Infinite Mixture Modeling of Streaming Data with Online Bayesian Moment Matching

Wilson Hsu

June 2015

Abstract

We investigate the problem of mixture modeling in the context of streaming data, where the number of components is not known a priori and might be infinite. The basic finite mixture model is extended to allow the number of components to vary, and we propose an online Bayesian moment matching algorithm to estimate the posterior distribution. The proposed method is tested on three types of mixtures: mixture of Gaussians, multinomial naive Bayes, and latent Dirichlet allocation.

1 Introduction

Mixture model is a probabilistic model commonly used in unsupervised learning and clustering. One of its drawbacks is that the number of components needs to be fixed before data are observed. Traditionally, setting the number of components is treated as a model selection problem. One solution is to train a model multiple times, each time with a different number of components, and choose the number that minimizes some cost function on a heldout test set.

More recently nonparametric Bayesian models have gained popularity as a mathematically elegant way to bypass the model selection problem. In particular, the Dirichlet process mixture model [1] can be used to model a mixture when the number of components is unknown.

However, nonparametric Bayesian methods can be difficult to understand for practitioners without an advanced mathematics background. We propose a more intuitive model that is a simple extension to the basic finite mixture model. We also describe an online Bayesian moment matching method to infer the posterior distribution after each observation from a stream of data.

2 Model

In a finite mixture model, the data are assumed to come from K mixture components, with the kth component having a probability distribution specified by parameter ϕ_k. The generative process starts by first sampling from the mixing proportion $\theta = (\theta_1, \theta_2, \dots, \theta_K)$ to obtain the component indicator $z \in \{1, 2, \dots, K\}$. The observation x is then sampled from the distribution of the zth component $p(x|\phi_z)$.

In a Bayesian setting, we further assume that the parameters θ and ϕ_k are also random variables drawn from distributions with hyperparameters ν and η, respectively. This generative process is depicted in Figure 1a.

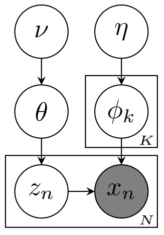
The distributions over the parameters are typically chosen to be the conjugate prior distributions. In the case of mixing proportion θ, the conjugate prior is the Dirichlet distribution:

$$p(\theta|\nu) = \mathrm{Dir}(\theta; \nu)$$

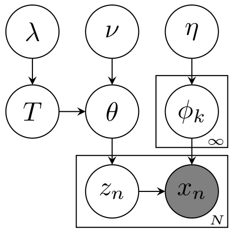
We extend the finite mixture model by introducing a new parameter T which models the number of components. The proposed model is shown in Figure 1b.

The parameter T is generated according to a Poisson distribution. To make sure T is positive, we shift the Poisson distribution by one:

$$p(T|\lambda) = \mathrm{Poi}(T - 1; \lambda - 1)$$



(a)



(b)

Figure 1: Graphical representations of (a) a finite mixture model with fixed number of topics and (b) the proposed extension which models unknown and possibly infinite number of components.

The mixing proportion θ is now assumed to be sampled from the conditional distribution

$$p(\theta|\nu, T) = \mathrm{Dir}(\theta; \nu, T)$$

where $\mathrm{Dir}(\theta; \nu, T)$ denotes a degenerate Dirichlet distribution $\mathrm{Dir}(\theta; \nu')$ with

$$\nu'_k = \begin{cases} \nu_k & \text{for } k \leq T \\ 0 & \text{for } k > T \end{cases}$$

In effect we are assuming θ and ν to be infinite-dimensional, but only a finite number of θ_k's will be positive. By varying the value of T we can model mixtures of any number of component.

3 Online inference by Bayesian moment matching

In this section we describe an online posterior inference algorithm using Bayesian moment matching (BMM) to handle streaming data.

