Coursework 3 - Latent Dirichlet Allocation (LDA)

Candidate number: H801L
Word count: 1000

10th December 2020

1 Part A

1.1 Maximum likelihood

For a multinomial distribution over words parameterised by $\boldsymbol{\beta} = [\beta_1, ..., \beta_M]^T$, we can estimate optimal $\boldsymbol{\beta}$ by maximising log likelihood, Equation 1.

$$\hat{\boldsymbol{\beta}} = \underset{\beta}{\text{argmax}} \log\left(p(\boldsymbol{w}|\boldsymbol{\beta})\right) = \underset{\beta}{\text{argmax}} \log\left(\frac{n!}{\prod_{m=1}^{M} c_m!} \prod_{m=1}^{M} \beta_m^{c_m}\right) = \underset{\beta}{\text{argmax}} \sum_{m=1}^{M} c_m \log \beta_m \tag{1}$$

This is equivalent to computing frequency of each vocabulary word in the text corpus, see Equation 2 and Code-listing 1. Where c_m is total count for m^{th} word in the vocabulary, n is total number of words.

$$\hat{\beta}_m = \frac{c_m}{n} \tag{2}$$

```python
D = np.max(A[:, 0])  # number of documents in A
beta_m = np.zeros(V.shape[0])  # multinomial distribution parameters
N = np.sum(A[:,2])  # total number of words
for m in range(V.shape[0]):
    all_words_m = np.where(A[:, 1] == m+1)
    c = np.array(A[all_words_m, 2])
    beta_m[m] = np.sum(c)/N
```

Listing 1: Estimating $\hat{\boldsymbol{\beta}}$ by maximum likelihood

Top 20 words with highest probability (highest frequency) are presented in Figure 1.

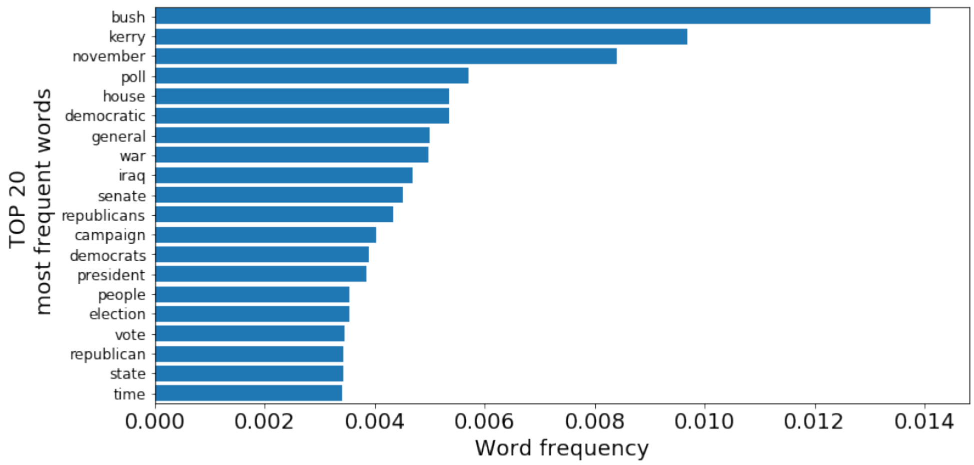


Figure 1: Top 20 words with highest frequency in corpus A.

1.2 Log probability of arbitrary test set

Looking at the log probability Equation 3, we observe that log probability decreases with length of a test set (every next world adds uncertainty). Additionally, the higher the frequencies β_m (Equation 2) of words in the test set, the higher the log probability.

$$\log p(\boldsymbol{w}|\boldsymbol{\beta}) = \log\left(\frac{n!}{\prod_{m=1}^{M} c_m!}\prod_{m=1}^{M}\beta_m^{c_m}\right) \tag{3}$$

Therefore the test set with the highest log probability will contain a single word $'bush'$

$$log(p(\{bush\} \mid \boldsymbol{\beta}) = -4.28 \tag{4}$$

Because our training data contains a finite set of words, there could be a test set with a word not present in the training set, for which $\beta_m = 0$. In such case the log probability goes to $-\infty$. This phenomenon is called **burstiness** of words. We can fix that using Bayesian inference presented in Part B.

