-data for the cases above but here they only pessimistically bias the result.

We thoroughly investigated these issues empirically by comparing different models (different versions of PLSA, EPCA and LDA) with different objective measures (perplexity using folding-in based on all and half of the words in the test document, which we refer to as full folding-in and half folding-in respectively) on various datasets. As predicted, full folding-in gives much lower (that is, better) perplexity results than half folding-in. Moreover, the perplexity results with full folding-in keep improving as we increase the num-

[*] At the time of submission M. Welling was on sabbatical at the Radboud University Nijmegen, Netherlands, Dept. of Biophysics

[1] For PLSA this fact was recently shown in [3, 4].

ber of topics, never showing any sign of overfitting (unlike half folding-in where we can not “cheat” in any way).

2 Variational EM with Point Estimates

Let $\mathbf{x}$ be a collection of observable (visible) random variables and $\mathbf{h}$ be a collection of unobservable (hidden) random variables[2]. The joint probability distribution of a two-layer directed model is defined as follows,

$$p_\theta(\mathbf{x}, \mathbf{h}) = p_\nu(\mathbf{x}|\mathbf{h})\, p_\gamma(\mathbf{h}). \tag{2.1}$$

We have introduced parameters $\theta = \{\nu, \gamma\}$ which could be vector valued. Estimation of these parameters from the data can be done conveniently in the expectation maximization (EM) framework. It can be shown that the following objective is a lower bound on the log-likelihood [6],

$$\begin{aligned}\mathcal{B}(\theta_t|\theta_{t-1}, X) &= \mathcal{Q}(\theta_t|\theta_{t-1}) + \mathcal{H}(\theta_{t-1}) = \\ &\sum_n \sum_{\mathbf{h}_n} p_{\theta_{t-1}}(\mathbf{h}_n|\mathbf{x}_n) \log\left[p_{\nu_t}(\mathbf{x}_n|\mathbf{h}_n) p_{\gamma_t}(\mathbf{h}_n)\right] \\ &- \sum_n \sum_{\mathbf{h}_n} p_{\theta_{t-1}}(\mathbf{h}_n|\mathbf{x}_n) \log\left[p_{\theta_{t-1}}(\mathbf{h}_n|\mathbf{x}_n)\right]\end{aligned} \tag{2.2}$$

where X indicates the data-matrix. This bound is iteratively maximized over the parameters $\theta_t = \{\nu_t, \gamma_t\}$, keeping θ_{t-1} fixed. Since the last term, which is the entropy of the posterior at time $t-1$, does not depend on θ_t, it can be ignored in the optimization process. It is not hard to show that the following identity holds,

$$\mathcal{L}(\theta_t) = \mathcal{B}(\theta_t|\theta_{t-1}) + KL[p_{\theta_{t-1}}(\mathbf{h}_n|\mathbf{x}_n)||p_{\theta_t}(\mathbf{h}_n|\mathbf{x}_n)] \tag{2.3}$$

where KL is the Kullback-Leibler divergence. This equation confirms that the log-likelihood is larger than or equal to the bound, $\mathcal{L} \geq \mathcal{B}$ and that the bound saturates when $p_{\theta_{t-1}}(\mathbf{h}_n|\mathbf{x}_n) = p_{\theta_t}(\mathbf{h}_n|\mathbf{x}_n)$, i.e. $\mathcal{L}(\theta_t) = \mathcal{B}(\theta_t|\theta_t)$. The above analysis is easily repeated in a slightly more general setting where the posterior at time $t-1$ is replaced with an arbitrary variational distribution over the hidden variables [7],

$$p_\theta(\mathbf{h}_n|\mathbf{x}_n) \rightarrow q_\alpha(\mathbf{h}_n|\mathbf{x}_n) \tag{2.4}$$

The parameters α will be optimized in the variational estimation (VE) step in such a way that the bound is made as tight as possible. Note that this now also involves the entropy term in Equation 2.2 because it explicitly depends on q_α. Often, the family of distributions q_α does not contain the actual posterior distribution, in which case the bound is never tight. Given the updated q_α distributions, we can do a variational maximization (VM) step where $\mathcal{Q}(\theta_t|\alpha_{t-1})$ is maximized over θ_t. In the following we will make a very special choice of this variational distribution, namely,

$$q_{\hat{\mathbf{h}}_n}(\mathbf{h}_n|\mathbf{x}_n) = \delta(\mathbf{h}_n - \hat{\mathbf{h}}_n) = \prod_j \delta(h_{jn} - \hat{h}_{jn}) \tag{2.5}$$

where the variational parameters are given by $\{\hat{h}_{in}\}$. Note that there is a separate parameter for every hidden variable *and for every data-case*.

If $\mathbf{h}$ takes discrete values then the entropy of the distribution in Equation 2.5 is zero. This, however, is not true for continuous variables. In cases where the domain of $\mathbf{h}$ is unbounded we can, for instance, define the delta function as the limit of an isotropic Gaussian distribution with decreasing variance. Since the entropy of a Gaussian distribution is given by $D \log(\sigma\sqrt{2\pi e})$, the entropy converges to $-\infty$ as $\sigma \rightarrow 0$. Similarly, if $\mathbf{h}$ is continuous but normalized, $\sum_j h_j = 1$, we can model a delta-function as the limit of a Dirichlet distribution $\mathcal{D}(\varepsilon\boldsymbol{\alpha})$ when $\varepsilon \rightarrow \infty$. Using Sterling’s approximation one can then determine that the entropy converges to $-\infty$ as $\frac{1-J}{2} \log \varepsilon$ when $\varepsilon \rightarrow \infty$ where J is the number of latent variables (i.e. the dimension of $\mathbf{h}$). From this we conclude that for continuous latent variables the MAP approximation leads to an infinitely negative entropy contribution to the variational objective. Ignoring it renders the resulting objective numerically unrelated to the log-likelihood. As we will show in the next sections, well known learning algorithms for EPCA [2] and PLSA [1] can precisely be interpreted as maximizing the variational objective with variational point posteriors and *ignoring the infinitely negative entropy contribution*, i.e. maximizing

$$\mathcal{O}(\theta, \hat{\mathbf{h}}) = \sum_n \log\left[p_\nu(\mathbf{x}_n|\hat{\mathbf{h}}_n)\right] + \sum_n \log\left[p_\gamma(\hat{\mathbf{h}}_n)\right] \tag{2.6}$$

The VEM algorithm then consists of a VE-step where we maximize $\mathcal{O}$ over $\{\hat{\mathbf{h}}_n\}$ and a VM-step where we maximize it over $\theta = \{\nu, \gamma\}$.

In the following we will provide some examples of algorithms that fall under this umbrella.

3 Examples of Deterministic Latent Variable Models

To appreciate the unifying principle of VEM, we derive the following well known algorithms in the machine learning literature as VEM approximations of other algorithms: K-means, Pricipal Component Analysis (PCA), Independent Component Analysis (ICA), Non-negative Maxtrix Factorization (NMF), Probabilistic Latent Semantic Analysis (PLSA) and Exponential family PCA (EPCA).

