

# Topic Models over Text Streams: A Study of Batch and Online Unsupervised Learning

Arindam Banerjee\*

Sugato Basu†

## Abstract

Automated unsupervised learning of topic-based clusters is used in various text data mining applications, e.g., document organization in content management, information retrieval and filtering in news aggregation services. Typically batch models are used for this purpose, which perform clustering on the document collection in aggregate. In this paper, we first analyze three batch topic models that have been recently proposed in the machine learning and data mining community – Latent Dirichlet Allocation (LDA), Dirichlet Compound Multinomial (DCM) mixtures and von-Mises Fisher (vMF) mixture models. Our discussion uses a common framework based on the particular assumptions made regarding the conditional distributions corresponding to each component and the topic priors. Our experiments on large real-world document collections demonstrate that though LDA is a good model for finding word-level topics, vMF finds better document-level topic clusters more efficiently, which is often important in text mining applications. In cases where offline clustering on complete document collections is infeasible due to resource constraints, online unsupervised clustering methods that process incoming data incrementally are necessary. To this end, we propose online variants of vMF, EDCM and LDA. Experiments on real-world streaming text illustrate the speed and performance benefits of online vMF. Finally, we propose a practical heuristic for hybrid topic modeling, which learns online topic models on streaming text data and intermittently runs batch topic models on aggregated documents offline. Such a hybrid model is useful for applications (e.g., dynamic topic-based aggregation of consumer-generated content in social networking sites) that need a good tradeoff between the performance of batch offline algorithms and efficiency of incremental online algorithms.

## 1 Introduction

Text data mining techniques have widespread application in today’s internet technologies. Automated unsupervised learning of latent topics from text documents is useful for different applications, like document organization (e.g., topic-based clustering of news articles), better retrieval and filtering of information (e.g., personalization of search results based on user interests), etc. Typically batch algorithms are used for this purpose, which perform clustering on the document collection in aggregate. Due to the large volume of text data that needs to be processed in these applications, scaling up of topic modeling algorithms to large datasets is very crucial.

In this paper, we first analyze three batch topic models that have been recently proposed in the machine learning and data mining community – Latent Dirichlet Allocation (LDA), Dirichlet Compound Multinomial (DCM) mixtures and von-Mises Fisher (vMF) mixture models – using a common framework based on the particular assumptions made regarding the conditional distributions corresponding to each component and the topic priors. vMF and DCM are essentially mixture models, which model topics at the document-level, while LDA is a more complex Bayesian model that considers per-word topic distributions. Since topic-based clustering at the document level is important in many text mining applications, we compare the efficiency and performance tradeoffs of these batch models in the task of document clustering.

Many applications also need the ability to process large volumes of data arriving over time in a stream, e.g., news articles arriving continually over a newswire. There are various challenges in analyzing such data – the whole data cannot be fit into memory at once and has to be processed incrementally, multiple scans of the data kept in secondary storage is not always possible due to real-time response rate requirements, etc. [48]. This necessitates the use of incremental topic models while performing unsupervised learning over streaming text, to efficiently handle the scale and response-rate requirements in unsupervised text mining applications on the web. To this end, we propose online variants of the three topic models – LDA, vMF and DCM, which would be very useful in inferring underlying topics on fast incoming streams of text where aggregate data analysis on the complete document corpus is not always feasible due to time and resource constraints.

Another phenomenon that is occurring on the web is its movement from being a mechanism for delivering static web-content in the extant Web 1.0 model to a platform facilitating dynamic collaborative content creation in the emerging Web 2.0 paradigm. This trend is reflected in the growing popularity of new web services (e.g., Wikipedia, Slashdot, Webwag, Blogger) hosting consumer-generated media, and has created an explosion of text content being continually generated by users

---

\*Department of CSE, University of Minnesota

†AI Center, SRI International

online. A lot of Web 2.0 applications are facing the need to process incoming data streams incrementally during peak load, with the option of doing offline processing on non-peak hours. This motivated us to create a practical hybrid topic model scheme: learning online topic models on streaming data, with intermittent background batch topic models on offline aggregated text documents. The online component is necessary for categorizing documents into topic-based clusters in real-time, whereas the intermittent batch processing is required for improved unsupervised mining on larger offline text collections.

The main contributions of the paper are:

- Comparing the performance of different offline topic modeling algorithms, and demonstrating that while LDA is good at finding word-level topics, vMF is more effective and efficient at finding document-level clusters;
- Proposing a new online vMF algorithm, that outperforms online versions of LDA and DCM in efficiency and performance;
- Presenting a practical hybrid scheme for topic modeling over document streams, which provides a good tradeoff between speed and accuracy while performing unsupervised learning over large volumes of text.

The paper is organized as follows: Section 2 gives an overview of the three batch topic models we consider in this paper, while Section 3 discusses their online variants. The experiments in Section 4 empirically compare the algorithms, while Section 5 discusses related research in this area. Finally, Section 6 concludes the paper with discussions about possible areas of future work.

## 2 Batch Topic Models

Unsupervised text mining and topic modeling has been a focus of active research over the past few years. The popular generative clustering and topic models for text analysis can be broadly divided into a few categories, depending on the particular assumptions made regarding the conditional distributions corresponding to each component and the cluster priors. The conditional assumptions are typically from one of two classes of distributions: multinomial distributions on the unit simplex [36], or the von-Mises Fisher distribution on the unit hypersphere [5]. Further, the probability of an observation can be computed from the conditional distribution using a point estimate of the distribution, as is typical in vMF models [5] and was used originally in multinomial models [36], or from a full Bayesian

model, as is becoming increasingly common for multinomial distributions [21]. The cluster priors were traditionally modeled using a distribution that was fixed across all documents, leading to the mixture of unigrams model [7, 45]. Recent years have seen development of non-parametric Bayesian modeling of the priors [10, 21]. Table 1 summarizes the main unsupervised models for text analysis, based on the above discussion.

In this paper, we focus on 3 representative models based on different assumptions on the conditional and prior:

1. Point Prior, Point Conditional: The first model we consider is the mixture of von Mises-Fisher distributions [6, 5] that uses a point estimate for the prior and a point estimate for the conditional distribution corresponding to each cluster.
2. Point Prior, Bayesian Conditional: The second model is the mixture of Dirichlet compound multinomial distribution [45, 29, 18] that uses a point estimate for the prior and a full Bayesian model for computing probability of an observation from a cluster.
3. Bayesian Prior, Bayesian Conditional: The third model is the Bayesian latent dirichlet allocation model [21, 10] that uses a non-parametric Bayesian model for the priors and a full Bayesian model for computing probability of an observation from a cluster.

We describe the details of each model briefly in the following subsections.

**2.1 vMF Models** The first model is a classic example of a mixture model [7, 5] that uses von Mises-Fisher distributions as the components. Figure 1 shows the graphical model representation of such a mixture model. The use of von Mises-Fisher distributions for text analysis was largely motivated by the extensive use of cosine-similarity in the information retrieval community [20, 40]. The spherical kmeans algorithm, a special case of the mixture of von Mises-Fisher distribution model, was one of the earlier successful models for text clustering. In the mixture of von Mises-Fisher (movMF) distributions model, documents are represented as an unit vector which is simply the  $L_2$  normalized version of the TFIDF vector [20] corresponding to the document. Thus, all documents lie on the surface of the unit hypersphere. Further, since all components of all the unit vectors are positive, the data actually lies only on the portion of the hypersphere on the positive orthant.