Let $\Phi = \{\phi_k\}$ be the set of parameters of all components, and let P_n denote the posterior distribution after seeing the first n observations $x_{1:n}$. Then

$$\begin{aligned} P_n(\theta, \Phi, T) &= p(\theta, \Phi, T|x_{1:n}) \\ &= \frac{1}{c_n} \sum_{z_n=1}^{T} p(\theta, \Phi, T, z_n, x_n|x_{1:n-1}) \\ &= \frac{1}{c_n} \sum_{z_n=1}^{T} p(z_n|\theta) p(x_n|\Phi, z_n) p(\theta, \Phi, T|x_{1:n-1}) \\ &= \frac{1}{c_n} \sum_{z_n=1}^{T} \theta_{z_n} p(x_n|\phi_{z_n}) P_{n-1}(\theta, \Phi, T) \end{aligned} \tag{1}$$

where $c_n = P(x_n|x_{1:n-1})$.

Equation 1 can be viewed as an update equation, where the posterior at the $(n-1)$th step is used as the prior in the nth step to compute the posterior after the nth observation. Therefore, starting from an original prior distribution $P_0(\theta, \Phi, T)$, we can successively update the posterior after each observation.

Nonetheless, exact update according to Equation 1 is intractable since the number of terms grows exponentially with respect to the number of observations. Instead, we will approximate $P_n(\theta, \Phi, T)$ with a different distribution $q_n(\theta, \Phi, T)$, whose parameters are estimated by matching its sufficient moments with the moments of P_n in Equation 1. Algorithm 1 depicts the general, high-level process of online Bayesian moment matching.

3.1 Approximate prior

First, to facilitate inference, we will assume the prior $P_{n-1}(\Theta, \Phi, T)$ takes the form shown in Figure 2a, which is the same as the original model except that the component distributions are now factorized. Therefore,

Algorithm 1 Online Bayesian moment matching algorithm

```
1: Initialize prior distribution $P_0(\theta, \Phi, T)$.
2: for $n = 1, \dots, N$ do
3:     Read the $n$-th observation $x_n$.
4:     Compute moments of $P_n(\theta, \Phi, T)$ by taking expectations with respect to Equation 1.
5:     Compute parameters of $q_n(\theta, \Phi, T)$ by assuming it has the same moments as $P_n$.
6:     Set $P_n(\theta, \Phi, T) = q_n(\theta, \Phi, T)$.
7: end for
```

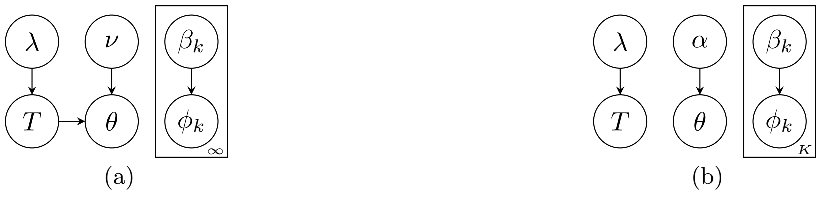


Figure 2: (a) The prior distribution P_{n-1} and (b) the approximate posterior distribution q_n.

the prior has the form

$$P_{n-1}(\Theta, \Phi, T) = \text{Poi}(T-1; \lambda - 1)\text{Dir}(\theta; \nu, T) \prod_{k=1}^{\infty} p(\phi_k|\beta_k). \tag{2}$$

3.2 Approximate posterior

As shown in Figure 2b, we use a fully factorized, finite-dimensional distribution q to approximate the posterior:

$$q_n(\theta, \Phi, T) = q_n(T|\lambda) q_n(\theta|\alpha) \prod_{k=1}^{K} q_n(\phi_k|\beta_k),$$

where[1]

$$q_n(T|\lambda) = \text{Poi}(T-1; \lambda - 1)$$

and

$$q_n(\theta|\alpha) = \text{Dir}(\theta; \alpha).$$

The specific form of $q_n(\phi_k|\beta_k)$ depends on the type of mixture model being considered. We discuss three examples in Section 4.