3.1 K-Means We start with the mixture of Gaussians (MoG) model where the discrete hidden variables take values in $1, \ldots, K$ with K being the total number of mixture

[2] In the following we will use the words “latent”, “hidden” and “unobservable” interchangeably.

components,

$$p_{\text{mog}}(\mathbf{x}, h = k) = \mathcal{N}_{\boldsymbol{\mu}_k, \Sigma_k}(\mathbf{x}|h = k) p_{\pi_k}(h = k) \tag{3.7}$$

If we use point posteriors of the form shown in Equation 2.5 then, since the latent variables are discrete, the entropy contribution of the variational posterior is zero, $S = 0$. Hence, we find the following VEM *bound* on the log-likelihood,

$$\begin{aligned}\mathcal{B} = \sum_n \sum_k \delta_{\hat{h}_n,k} \Big[&-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) \\ &-\frac{1}{2} \log\det\Sigma_k + \log(\pi_k) \Big]\end{aligned} \tag{3.8}$$

Alternating optimization over $\{\hat{\mathbf{h}}_n\}$ and $\{\boldsymbol{\mu}_k, \Sigma_k, \pi_k\}$ represents a generalized K-means algorithm where we have the opportunity to fit full covariance matrices and mixture weights for each cluster. If we choose $\pi_k = 1/K$ and $\Sigma_k = \sigma I$ with some fixed value for σ, then the non-constant terms in the bound are proportional to,

$$\mathcal{B}' = -\frac{1}{2} \sum_n \sum_k \delta_{\hat{h}_n,k} \left[(\mathbf{x}_n - \boldsymbol{\mu}_k)^2 \right] \tag{3.9}$$

which is, of course, the (negative) K-means objective. VE and VM steps then indeed correspond to K-means. This shows that K-means is a VEM approximation of MoG.

A related example is given by "Viterbi-HMM". In this case, one alternates Viterbi decoding in the E-step with the usual parameter updating in the M-step. This algorithm also maximizes a proper lower bound on the log-likelihood.

3.2 PCA and ICA We start with the Factor analysis model with normal variables both in the hidden and in the visible layer. The visible variables are noisy versions of linear combinations of fewer hidden factors, $\mathbf{x} = W\mathbf{h} + \varepsilon$ with ε an axis aligned Gaussian noise variable with diagonal covariance Σ. $\mathbf{h}$ is also distributed according to a standard normal distribution,

$$p(\mathbf{x}, \mathbf{h}) = \mathcal{G}_{W,\Sigma}(\mathbf{x}|\mathbf{h}) \prod_j \mathcal{G}(h_j) \tag{3.10}$$

In this case the latent variables are continuous, hence variational point posteriors lead to an infinitely negative entropy contribution. Simply ignoring this term leads to the following objective,

$$\begin{aligned}\mathcal{O} = -\frac{1}{2} \sum_n \Big[&(\mathbf{x}_n - W\hat{\mathbf{h}}_n)^T \Sigma^{-1} (\mathbf{x}_n - W\hat{\mathbf{h}}_n) \\ &+ \log\det\Sigma + ||\hat{\mathbf{h}}_n||^2_{L_2} \Big]\end{aligned} \tag{3.11}$$

Using $\Sigma = \lambda I$, defining $\mathcal{O}' = \lambda\mathcal{O}$ and taking the limit $\lambda \to 0$, we find,

$$\mathcal{O}' = -\frac{1}{2} \sum_n \left[(\mathbf{x}_n - W\hat{\mathbf{h}}_n)^T (\mathbf{x}_n - W\hat{\mathbf{h}}_n) \right] \tag{3.12}$$

as our objective. The VE-step is obtained by maximizing this w.r.t. the variational variables $H = \{\hat{h}_{jn}\}$ while the VM-step optimizes this bound w.r.t. W,

$$W \leftarrow (XH^T)(HH^T)^{-1} \tag{3.13}$$

$$H \leftarrow (WW^T)^{-1}(W^T X) \tag{3.14}$$

where $X = \{x_{in}\}$. This is precisely the SPCA algorithm derived in [12].

Above we have taken the limit $\lambda \to 0$ which has the effect of nullifying the prior. For independent component analysis, the prior is essential. In this case we should use a non-Gaussian, factorized prior for $\mathbf{h}$ with high kurtosis, $p(\mathbf{h}) = \prod_j p_j(h_j)$. If we choose a Laplace prior we should simply replace the L_2 norm in Equation 3.11 with an L_1 norm. In trying to maximize this bound one quickly finds out that $H = 0$ is the optimal solution due to the fact that a transformation of the form $W \to WT$ and $H \to T^{-1}H$ leaves the first term invariant but can be used to set the prior term on $\hat{\mathbf{h}}$ to zero. This phenomenon is a direct consequence of ignoring the entropy. The entropy term, if included, would have induced a counteracting "force" that would have prevented the collapse of $\hat{\mathbf{h}}$.

In this case we can however salvage the algorithm by requiring that the columns of W be normalized. This was indeed the approach taken in [11]. Their approximation translates to an even simpler VEM approximation, namely $q(\mathbf{h}) = \delta(\mathbf{h} - \hat{\mathbf{h}})$ independent of n. For ICA, the VE and VM steps can not be solved analytically. However, one can alternate gradient steps for W and H until convergence.

3.3 Non-negative Matrix Factorization Non-negative matrix factorization (NMF) [10] is a method where a positive matrix is factorized into two positive lower rank matrices. To derive it from VEM, we use a conditional Poisson distribution for the observed variables and write the Poisson rate as a linear combination of hidden factors,

$$p(\mathbf{x}|\mathbf{h}) = \prod_i \frac{(\sum_j W_{ij} h_j)^{x_i}}{x_i!} \exp[-\sum_j W_{ij} h_j] \tag{3.15}$$

Applying the point-VEM approximation and again ignoring the infinitely negative entropy we derive the objective,

(3.16)

$$\mathcal{O} = \sum_{i,n} \left(x_{in} \log[\sum_j W_{ij} \hat{h}_{jn}] - \sum_j W_{ij} \hat{h}_{jn} - \log(x_{in}!) \right)$$

Since the Poisson rate is positive, the same must be true for WH which is achieved by separately constraining both to be positive. We now bound the non-constant part of the objective $\mathcal{B} \leq \mathcal{O} + \log(x_i!)$, with

$$\mathcal{B} = \sum_{i,n} \left(x_{in} \left(\sum_j Q_{in}(j) \log(W_{ij}\hat{h}_{jn}) + S(Q_{in}) \right) - \sum_j W_{ij}\hat{h}_{jn} \right)$$

where

$$Q_{in}(j) = \frac{W_{ij}\hat{h}_{jn}}{\sum_{j'} W_{ij'}\hat{h}_{j'n}} \tag{3.17}$$

and $S(Q)$ is its entropy. In [10], the authors add the constraint $\sum_i W_{ij} = 1$ to reduce the degeneracy associated with $W \to WT$ and $H \to T^{-1}H$. Adding the appropriate Lagrange multipliers we find the following EM updates[3], which are equal to the NMF update rules of [10],

$$W_{ij} \quad \leftarrow \quad \frac{1}{\gamma_j} W_{ij} \sum_n \frac{X_{in}H_{jn}}{[WH]_{in}} \tag{3.18}$$

$$H_{jn} \quad \leftarrow \quad H_{jn} \sum_i \frac{X_{in}W_{ij}}{[WH]_{in}} \tag{3.19}$$

where γ_j is a constant to normalize W_{ij} over i.

