UNIVERSITY OF CAMBRIDGE

MPhil in Machine Learning, Speech and Language Technology
2015-2016

MODULE COURSEWORK FEEDBACK

Student Name: Riashat Islam

Module Title: Machine Learning

CRSiD: ri258

Module Code: 4F13

College: St John's

Coursework Number: 3

I confirm that this piece of work is my own unaided effort and conforms with the Department of Engineering guidelines on plagiarism Confirm

I declare the word count for this piece is: 3500

Student's Signature: Riashat Islam

Date Marked:

Marker's Name:

This piece of work has been completed to a standard which is *(Please give mark as appropriate)*:

Marker's Comments:

4F13 Machine Learning Coursework 3: Latent Dirichlet Allocation

Riashat Islam RI258@CAM.AC.UK
University of Cambridge, Department of Engineering

1. Question A

In the provided dataset, the training set A contains 205211 data points, and the test set contains 147949 data points, with columns representing document ID, word ID and word count. For example, the word $V(89) - administration$ appears twice in document 1, while the word $V(4625) - policy$ appears 8 times in document 1. The word 'bush' ($V(841)$) appears in 1222 of the 2000 documents in training set A, as can be found by $x = find(A(:,2) == 841)$, and using the Matlab command $max(A\ :,3)$, we find that this word appears 29 times (as found by $find(A(:,3) == 29)$) in document 1158. The table below shows the count of documents, words and unique words.

	Set A	Set B	Union A U B
Num Docs	2000	3430	5430
Num Words	271898	195816	467714
Num Uniq Words	6892	6870	6906

```
A_docs = max(A(:,1));
B_docs = max(B(:,1));
Tot_docs = A_docs + B_docs;
A_total_words = sum(A(:,3)); B_total_words = sum(B(:,3));
Tot_words = A_total_words + B_total_words;
A_unique_words = unique(A(:,2));
B_unique_words = unique(B(:,2));
Union = [A ; B];
U_unique_words = unique(Union(:,2));
```

2. Question B

Our document model considers that each word in the document is drawn from a discrete categorical distribution that is parameterized by β. This model is based on the word frequencies across all the documents in the training set A. In this parameter based approach, to find the maximum likelihood multinomial over the words, we want to estimate values for a set of distribution parameters β such that it can best explain our observations of the words. We fit β by maximising the likelihood

$$\hat{\beta} = \arg\max_{\beta} \prod_{d=1}^{D} \prod_{n=1}^{N_d} Cat(w_{nd}|\beta) \tag{1}$$

$$Cat(w_{nd}|\boldsymbol{\beta}) = \prod_{m=1}^{M} \beta_m^{\mathbb{1}(w_{nd}=m)} \tag{2}$$

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \prod_{d=1}^{D} \prod_{n=1}^{N_d} \prod_{m=1}^{M} \beta_m^{\mathbb{1}(w_{nd}=m)} \tag{3}$$

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \prod_{m=1}^{M} \beta_m^{\sum_d \sum_n \mathbb{1}(w_{nd}=m)} \tag{4}$$

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \prod_{m=1}^{M} \beta_m^{c_m} = \arg\max_{\boldsymbol{\beta}} Mult(c_1, c_2, c_3, ...c_m|\beta, N) \tag{5}$$

Therefore, the maximum likelihood estimates of β_m is given as the ratio of the total count of vocabulary word m (given by c_m) divided by the total number of words (N) available in the collection of training set A, as given in equation 6 below

$$\hat{\beta} = \frac{c_m}{N} = \frac{c_m}{\sum_{l=1}^{M} c_l} \tag{6}$$

Using the estimates of β, ie, the maximum likelihood multinomial over the words for document set A, we can therefore plot a histogram showing the first 20 largest probability values (β).

```
beta = zeros(1,W); word_id = [1:W]';
for i = 1:W %total words in document A
j = word_id(i) == A(:,2); c = sum(A(j,3));
beta(i) = c/A_total_words; end
[R,I] = sort(beta,'descend'); b = R(1:20) i = I(1:20)
for elm = i names = [names V(i)];
axis([0 1 0 20]) barh(b) set(gca, 'ytick', 1:20)
set(gca,'yticklabel',names)
```

However, such a model does not specialize well since we only observe the most likely words across the entire training set A. Such a model does not give any specific information about the topics that each document represents. It is based on global word frequency distribution that does not give any specific information about individual documents in the set.

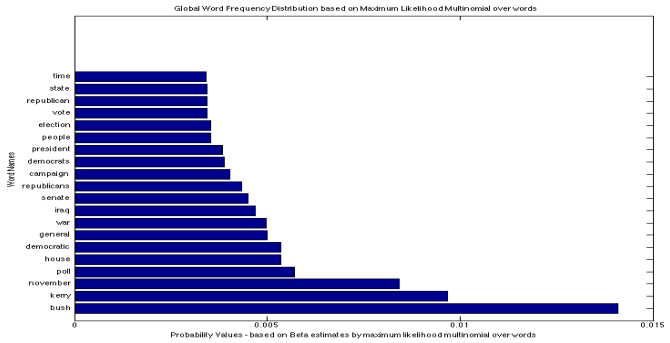

Figure 1. 20 Largest probability items in training set A based on β estimated by maximum likelihood

3. Question C

With the calculated probability values or β estimates, we can then calculate the probability of a new observation (word) $\hat{x}$ in the test document set B given that we know the maximum likelihood estimates as given by the equation 7.

$$p(\hat{x}|B) = \int_{\hat{\beta}} p(\hat{x}|\beta)p(\beta|B)d\beta = p(\hat{x}|\beta_{ML}) \tag{7}$$