A  $d$ -dimensional unit random vector  $\mathbf{x}$  (i.e.,  $\mathbf{x} \in \mathbb{R}^d$  and  $\|\mathbf{x}\| = 1$ , or equivalently  $\mathbf{x} \in \mathbb{S}^{d-1}$ ) is said to

|       |          | Conditional      |                           |                  |
|-------|----------|------------------|---------------------------|------------------|
|       |          | Multinomial      |                           | von-Mises Fisher |
|       |          | Point            | Bayesian                  | Point            |
| Prior | Point    | naive-Bayes [36] | DM [45, 46], DCM [29, 18] | movMF [5]        |
|       | NP Bayes | LDA [10]         | Bayesian LDA [21]         | -                |

Table 1: Unsupervised models for text. The conditional is typically from the multinomial or vMF distribution, and the observation probability can be computed from a point estimate [36, 5] or a Bayesian model [45]. The prior is typically either a point estimate [7, 45, 18] or a non-parametric Bayesian model [10, 21].

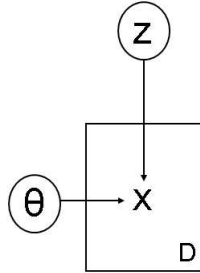


Figure 1: Graphical plate model of the vMF mixture model.  $\mathbf{z}$  are the latent cluster variables,  $\Theta$  are the model parameters, and  $\mathbf{x}$  are the observed unit vectors.

have  $d$ -variate von Mises-Fisher (vMF) distribution if its probability density function is given by

$$(2.1) \quad f(\mathbf{x}|\boldsymbol{\mu}, \kappa) = c_d(\kappa) e^{\kappa \boldsymbol{\mu}^T \mathbf{x}},$$

where  $\|\boldsymbol{\mu}\| = 1$ ,  $\kappa \geq 0$  and  $d \geq 2$ . The normalizing constant  $c_d(\kappa)$  is given by

$$(2.2) \quad c_d(\kappa) = \frac{\kappa^{d/2-1}}{(2\pi)^{d/2} I_{d/2-1}(\kappa)},$$

where  $I_r(\cdot)$  represents the modified Bessel function of the first kind and order  $r$ . The density  $f(\mathbf{x}|\boldsymbol{\mu}, \kappa)$  is parameterized by the mean direction  $\boldsymbol{\mu}$ , and the *concentration* parameter  $\kappa$ , so-called because it characterizes how strongly the unit vectors drawn according to  $f(\mathbf{x}|\boldsymbol{\mu}, \kappa)$  are concentrated about the mean direction  $\boldsymbol{\mu}$ . Larger values of  $\kappa$  imply stronger concentration about the mean direction. In particular when  $\kappa = 0$ ,  $f(\mathbf{x}|\boldsymbol{\mu}, \kappa)$  reduces to the uniform density on  $\mathbb{S}^{d-1}$ , and as  $\kappa \rightarrow \infty$ ,  $f(\mathbf{x}|\boldsymbol{\mu}, \kappa)$  tends to a point density. The interested reader is referred to [31] or [16] for details on vMF distributions.

Consider a mixture model over  $k$  vMF (movMF) distributions

$$(2.3) \quad f(\mathbf{x}|\Theta) = \sum_{h=1}^k \alpha_h f_h(\mathbf{x}|\theta_h),$$

where  $\Theta = \{\alpha_1, \dots, \alpha_k, \theta_1, \dots, \theta_k\}$  and the  $\alpha_h$  are non-negative and sum to one, and  $f_h(\mathbf{x}|\theta_h)$  is a vMF distribution with parameter  $\theta_h = (\boldsymbol{\mu}_h, \kappa_h)$  for  $1 \leq h \leq k$ . Given a set  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$  of data points assumed to have been sampled i.i.d. following the mixture distribution, one can use the EM algorithm and appropriate approximations to estimate the parameters of the model. Following [5], the M-step involves the following parameter updates:

$$(2.4) \quad \alpha_h = \frac{1}{m} \sum_{i=1}^m p(h|\mathbf{x}_i, \Theta),$$

$$(2.5) \quad \mathbf{r}_h = \sum_{i=1}^m \mathbf{x}_i p(h|\mathbf{x}_i, \Theta), \quad \bar{r}_h = \frac{\|\mathbf{r}_h\|}{\sum_{i=1}^m p(h|\mathbf{x}_i)},$$

$$(2.6) \quad \hat{\boldsymbol{\mu}}_h = \frac{\mathbf{r}_h}{\|\mathbf{r}_h\|},$$

$$(2.7) \quad \hat{\kappa}_h = \frac{\bar{r}_h d - \bar{r}_h^3}{1 - \bar{r}_h^2}.$$

In the E-step, the distribution of the hidden variables [34, 8] is computed as

$$(2.8) \quad p(h|\mathbf{x}_i, \Theta) = \frac{\alpha_h f_h(\mathbf{x}_i|\Theta)}{\sum_{l=1}^k \alpha_l f_l(\mathbf{x}_i|\Theta)}.$$

It can be shown [14] that the incomplete data log-likelihood,  $\log p(\mathcal{X}|\Theta)$ , is non-decreasing at each iteration of the parameter and distribution updates. Iteration over these two updates till convergence constitutes the movMF algorithm.

One can consider a second update scheme based on hard assignment, instead of a probabilistic assignment, given by

$$(2.9) \quad q(h|\mathbf{x}_i, \Theta) = \begin{cases} 1, & \text{if } h = \underset{h'}{\operatorname{argmax}} p(h'|\mathbf{x}_i, \Theta), \\ 0, & \text{otherwise.} \end{cases}$$

Spherical kmeans is a useful special case of the hard assignment model corresponding to the case when all  $\kappa_h$  are assumed to be the same [5, 15].

**2.2 DM/DCM models** The second model we consider is the Dirichlet mixture (DM) [45, 46], which is also known as the mixture of Dirichlet compound multinomial (DCM) distributions [29, 18]. The model is similar to a mixture model, such as movMF, but uses a full Bayesian model on the conditional distribution corresponding to each cluster. In particular, the model uses a Dirichlet prior over multinomial conditionals, where the parameters of the Dirichlet are different for every cluster. Figure 2 shows a graphical model representation of the mixture of DCM model.

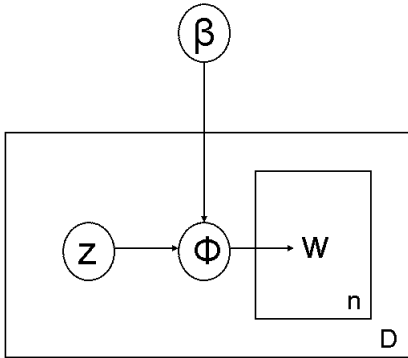


Figure 2: Graphical plate model of DCM mixture model.  $\mathbf{z}$  are the latent cluster variables,  $\phi$  and  $\beta$  are the model parameters, and  $\mathbf{w}$  are the observed words.