The authors actually relax the constraint on the discreteness of $\mathbf{x}$ by noting that the objective $\mathcal{B}$ still makes sense in this more general setting.

3.4 Probabilistic Latent Semantic Analysis

The insight that PLSA can be viewed as a MAP estimate of LDA was published in [4], while very similar remarks on the relation between LDA and PLSA were also noted in [3]. Here, we will re-derive those results in the variational EM framework and show the striking similarity to NMF.

Let us consider an LDA model ([5]). The conditional probability in this case is a multinomial distribution while the prior on the factors is given by a Dirichlet distribution,

$$p(\mathbf{x}, \mathbf{h}) = \left[\prod_i \left(\sum_j W_{ij} h_j \right)^{x_i} \right] \mathcal{D}_{\mathbf{h}}[\alpha] \tag{3.20}$$

where W_{ij} is the probability of word i for topic j. x_i is the count of the number of times word i is sampled and h_j is the prior probability that topic j is used. To make a connection to PLSA we use the variational approximation with point posteriors and choose a constant prior, $\boldsymbol{\alpha} = 1$. Note that since we have continuous latent variables we will ignore an infinitely negative entropy contribution in the variational objective to arrive at the following objective,

$$\mathcal{O} = \sum_{i,n} x_{in} \log \left(\sum_j W_{ij}\hat{h}_{jn} \right) \tag{3.21}$$

plus two Lagrange multiplier terms to enforce $\sum_i W_{ij} = 1$ and $\sum_j \hat{h}_{jn} = 1$.

Analogous to the derivation for NMF, we can now bound $\mathcal{O}$ with $\mathcal{B} \leq \mathcal{O}$, and

$$\mathcal{B} = \sum_{i,n} x_{in} \left(\sum_j Q_{in}(j) \log(W_{ij}\hat{h}_{jn}) + S(Q_{in}) \right) \tag{3.22}$$

where $S(Q)$ is the entropy of Q. Again, we should also include the Lagrange multiplier terms which we left out for convenience. This bound is valid for any variational distribution Q, but the expression that maximizes (in fact saturates) the bound is given by,

$$Q_{in}(j) = \frac{W_{ij}\hat{h}_{jn}}{\sum_{j'} W_{ij'}\hat{h}_{j'n}} \tag{3.23}$$

This can be viewed as the E-step of an EM algorithm. The M-step is then given by fixing Q and maximizing over $W, \hat{h}$. Combining E and M steps, we find the following update equations that are guaranteed to improve $\mathcal{O}$,

$$\begin{aligned} W_{ij} &\leftarrow \frac{1}{\gamma_j} W_{ij} \sum_n \frac{X_{in}H_{jn}}{[WH]_{in}} \\ H_{jn} &\leftarrow \frac{1}{\lambda_n} H_{jn} \sum_i \frac{X_{in}W_{ij}}{[WH]_{in}} \end{aligned} \tag{3.24}$$

which are identical to NMF except for an extra normalization of H. Retaining Dirichlet priors in the derivation for both W and H would translate into constant off-sets in the update equation 3.24.

To connect this to the PLSA updates derived in [1] we note that this model contains three factors: $p(w|z)$, $p(d|z)$ and $p(z)$. We can absorb $p(z)$ into $p(d|z)$ and re-parameterize as $p(z|d) \propto p(d|z)p(z)$. The factor $p(d)$ that remains in this parametrization is chosen constant (each document has the same weight). If we make the identification $W_{ij} \leftrightarrow p(w|z)$ and $H_{jn} \leftrightarrow p(z|d)$, then the model and the learning updates become identical to those in [1]. In figures 1,2 and 3 we have verified that the standard PLSA implementation of [1] indeed gives almost identical results to the updates derived above.

[3] Note that this EM algorithm does not refer to the same variational EM argument to derive the objective $\mathcal{O}$. So, one could say that this an EM algorithm within a VEM algorithm.

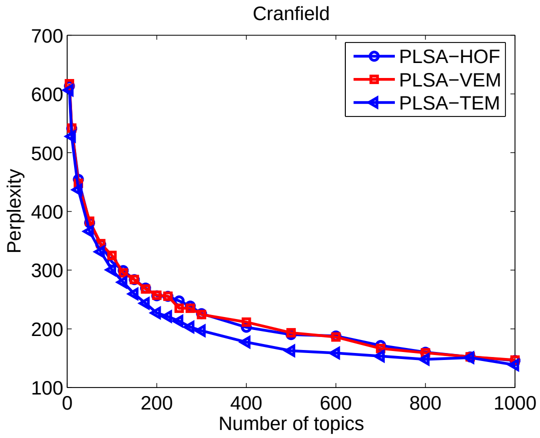

Figure 1: Comparison of test set perplexity of PLSA-HOF, PLSA-VEM and PLSA-TEM on the Cranfield dataset

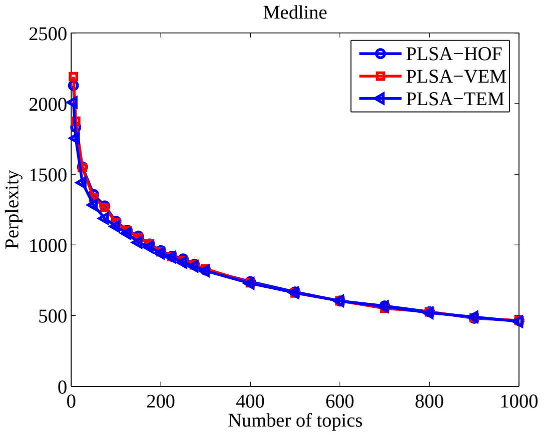

Figure 2: Comparison of test set perplexity of PLSA-HOF, PLSA-VEM and PLSA-TEM on the Medline dataset

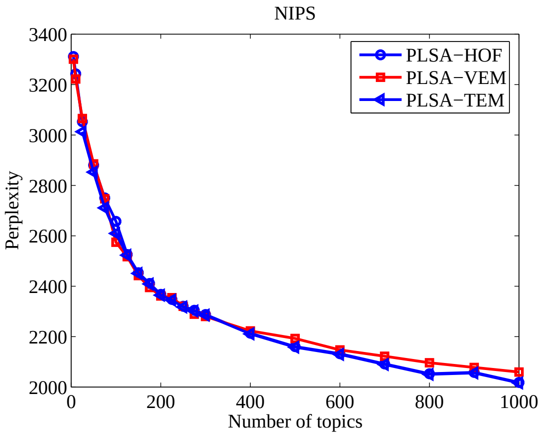

Figure 3: Comparison of test set perplexity of PLSA-HOF, PLSA-VEM and PLSA-TEM on the NIPS dataset