Equation 8 then shows the log likelihood that we want to maximize given the training set, to generalize well for unseen test documents:

$$\log(\hat{\boldsymbol{\beta}}) = L(\beta) = \arg\max_{\boldsymbol{\beta}} \log \prod_{m=1}^{M} \beta_m^{c_m} = \arg\max \boldsymbol{\beta} \sum_{m=1}^{M} c_m \log(\beta_m) \tag{8}$$

Implications of words not in training set A for this model: Based on equation 6 we know that $\hat{\beta}$ is specific to the word m in the training set. If a word is not contained in training set A, it implies that c_m representing the total count of vocabulary word m will be 0, leading to a β value of 0. A bag of words containing unseen words will therefore be assigned a probability of zero. For example, test set B contains 14 new words (such as 'bulb', 'cats') that are not available in test set A. Such a model based on word frequency distribution will generalize poorly for words not available in the training set. Due to this, the log likelihood of the training set will be negative infinity, since as given in equation 8, a 0 value of β_m implies log(0) leading to negative infinity. To fix this model, one possible alternative is to increase word counts by one and normalize. This is the same as incorporating a Dirichlet prior (with α= 1) on our model to obtain a posterior distribution of β instead of doing maximum likelihood estimates (discussions later). This can make sure that every word is assigned a certain probability such that the log likelihood is not negative infinity. **Perplexity and Test set log probability**: We find that document 2001 in test set B does not contain any words that are not in A. However, set B does contain 14 unique words that are not present in A. Matlab:

```
uniq_doc2001 = unique(B(1:232,2));
A_unique_words = unique(A(:,2));
diff_w=setdiff(uniq_doc2001, A_unique_words); %result is 0
n_uni_w_B = setdiff(unique(B(:,2)), unique(A(:,2))); %result is 14
```

We can then calculate the test set log probability and perplexity, for document 2001 and entire test document set.

```
logp_B = 0; B_unique = unique(B(:,2));
for i = 1:size(B_unique,1)
logp_B=logp_B+sum(B(B_unique(i)==B(:,2),3))* log(beta(B_unique(i)));
end
perB =exp( logp_B/B_total_words);
```

	Log Probability	Perplexity
Test Set B	-inf	inf
Document 2001	-3692	4402

The results in table above shows that when test set contains words also available in training set (document 2001), the perplexity and

log probability values are lower (4402) compared to the case when test set B contains words that were not available in the training set (case of taking entire test set B)

4. Question D

Instead of maximum likelihood, when we do Bayesian inference, we obtain a posterior distribution $p(\beta|w_A, \alpha)$ over the parameter set β given the likelihood and priors, instead of calculating a point estimate. The prior belief is considered as a symmetric Dirichlet distribution with parameter $\alpha = 0.1$. A symmetric Dirichlet is a Dirichlet where each component of the parameter is equal to the same value. The Dirichlet is used as a distribution over discrete distributions where each component in the random vector is the probability of drawing the item associated with that component. Below we derive the predictive probability for a word vector from set B by using Bayesian inference.

$$p(\boldsymbol{w_B}|w_A, \alpha) = \int_{|\beta|=1} p(\boldsymbol{w_B}|\boldsymbol{\beta}) p(\boldsymbol{\beta}|w_A, \alpha) d\boldsymbol{\beta} \tag{9}$$

$$p(\boldsymbol{\beta}|w_A, \alpha) = \frac{p(w_A|\boldsymbol{\beta}) p(\boldsymbol{\beta}|\alpha)}{p(w_A)} \tag{10}$$

$$p(\boldsymbol{w_B}|w_A, \alpha) = \int_{|\beta|=1} p(\boldsymbol{w_B}|\boldsymbol{\beta}) \frac{p(w_A|\beta) p(\beta|\alpha)}{p(w_A)} \tag{11}$$

where $p(\beta|\alpha)$ and $p(w_A|\beta)$ are given by:

$$p(\boldsymbol{\beta}|\boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{m=1}^{M} \beta_m^{\alpha_m - 1} \quad p(w_A|\boldsymbol{\beta}) = \prod_{m=1}^{M} \beta_m^{C_{m,A}} \tag{12}$$

Based on the Dirichlet prior and likelihood terms from above, the predictive probability for w_B in equation 9 is therefore given as:

$$p(w_B|w_A, \alpha) = \frac{1}{p(w_A) B(\alpha)} \int_{\sum \beta_m = 1} \prod_{m=1}^{M} \beta_m^{c_{m,B} + c_{m,A} + \alpha_m - 1} d\beta \tag{13}$$

We can solve the above intergral by multiplying with the term $\delta(1 - \sum_m^M \beta_m) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-ik(1 - \sum_m^M \beta_m)}$. For equation 13, the term $p(w_A)$ has the integral of the form:

$$p(w_A) = \frac{1}{B(\alpha)} \int_{\sum \beta_m = 1} \prod_{m=1}^{M} \beta_m^{c_{m,A} + \alpha_m - 1} d\beta \tag{14}$$

We solve the above two integrals by using Laplace and inverse laplace transform expressions to get the following expression for the predictive probability:

$$p(w_B|w_A, \alpha) = \frac{\prod_m^M \Gamma(\gamma_m)}{\Gamma(\sum_m^M \eta_m)} \frac{\Gamma(\sum_m M\eta_m)}{\prod_m^M \Gamma(\gamma_m)} = \frac{B(\gamma)}{B(\eta)} \tag{15}$$

where $\gamma = c_{m,B} + c_{m,A} + \alpha_m$ and $\eta = c_{m,A} + \alpha_m$
Equation 15 further shows that the predictive distribution can simply be obtained from Beta distributions. Note that since we are integrating over β, the prior belief is automatically incorporated into the predictive distribution. For unseen words in the training set that are available in the test set, we would avoid problems that were encountered before, since the unseen words are now given priors probabilities. In section 5 below, we compute the log probability and perplexity given the entire test document set B.