2 Part B

In the Bayesian approach we form posterior belief about the system by combining the observation likelihood with prior belief. Dirichlet distribution is a **conjugate prior** of multinomial distribution, thus posterior is also a Dirichlet distribution, and it can be simplified as shown in Equation 5.

$$\begin{gathered}posterior = \frac{likelihood \times prior}{p(\boldsymbol{w})}\\ p(\boldsymbol{\beta} \mid \boldsymbol{w}) = \frac{p(\boldsymbol{w} \mid \boldsymbol{\beta})p(\boldsymbol{\beta} \mid \alpha)}{p(\boldsymbol{w})} \propto Z\prod_{m=1}^{M}\beta_i^{c_m} \times \frac{1}{B(\alpha)}\prod_{m=1}^{M}\beta_m^{\alpha_m-1} = C\prod_{m=1}^{M}\beta_m^{c_m+\alpha_m-1}\end{gathered} \tag{5}$$

The posterior probability of word w_m* can now be computed by integrating over all possible β, Equation 6, which is the expectation of β under posterior distribution.

$$\begin{aligned} p\left(w_m^* \mid \boldsymbol{w}\right) &= \int p\left(w_m^* \mid \beta\right) p(\beta \mid \boldsymbol{w}) d\beta \\ &= \int \beta_m \times C\prod_{m=1}^{M}\beta_m^{c_m+\alpha-1} d\beta = E[\beta_m \mid \boldsymbol{w}] = \frac{c_m + \alpha_m}{n + \sum^M \alpha_m} = \hat{\beta}_m \end{aligned} \tag{6}$$

Multinomial distribution derived with Bayesian inference looks similar to the one obtained with maximum likelihood estimate. The difference is in the extra α_m term in the nominator and $\sum \alpha_m$ in denominator. Where α_m can be interpreted as a psudo-count for word w_m.

When $\alpha > 0$, **Rare** words (not present in the training data) have a non-zero probability, thus allowing computation of log probability of any test sequence. Alpha has little effect on probability of common words given that $c_m >> \alpha$. The higher the value of α the stronger our prior belief about word frequency and the probability difference between common and rare words decreases.

3 Part C

In this coursework we assume a **unigram** language model (independent words), where the joint probability of words in a document factorises as in Equation 7.

$$p(\{w_1, w_2, ..., w_n\}) = p(w_1)p(w_2)...p(w_n) = \prod_{i=1}^{n} p(w_i) \tag{7}$$

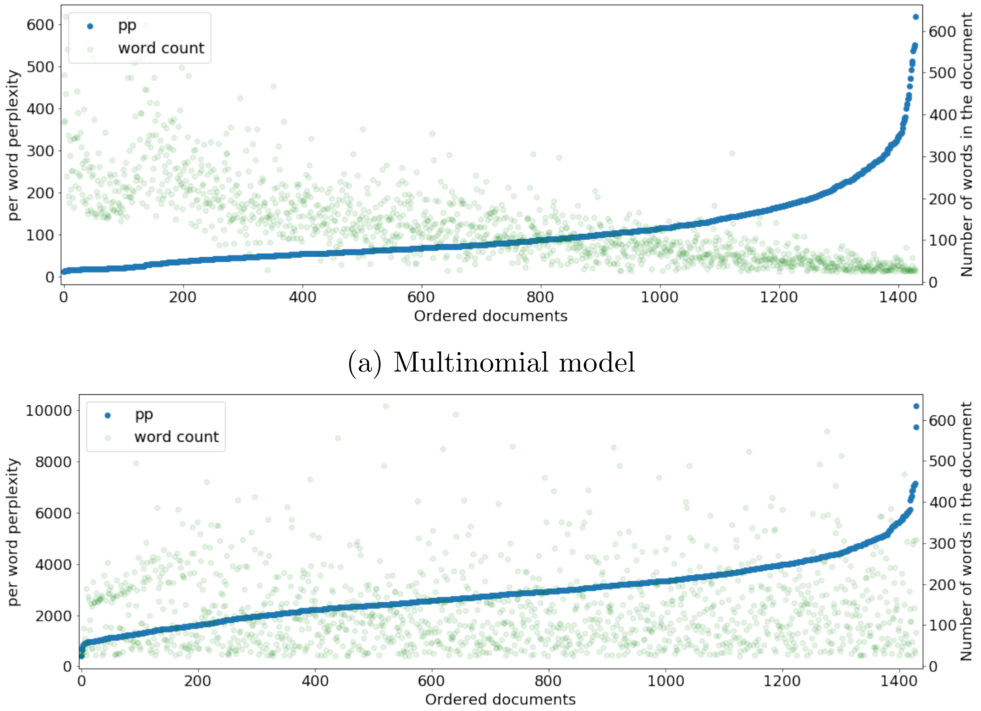
We can think of this factorisation as a product of $\boldsymbol{n}$ multinomial distributions of every word in a document. This is equivalent to computing **categorical distribution** of the sequence of words in a document and the log of this probability equals **-3689** for document ID 2001.