3.5 Exponential Family PCA (EPCA) A general class of PCA algorithms is given by exponential family PCA (EPCA) [2]. To derive this class of algorithms from VEM, we start with a conditional distribution $p(\mathbf{x}|\mathbf{h})$ parameterized in the canonical representation, i.e. $\log p(\mathbf{x}|\mathbf{h}) = \mathbf{x}^T\boldsymbol{\theta} + c$. The canonical parameters $\boldsymbol{\theta}$ are in turn written as linear combinations of hidden variables: $\theta_i = \sum_j W_{ij} h_j$. Ignoring the prior term we have the following model,

$$p(\mathbf{x},\mathbf{h}) = \exp[\sum_{i,j} x_i W_{ij} h_j - F(W\mathbf{h})] \tag{3.25}$$

where F is the log-partition function for the exponential family distribution under consideration. By applying the point-VEM approximation (ignoring the infinitely negative entropy) we derive the following objective,

$$\mathcal{O} = \sum_{i,n}\left(\sum_j x_{in} W_{ij} \hat{h}_{jn} - F_n(WH)\right) \tag{3.26}$$

which should be optimized over $W = \{W_{ij}\}$ and $H = \{\hat{h}_{jn}\}$. The two-phase algorithm proposed in [2] precisely corresponds to the VE-step and VM-steps. Here, we will simply follow gradients alternatingly for W and H. The gradients of $\mathcal{O}$ are given by,

$$\begin{aligned} \frac{\partial\mathcal{O}}{\partial W} &= XH^T - F'(WH)H^T \\ \frac{\partial\mathcal{O}}{\partial H} &= W^T X - W^T F'(WH) \end{aligned} \tag{3.27}$$

where F' is the gradient of F. We note that the VEM algorithm for PCA in section 3.2 is a special case of EPCA where F is quadratic and hence F' is linear.

In the experiments we will compare EPCA and PLSA on text. For this purpose $\mathbf{x}$ represents a word from a finite vocabulary and is given as an indicator variable consisting of zeros and a single 1 at the vocabulary index for that word. In this case we find that,

$$F_n = \log \sum_i \exp\left[\sum_j W_{ij}\hat{h}_{jn}\right] \tag{3.28}$$

Another example for which the above derivation could be helpful is in formulating a simple learning algorithm for the "multiple multiplicative factor model" proposed in [13], but we do not pursue this any further here.

4 Pitfalls For Deterministic Latent Variable Models

In the previous section, we gave examples of Deterministic Latent Variable models that are derived as VEM approximations of other models. We will now discuss two perspectives that will prove the central thesis of this paper, namely that estimates of the test-data log-likelihood (or equivalently test-data perplexity) based on "folding-in" can be hugely optimistic for continuous latent variables, especially in high dimensions.

4.1 Estimating Test-Set Log-Likelihood Assume we wish to approximate the log-likelihood of the original probabilistic model $p(\mathbf{x}) = \sum_{\mathbf{h}} p(\mathbf{x}, \mathbf{h})$. Analogous to the variational bound on training data we can also lower bound the log-likelihood of a *test* data case $\mathbf{y}$ using,

(4.29)
$$\mathcal{B}_1 = \max_q \left[\sum_{\mathbf{h}} q(\mathbf{h}) \log p(\mathbf{y}, \mathbf{h}) + S(q) \right] \leq \log p(\mathbf{y})$$

where $S(q)$ is the entropy of $q(\mathbf{h})$. Note that we compute q by maximizing $\mathcal{B}_1$ for the test-data case under consideration which is what we will also call "folding-in".

This equation tells us that mean-field/variational bounds based on folding-in will provide a *pessimistic* estimate of the true log-likelihood of the model p. Moreover, the closer q is in KL-divergence to the true posterior, the smaller the error. Note that we separate model evaluation and model learning, so p could have been obtained using any method other than VEM, although this approximation is particularly convenient in the context of VEM.

If we restrict the family of variational distributions q to point-posteriors , i.e. to be of the form shown in Equation 2.5, then in the case of discrete latent variables the entropy is zero. Therefore, the following expression also constitutes a proper lower bound in this case,

$$\mathcal{B}_2 = \max_{\hat{\mathbf{h}}} \left[\log p(\mathbf{y}, \hat{\mathbf{h}}) \right] \leq \log p(\mathbf{y}) \tag{4.30}$$

where $\hat{\mathbf{h}}$ is now obtained by maximizing $\log p(\mathbf{y}, \hat{\mathbf{h}})$ over $\hat{\mathbf{h}}$. We will also call this a form of proper folding-in in the sense that although we "fit" something on test-data it still provides a *pessimistic* estimate of the test-log-likelihood.

Finally, let us look at what happens when $\mathbf{h}$ is continuous. As argued in the previous section, for this case the entropy is no longer positive, and in fact converges to $S \rightarrow -\infty$. Hence, ignoring it will no longer guarantee the bound. In fact, it is not hard to show that the objective in Equation 4.30 can be much larger than the test data log likelihood and hence can be *optimistic*. To see that let us rewrite Equation 4.30 as,

$$\mathcal{B}_2 = \log p(\mathbf{y}) + \left[\max_{\hat{\mathbf{h}}} \log p(\hat{\mathbf{h}}|\mathbf{y}) \right] \tag{4.31}$$

For discrete latent variables we have $p(\hat{\mathbf{h}}|\mathbf{y}) \leq 1$ and hence the logarithm is negative which confirms the claim that $\mathcal{B}_2$ is a lower bound of $\log p(\mathbf{y})$ in this case. However, when $p(\hat{\mathbf{h}}|\mathbf{y})$ is a *density* it is highly likely that $\max_{\hat{\mathbf{h}}} p(\hat{\mathbf{h}}|\mathbf{y}) \gg 1$. For high dimensional spaces these values can be extremely large, particularly for cases where the posterior is concentrated. In such cases we can thus conclude that $\mathcal{B}_2$ can in fact be much larger than the true log-likelihood resulting in unfounded optimism.

These observations are in particular true for the popular PLSA model [1] for which $\hat{\mathbf{h}} = p(z|d)$. This fact is not much appreciated by the research community and has led some researchers to compute test-set perplexity in the wrong way probably leading to overly optimistic estimates. Other than the original PLSA paper [1] where folding-in was first introduced, a number of publications can be found in the literature that use folding-in to compute test-set perplexity (e.g. some recent publications like [15] and [16] follow the folding-in procedure to compute test-set perplexity).

It should be noted that [8] discusses some inconsistencies in the PLSA framework (namely, that PLSA as described in [1] assumes zero probability for test documents) but the solution offered is again based on folding-in and it therefore suffers from the same problem as described above.

4.2 An Energy-Based Model The approach until now has been to assume that we are interested in the model $p(\mathbf{x}, \mathbf{h})$, but that we need a variational approximation of the posterior $p(\mathbf{h}|\mathbf{x})$ to make learning and prediction tractable. In this section, we will take a slightly different perspective. Instead of changing the algorithm, we will change the model so that it matches well with the folding-in procedure.

We first notice that folding-in can be defined as the following optimization for a data-case $\mathbf{y}$

$$f(\mathbf{y}) = \max_{\mathbf{h}} p(\mathbf{y}, \mathbf{h}) \tag{4.32}$$

Note that this is done *for every data-case separately*. During learning we wish to adjust parameters such that this goodness function $f(\mathbf{x})$ is large for every data-case. However, as we will show in an example in the next section, this can be achieved without learning anything. We need to first define a proper probability distribution over the entire input space and maximize its log-likelihood. This is achieved by normalizing the expression $f(\mathbf{x})$ over its input domain,

$$g(\mathbf{x}) = \frac{\max_{\mathbf{h}} p(\mathbf{x}, \mathbf{h})}{\sum_{\mathbf{x}'} \max_{\mathbf{h}} p(\mathbf{x}', \mathbf{h})} \tag{4.33}$$

If we rewrite this in the form $g = e^{-E}/Z$, the negative energy is exactly given by the expression shown in Equation 4.30. This type of a model is called an "energy-based model" and has been studied in [9].