5. Question E

Based on equation 15, and taking the log, the log probability for the test document with ID2001 is given by:

$$L(\mathbf{w'ID2001}) = \log p(\boldsymbol{w_{ID2001}}|A, \alpha) = \Big(\sum_{m=1}^{M} log\Gamma(\gamma_m) - log\Gamma(\sum_{m-1}^{M} \gamma_m) \Big) - \Big(\sum_{m=1}^{M} log\Gamma(\eta_m) - log\Gamma(\sum_{m-1}^{M} \eta_m) \Big) \tag{16}$$

	Log Probability	Perplexity
Test Set B	-1.5437e+06	2653
Document 2001	-3.6801e+03	4289

```
document_2001 = B(1:232,:); id_word = [1:W]'; CA = zeros(1,W);
CD = zeros(1,W); CB = zeros(1,W);
for i = 1:W
j = id_word(i) == A(:,2); s = sum(A(j,3));
CA(i) = s; j = id_word(i) == document_2001(:,2);
s = sum(document_2001(j,3)); CD(i) = s; j = id_word(i) == B(:,2);
s = sum(B(j,3)); CB(i) = s;
%calculating per word perplexity for doc 2001
gamma_2001 = CA+CD +0.1;
n = CA + 0.1;
logB_gamma = sum(gammaln(gamma_2001)) gammaln(sum(gamma_2001));
logB_eta = sum(gammaln(n)) gammaln(sum(n));
log_prob_2001 = logB_gamma logB_eta;
perplexity_2001 = exp( log_prob_2001/440);
%calculating per word perplexity for all documents in B
gamma_AllB = CA +CB + 0.1
logB_gamma = sum(gammaln(gamma_AllB)) gammaln(sum(gamma_AllB));
logB_eta = sum(gammaln(n)) gammaln(sum(n));
log_prob_AllB = logB_gamma logB_eta;
perplexity_AllDocs = exp( log_prob_AllB/B_total_words);
```

Using the Maltab code given above, we compute the log probabilities. It is preferred to use the Multinomial without the combinatorial factor and therefore this factor is not included in calculation of the log probability. This is because it is only important to find the probability of the collection of the words under the model. The log probability is calculated for a bag or group or words instead of word collections. Such a model ignores the arrangement or sequence of words and considers only counts of words. Another possible issue with including the combinatorial factor is that the factor might become too large for the large document collections and word collections we use in practice for such problems. This combinatorial factor is also independent of α in the Dirichlet distribution.

The per-word perplexity value is an indicator of the confidence or certainty of the model about a particular word. In other words, the perplexity indicates how uncertain a model is about the particular word in the document. A high value of perplexity suggests that the model is as highly uncertain about a particular word as any other words in the document. Therefore, a low value of perplexity indicates a good model.

Comparison of Perplexity between ML and Bayesian infernce: The **Maximum Likelihood (ML)** Considering entire **test set B**, the model perplexity was infinity compared to the **Bayesian inference** model with a perplexity value of 2653. This suggests that the Bayesian model can generalize better for unseen words in test document, since it assigned Dirichlet priors to words, compared to the ML model that is infinitely uncertain about unseen words during testing.

Considering document ID 2001 where B does not contain any unseen words compared to A (unlike above case). The ML model had a perplexity of 4402 compared to Bayesian model with perplexity 4289 as shown in tables. This again suggests that the Bayesian model generalizes better to test documents and is a better model since it is not based simply on global word frequency. The Bayesian model has a lower uncertainty measure during document modelling as indicated by per-word perplexity. However, both the ML and Bayesian model still does not contain any specific information about particular topics for each document. None of the generative models mentioned above specializes well to model the different topics for different documents.

6. Question F

Considering a uniform multinomial distribution where the probabilities of each of the words are the same, ie, the β values are considered equal for all words, instead of considering a count of words. $\forall_m \beta_m = \frac{1}{M}$. The prior under a uniform multinomial is given by:

$$p(\boldsymbol{w}|\boldsymbol{\beta}) = \prod_d^D \prod_m^M \beta_m^{C_m} = \prod_d^D \prod_m^M \frac{1}{M^{C_m}} = \prod_d^D \frac{1}{M^{N_d}} = \frac{1}{M^{N_D}} \tag{17}$$

Therefore the log likelihood is now given by:

$$L(\boldsymbol{w}) = log(\frac{1}{M^{N_D}}) = -log(M^{N_D}) = -N_D log(M) \tag{18}$$

and hence the perplexity measure can be expressed as:

$$perplexity(\boldsymbol{w}) = exp(-\frac{-N_D log(M)}{N_D}) = M \tag{19}$$

```
log_Mult2001 = 440*log(W);
perplexity_2001_Mult = exp ( log_Mult2001/440);
log_Mult_AllB = B_total_words * log(W);
perplexity_AllB_Mult = exp( log_Mult_AllB/B_total_words);
```

	Log Probability	Perplexity
Test Set B	-3890	6906
Document 2001	-1.7310e+06	6906

The **log probability** for document 2001 is much lower than the log probability for all the documents in set B - we have assigned equal probabilities to each word, the set of words in document 2001 is more likely to be observed (higher log probability) compared to the large diverse words available in the entire test set B. **Perplexity under uniform multinomial**: The results above shows that

the perplexity values are equal for both document 2001 and test B, unlike before where perplexity was lower for B. This is because the uniform model does not use any relevant information captured from the training set. Since β is assigned equal probability values, such a uniform model does not distinguish between words observed or unobserved in the training set. The test set, even if contains unseen words, will still assign equal probability values to β irrespectively of whether the word was observed in the training set or not. Under uniform multinomial model, the likelihood only depends on size of vacabulary and document and the model is equally uncertain about the generative process for the document irrespective of the words it contains.