The mixture of multinomial model [36] is one of the earlier models for text analysis and is an example of a naive-Bayes model [35]. In the basic model, corresponding to each cluster, one assumes a probability distribution over all words, i.e., for a word  $w_j$ ,  $\phi_j$  is the probability of emitting  $w_j$ , so that  $\sum_{j=1}^d \phi_j = 1$ . Then, a document is generated by sampling repeatedly from the word distribution. The naive-Bayes assumption posits conditional independence of subsequent draws, so that for a document  $\mathbf{x}$  with  $x_j$  occurrences of word  $w_j$ , the probability of observing  $\mathbf{x}$  given the model is

$$(2.10) \quad p(\mathbf{x}|\phi) = \frac{n!}{\prod_{j=1}^d x_j} \prod_{j=1}^d \phi_j^{x_j}.$$

Instead of using a single multinomial for each cluster, the DCM model assumes a Dirichlet prior over all multinomials. In particular, the prior probability of the multinomial with parameter  $\phi$  is

$$(2.11) \quad D(\phi|\beta) = \frac{\Gamma\left(\sum_{j=1}^d \beta_j\right)}{\prod_{j=1}^d \Gamma(\beta_j)} \prod_{j=1}^d \phi_j^{\beta_j-1},$$

where  $\beta$  is the parameter of the Dirichlet distribution.

Then, the probability of observing document  $\mathbf{x}$  is obtained by integrating the probability contributions of individual multinomials over the prior so that

$$\begin{aligned} p(\mathbf{x}|\beta) &= \int_{\phi} p(\mathbf{x}|\phi)p(\phi|\beta)d\phi \\ &= \frac{n!}{\prod_{j=1}^d x_j} \frac{\Gamma\left(\sum_{j=1}^d \beta_j\right)}{\Gamma\left(\sum_{j=1}^d (x_j + \beta_j)\right)} \prod_{j=1}^d \frac{\Gamma(x_j + \beta_j)}{\Gamma(\beta_j)}. \end{aligned}$$

The DCM distribution does not belong to the exponential family [18] and the maximum likelihood parameters estimates need non-trivial iterative computations [32, 45]. Motivated by such computational bottlenecks, Elkan [18] recently proposed an exponential family approximation of the DCM model, known as the EDCM model, for which the computations are comparatively reasonable. In particular, the probability of the document  $\mathbf{x}$  is given by

$$(2.12) \quad q(\mathbf{x}|\beta) = n! \frac{\Gamma(s)}{\Gamma(s+n)} \prod_{j:x_j \geq 1} \frac{\beta_j}{x_j},$$

where  $\beta$  is the parameter of the EDCM model, and  $s = \sum_{j=1}^d \beta_j$ . Given a set of  $m$  documents  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ , one can run an EM algorithm for the mixture of EDCM models in order to estimate the parameters as well as get a clustering of the documents. The E-step involves computing  $p(h|\mathbf{x})$  and is similar to that of the mixture of vMF distributions. The M-step parameter estimates  $(s_h, \beta_h)$ ,  $h = 1, \dots, k$  are given by

$$(2.13) \quad s_h = \frac{\sum_{j=1}^d \sum_{i=1}^m p(h|\mathbf{x}_i) \mathbb{I}(x_{ij} \geq 1)}{\sum_{i=1}^m p(h|\mathbf{x}_i) \Psi(s_h + n_i) - M \Psi(s_h)},$$

$$(2.14) \quad \beta_{hj} = \frac{\sum_{i=1}^m p(h|\mathbf{x}_i) \mathbb{I}(x_{ij} \geq 1)}{\sum_{i=1}^m p(h|\mathbf{x}_i) \psi(s_h + n_i) - M \Psi(s_h)},$$

where  $n_i$  is the number of words in document  $\mathbf{x}_i$ .<sup>1</sup> Elkan [18] recommends several other practical ways to get high quality clustering results from the mixture of EDCM model.

**2.3 LDA models** The third model is a full Bayesian version of latent Dirichlet allocation (LDA) [21, 10]. The fundamental difference between the LDA model and the vMF and DCM models is that LDA uses a detailed non-parametric Bayesian model of the prior probability over all the clusters. Similar to the naive-Bayes model [36], LDA uses point estimates of multinomials as conditional distributions for each cluster. In this paper, we focus on the full Bayesian version of LDA that uses a Bayesian

<sup>1</sup>Note that the equation for  $s_h$  is a nonlinear equation in one variable, and needs to be first solved before estimating the  $\beta_{hj}$  values.

model for computing the probability of an observation given each cluster [21]. Similar to the DCM model, the Bayesian LDA models assumes a Dirichlet prior over all multinomials, where the Dirichlet parameter is different for every cluster. Figure 3 shows the graphical model representation of the Bayesian LDA model.

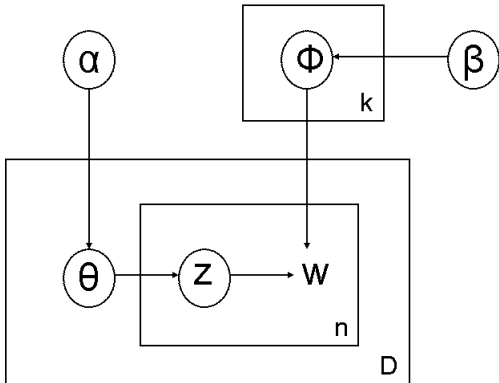


Figure 3: Graphical plate model of LDA.  $\mathbf{z}$  are the latent cluster variables,  $\alpha$ ,  $\beta$ ,  $\phi$  and  $\theta$  are the model parameters, and  $\mathbf{w}$  are the observed words.

Unlike the previous models, LDA assumes a different topic distribution for every document. In particular, in order to generate a document  $\mathbf{w}$ ,<sup>2</sup> a topic distribution  $\theta$  is first sampled from a Dirichlet prior, with parameter  $\alpha$ , on the topic simplex. To generate the  $\ell^{\text{th}}$  word of the document, a topic  $z_\ell$  is first sampled at random from the topic distribution  $\theta$ . Then a topic-specific multinomial  $\phi_{z_\ell}$  for the word distribution is sampled from the Dirichlet prior with parameter  $\beta$ , corresponding to the topic  $z_\ell$ . Finally, the word  $w_\ell$  is sampled according to the multinomial  $\phi_{z_\ell}$ . Thus, the probability of observing the document  $\mathbf{w}$  is given by

$$p(\mathbf{w}|\alpha, \beta) = \int_{\theta} p(\theta|\alpha) \left( \prod_{\ell=1}^n \sum_{z_\ell} p(z_\ell|\theta) \int_{\phi} p(w_\ell|\phi) p(\phi|\beta_{z_\ell}) d\phi \right) d\theta.$$

For a  $k$  component topic model, each  $z_\ell \in \{1, \dots, k\}$ .

Consider a corpus  $\mathbf{w} = \{w_1, \dots, w_n\}$ , where word  $w_\ell$  is from document  $d_\ell$ . Given particular values of  $\alpha$  and  $\beta$ , the main inference problem to be solved involves estimating  $z_\ell$  for each word  $w_\ell$ . As [21] showed, the inference problem can be solved using Gibbs sampling.