The normalization constant $Z = \sum_{\mathbf{x}} \max_{\mathbf{h}} p(\mathbf{x}, \mathbf{h})$ is unfortunately intractable for all but the simplest of models. For instance, for documents, we would have to sum over all possible valid documents. We also note that the "folding-in" heuristic is precisely equivalent to *ignoring* the normalization constant in the log-likelihood for the model $g(\mathbf{x})$. We can now ask the central question: What will the effect of ignoring the normalization term be on the test-set log likelihood of the modified model $g(\mathbf{x})$?

To study that, first rewrite,

$$\log Z = \log \sum_{\mathbf{x}} \left[\max_{\mathbf{h}} p(\mathbf{h}|\mathbf{x})\right] p(\mathbf{x}) \tag{4.34}$$

For discrete latent variables, we can show that $\alpha(\mathbf{x}) \doteq \max_{\mathbf{h}} p(\mathbf{h}|\mathbf{x}) \leq 1$ which implies that $\sum_{\mathbf{x}} \alpha(\mathbf{x}) p(\mathbf{x}) \leq 1$ (since $\sum_{\mathbf{x}} p(\mathbf{x}) = 1$) and therefore we have $\log Z \leq 0$. In other words, by reporting Equation 4.30 we report a lower bound on the test log-likelihood of the model g. However, for continuous latent variables, $\alpha(\mathbf{x})$ can take arbitrary large values resulting in test log-likelihood estimates which may be highly optimistic. Again, this behavior is much more pronounced for concentrated distributions in high dimensional spaces. Therefore, even in this interpretation the conclusion remains unchanged: for continuous latent variables, folding-in to compute an estimate of test set log-likelihood may result in values that are highly optimistic.

4.3 An Example Let us look at a very simple PLSA example given by,

$$p(\mathbf{x}, \mathbf{h}) = p(\mathbf{x}|\mathbf{h})\ p(\mathbf{h}) = \left(\sum_j p_j(\mathbf{x})\ h_j\right)\ \mathcal{D}_{\mathbf{h}}(1) \tag{4.35}$$

where $\mathcal{D}_{\mathbf{h}}(1)$ is a Dirichlet distribution with prior strengths equal to 1. In this example the conditional distribution $p(\mathbf{x}|\mathbf{h})$ is a mixture model with *data-dependent* mixture weights h_j (similar to $p(z|d)$ in PLSA) and where $\sum_j h_j = 1$. Also, $p_j(x)$ is some probability distribution for component j (similar to $p(w|z)$ for PLSA). In this example we will assume that $\mathbf{x} = 1, .., V$ and that there are exactly V mixture components such that $p_j(\mathbf{x}) = \delta_{\mathbf{x},j}$, i.e. a delta-peak at one of the possible values.

It is not hard to compute the true log-likelihood of this model for any data-case. For that we compute,

$$\log p(\mathbf{y}) = \log \int \mathbf{dh}\ p(\mathbf{y}, \mathbf{h}) = \log 1/V \tag{4.36}$$

If we instead compute the folding-in expression in Equation 4.30 we observe that we can shift all the weight $\{h_j\}$ to the component which has the delta-peak located at the data-case, $h_{j*} = 1,\ h_j = 0, j \neq j*$ with $j* = \mathbf{y}$. This results in an expression $\log p(\mathbf{y}) = \log 1$ which confirms that it over-estimates the true log-likelihood.

In the other view, we should include a normalization factor (Eqn.4.34) which equals $\log Z = \log 1/V$ and confirms again that we would report overly optimistic results using the folding-in heuristic.

Note that neither the true log-likelihood of p nor the normalization factor of g are tractable for real world problems.

4.4 Implications For Learning The observations in this section also have implications for learning. The standard learning algorithms for EPCA and PLSA maximize the approximate objective function for model p that ignores the entropy term, i.e. Equation 2.6. We have seen above that this is equivalent to minimizing the energy function for another model g while ignoring its normalization constant. Learning based on folding-in assumes and benefits from the fact that the "quality" of a model is evaluated using the folding-in recipe, both during training and during testing. Therefore, it is possible to misinterpret these results and think that we have learned a great model (since both training and testing are done using folding-in) when in reality what we have learned could be very poor. Consider the toy example above: every data-point is assigned the maximal possible value of 1, so according to the objective there is nothing left to learn. However, after including the normalization constant for model g we find that every data-point really has a probability of $1/V$ (uniform) which means that the model has not learned anything whatsoever. The same conclusion was reached by computing the correct test log-likelihood for model p.

The fact that learning algorithms for PLSA and EPCA still seem to produce reasonable results in many cases may be explained by two factors. 1) The toy example above was designed to exaggerate the effect. With fewer data-dependent adjustable parameters the influence of the normalization constant is less pronounced and the effect may be more subtle. This leads to the prediction that the effect is stronger for PLSA models with many topics (and indeed this is what we observe in our experiments in the next section). 2) If one trains a model to optimize the wrong objective and evaluates the model using the same wrong objective the results may look very good whereas in reality they might be very poor.

5 Experiments

Dataset	D_train	D_test	Vocab size	Avg doc length
Cranfield	979	419	3763	84.3
Medline	724	309	7014	79.6
NIPS	150	50	10780	1381.1

Table 1: General characteristics of datasets used in experiments.

We report results on 3 datasets: Cranfield, Medline and NIPS. Details of these datasets are shown in table 1.

We trained LDA, EPCA and PLSA models on all three datasets. We computed perplexity in two different ways: (i) folding-in using the full test documents (simlar to [1]) (full folding-in); and (ii) folding-in using 50% of words of

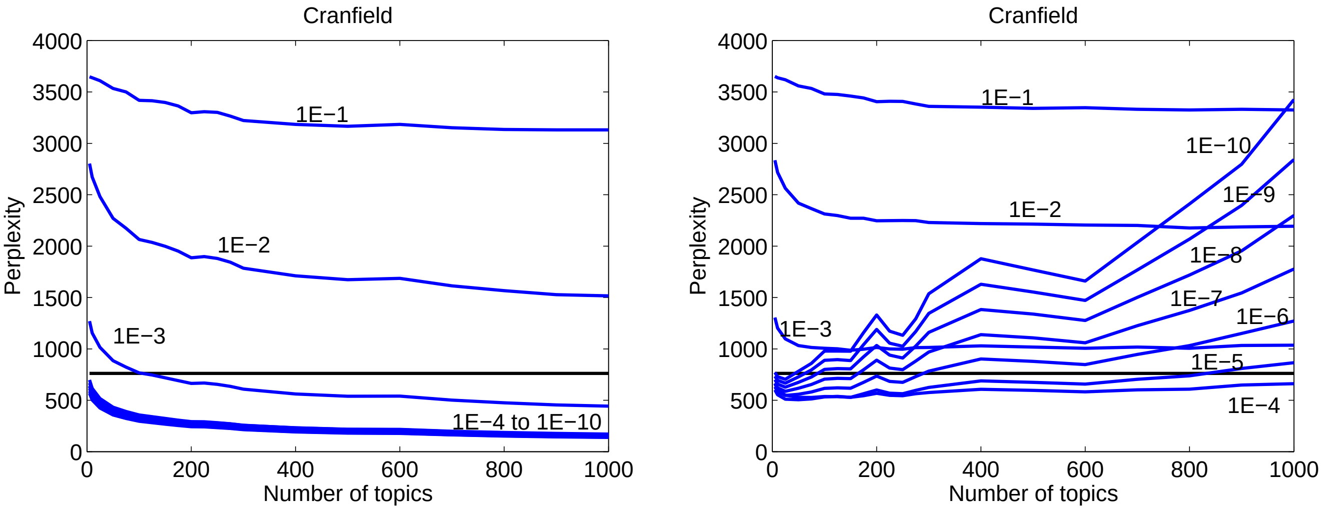

Figure 4: Test set perplexity of PLSA on the Cranfield dataset with various regularization values using full folding-in (left) and half folding-in (right)