Comparing Uniform Multinomial Model with ML and Bayesian Inference model: The UM model performs better than the ML model in cases where there are unseen words in the test set, as suggested by the lower perplexity values. This is because the UM model assigns equal probabilities to each word, even if for unseen words. However, for document 2001, the perplexity value for the ML model (4402) is lower than the UM model (6906). This is because the ML model is more informative for seen words, and is more certain about the document generative process, compared to the UM model which does not include any information at all about the frequency or word counts of the words in the document set. Comparing the UM model with the Bayesian inference model, it is immediately obvious that the Bayesian model always performs better than the UM model. The UM model does not contain any informative measures of β. For Bayesian model, obtaining posterior distribuions based on assigned priors is more informative than simply assuming a uniform distribution of words for generalization on test document set.

7. Question G

In BMM, a document generative process based on bag of words, the model takes into account the document structure, and assigns a topic to each document ($z_d = k$), while the words for the document are being generated by the word distribution of the topic (β_k). We explore the use of collapsed Gibbs sampler on a mixture of multinomials document model. We consider a document d from cluster j consisting of a vector of N words w_d distributed according to a multinomial distribution parameterized by the vector θ_j. For the latent variables θ and β in the model, we choose a uniform Dirichlet prior distribution because we assume that a priori nothing is known about the word distributions for each class. The model includes K mixture components. For clustering, the only specific relevant variables are the hidden document topic labels z ($z_i d$ being the per word topic assignment for each document). The conjugacy between the Dirichlet and Multinomial distributions make it possible for collapsed Gibbs sampling, where the variables to be marginalized out must be marginalized in closed form. Collapsed Gibbs sampling is used to sample from the complete conditional of the label of document d $p(z_d = k|w, z_{-d})$. Using the samples of z, we can then find the posterior distributions of θ and β given by:

$$p(\theta|w, z, \alpha) = \frac{1}{Z_\theta} \prod_{d=1}^{D} p(z_d|\theta)p(\theta|\alpha) = Dirichlet(\theta|c + \alpha) \tag{20}$$

$$p(\beta_k|w, z, \alpha) = \frac{1}{Z_{\beta_k}} \prod_{m=1}^{M} p(w_m|\beta_k)p(\beta_k|\gamma) = Dirichlet(\beta_k|c^k, \alpha) \tag{21}$$

We can then compute the means or expected values of the mixture proportions by taking the expectation for the Dirichlet distributio . The expected values of the posterior probabilities of θ and β are given by:

$$\hat{\theta_j} = \frac{\hat{c_j} + \alpha}{\sum_{d=1}^{D}(\hat{c_d} + \alpha)} \quad \hat{\beta_{k,j}} = \frac{\hat{c_j^k} + \gamma}{\sum_{m=1}^{M}(\hat{c_m^k} + \gamma)} \tag{22}$$

The following MATLAB lines of code was added to given BMM code:

```
perplexity_K = zeros(length(K), 1);
%sk_docs contains documents assigned to each mixture component
theta(:,iter)=(sk_docs(:)+alpha)/(sum( sk_docs(:) + alpha));
plot(theta); plot(theta')
perplexity = exp( lp/nd);
perplexity_K(t,1) = perplexity; %for each K mixtures
plot(perplexity_K);
```

Figure 2 shows how the posterior probabilities for each mixture component ($K = 20$) evolves with number of Gibbs iterations. With higher Gibbs iterations, the documents are clustered with two different topics (components with higher posterior probabilities), showing that the generative model is becoming more certain across iterations about which two topics are more likely to have generated the document set. The other mixture components stay almost constant showing that those topics are not suitable to cluster the document set.A different representation of this figure is given in figure 3. This plot shows that topics 14 and 11 (topics with highest posterior probabilities) are assigned higher posteriors such that they are more responsible for the document structure. As shown in section 11, we also considered 50 Gibbs iterations to analyze the convergene of Gibbs sampler and the topic based clustering (details in section 11).

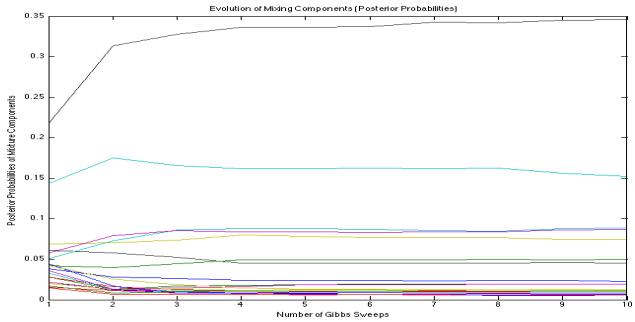

Figure 2. Posterior Probabilities of Each Mixture Component as a function of Gibbs sweeps

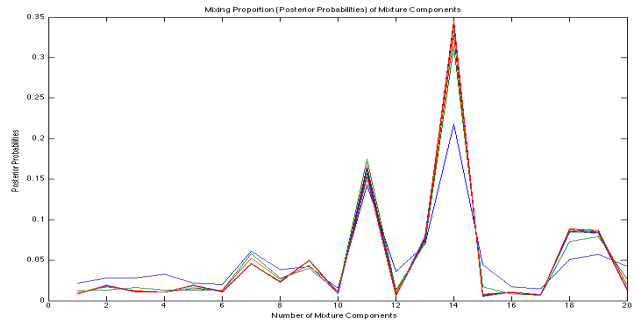

Figure 3. Posterior Probabilities for each Mixture Component

Perplexity: After 10 Gibbs iterations, the perplexity was found to be **2124** with log probability of **-1.5e+06**. We also considered how the perplexity values change as we include more topics (explanation given in solution of Question K) as shown in figure 13 in section 11. Additionally, we considered how the generalization of the BMM model changes as we change the prior of the Dirichlet (from $\alpha = 10$ to $\alpha = 0.1$. Using 50 Gibbs iterations, the results showed that with $\alpha = 0.1$ the perplexity value was **2128**, while with $\alpha = 10$, the perplexity value was **2096**. This suggests that, with higher Gibbs iterations, the model generalizes better to unseen test document (2096 instead of 2124) for $\alpha = 10$, as the mixing distributions are likely to converge better. Also, with lower value of $\alpha = 0.1$, we are assigning a very small prior distribution to the BMM which makes it more likely to perform poorly to unseen words in the test document. The low value of α uses less constraint on the θ distributions to converge to a single topic for each document leading to a poorer document clustering.