<sup>2</sup>We denote a document as a sequence  $\mathbf{w}$  of words rather than a feature vector  $\mathbf{x}$  as before. Note that vMF uses the feature vector representation, but DCM actually uses a sequence representation that can be compiled into a feature vector form.

In particular, from Bayes rule, the conditional posterior distribution for  $z_\ell$  is given by

$$(2.15) \quad p(z_\ell = h|\mathbf{z}_{-\ell}, \mathbf{w}) \propto p(w_\ell|z_\ell = h, z_{-\ell}, \mathbf{w}_{-\ell}) p(z_\ell = h|\mathbf{z}_{-\ell}).$$

A careful calculation [21] shows that both terms in the right hand side can be computed in closed form. In particular, we have

$$p(w_\ell|z_\ell = h, z_{-\ell}, \mathbf{w}_{-\ell}) = \frac{n_{-\ell,h}^{(w_\ell)} + \beta}{n_{-\ell,h}^{(\cdot)} + d\beta},$$

where  $n_{-\ell,h}^{(w_\ell)}$  is the number of instances of word  $w_\ell$  assigned to topic  $h$  not including the current word,  $n_{-\ell,h}^{(\cdot)}$  is the total number of words assigned to topic  $h$  not including the current word, and  $\beta$  is assumed to be uniform across all words, i.e., all components of  $\beta$  are the same. Note that  $\beta$  is a hyper-parameter that determines how heavily the estimates are smoothed. The second component of the (2.15) can be expressed as

$$p(z_\ell = h|\mathbf{z}_{-\ell}) = \frac{n_{-\ell,h}^{(d_\ell)} + \alpha}{n_{-\ell,\cdot}^{(d_\ell)} + k\alpha},$$

where  $n_{-\ell,h}^{(d_\ell)}$  is the number of words from document  $d_\ell$  assigned to topic  $h$ , not including the current one, and  $n_{-\ell,\cdot}^{(d_\ell)}$  is the number of words in document  $d_\ell$ , not including the current one.

Based on the above discussion, the conditional posterior probabilities of the topic assignments of each word is given by

$$(2.16) \quad p(z_\ell = h|\mathbf{z}_{-\ell}, \mathbf{w}) = \frac{n_{-\ell,h}^{(w_\ell)} + \beta}{n_{-\ell,h}^{(\cdot)} + d\beta} \frac{n_{-\ell,h}^{(d_\ell)} + \alpha}{n_{-\ell,\cdot}^{(d_\ell)} + k\alpha}.$$

A Markov Chain Monte Carlo algorithm based on the above equation can then be used to get samples from the topic distribution [21].

### 3 Online Topic Models

Unlike batch topic models, the literature on online topic models is not as extensive. However, there is growing interest in online topic models for text, since several applications of topic modeling have to deal with content that is generated over time. In particular, the explosion of consumer generated media on the web provides a strong motivation for an detailed study of online topic models. In this section, we present online versions of the three topic models discussed in Section 2. We also discuss a hybrid scheme of interleaving online topic modeling on streaming text with intermittent batch processing, to combine the higher efficiency of the online algorithm with the higher performance of the offline algorithm.

**3.1 Online vMF** The mixture of vMF distributions model is the simplest of the three models discussed in Section 2. We focus on the spherical kmeans algorithm, which is a popular special case of the general vMF model, and propose a version that is fully online. Since the batch vMF model uses the EM algorithm, our extension is partly motivated by the analysis of [34], and an application of a similar analysis on frequency sensitive clustering due to [6].

Given an existing mixture of vMF model based on a stream of  $t$  documents, and given a new document  $\mathbf{x}_{t+1}$ , the document can be assigned to the best cluster, i.e., the cluster having highest posterior probability  $p(h|\mathbf{x}_{t+1})$ , following (2.9). Now, the parameters of the model need to be updated based on the new document. While there are several choices of doing such an update, we choose a simple approach of only updating the parameters of the mixture component to which the current document got assigned to. Such a choice is partly motivated by theoretical results on online learning of exponential family distributions [7, 3]. In particular, for exponential family distributions, one can show a strong relative loss bound on streaming data based on the following simple recursive update of the mean parameter

$$\boldsymbol{\mu}^{(t+1)} = \boldsymbol{\mu}^{(t)} + \frac{1}{t+1}(\mathbf{x}_{t+1} - \boldsymbol{\mu}^{(t)}) .$$

In practice, as  $t$  increases, the last term becomes vanishingly small, which may not be desirable particularly in non-stationary environments. A practical trade-off is to maintain an effective count  $c_{t+1} = (1 - 1/L)c_t + 1$ , where  $L$  is the effective memory length [6]. Note that as  $t \rightarrow \infty$ ,  $c_t \rightarrow L$  from below. In case of von Mises-Fisher distributions, the estimate of the mean has to be normalized to lie on the unit hypersphere [6, 5], so that the recursive update becomes

$$(3.17) \quad \boldsymbol{\mu}^{(t+1)} = \frac{\boldsymbol{\mu}^{(t)} + \frac{1}{c_{t+1}}(\mathbf{x}_{t+1} - \boldsymbol{\mu}^{(t)})}{\|\boldsymbol{\mu}^{(t)} + \frac{1}{c_{t+1}}(\mathbf{x}_{t+1} - \boldsymbol{\mu}^{(t)})\|} .$$

Thus, the online vMF model is a truly online model that processes one point at a time and does not need to store any additional information other than the current set of parameters.

**3.2 Online DCM** The basic DCM model is not an exponential family distribution, so the simple recursive update is not appropriate for the mixture of DCM model. In fact, the EDCM model, which is an exponential family approximation to DCM, is actually not an exponential family model in the form it appears in [18], since the cumulant function has not been determined

exactly. The knowledge of the cumulant function is crucial to applying the simple recursive exponential family update [7, 3] because of the following reasons:

- (i) The EDCM is expressed in terms of its natural parameters. The maximum likelihood estimate of the natural parameter  $\boldsymbol{\theta}$  for any exponential family distribution is given by  $\hat{\boldsymbol{\theta}} = \nabla\psi^{-1}(\frac{1}{n}\sum_{i=1}^n \mathbf{x}_i)$ , where  $\psi$  is the cumulant and  $\mathbf{x}_i$  are the samples. Thus, such an estimation is not possible without the knowledge of the cumulant  $\psi$ . Further, other than constant variance Gaussians, the estimation equation cannot be written in a recursive form suitable for online updates.
- (ii) The maximum likelihood estimate of the expectation parameter  $\boldsymbol{\mu}$  for any exponential family distribution is given by  $\hat{\boldsymbol{\mu}} = \sum_{i=1}^n \mathbf{x}_i$ , which readily allows a recursive form that is appropriate for online updates. However, in order to compute the probability of a document  $p(\mathbf{x}|\boldsymbol{\mu})$  given the EDCM model and the estimated mean parameter, one needs to be able to either express the EDCM distribution in terms of its mean parameters, or get the corresponding estimate of the natural parameters, both of which require the knowledge of the cumulant [7].