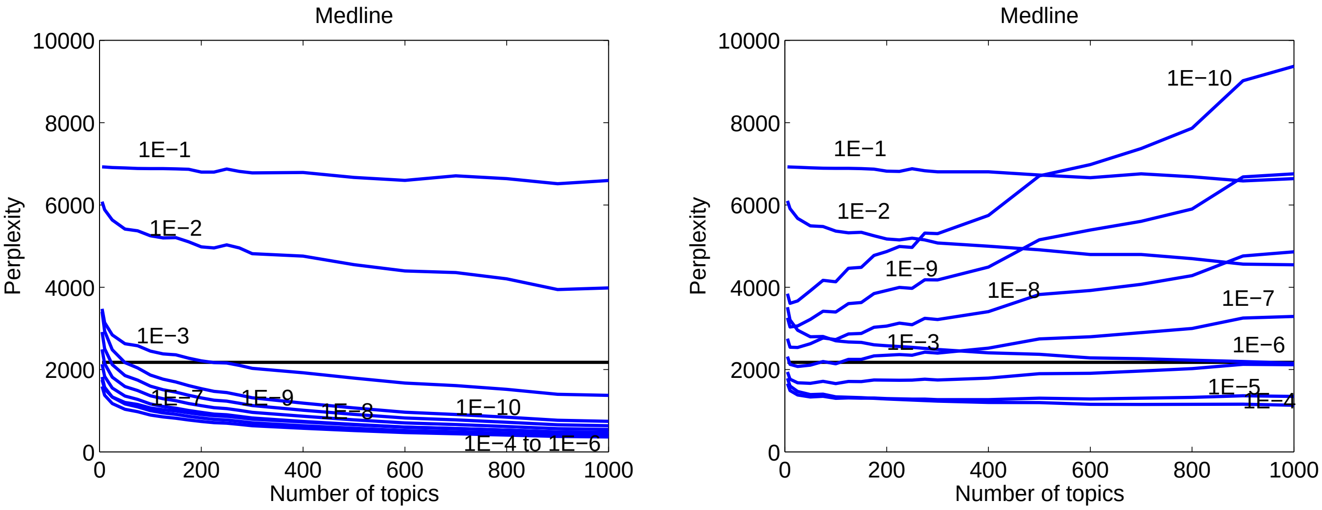

Figure 5: Test set perplexity of PLSA on the Medline dataset with various regularization values using full folding-in (left) and half folding-in (right)

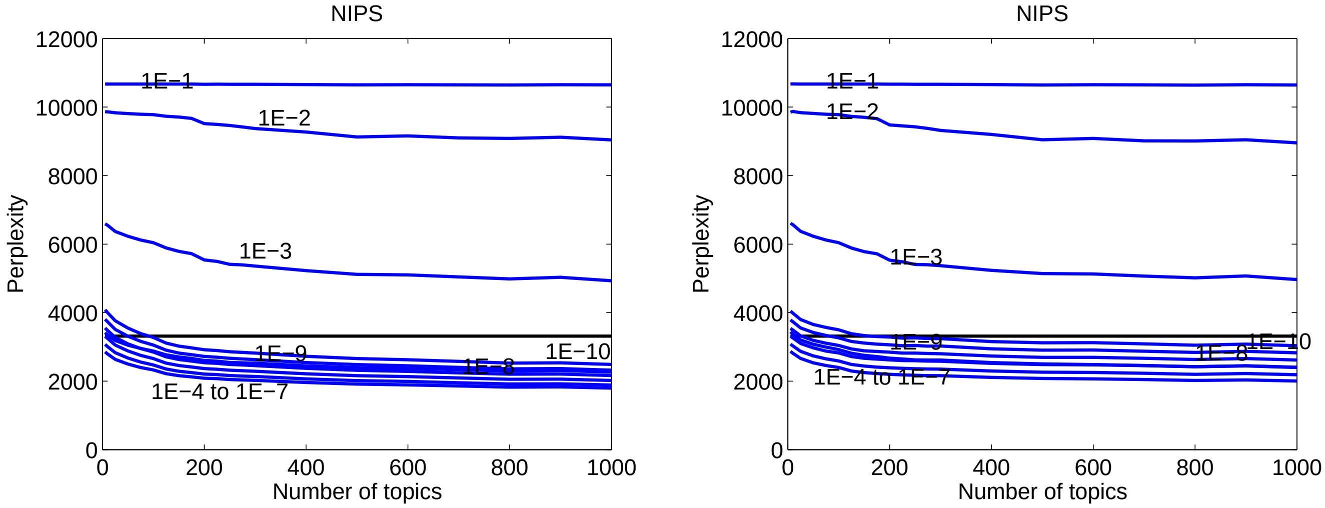

Figure 6: Test set perplexity of PLSA on the NIPS dataset with various regularization values using full folding-in (left) and half folding-in (right)

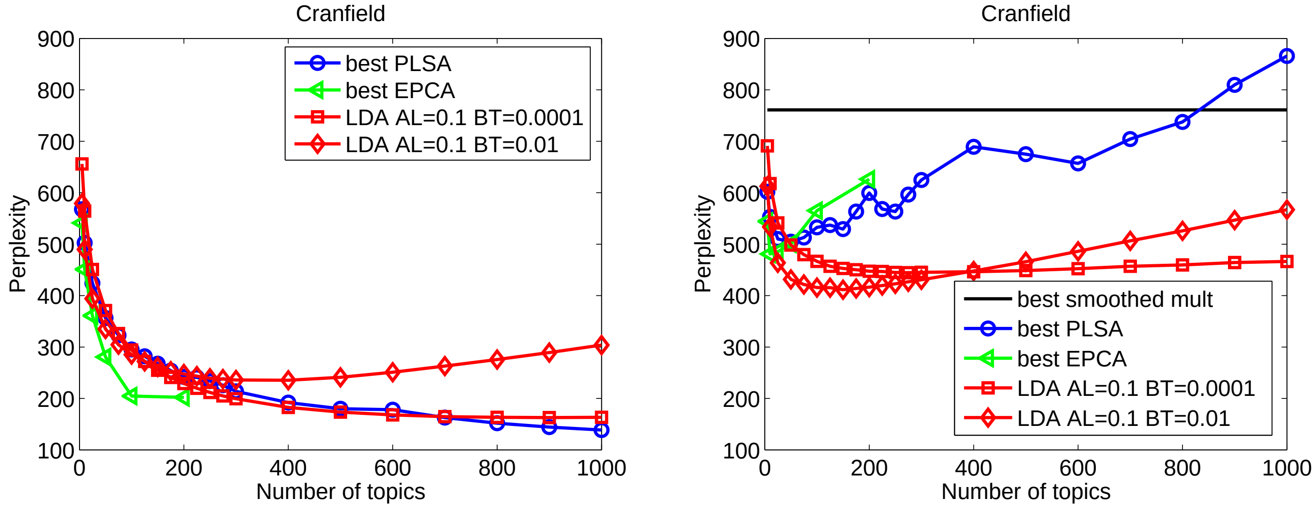

Figure 7: Comparison of test set perplexity of PLSA, EPCA and LDA on the Cranfield dataset using full folding-in (left) and half folding-in (right)