3.3 Moment matching for $q_n(T)$

A Poisson distribution can be completely determined given its mean; therefore, we can estimate $q_n(T|\lambda)$ by computing the expected value $E_{P_n}[T]$. First, we can compute the marginal posterior over T by integrating out θ and Φ in Equation 1:

$$\begin{aligned} P_n(T) &= \int \int P_n(\theta, \Phi, T)\ \mathrm{d}\theta\ \mathrm{d}\Phi \\ &= \frac{1}{c_n} \frac{\sum_{z_n=1}^{T} \nu_{z_n} b_{z_n}(x_n)}{\sum_{k=1}^{T} \nu_k} \text{Poi}(T-1; \lambda - 1), \end{aligned}$$

where

$$b_z(x) = \int p(x|\phi_z) p(\phi_z|\beta_z)\ \mathrm{d}\phi_z.$$

[1]We are overloading the notation q_n and will rely on the arguments to the function to determine the particular distribution being discussed.

Note that we only need to determine $b_z(x)$ up to a constant factor with respect to z, since the same factor is also present in c_n and will thus cancel out.

We can then estimate $q_n(T|\lambda)$ by importance sampling as follows:

1. Draw M samples $T_s \sim \text{Poi}(T-1; \lambda - 1), s = 1, \ldots, M$.
2. Compute $E_{P_n}[T] \approx \frac{\sum_s h(T_s)T_s}{\sum_s h(T_s)}$, where $h(T) = \frac{\sum_{z_n=1}^{T} \nu_{z_n} b_{z_n}(x_n)}{\sum_{k=1}^{T} \nu_k}$.
3. update $\lambda \leftarrow E_{P_n}[T]$.

Furthermore, using the samples T_s we can approximate $P_{n-1}(T)$ as the fraction of samples with value T. We denote these estimates as $\tilde{P}_{n-1}(T)$, i.e.,

$$\tilde{P}_{n-1}(T) = \frac{1}{M} \sum_{s=1}^{M} \mathbf{1}(T_s = T)$$

where $\mathbf{1}(\cdot)$ returns 1 if the argument is true and 0 otherwise.

We will use $\tilde{P}_{n-1}(T)$ in place of $P_{n-1}(T)$ in Equation 1 when we estimate the other parameters. Let $K = \max_s T_s$, then we have $\tilde{P}_{n-1}(T) = 0$ for $T > K$; therefore, this allows us to marginalize out T with only a finite sum.

3.4 Moment matching for $q_n(\theta)$

For a K-dimensional Dirichlet distribution $q_n(\theta|\alpha) = \text{Dir}(\theta_1, \ldots, \theta_K; \alpha_1, \ldots, \alpha_K)$, we can uniquely solve for the parameters $\alpha_1, \ldots, \alpha_K$ if we have the K first-order moments, $E[\theta_1], \ldots, E[\theta_K]$[2], and one second-order moment, $E[\theta_j^2]$ for any $j \in \{1, \ldots, K\}$. Given the moments, we can determine the Dirichlet parameters as

$$\alpha_k = E[\theta_k] \frac{E[\theta_j] - E[\theta_j^2]}{E[\theta_j^2] - E[\theta_j]^2}$$

for $k = 1, \ldots, K$.

We thus need to compute the first and second order moments $E_{P_n}[\theta_k]$ and $E_{P_n}[\theta_k^2]$:

$$E_{P_n}[\theta_z] = \frac{1}{c_n} \alpha_z \sum_{T=z}^{K} \tilde{P}_{n-1}(T) \frac{h(T) S_{\nu,T} + b_z(x_n)}{S_{\nu,T}(S_{\nu,T} + 1)}$$

$$E_{P_n}[\theta_z^2] = \frac{1}{c_n} \alpha_z(\alpha_z + 1) \sum_{T=z}^{K} \tilde{P}_{n-1}(T) \frac{h(T) S_{\nu,T} + 2b_z(x_n)}{S_{\nu,T}(S_{\nu,T} + 1)(S_{\nu,T} + 2)}$$

where $S_{\nu,T} = \sum_{k=1}^{T} \nu_k$.