As a result, we resort to a more explicit windowed update that we describe next. Consider an existing mixture of EDCM models based on a stream of  $t$  documents. Given a new document  $\mathbf{x}_{t+1}$ , it is straightforward to compute  $p(h|\mathbf{x}_{t+1})$  from the existing model components, by applying Bayes' Rule. After assigning the document to the most likely component, we update the component parameters as follows. Let  $M_h^L$  be the set of the last  $L$  documents that have been assigned to topic  $h$ . If the new document  $\mathbf{x}_{t+1}$  is assigned to topic  $h$ , then  $M_h^L$  is updated by inserting  $\mathbf{x}_{t+1}$  in, and deleting the oldest document in the set. Then, the documents in  $M_h^L$  is used to estimate a new sets of parameters following (2.13) and (2.14). The parameters of the EDCM components are updated as a moving average of the new estimated parameters and the existing parameter values over the sliding window.

**3.3 Online LDA** For learning the LDA model online, we use the incremental LDA model proposed in [41]. In the incremental LDA algorithm, batch LDA is initially run on a small window of the incoming data stream and the LDA parameters  $\phi$  and  $\theta$  are initialized using the MAP estimates  $\hat{\phi}$  and  $\hat{\theta}$  estimated from this

window:

$$(3.18) \quad \phi_j^{w_l} = \frac{n_j^{w_l} + \beta}{n_j^{(\cdot)} + W\beta},$$

$$(3.19) \quad \theta_j^{d_l} = \frac{n_j^{d_l} + \alpha}{n_{(\cdot)}^{d_l} + T\alpha},$$

where  $n_j^{w_l}$  is the number of times the word  $w_l$  is assigned to topic  $j$ ,  $n_j^{(\cdot)}$  is the sum of  $n_j^{w_l}$  over all words,  $n_j^{d_l}$  is the number of times a word from document  $d_l$  has been assigned to topic  $j$ , and  $n_{(\cdot)}^{d_l}$  is the sum of  $n_j^{d_l}$  over all topics.

Henceforth, with the arrival of every new document  $d$ , the topic assignment of the  $i^{\text{th}}$  word in the document is estimated as:

$$(3.20) \quad P(z_i = j | z_{-i}, w) \propto \hat{\phi}_j^{w_i} \frac{n_{-i,j}^d + \alpha}{n_{-i,\cdot}^d + T\alpha}.$$

Subsequently, the MAP estimates  $\hat{\phi}$  and  $\hat{\theta}$  are updated using the expected assignments of words to topics in  $d$ . This process of assignment of incoming topics and update of the MAP estimates of the parameters is continued till the end of the document stream. Note that this is not the true online Bayesian version of the LDA algorithm, since it does not update the posterior distribution over the parameters  $\phi$  and  $\theta$  – instead, it works with their MAP estimates. Nonetheless, the incremental LDA algorithm is efficient, since the topic assignments and parameter updates with every new incoming document depends only on the accumulated counts and the words in the current document.

**3.4 Hybrid Scheme** In the different motivating examples outlined in Section 1, online topic modeling is necessary for real-time topic analysis of an incoming document in the data stream. But at the same time it may be required to run offline topic models intermittently on the repository where the incoming data is stored, in order to get robust statistics of the overall topic model (and hence better clustering) from collective inference over a large text corpus. As a result, what we need in such applications is a hybrid topic modeling scheme that alternates between two phases:

- **STREAM** phase: run an online topic algorithm on streaming data;
- **OFFLINE** phase: run a batch algorithm on offline repository data.

A hybrid algorithm can operate on different schedules of alternation between the **STREAM** and **OFFLINE**

phases. Similar schemes have been used successfully in clustering evolving data streams [2]. In our experiments, we consider a schedule where we run the **STREAM** phase over a fixed number of elements in the text stream, followed by one iteration of the **OFFLINE** phase. This would be useful in settings where the data load is high during peak time periods (e.g., the daytime) and low during off-peak periods (e.g., at night). We study the vMF model in the hybrid scheme, using online vMF in the **STREAM** phase and the batch vMF in the **OFFLINE** phase.

## 4 Experiments

We performed experiments on several large real-world text datasets, in order compare the performances of the algorithms discussed in this paper. This section outlines the details of the datasets and evaluation measures, describes the experimental methodology, and finally discusses the results of the experiments.

**4.1 Datasets** We used the *20 Newsgroups* collection and four subsets derived from it.<sup>3</sup> The *news-20* collection has about 20,000 messages harvested from 20 different Usenet newsgroups, with about 1000 messages from each newsgroup. From the original dataset, four reduced datasets were created:

- **subset-20**: consists of 100 documents sampled randomly from each of the 20 newsgroups;
- **sim-3**: consists of 3 newsgroups on similar topics (comp.graphics, comp.os.ms-windows, comp.windows.x) with significant overlap between clusters due to cross-posting;
- **rel-3**: consists of 3 newsgroups on related topics (talk.politics.misc, talk.politics.guns, talk.politics.mideast);
- **diff-3**: consists of articles posted in 3 newsgroups that cover different topics (alt.atheism, rec.sport.baseball, sci.space) with well-separated clusters.

Since the overlap between topics in **sim-3** and **rel-3** is significant, they are more challenging datasets to cluster than **diff-3**. **subset-20** is a more difficult datasets than **news-20**, since clustering lesser number of documents embedded in a high-dimensional space is generally a more difficult task. So, we see that the different datasets considered for the experiments cover a wide spectrum, in terms of data set size and difficulty of clustering.

<sup>3</sup><http://people.csail.mit.edu/jrennie/20Newsgroups/>

We also harvested news articles from the Slashdot website<sup>4</sup> and created 2 datasets. For each category in these datasets, we collected 1000 articles primarily tagged with the category label, and then removed articles that were posted to multiple categories.

- **slash-7** contains 6714 news articles posted to 7 Slashdot categories: Business, Education, Entertainment, Games, Music, Science and Internet.
- **slash-6** contains 5182 articles posted to the 6 categories: Biotech, Microsoft, Privacy, Google, Security, Space.

All the datasets used the bag-of-words representation with word-level features, and were pre-processed using stop-word removal, TFIDF weighting (for vMF only, since LDA and DCM can handle only counts), removal of very high-frequency and low-frequency words, etc., following the methodology of Dhillon et al. [15]. The size and dimensionality of the datasets after pre-processing is listed in Table 2.

| Dataset   | Size  | Dimensions | Clusters |
|-----------|-------|------------|----------|
| news-20   | 19941 | 25936      | 20       |
| subset-20 | 1997  | 13341      | 20       |
| rel-3     | 2996  | 10091      | 3        |
| sim-3     | 2980  | 5950       | 3        |
| diff-3    | 2995  | 7670       | 3        |
| slash-7   | 6714  | 5769       | 7        |
| slash-6   | 5182  | 4498       | 6        |

Table 2: Datasets used in experiments.

**4.2 Evaluation** The following evaluation measures were used in the experiments:

- *Cluster quality*: We used normalized mutual information (nMI) as our quantitative evaluation measure of the final clustering output by the topic modeling algorithms. nMI is an external clustering validation metric that estimates the quality of the clustering with respect to a given underlying category labeling of the data: it measures how closely the clustering algorithm could reconstruct the underlying label distribution in the data [42, 17]. If  $C$  is the random variable denoting the cluster assignments of the points and  $K$  is the random variable denoting the underlying class labels on the points, then the nMI measure is defined as:

$$(4.21) \quad nMI = \frac{I(C; K)}{\sqrt{H(C) \times H(K)}},$$

where  $I(X; Y) = H(X) - H(X|Y)$  is the mutual information between the random variables  $X$  and  $Y$ ,  $H(X)$  is the Shannon entropy of  $X$ , and  $H(X|Y)$  is the conditional entropy of  $X$  given  $Y$  [13]. nMI effectively measures the amount of statistical information shared by the random variables representing the cluster assignments and the category labels of the data points.