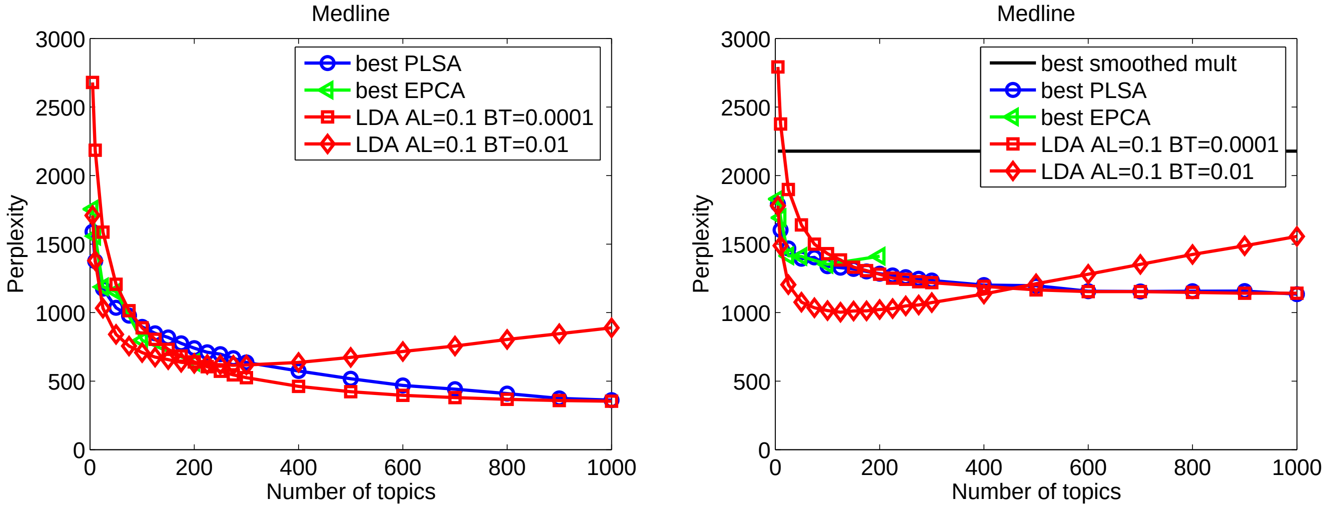

Figure 8: Comparison of test set perplexity of PLSA, EPCA and LDA on the Medline dataset using full folding-in (left) and half folding-in (right)

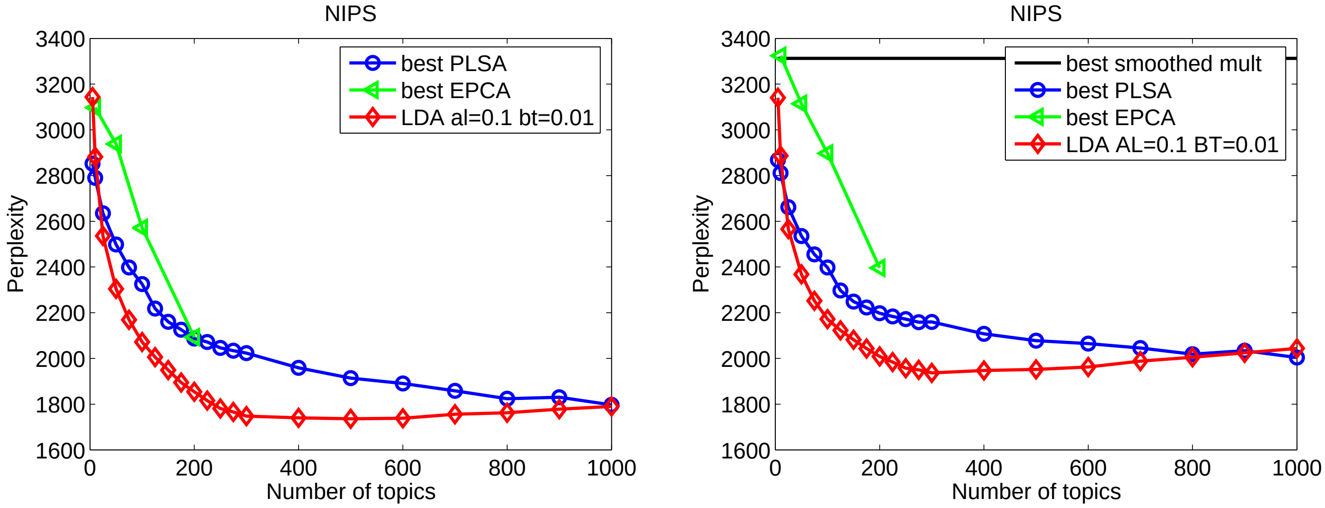

Figure 9: Comparison of test set perplexity of PLSA, EPCA and LDA on the NIPS dataset using full folding-in (left) and half folding-in (right)

a test document, and computing perplexity on the remaining words (half folding-in). We argue that the half folding-in approach used in our experiments to measure perplexity is a fair measure because unlike the full folding-in approach, the half folding-in approach does not see the part of the data on which we measure the test likelihood. It, therefore, cannot overfit on the test data and lead to incorrect and optimistic perplexity values.

Perplexity for PLSA is computed by first folding-in: computing $\hat{h}_{jn}$ (or $p(z|d)$) by iterating only the E-step of EM. After that we compute the log-probability of the test-data and transform it to perplexity as follows:

$$\text{Perplexity} = \exp\left[-\frac{\sum_{i,d}\log\sum_k W_{x_{id}^{\text{test}},k}H_{kd}}{\sum_d N_d}\right] \tag{5.37}$$

A similar procedure was followed for EPCA.

For LDA we implemented the collapsed Gibbs sampler described in [14]. Test-set perplexity was computed by averaging over 10 independent Gibbs samplers. After the Markov chains converged on the training set we fixed the assignments to the last iteration on the training data and sampled the assignments on the test data until convergence (this is analogous to folding-in). For the last sample on the training set we then compute,

$$\phi^s_{wk} = \frac{N^s_{wk}+\beta}{N^s_k+V\beta} \tag{5.38}$$

where $N^s_{wk} = \sum_{i,j}\mathbb{I}[x_{ij} = w]\mathbb{I}[z^s_{ij} = k]$, $N^s_k = \sum_i N^s_{ik}$ and V is the vocabulary size. Using the last sample from the Gibbs sampling chain on the test-set we compute,

$$\theta^s_{kd} = \frac{N^s_{kd}+\alpha}{N^s_d+K\alpha} \tag{5.39}$$

where $N^s_{kd} = \sum_i \mathbb{I}[z^s_{id} = k]$ and K is the number of topics. These are then used to compute the test-set perplexity. In the case of half folding-in the topic assignments for only 50% of the words of the test data were sampled and perplexity is measured on the rest of the words of the test documents (this is analogous to half folding-in of PLSA described earlier).