Comparison of Perplexity for BMM with Bayesian Model and Uniform Multinomials model:
With the BMM model, the perplexity value was **2124** compared to the Bayesian inference model with perplexity of **2653** and the uniform multinomial with perplexity **6906**. This suggests that the BMM model generalizes best to unseen test documents and is a better model (maximizing the log probability based on posterior distributions of β and θ) compared to the other two models.
Explanation: This is because the BMM considers topic structure of the documents, and assigns a soft topic assignment to each document. In other words, BMM assigns a specific topic to each document with certain probability using a clustering approach. For testing with B using BMM, with posterior estimates of β and θ we can assign unseen documents to a specific cluster (topic) with a certain uncertainty. The BMM learns the document set structure by assigning a topic to each document (using knowledge

of topic's word distribution)unlike the other two models which us simply based on the word probabilities (point estimates) with no knowledge of the topic structure present across the document set. BMM performs better as it can identify the generative process of which topics are more likely to have generated the document set.

8. Question H

Convergence of a MCMC method to a stationary distribution means that we can successively sample from a convergent Markov chain, the limiting distribution of which is the true distribution. At convergence to the stationary distribution, the samples are truly representative of the underlying stationary distribution of the Markov chain. The transition probabilities between the states of the Markov chain become constant. However, often the samples are correlated with each other slowing the sampling algorithm. This means that the samples from the beginning of the chain can be highly influenced by the random initialization state. To correct this, MCMC methods often include a **"burn-in"**such that the initial samples at the beginning of the chain are discarded to reduce the influence of the random initializations on the samples.

```
%changing the seed in sampDiscrete.m
randn('seed', 46)
r = sum(b)*rand();
```

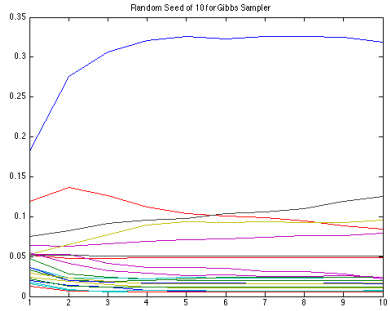

Figure 4. Random Seed of 10 for Gibbs sampler

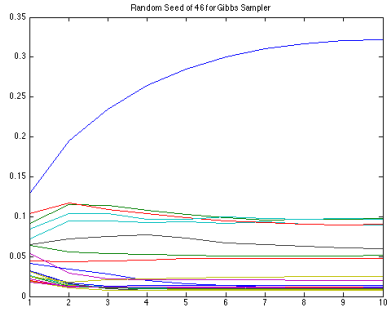

Figure 5. Random Seed of 46 for Gibbs sampler

Figure 6. Differnet random seeds of 10 Gibbs sampling sweeps for BMM

Using 10 iterations of Gibbs sampling, for the BMM, the sampler does seem to converge. The Gibbs sampler samples from the posterior distribution of the topics given the words, where the samples produced occur in proportion to their posterior density, ie, the sampler has converged in distribution. The converged Gibbs sampler samples more frequently around the modes of the posterior distribution. Local convergence in distribution of Gibbs sampling can be understood based on figures 4 and 5 as the sampler is clustering documents in almost similar ways - in both plots we observe that the document clustering is similar, where one particular topic is assigned higher posteriors. However, even though the Gibbs sampler converges to assigning a single specific topic to the documents (converges to a single topic assignment), it does not seem to explore the posterior distribution well. In other words, once the sampler assigns a particular topic to the document set, at convergence, it assigns higher posterior probability to a single topic, without exploring the posterior distribution well enough. The Gibbs sampler is not exploring the full space and is confined to a local region that includes multiple maxima, and hence is not sampling around the global optima. Gibbs sampling, when fully converged, is good at exploring posterior distribution since it frequents from the regions of high probability. However, because variables can be highly correlated, Gibbs sampling may not actually sample from the full distribution. This may be also be because we are taking only 10 samples - for robust approximation to true distribution, we need to take more samples to avoid correlated samples. One additional possible reason for the lack of global convergence of distribution in MCMC methods like Gibbs sampling is **'label switching'** that can occur mid-chain due to which averaging across several samples is often worse than taking individual samples as the chain summary since the meaning of each label can change across multiple samples.