- *Time*: For the batch algorithms, we measured the time taken to converge to the final clustering solution. In the online case, we report the average time to cluster each incoming document, which is the natural performance measure for an online algorithm that incrementally processes incoming streaming data.

Apart from this, we show sample topics output by these algorithms, for illustration purposes. We also show the confusion matrix from the clustering, which shows how the data points, assigned to different underlying labels, are actually distributed across the various clusters.

**4.3 Results** We performed the following three experiments. For each experiment, the category labels were removed from the datasets before clustering. After clustering, the nMI of each cluster partitioning was calculated with respect to the known underlying categorization. Time was calculated as the total system time in seconds (without I/O), and the experiments were run on a 1.7GHz-CPU 2MB-RAM machine using the Matlab platform.

**4.3.1 Experiment 1** This compares the performance of the three batch algorithms – LDA, EDCM and vMF – on the 7 datasets. Table 3 shows the nMI and run time results averaged across 5 runs, where NMF has the highest nMI accuracy and lowest run time for most of the datasets. LDA and EDCM outperformed vMF on **subset-20** and **news-20** respectively, but in those cases vMF achieved close to the best nMI values in a fraction of the time.

Tables 4 and 5 show the top dozen highest weighted words in clusters obtained by vMF on different datasets, illustrating that vMF is capable of finding good topics. Table 6 shows two sample confusion matrices representing the results of batch vMF, showing that vMF gets coherent clusters.

**4.3.2 Experiment 2** This experiment compares the online algorithms – o-LDA, o-EDCM and o-vMF – on the 7 datasets. The nMI and time results shown in Table 7 are averaged across 5 epochs. In each epoch, we

<sup>4</sup><http://www.slashdot.org>

| Dataset   | nMI         |             |             | Run Time (sec) |      |     |
|-----------|-------------|-------------|-------------|----------------|------|-----|
|           | vMF         | EDCM        | LDA         | vMF            | EDCM | LDA |
| news-20   | 0.51        | <b>0.54</b> | 0.53        | 204            | 934  | 352 |
| subset-20 | 0.41        | 0.36        | <b>0.43</b> | 14             | 25   | 34  |
| sim-3     | <b>0.27</b> | 0.12        | 0.11        | 2              | 4    | 15  |
| rel-3     | <b>0.38</b> | 0.30        | 0.28        | 3              | 9    | 17  |
| diff-3    | <b>0.82</b> | 0.81        | 0.74        | 1              | 7    | 16  |
| slash-7   | <b>0.39</b> | 0.22        | 0.31        | 15             | 40   | 47  |
| slash-6   | <b>0.65</b> | 0.36        | 0.46        | 6              | 26   | 36  |

Table 3: Performance of batch LDA, vMF and EDCM algorithms, w.r.t. nMI of final clustering result and time to convergence of the algorithms, averaged over 5 runs. Best nMI for every dataset is highlighted.

|           |         |             |            |           |
|-----------|---------|-------------|------------|-----------|
| music     | web     | scientists  | internet   | games     |
| apple     | google  | nasa        | broadband  | gaming    |
| itunes    | search  | space       | domain     | game      |
| riaa      | yahoo   | researchers | net        | nintendo  |
| ipod      | site    | science     | network    | sony      |
| wikipedia | online  | years       | verisign   | xbox      |
| digital   | sites   | earth       | bittorrent | gamers    |
| napster   | ebay    | found       | icann      | wii       |
| file      | amazon  | brain       | service    | console   |
| drm       | engine  | university  | access     | video     |
| songs     | users   | human       | voip       | article   |
| industry  | browser | research    | dns        | microsoft |

Table 4: Five of the topics obtained by running batch vMF on slash-6.

took a random permutation of the documents in the data and streamed it through the online algorithms. o-vMF substantially outperforms o-LDA and o-EDCM for all datasets, and for some datasets, e.g., news-20, subset-20, online vMF even outperforms batch vMF. This is an interesting phenomenon that has been observed in previous work [6]. In general, the online algorithms give worse nMI results than the corresponding batch algorithms, which is expected since the online algorithms can only update the cluster statistics incrementally. The incremental o-LDA algorithm was given the first 5% of the data stream to perform the initial windowed Gibbs update. o-LDA is very fast but does not give comparable results to o-vMF. The windowed online EDCM algorithm takes an order of magnitude more time than o-LDA but gives comparable results.

**4.3.3 Experiment 3** Since vMF had overall the best performance in Experiments 1 and 2, in this experiment we evaluate the performance of h-vMF, the hybrid scheme for vMF. h-vMF was run on the news datasets 5 times – in each run, we took a random permutation of the documents in the data and streamed it through

|         |           |          |              |           |
|---------|-----------|----------|--------------|-----------|
| windows | turkish   | game     | god          | israeli   |
| dos     | armenian  | team     | bible        | israel    |
| files   | armenia   | games    | christian    | moral     |
| file    | genocide  | hockey   | jesus        | arabs     |
| disk    | turkey    | year     | church       | arab      |
| drive   | radar     | play     | christians   | absolute  |
| port    | armenians | season   | atheism      | killed    |
| program | soviet    | baseball | religion     | morality  |
| irq     | list      | pens     | people       | lebanon   |
| ftp     | turks     | players  | faith        | lebanese  |
| modem   | detector  | league   | life         | people    |
| ibm     | people    | player   | christianity | civilians |

Table 5: Five of the topics obtained by running batch vMF on subset-20.

| diff-3     |            |            | slash-6    |            |            |            |            |             |
|------------|------------|------------|------------|------------|------------|------------|------------|-------------|
| 10         | 28         | <b>932</b> | 9          | 0          | 3          | 1          | 10         | <b>1178</b> |
| 11         | <b>958</b> | 54         | 11         | 8          | 40         | 31         | <b>753</b> | 6           |
| <b>976</b> | 13         | 13         | 22         | 1          | 15         | <b>580</b> | 309        | 17          |
|            |            |            | 2          | 0          | <b>808</b> | 10         | 52         | 0           |
|            |            |            | 2          | <b>520</b> | 44         | 16         | 13         | 2           |
|            |            |            | <b>505</b> | 0          | 8          | 49         | 138        | 19          |

Table 6: Confusion matrices after running batch vMF on diff-3 and slash-6. Rows correspond to cluster labels, columns to class labels, entry(i, j) is the number of points in cluster i with class label j. A diagonally dominant confusion matrix  $\Rightarrow$  cluster partitioning corresponds well to the underlying class labeling.

h-vMF. The STREAM phase was applied on the data stream, and after every 20% of the stream, one iteration of the OFFLINE phase was run. Table 8 shows the final nMI results and the average run time across the different OFFLINE phases, averaged over 5 runs. As expected, the hybrid algorithm is faster than the batch vMF and gives better performance than the online vMF algorithms, thereby giving a good tradeoff between speed and accuracy.