We experimented with three different versions of PLSA: 1) PLSA-HOF: the standard approach described in [1]; 2) PLSA-VEM using Equation 3.25 as described in Section 3.4; and 3) PLSA-TEM: "tempered EM" as described in [1] but searching for the annealing parameter, β, between $[0.5, 1]$ in increments of 0.02 on the test set directly (hence, this represents an upper bound on performance). We compared perplexity of PLSA-HOF, PLSA-VEM and PLSA-TEM on Cranfield, Medline and NIPS datasets for both full folding-in and half folding-in. From Figures 1, 2 and 3, it can be observed that all three approaches give similar results on Cranfield, Medline and NIPS. Unlike in [1], we did not find any advantage of "tempering". A similar result was also found in [8]. As the performance is similar for all three version of PLSA, in the subsequent experimental results we only show the results for PLSA-HOF.

To avoid overfitting both PLSA and EPCA were regularized by adding a constant α to $p(w|z)$ and renormalized. We tried α = [1E-10, 1E-9, ..., 1E-1] and show results for all values in the PLSA plots and best values when comparing with other models.

We now compare the perplexity results of PLSA-HOF using full folding-in and half folding-in for a range of regularization parameters, α. Figures 4, 5, and 6 show the perplexity results using full folding-in and half folding-in for Cranfield, Medline and NIPS respectively. For the smaller datasets (Cranfield and Medline), it can be seen that the perplexity values are significantly higher for half folding-in compared to full folding-in. Additionally, it can also be seen that in both these datasets perplexity results for half folding-in are very sensitive to the regularization parameter, α.

Since full folding-in is given more information to base its predictive probabilities on, we expect full folding-in to produce somewhat better perplexity than half folding-in. This in itself does therefore not prove the claim that full folding-in is overly optimistic. Observe however that for full folding-in perplexity always improves with increasing number of topics, irrespective of the regularization value applied. In fact, in some of our experiments we increased the number of topics, T, to a value greater than the vocabulary size and the perplexity value still kept decreasing. Hence, we see no sign of overfitting with full folding-in. This is not, however, true for half folding-in where for small regularization parameters one can clearly observe overfitting as the model complexity grows. These results support our claim about the overly optimistic perplexity estimates of full folding-in, especially for large T. These effects are alleviated in NIPS as NIPS documents are significantly longer (by more than a factor of 10, see Table 1) and hence folding-in on 50% of words (half folding-in case) gives a better estimate of H resulting in a relatively lower perplexity.

Additionally, we compare LDA with the best perplexity (best is defined as the curve with the lowest perplexity value) results for PLSA and EPCA using both full folding-in and half folding-in approaches. We show EPCA results only until 200 topics as EPCA was extremely slow to run and prone to numerical instabilities.

Figures 7, 8, and 9 show the results of these experiments for Cranfield, Medline and NIPS respectively. It can be noted that the perplexity results of LDA are generally better than those of the best perplexity results of PLSA and EPCA, more so for the half folding-in case. The perplexity results of PLSA and EPCA are similar to each other but we found that PLSA is easier to train than EPCA.

6 Discussion

In this paper we discuss a class of deterministic latent variable models and their learning algorithms under a unifying framework. We classify these models into two categories based on the type of latent variables used, namely, (i) discrete variables (e.g. K-means and Viterbi on HMM) or (ii) continuous variables (e.g. NMF, PLSA, PCA, ICA and EPCA). Hence NMF, PLSA, PCA, ICA and EPCA fall under the same umbrella within this framework and are different only in the way they mix the latent variables: NMF and PLSA mix latent variables in the probability domain while EPCA, PCA and ICA mix latent variables in the log-probability domain. In our experiments, we found that EPCA and PLSA produce models with similar performance and that learning for EPCA is relatively difficult. We also found that LDA produced better models than PLSA or EPCA.

Our main contribution, however, is the observation that the standard learning algorithms for the category (ii) models optimize a questionable objective. To derive the objective from the variational objective one has to dismiss an entropy term with a value of $-\infty$ which renders the resulting objective numerically unrelated (and certainly not a bound) to the log-probability. While these models, in general, can learn reasonable parameters in training, the test data probability using folding-in can be highly optimistic, especially when there are a large number of latent variables. Models with discrete latent variables do not suffer from these issues as their entropy contribution is 0 in the MAP approximation.

One can try to fix this problem, in theory, by switching to a mean field approach which incorporates an approximation to the dismissed entropy, or by changing the model and incorporating a normalization factor. The latter approach naturally leads to the view of an “energy-based” model. We note, however, that computing the normalization factor is usually intractable.

We have also experimentally verified that computing test-set perplexity using the “folding-in” recipe can easily lead to overly optimistic results.

Acknowledgments

We thank Geoff Hinton for his deep insights that helped shape this paper.

Max Welling was supported by NSF grants IIS-0535278 and IIS-0447903.

The research reported here is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant BSIK03024.

We acknowledge use of the computer clusters supported by NIH grant LM-07443-01 and NSF grant EIA-0321390 to Pierre Baldi and the Institute of Genomics and Bioinformatics.

References

[1] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm, 1999.

[2] M. Collins, S. Dasgupta, and R. E. Schapire. A generalization of principal components analysis to the exponential family. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Neural Information Processing Systems 14*. MIT Press, 2002.

[3] Wray Buntine. Variational Extensions to EM and Multinomial PCA Volume 2430 of *Lecture Notes in Computer Science*, Helsinki, Finland, 2002. Springer.

[4] M. Girolami and A. Kaban. On an equivalence between PLSI and LDA. In *Proceedings of SIGIR 2003*, 2003.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[6] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* **B**, 39:1–38, 1977.

[7] R.M. Neal and G.E. Hinton. A view of the EM algorithm that justifies incremental, sparse and other variants. 1999.

[8] T. Brants. Test data likelihood for plsa models. *Inf. Retr.*, 8(2):181–196, 2005.

[9] Y.W. Teh, M. Welling, S. Osindero, and G.E. Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research - Special Issue on ICA*, 4:1235–1260, 2003.

[10] D.D. Lee and H.S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[11] A. Olshausen and D. Field. Sparse coding with over-complete basis set: A strategy employed by V1? *Vision Research*, 37:3311–3325, 1997.

[12] S.T. Roweis. EM algorithms for PCA and SPCA. In *Neural Information Processing Systems*, volume 10, pages 626–632, 1997.

[13] B. Marlin and R. Zemel. The multiple multiplicative factor model for collaborative filtering. In *Proceedings of the 21st International Conference on Machine Learning*, volume 21, 2004.

[14] T.L. Griffiths and M. Steyvers. A probabilistic approach to semantic representation. In *Proceedings of the 24th Annual Conference of the Cognitive Science Society*, 2002.

[15] D. Downey, S. Dumais and E. Horvitz Head and tails: Studies of Web Search with Common and Rare Queries In *Proceedings of the 30th Annual International ACM SIGIR Conference*, 847-848, 2007

[16] Y. Akita and T. Kawahara. Language Model Adaptation Based on PLSA of Topics and Speakers for Automatic Transcription of Panel Discussions. In *Journal of IEICE Transactions*, 3:439–445, 2005.