One thing to note is that the approximate posterior $q_n(\theta)$ does not have the same form as the prior $P_{n-1}(\theta)$; therefore, we cannot directly apply $q_n(\theta)$ in the next iteration as the prior for the $(n+1)$th observation. Here we introduce two heuristics to match the two distributions.

First, we use the Dirichlet parameters α of $q_n(\theta)$ as the degenerate Dirichlet parameters ν of the prior distribution, i.e., we set $\nu = \alpha$. However, ν would then be truncated since α only has finite dimension. We thus apply the second heuristic, which assigns $\nu_t = \min_k \alpha_k$ for $t > K$. This way we keep a small probability for growing the number of topics for future observations.

[2]Only $K - 1$ of these are linearly independent since $\sum_{k=1}^{K} E[\theta_k] = 1$.

3.5 Estimating $p(z_n|x_{1:n})$

If we marginalize out the parameters θ, Φ, and T in the joint posterior, we obtain the posterior probability of the component indicator z_n:

$$\begin{aligned} p(z_n|x_{1:n}) &= \frac{1}{c_n} \sum_{T=1}^{\infty} \int \int p(\theta, \Phi, T, z_n, x_n | x_{1:n-1}) \, \mathrm{d}\theta \, \mathrm{d}\Phi \\ &\approx \frac{1}{c_n} \nu_{z_n} b_{z_n}(x_n) \sum_{T=z_n}^{K} \frac{\tilde{P}_{n-1}(T)}{\sum_{k=1}^{T} \nu_k} \end{aligned}$$

4 Example distributions

Everything we have considered so far does not depend on the specific distribution used by the mixture components and can therefore be implemented as a framework into which we plug specific mixture distributions we need. In order to use a new mixture distribution, we simply have to specify how to compute $b_z(x)$ and how to update the posterior hyperparameters β_z after each observation. We now consider three specific examples of mixture models and show how this can be done.

4.1 One-dimensional GMM with equal and known variance

We start with a simple one-dimensional Gaussian mixture model to illustrate the idea. The Gaussian of each component is assumed to have equal variance τ^2 which is known. In this case, the component parameter ϕ_z is the mean of the Gaussian:

$$p(x|\phi_z) = \mathcal{N}\left(x; \phi_z, \tau^2\right)$$

The conjugate prior over Gaussian mean is again a Gaussian, and the hyperparameters $\beta_z = (\mu_z, \sigma_z^2)$ are the mean and variance:

$$p(\phi_z|\beta_z) = \mathcal{N}\left(\phi_z; \mu_z, \sigma_z^2\right)$$

For this simple distribution we obtain the distributions we need as follows:

$$b_z(x) = \mathcal{N}\left(x; \mu_z, \sigma_z^2 + \tau^2\right)$$

and

$$P_n(\phi_z) = (1 - p_z)\mathcal{N}\left(\phi_z; \mu_z, \sigma_z^2\right) + p_z \mathcal{N}\left(\phi_z; \frac{\sigma_z^2 x + \tau^2 \mu_z}{\sigma_z^2 + \tau^2}, \frac{\sigma_z^2 \tau^2}{\sigma_z^2 + \tau^2}\right)$$

where $p_z = p(z_n = z|x_{1:n})$.

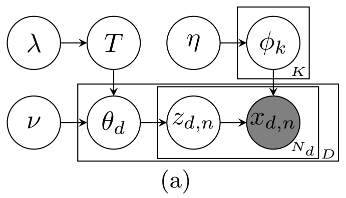
We choose the approximate posterior $q(\phi_z|\beta_z)$ to be Gaussian like the prior, and thus the posterior can be estimated by matching the mean and variance. We thus arrive at the following updates:

$$\begin{aligned} \mu_z &\leftarrow (1 - p_z)\mu_z + p_z \frac{\sigma_z^2 x + \tau^2 \mu_z}{\sigma_z^2 + \tau^2} \\ \sigma_z^2 &\leftarrow (1 - p_z)^2 \sigma_z^2 + p_z^2 \frac{\sigma_z^2 \tau^2}{\sigma_z^2 + \tau^2} \end{aligned}$$

for $z = 1, \ldots, K$.