3.1 Per word perplexity

Per-word perplexity, Equation 8, is a metric which measures how well a given language model has captured or identified the language used in a particular document. The lower the per-word perplexity, the lower the uncertainty about the next generated word.

$$pp(\boldsymbol{w}) = exp\frac{-log(p(w_1, w_2, ..., w_n))}{n} \tag{8}$$

Per word perplexity normalises over word count only if we compute the log probability of a test sequence using **categorical function**. If the multinomial function was used, the combinatorial factor would make the per-word perplexity grow with decreasing document length, as shown in Figure 2.a.



(a) Multinomial model

(b) Categorical model

Figure 2: Per word perplexity and word count for sorted documents by ascending per word perplexity. PP calculated using multinomial function is not size invariant.

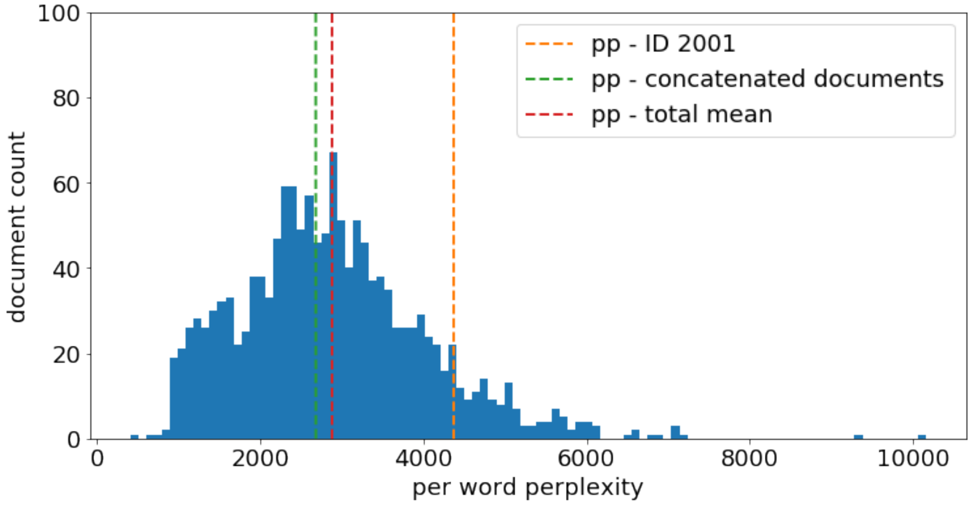


Figure 3: Distribution of per-word perplexities for categorical model.

	Document ID 2001	Mean for all documents	All documents
per-word perplexity	4373	2888	2683

Table 1: Per word perplexity for categorical model. Evaluated for document with ID 2001, all documents separately, all documents together.

Figure 3 shows the distribution of per-word perplexities for all 1430 test-documents. The perplexity for document ID 2001 is relatively high, because it uses rare words, which have low probability. In Figure 9, ID 2001 (high perplexity) is contrasted with ID 2012 which has low perplexity as it uses words more commonly found in the training corpus A. If we assume that the only possible words are the words from a vocabulary, then for a uniform distribution the perplexity would be equal to the number of words $exp(n * log(1/n)/n) = n = 6906$.