Figure 4 shows how the nMI values improves with increasing fraction of the dataset being processed by h-vMF. As expected, there are sharp jumps in the plot at intermittent points along the data stream – these are where h-vMF switched to the OFFLINE phase from the STREAM phase and improved the clustering, validating our claim that intermittent batch processing improves the clustering performance. Note that on more difficult datasets, e.g., sim-3, the STREAM phase can accumulate errors along the way, as previously noted by [2] – running intermittent OFFLINE phases can correct these errors and improve the overall performance.

| Dataset   | nMI         |        |       | Time per doc (sec) |        |       |
|-----------|-------------|--------|-------|--------------------|--------|-------|
|           | o-vMF       | o-EDCM | o-LDA | o-vMF              | o-EDCM | o-LDA |
| news-20   | <b>0.54</b> | 0.39   | 0.30  | 0.011              | 0.565  | 0.010 |
| subset-20 | <b>0.42</b> | 0.27   | 0.29  | 0.032              | 0.361  | 0.041 |
| sim-3     | <b>0.17</b> | 0.09   | 0.08  | 0.014              | 0.053  | 0.011 |
| rel-3     | <b>0.31</b> | 0.16   | 0.19  | 0.019              | 0.092  | 0.012 |
| diff-3    | <b>0.72</b> | 0.62   | 0.60  | 0.009              | 0.061  | 0.008 |
| slash-7   | <b>0.34</b> | 0.16   | 0.12  | 0.007              | 0.048  | 0.006 |
| slash-6   | <b>0.54</b> | 0.25   | 0.30  | 0.005              | 0.035  | 0.004 |

Table 7: Performance of online LDA, vMF and EDCM algorithms, w.r.t. nMI of final clustering result and average time to process one incoming document, averaged over 5 epochs. Best nMI for every dataset is highlighted.

| Dataset   | Time (sec) |     | nMI   |       |      |
|-----------|------------|-----|-------|-------|------|
|           | h-vMF      | vMF | o-vMF | h-vMF | vMF  |
| news-20   | 17.8       | 204 | 0.54  | 0.51  | 0.51 |
| subset-20 | 4.6        | 14  | 0.42  | 0.42  | 0.41 |
| sim-3     | 1.3        | 2   | 0.17  | 0.21  | 0.27 |
| rel-3     | 1.7        | 3   | 0.31  | 0.33  | 0.38 |
| diff-3    | 1.1        | 1   | 0.72  | 0.81  | 0.82 |
| slash-7   | 6.2        | 15  | 0.34  | 0.37  | 0.39 |
| slash-6   | 2.1        | 6   | 0.54  | 0.60  | 0.65 |

Table 8: Performance of h-vMF. Time to process the data stream is less than the batch algorithm, while the performance is in between the batch and the online algorithms. The time and nMI values of vMF and o-vMF are the same as previous tables.

## 5 Related Work

Our work is related to two different existing research directions: unsupervised models for text analysis, and online/streaming models for data analysis. Significant work has been done on both the directions, especially over the past few years. In this section, we briefly review some of the existing work and how they relate to our current work.

As discussed in Section 2, currently there are several popular generative models for unsupervised text analysis, which can be categorized into broad groups based on the assumptions made about the prior as well as the conditional distributions. The first category of models used point estimates for both priors and conditionals. The two most popular examples are the naive-Bayes with multinomials [36] and the mixture of von Mises-Fisher (vMF) models [5]. Experimental comparisons have shown that the basic vMF model tends to perform marginally better than the basic naive-Bayes model [49], although the performance of naive-Bayes can be improved using annealing [18] and other heuristics [39]. The second category of models use a point estimate for the prior, but a Bayesian model for the conditional. The

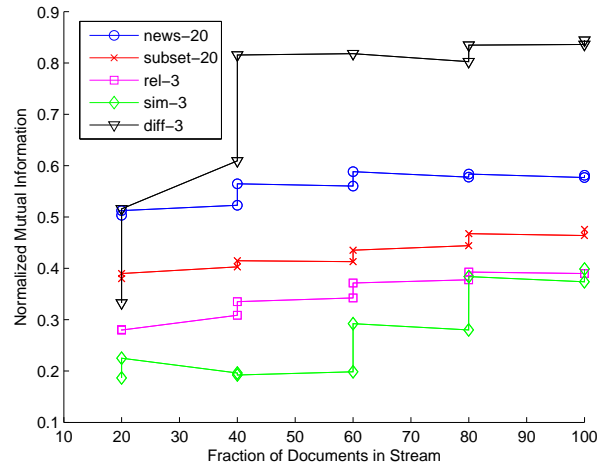


Figure 4: nMI results of h-vMF algorithm on the news datasets, along increasing fractions of the data stream. The STREAM phase is run on the data incrementally, with one iteration of the OFFLINE phase at every 20% of the stream.

most prominent of such models is the Dirichlet mixture (DM) model [45, 46], also known as the Dirichlet compound multinomial (DCM) model [29, 18]. Learning of both the first two categories of models use a variant of the EM algorithm, where approximations [5, 18] and/or fast iterative algorithms [32] are used for the parameter learning step. Further, both categories of models assume all documents in a corpus to have a single fixed distribution over topics. The third category of models relaxes this assumption, and allows each document in a corpus to have a different topic distributions. While probabilistic latent semantic indexing (PLSI) [22] was one of the first models in this category, latent Dirichlet allocation (LDA) [10] as well as its full Bayesian variants [21] have become significantly more popular over time. Such approaches use a non-parametric Bayesian modeling of the prior and a full Bayesian model of the conditional [21]. Extensions to such models to include author, role, communities, as well as sub-topics have been studied in the recent years [44, 26, 50].

Extensive research on analysis of data streams has been done in the database and data mining communities. One important part of the research has focussed on the core data management issues for data streams, including extensions of query languages and data models to handle streams [4, 25]. A large part of the research has been motivated by specific problems and applications, including novelty detection [1, 28, 51, 19, 47], frequent pattern mining [12, 30], as well as cluster-

ing [38, 24, 2, 37, 6]. Classification, regression and related learning methods have been independently studied by the database/data mining communities [43, 23] as well as the machine learning community [27, 11].

Although unsupervised online clustering and its applications to text analysis is becoming increasingly important, except for a few important ideas, the domain seems to be largely unexplored. One of the important earlier ideas on clustering evolving data streams suggested using a hybrid online-offline strategy, rather than a one pass algorithm, based on practical considerations [2]. It was argued that a simple one pass algorithm may carry forward unimportant past information, and may miss important information on how the clusters are evolving without looking at time windows. Further, as we note in our current work, offline aggregation after a phase of fast online updates may be desirable for accuracy reasons [2]. Some of the other important ideas include a class of online clustering algorithms on data streams with performance guarantees [37], and extensions to kmeans to work on binary data streams [38].

The literature on online models for unsupervised text analysis and topic modeling is surprisingly small, which was part of the motivation for our present study. One of the earliest ideas was an extension of a variant of the spherical kmeans algorithm [15] to work on text streams [6], and is an example of an online extension of the first category of text models we discussed earlier. Online extensions of the text models of the second and third category have also happened over the past year. The online extension of the Dirichlet mixture model proposes to use a multinomial particle filter [33]. A recently proposed dynamic extension of the original latent dirichlet allocation (LDA) model uses ideas from linear dynamical systems, and have given encouraging results based on variational inference [9]. Another recent paper proposes an online extension of the full Bayesian version of LDA [41], which is the incremental LDA algorithm used in this paper.