9. Question I

LDA is a generative probabilistic model in which each document is represented as a mixture over latent topics, ie, each document contains proportions of multiple topics, where each topic is characterized by a distribution over words. Every document in LDA has its own distribution over topics, and we want to estimate the posterior probabilities of the proportions of topics within each document. This would give insights into which topics are more responsible (within the mixture) to have generated a particular document, to get overall better insights about the entire training document set structure. **Difference between LDA and BMM**: In BMM, the multinomial clustering variable is selected once for each document, and a set of words are then selected for the document conditional on the clustering variable or topic. The BMM model restricts the document to be associated to a single topic. However, in LDA, the topic node is sampled repeatedly within the document and the documents are associated to multiple topics. In LDA, we need to integrate out both θ_d and β and we obtain a predictive distribution for a single $z_n d$ as shown in equation 23 and the Gibbs update contains a word from the topic distribution and another from the word distribution.

$$p(z_{nd} = k|z_{-nd}, w, \gamma, \alpha) \propto p(z_{nd} = k|z_{-nd}, \alpha)p(w_{nd}|z_{nd} = k, w_{-nd}, z_{-nd}, \gamma) \quad (23)$$

For each sample of z from Gibbs sampler, and given the word vector w, the posteriors of θ and β are given as:

$$p(\theta_d|w, z, \alpha) = \frac{1}{Z_{\theta_d}} \prod_{n=1}^{N_d} p(z_{n,d}|\theta)p(\theta_d|\alpha) = Dirichlet(\theta_d|c^d + \alpha) \quad (24)$$

$$p(\beta_k|w, z, \alpha) = \frac{1}{Z_{\beta_k}} \prod_{m=1}^{M} p(w_m|\beta_k)p(\beta_k|\gamma) = Dirichlet(\beta_k|c^k, \alpha) \quad (25)$$

From there, we can obtain expected or means of the posterior estimates of θ and β given by:

$$\hat{\theta_{d,j}} = \frac{c_j^d + \alpha}{\sum_{k=1}^{K}(c_k^d + \alpha)} \quad \hat{\beta_{k,j}} = \frac{c_j^k + \gamma}{\sum_{m=1}^{M}(c_m^k + \gamma)} \quad (26)$$

The following MATLAB codes were added to lda.m to get the figures below:

```
iterations = 10;
theta = zeros(K(t), size(iterations,2));
num_topic_perplexity = zeros(length(iterations), length(K));
%theta for each document
theta(:,iter) = (skd(:,2)+alpha)/(sum( skd(:,2)+alpha));
plot(theta);
```

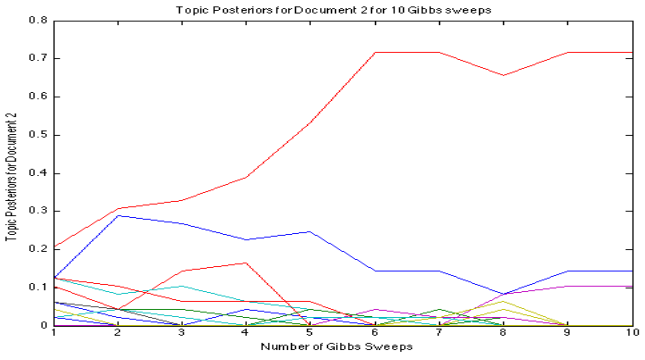

Figure 7. LDA with 10 Gibbs Iterations for Document 2

Topic Posteriors vs Gibbs sweeps: Figure 7 and 8 shows the expected topic posteriors across 10 Gibbs iterations for a particular document(ID2). Note that here we are plotting for a single document, and showing how the topic posteriors change across number of Gibbs iterations. Plots show how the expected topic

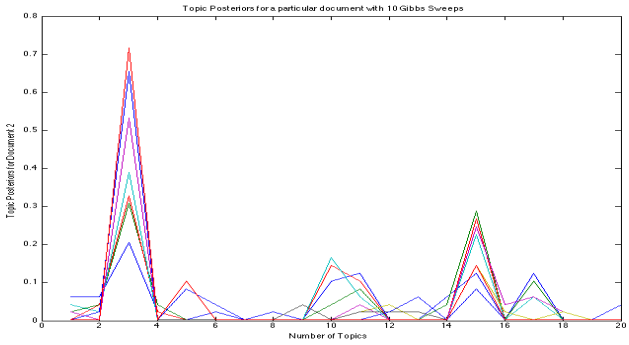

Figure 8. LDA for Document 2 showing proportions of topics

posteriors for a document evolves with sampling. Initially, one or more topics within the document may reach a higher expected posterior, showing that those topics are more likely to generate the given document. Figure 7 shows that initially our belief about mixture of topics within the document are almost same (similar probabilities). However, as we sample for a single $z_n d$, one topic obtains higher posterior compared to other topics. The document still contains a mixture of topics, except that, as we converge for Gibbs sampling, we assign a higher topic proportion and consider this topic to have contributed more towards generating this document.

Convergence of Gibbs Sampler for LDA: Figure 7 suggests that the Gibbs sampler does not converge with 10 Gibbs iterations. This is because even though one topic has a much higher posterior, other topics are not completely negligible and has probabilities of around 0.1. Therefore, those topics may also have been responsible for generating this document. At convergence of Gibbs sampler, we would expect a noticeable difference between expected topic posteriors - where one or more topics would have very higher posterior probabilities compared to the others. This can be illustrated using figure 12 in section 11. Figure 12 shows one topic to be have a significantly higher topic posterior compared to other topics as we increased Gibbs sweeps to 50. In LDA, the Gibbs sampler is not guaranteed to converge to a global optima. The Gibbs sampler samples from the posterior distributions of the topics given the words. Therefore, the Gibbs sampler is not good at exploring the posterior distribution. We notice that even with 50 Gibbs iterations as shown in figure 12 in section 11, the topic posterior (one with highest posterior) still does not reach a steady value. LDA with Gibbs sampling is not always guaranteed to converge, unlike using variational inference or Variational EM (not discussed here).