4.2 Multinomial naive Bayes

We next consider the multinomial naive Bayes model used in document modeling. In this model, each mixture component has a multinomial distribution over words in the vocabulary and can be thought of as a topic. Each observation is a document of a certain topic, represented as a bag of words. Given the topic, the words are generated independently according to the multinomial distribution associated with the topic.



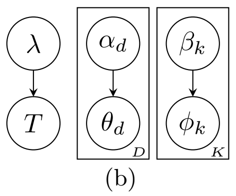


Figure 3: (a) LDA with varying number of topics and (b) the approximate posterior distribution.

Under this model, we can represent each observation x_n as a length-V vector of word counts, with V being the vocabulary size. Thus the vth element $x_{n,v}$ is the number of times the vth word in the vocabulary occurs in the nth document.

We can define the distributions associated with the zth component as follows:

$$p(x_n|\phi_z) = \prod_{v=1}^{V} \phi_{z,v}^{x_{n,v}}$$
$$p(\phi_z|\beta_z) = \mathrm{Dir}(\phi_z; \beta_z).$$

Here a Dirichlet prior is chosen since it is the conjugate prior of multinomial distribution.

As discussed in Section 3.4, we need the first-order moments as well as one second-order moment to estimate a Dirichlet distribution by moment matching. We can compute these moments as

$$E[\phi_{z,v}] = (1 - p_z)\frac{\beta_{z,v}}{\sum_{v'} \beta_{z,v'}} + p_z \frac{\beta_{z,v} + x_{n,v}}{\sum_{v'}(\beta_{z,v'} + x_{n,v'})}$$

and

$$E[\phi_{z,v}^2] = (1 - p_z)\frac{\beta_{z,v}}{\sum_{v'} \beta_{z,v'}} \frac{\beta_{z,v} + 1}{\sum_{v'} \beta_{z,v'} + 1} + p_z \frac{\beta_{z,v} + x_{n,v}}{\sum_{v'}(\beta_{z,v'} + x_{n,v'})} \frac{\beta_{z,v} + x_{n,v} + 1}{\sum_{v'}(\beta_{z,v'} + x_{n,v'}) + 1}.$$

Finally, we can compute $b_z(x)$ as the ratio

$$b_z(x) = \frac{\Delta(\beta_z + x)}{\Delta(\beta_z)}$$

where $\Delta(a) = \frac{\prod_i \Gamma(a_i)}{\Gamma(\sum_i a_i)}$ is the normalization constant of a Dirichlet distribution with parameter a.

Note that this value can become very small for larger documents. To avoid undesirable numerical effects, this should be implemented in log space. Furthermore, since we only need $b_z(x)$ up to a constant factor, $\max_z b_z(x)$ should be factored out before converting $b_z(x)$ back to linear scale.

4.3 Latent Dirichlet allocation

Latent Dirichlet allocation (LDA) [2] has an additional layer of hierarchy compared to a mixture model like naive Bayes. In LDA, each document has its own topic distribution, and different words within the same document can be sampled from different topics. However, if we look at only one document at a time, then LDA is essentially a mixture model with multinomial component distribution that always generates only one word for each observation.

Therefore, our model can be extended to LDA in a straightforward manner. Figure 3 shows how our extension applies to LDA as well as the approximate distribution used for posterior inference. The data can be seen as a stream of words, and the same moment matching technique can still be used except we do not update the posterior for θ across document boundaries. Instead, we restart with the original prior on θ when a new document arrives.