## 6 Conclusion

This paper first compares the performance of three popular topic models – LDA, vMF, EDCM. It empirically demonstrates, via thorough experiments, that vMF provides the best overall performance for document categorization in batch processing, discovering coherent underlying topics in the process. It also presents a new online algorithm for vMF, which outperforms corresponding online versions of LDA and EDCM. Finally, it proposes a practical hybrid scheme for topic modeling, which gives a good tradeoff of performance and efficiency for processing streaming text.

In future work, we would like to investigate other

hybrid topic model schemes, which can do a load-based switch between the online and batch algorithms, depending on the rate of incoming documents in the text stream. We would also like to incorporate model-selection into the online algorithms – this would enable the online component to detect new topics in the STREAM phase, and the batch algorithm to get better statistics for the newly discovered topic clusters in the OFFLINE phase. On the theoretical side, we would like to study the loss bounds for the online algorithms outlined in this paper in more detail, and compare the theoretical bounds to the empirical results.

## References

- [1] C. C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proceedings of the ACM International Conference on Management of Data*, 2003.
- [2] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In *Proceedings of the International Conference on Very Large Data Bases*, 2003.
- [3] K. S. Azoury and M. K. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.
- [4] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proceedings of the ACM Symposium on Principles of Database Systems*, 2002.
- [5] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6:1345–1382, 2005.
- [6] A. Banerjee and J. Ghosh. Frequency sensitive competitive learning for balanced clustering on high-dimensional hyperspheres. *IEEE Transactions on Neural Networks*, 15(3):702–719, May 2004.
- [7] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.
- [8] J. Bilmes. A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden markov models. Technical Report ICSI-TR-97-02, University of Berkeley, 1997.
- [9] D. Blei and J. Lafferty. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.
- [10] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [11] N. Cesa-Bianchi, Y. Freund, D. P. Helmbold, D. Haussler, R. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.
- [12] G. Cormode and S. Muthukrishnan. What’s hot and what’s not: Tracking most frequent items dynamically. In *Proceedings of the ACM Symposium on Principles of Database Systems*, 2003.
- [13] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.
- [14] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm.

- Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.
- [15] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, 2001.
- [16] I. S. Dhillon and S. Sra. Modeling data using directional distributions. Technical Report TR-03-06, Department of Computer Sciences, University of Texas at Austin, Austin, TX, 2003.
- [17] B. E. Dom. An information-theoretic external cluster- validity measure. Technical Report RJ 10219, IBM Research Report, 2001.
- [18] C. Elkan. Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.
- [19] W. Fan, Y. Huang, H. Wang, and P. Yu. Active mining of data streams. In *Proceedings of the 4th SIAM International Conference on Data Mining*, 2004.
- [20] W. B. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms*. Prentice Hall, 1992.
- [21] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Science*, 101(1):5228–5225, 2004.
- [22] T. Hoffman. Probabilistic latent semantic indexing. In *Proceedings of the 15th Conference in Uncertainty in Artificial Intelligence*, 1999.
- [23] J. Z. Kolter and M. A. Maloof. Dynamic weighted majority: A new ensemble method for tracking concept drift. In *Proceedings of the IEEE International Conference on Data Mining*, 2003.
- [24] F. Korn, S. Muthukrishnan, and D. Srivastava. Reverse nearest neighbor aggregates over data streams. In *Proceedings of the International Conference on Very Large Data Bases*, 2002.
- [25] Y. Law, H. Wang, and C. Zaniolo. Query languages and data models for database sequences and data streams. In *Proceedings of the International Conference on Very Large Data Bases*, 2004.
- [26] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.
- [27] N. Littlestone and M. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.
- [28] J. Ma and S. Perkins. Online novelty detection on temporal sequences. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.
- [29] R. Madsen, D. Kauchak, and C. Elkan. Modeling word burstiness using the Dirichlet distribution. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.
- [30] G. S. Manku and R. Motawani. Approximate frequency counts over data streams. In *Proceedings of the International Conference on Very Large Data Bases*, 2002.
- [31] K. V. Mardia and P. Jupp. *Directional Statistics*. John Wiley and Sons Ltd., 2nd edition, 2000.
- [32] T. Minka. Estimating a Dirichlet distribution, 2003.
- [33] D. Mochishashi and Y. Matsumoto. Context as filtering. In *Proceedings of the 18th Annual Conference on Neural Information Processing Systems*, 2006.
- [34] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, 1998.
- [35] A. Ng and M. Jordan. On discriminative vs generative classifiers: A comparison of logistic regression and naive Bayes. In *Proceedings of the 14th Annual Conference on Neural Information Processing Systems*, 2001.
- [36] K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.
- [37] L. O’Callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motawani. Streaming-data algorithms for high-quality clustering. In *Proceedings of the IEEE International Conference on Data Engineering*, 2001.
- [38] C. Ordonez. Clustering binary data streams with kmeans. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2003.
- [39] J. Rennie, L. Shih, J. Teevan, and D. Karger. Tackling the poor assumptions of naive bayes classifiers. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.
- [40] Gerard Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley (Reading MA), 1989.
- [41] X. Song, C.-Y. Lin, B. L. Tseng, and M.-T. Sun. Modeling and predicting personal information dissemination behavior. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.
- [42] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Proc 7th Natl Conf on Artificial Intelligence : Workshop of AI for Web Search (AAAI 2000)*, pages 58–64. AAAI, July 2000.
- [43] H. Wang, W. Fan, P. S. Yu, and J. Han. Mining concept drifting data streams using ensemble classifiers. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.
- [44] X. Wang, N. Mohanty, and A. McCallum. Groups and topic discovery from relations and text. In *KDD Workshop on Link Discovery: Issues, Approaches and Applications (LinkKDD)*, 2005.
- [45] M. Yamamoto and K. Sadamitsu. Dirichlet mixtures in text modeling. Technical Report CS-TR-05-1, University of Tsukuba, 2005.
- [46] M. Yamamoto, K. Sadamitsu, and T. Mishina. Context modeling using Dirichlet mixtures and its application to language models. In *Information Processing Society of Japan-SIGSLP*, volume 104, pages 29–34, 2003. In Japanese.
- [47] J. Zhang, Z. Ghahramani, and Y. Yang. A probabilistic model for online document clustering with application to novelty detection. In *Proceedings of the 17th Annual Conference on Neural Information Processing Systems*, 2004.
- [48] S. Zhong. Efficient streaming text clustering. *Neural Networks*, 18:790–798, August 2005.
- [49] S. Zhong and J. Ghosh. A comparative study of generative models for document clustering. In *Workshop on Clustering High Dimensional Data : Third SIAM Conference on Data Mining*, April 2003.
- [50] D. Zhou, E. Manavoglu, J. Li, C. Lee Giles, and H. Zha. Probabilistic models for discovering e-communities. In *Proceedings of the 15th International World Wide Web Conference*, 2006.
- [51] Y. Zhu and D. Shasha. Efficient elastic burst detection in data streams, 2003.