Perplexity: With 10 Gibbs iterations, the perplexity is **1890**. With 50 Gibbs iterations, the perplexity was calculated to be **1651**. **Comparison**: The perplexity values for LDA compared to the previous values suggests that LDA performs better during testing. The perplexity value after 10 Gibbs iterations for LDA (1890) is much lower compared to BMM model even with 50 Gibbs iterations (that is almost guaranteed convergence of BMM). Comparing LDA with the Bayesian inference, uniform multinomial and ML model, the perplexity values are significantly lower for LDA. After 50 Gibbs iterations, the value even decreases furhter (towards convergence in LDA). **Explanation**: Using LDA, we learn the topic structure of each document within the training corpus, and gain better understanding about topic proportions for each document within entire training set. For unseen documents, with better knowledge of each document topic proportions, LDA can gain better insights about the contents of test documents, even those containing unseen words. The topics recovered by the algorithm can pick up meaningful aspects of the structure of the document set. Figure 14 in section 11 (with explanation) shows performance of LDA depending on number of topics (K). We also include an explanation of requirement of **Burn In** and **Implication of Dirichlet Prior Parameters** in section 11.

10. Question J

In LDA, the word entropy is a measure of the disorder of the word distribution for a given topic. We consider the frequency of words, and its distribution for a given topic (which itself is a proportion of a document). Word entropy captures whether all the mass for the word distribution for topic is centred within a few words, or whether the mass is evenly spaced across a lot of words. Entropy, in this context, expresses the disorder of words within each topic (cluster) where for each cluster, the category distribution of the words is calculated first, and the probability that a member of topic j belongs to category i. We use the following MATLAB code to calculate and plot word entropy for each topic:

```
beta = zeros(1, K(t));
word_entropy = zeros(K(t),iterations); %word entropy for all
beta = zeros(W,K(t));
for o = 1:K(t)
beta(:,o) = (swk(:,o)+gamma)/(sum(swk(:,o)+gamma)) ;
beta_fixed_word = beta(10,:)';
beta_fixed_word_sweeps(:, iter) = beta_fixed_word;
for i = 1:K(t)
entropy = 0;   %word entropy for topic
for j = 1:size(beta,1)
entropy = entropy + ( beta(j,i)*log(beta(j,i)));
end     word_entropy(i,iter) = entropy;
plot(word_entropy')
```

As shown in figure 9, the entropy decreases as we have more Gibbs iterations for the LDA model. Similar to how topics posterior increase shows our informative measure of the document being likely to be generated by those topics - the word entropy decrease shows which words are more responsible for representing those topics. This means, the word disorder for each topic decreases with number of iterations. The words become more informative about a given topic. This is shown in figure 9. It shows which two topics are more likely to contain word distributions that are reliable informative measures for representing that topic. Out of 20, two word entropy for the topics are lower than the others. This suggests that the words representing topic 15 and topic 2 are more informative (less disordered) about that topic. Other topics (with higher entropy) indicates that the words responsible for those topics are more uncertain about the topic representation, compared to topics with lower entropy. Figure 10 shows a different perspective of this representation, showing which topics contain more certain words for representing these topics. This means, word entropy indicates our uncertainty about which words are more or less responsible for the topic. Entropy decreases when the words are more informative.

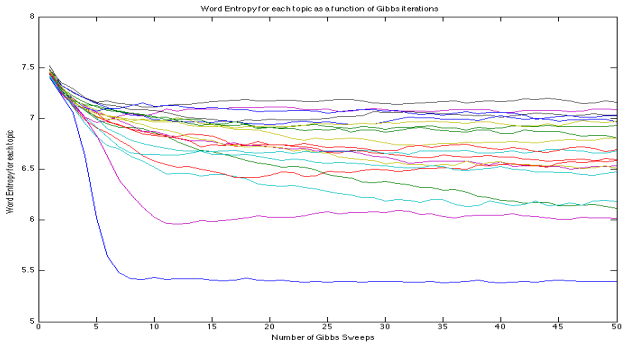

Figure 9. Decrease in Word Entropy for each topic across 50 Gibbs Iterations

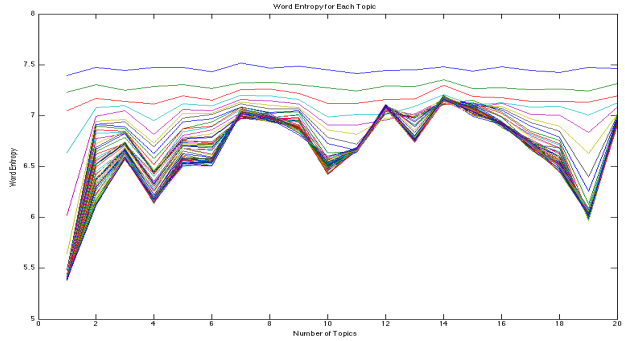

Figure 10. Word Entropy for each topic

11. Question K

Question K shown in PAGE 6 since no marks contained

Number of Gibbs sweeps

We considered using 50 Gibbs iterations instead of 10, to analyze whether the topic based clustering of the documents changes. Figure 11 shows results with 50 Gibbs iterations. Results again show that two topics or mixture components have much higher posterior probabilities compared to other topics, confirming that the document set A can be clustered using two different topics. In other words, two out of 20 topics are more likely to have generated the documents in A. Note that the labels of topics applied by the BMM algorithm does not have any particular meaning and are completely interchangeable. What is more important is to see that using a soft assignment, the document structure is more likely to have been generated by two topics compared to other topics. We still however, assign low posterior probabilities of the other 18 topics that may have also generated the document. In other words, BMM leads to a soft assignment of documents instead of hard assignments. Figure 11 shows the posterior probabilities for 50 Gibbs iterations. The plot shows that with higher iterations, the Gibbs sampler for the BMM is more likely to converge, leading to constant posterior probability values.