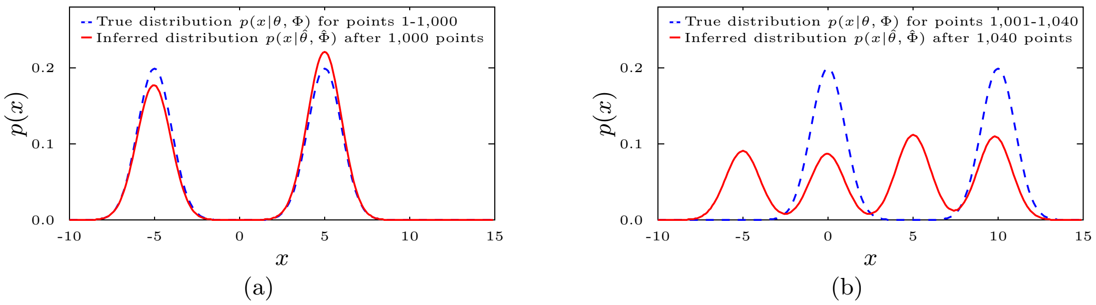


Figure 4: True and inferred data distributions with GMM

5 Experiments

5.1 One-dimensional GMM with equal and known variance

First we experimented with synthetic data generated from a one-dimensional GMM whose components all have a known variance equal to one. We initialized our model with $\lambda = 1.1$, so there was a priori a low probability of having more than one component. The Gaussian hyperparameters were initialized to $\mu_z = 0$ and $\sigma_z^2 = 1000$ for all z.

We generated 1,040 points, the first 1,000 of which came from two equally weighted Gaussians with means -5 and 5. After seeing the first 1,000 points, the algorithm successfully tracked the means of the true Gaussians, as shown in Figure 4a.

The next 40 points were generated by two equally weighted Gaussians as well but with means 0 and 10. After seeing these 40 points, the algorithm produced two more components centered around the true means as can be seen in Figure 4b.

One observation to note is that more recent points have more weight. With only 40 points, the new components have almost the same mixing proportions as the older ones that produced 1,000 points. Therefore, if the probability distribution changes midway through the data stream, components that are no longer producing data will quickly drop out, and the mixing proportions will be shifted towards the new components.

5.2 Multinomial naive Bayes

We tested the proposed method with naive Bayes model on two real text corpora, the Reuters-21578 dataset and the NIPS dataset. The Reuters corpus contains 21,578 Reuters news articles in 1987, and the NIPS dataset contains 2,742 articles from the NIPS conference for the years 1988–2004[3]. For both datasets the documents were passed through a stemmer, and a standard set of stopwords were removed. We also removed words appearing no more than 10 times in Reuters and no more than 50 times in NIPS. After preprocessing, we were left with vocabulary sizes of 6,344 for Reuters and 4,876 for NIPS.

We compared the method with variational Dirichlet process mixture model (VB-DP) as well as expectation maximization (EM) [3] with different number of components. For VB-DP, we used the truncated stick-breaking representations as described in [4] and set the truncation level to 100. We initialized our model with $\lambda = 5$, and we also initialized the Dirichlet hyperparameters for word distribution to $\beta = \frac{1}{\sqrt{V}}$, where V is the vocabulary size, to encourage the algorithm to find topics with more concentrated word distributions.

We ran ten-fold cross-validation on the datasets. Table 1 shows the average per-word log likelihood on the test set along with plus/minus one standard deviation. For these two datasets, our method produced similar performance as VB-DP and outperformed EM in terms of test likelihood despite being an online method which processes each document only once.

[3]We used the raw text versions available at `http://cs.nyu.edu/~roweis/data.html` (1988–1999) and `http://ai.stanford.edu/~gal/data.html` (2000–2004)

Table 1: Per-word log likelihood of naive Bayes models

	Online BMM	Batch VB-DP	Batch EM		
			$K = 10$	$K = 50$	$K = 100$
NIPS	**-6.89 ±0.05**	**-6.89 ±0.04**	-7.50 ±0.06	-7.44 ±0.06	-7.43 ±0.06
Reuters	**-6.54 ±0.09**	-6.59 ±0.08	-8.44 ±0.02	-8.38 ±0.02	-8.36 ±0.02