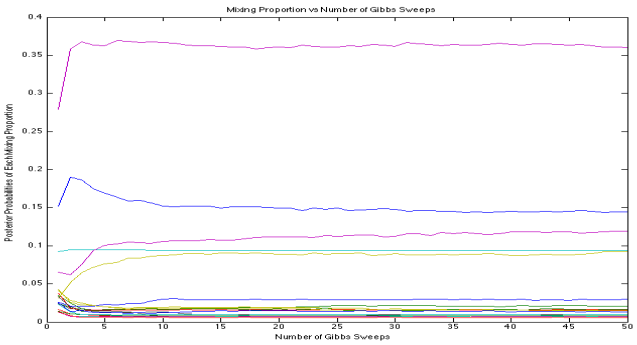

Figure 11. 50 Gibbs iterations for BMM

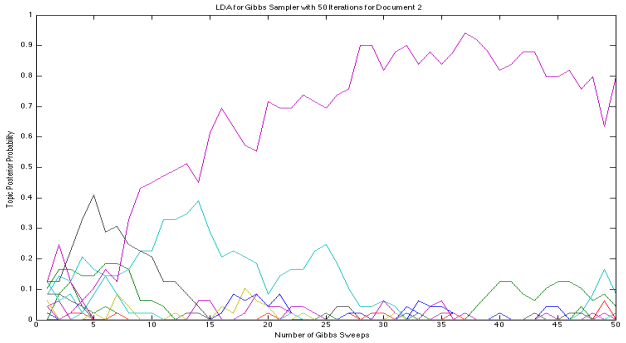

Figure 12. LDA with 50 Gibbs Iterations

Compared to that, for the LDA, with 50 Gibbs iterations as shown in figure 12, the Gibbs sampler is not guaranteed to converge to a global optima, although sufficient local optima is ensured. This can be understand based on the large gap in topic posteriors, which shows that as number of Gibbs sweeps are increased, the generative model becomes more certain about which topics are more likely to have generated the documents. Unlike BMM, the LDA model is not fully guaranteed to converge to a global convergent distribution for 50 Gibbs samples. However, using more samples, we obtain a much better robust approximation to the true distribution. Hence, for LDA, using higher number of Gibbs samples ensures better local conveegence of the distribution, and the Gibbs sampler is more likely to explore the posterior distribution.

Number of Topics K

Figure 13 shows how the performance depdends on the number of topics in the BMM. The figure shows that as we include more topics, the generalization performance gets better (as suggested by the lower perplexity values) upto 60 topics. However, as we include more topics in the model, the generalization becomes worse. The number of topics is a model selection process. If we have an optimum number of topics, then the model is likely to perform better in clustering the documents in the training set, leading to good training set performance. However, if we have too many topics, then the model is likely to overfit. This is because the high number of topics is more than required for the document clustering, and hence each cluster contains a lower proportion of documents assigned to it. Similar to the k-means analogy, too many clusters leads to poorer performance on the test set.

Similarly, the figure 14 in section 9 shows how the performance of LDA depends on the number of topics. This again shows that with higher number of topics, the LDA performs better in test set. Also, unlike the BMM, the LDA is less prone to overfitting. This is because in LDA, we estimate topic posteriors for each of the documents. Therefore, with high number of topics, for each document we still estimate the topic proportion, and at convergence, the topics that have not generated the document are assigned very low probabilities. Since the LDA model is document specific, even with very high number of topics, the topics that are irrelevant for the entire document corpra are always assigned low posterior probabilities and therefore does not affect the performance on the training set. The estimates of the posterior distributions automatically takes account of the complex model (Occam's Razor), and so the LDA model does not overfit. This is shown by the decreasing perplexity values with higher number of topics.

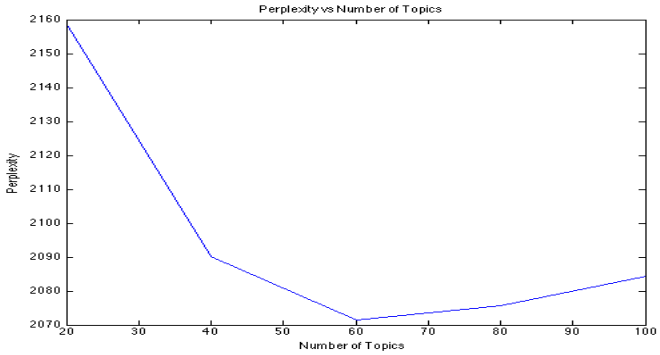

Figure 13. Perplexity vs Number of Topics for BMM

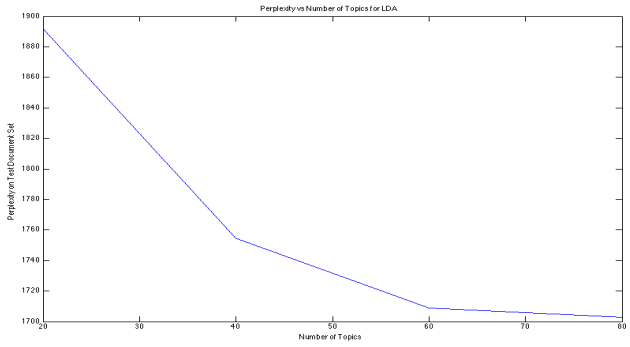

Figure 14. Perplexity vs Number of Topics for LDA

Dependence on Dirichlet Prior Parameters θ and β

We investigated the influence of the Dirichlet prior parameters on the LDA model. Note that, we always use symmetric Dirichlet priors (each element in the vector has the same value) such that we can reduce the number of parameters for the topic Dirichlet β, otherwise there would have been as many parameters as the vocabulary size. **What values to set**: If the parameters are set to 1, it means a uniform prior distribution such that all document-topic proportions are equally likely. However, when the parameters are set to a value greater than 1, it favors vecotrs such as (0.33, 0.34) with a higher probability than vectors (0.8, 0.1). When the parameters are set to a value less than 1, distribution is more likely to favor sparse vectors. If α is set to a relatively large value, then many topics will be activated per document.