Table 2: Top 20 words (after stemming) from four of the topics found by model on the Reuters dataset with human-annotated labels

Economy	Oil	Technology	Agriculture
mln	price	industri	loan
dlrs	oil	comput	tax
billion	product	product	payment
year	pct	develop	dlrs
pct	market	busi	corn
loss	cut	market	program
oper	barrel	unit	billion
net	bond	system	propos
note	crude	ibm	plant
profit	yield	time	usda
compani	corp	line	save
share	sourc	sale	bill
includ	opec	high	agricultur
cts	output	softwar	committe
sale	april	profit	texa
tax	industri	problem	support
march	cost	intern	mln
gain	bpd	announc	farm
quarter	demand	technolog	year
end	manag	design	canada

5.3 Latent Dirichlet allocation

We also tested our model with LDA by making a single run through the Reuters corpus. Table 2 gives the top 20 words from four of the topics found by the model. Each topic is given a human-annotated label. As can be seen, the algorithm is able to find reasonable clusters containing semantically related words.

Figure 5 shows the smoothed progression of λ as more documents were processed. There are still some random ups and downs due to the randomness of sampling, but the graph shows that there is a general upward trend that more topics are being discovered as more documents are seen.

In addition, the Reuters dataset comes with a set of topic labels, and we experimented with using the inferred Dirichlet hyperparameters α_d of each document as features in a classification task. We tested the classification performance on the six topics with the most positive examples in the corpus. The classification algorithm used was LinearSVC in scikit-learn 0.14.1 [5] with the `class_weight` option set to `'auto'`, which gives each class a weight that is inversely proportional to its frequency in the training set.

The first 70% of the documents were used for training and the rest used as the test set. Table 3 shows the area under the receiver operator curve (AUC) scores for using word frequencies and the inferred α_d's as features. The result indicates that the proposed model was able to capture the underlying topic distributions as it induced a feature set that had comparative performance with word features despite reducing the feature space from 6,344 dimensions to 56, which was the inferred value of λ.

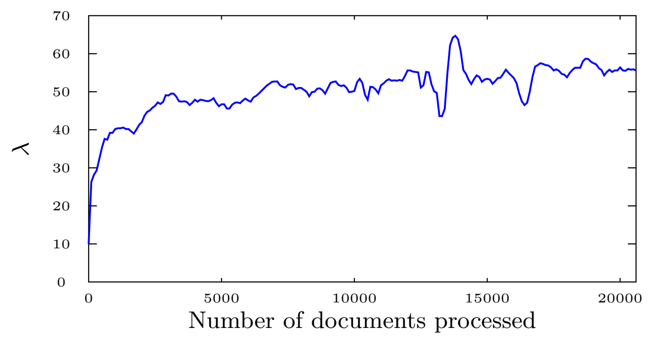


Figure 5: Progression of λ with LDA over the Reuters dataset

Table 3: Comparison of AUC scores on Reuters-21578 using linear SVM with different features: the word frequencies and the posterior hyperparameters α_d's learned in the proposed model

Feature \ Topic	earn	acq	money -fx	crude	grain	trade
Words	0.953	**0.951**	**0.932**	0.953	0.943	0.898
α_d's	**0.975**	0.921	0.919	**0.966**	**0.984**	**0.967**

6 Conclusion

We proposed a simple and intuitive extension to finite mixture model to handle streaming data with unknown number of components. We also proposed an online Bayesian moment matching technique to learn the parameters. The generality of the approach was demonstrated with Gaussian mixtures, multinomial naive Bayes, and latent Dirichlet allocation.

References

[1] Charles E Antoniak. "Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems". In: *The Annals of Statistics* (1974), pp. 1152–1174.

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

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the Royal Statistical Society* 39 (1977), pp. 1–38.

[4] David M. Blei and Michael I. Jordan. "Variational inference for Dirichlet process mixtures". In: *Journal of Bayesian Analysis* 1 (2006), pp. 121–144.

[5] F. Pedregosa et al. "Scikit-learn: machine learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.