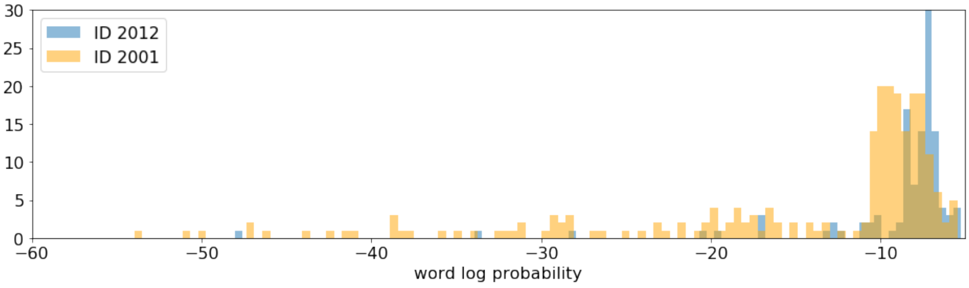


Figure 4: Distribution of word log probabilities of all words in documents ID 2001 (high perplexity: 4373) and ID 2012 (low perplexity: 947).

4 Part D

In BMM we hope to achieve **lower perplexity** by tailoring the language model to specific, latent topic-categories $z_d \in \{1, ..., K\}$. However, with K categories and D training documents, the posterior in Equation 9 is intractable, as there are K^D possible configurations for latent variables.

$$p(\{z_d\}, \boldsymbol{\theta}, \boldsymbol{\beta_k} \mid \{w_{nd}\}, \gamma, \alpha) = \frac{p(\{w_{nd}\} \mid \boldsymbol{\beta}, \{z_d\})p(\{z_d\} \mid \boldsymbol{\theta})p(\boldsymbol{\beta_k} \mid \gamma)p(\boldsymbol{\theta} \mid \alpha)}{p(\{w_{nd}\}} \tag{9}$$

Therefore, in BMM.py, we marginalise $\boldsymbol{\theta}$ and $\boldsymbol{\beta_k}$ and perform **Colapsed Gibbs sampling**. We sequentially sample category assignment for each document using conditional distribution in Equation 11.

$$\int\int p(\{z_d\}, \boldsymbol{\theta}, \boldsymbol{\beta_k} \mid \{w_{nd}\}, \gamma, \alpha) d\boldsymbol{\theta} d\boldsymbol{\beta_k} = p(\{z_d\} \mid \{w_{nd}\}, \gamma, \alpha) \tag{10}$$

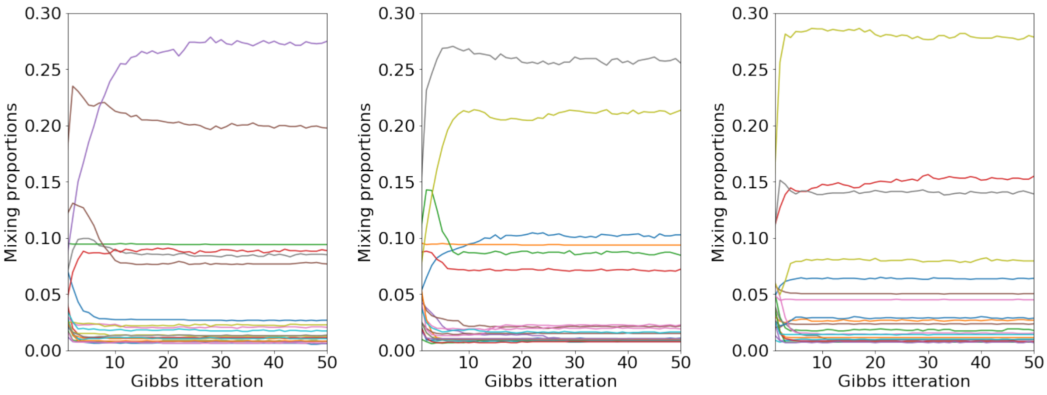
$$p(z_d = k \mid \{w_d\}, \{z_{-d}\}, \gamma, \alpha) \propto \underbrace{\prod_{m=1}^{M} \left(\frac{c^k_{-d,m} + \gamma}{\sum_{i=1}^{M} (c^k_{-d,i} + \gamma)} \right)^{c_{d,m}}}_{\textit{Language similarity}} \times \underbrace{\frac{c^k_{-d} + \alpha}{\sum_{j=1}^{K} (c^j_{-d} + \alpha)}}_{\textit{Category presence}} \tag{11}$$

Distribution of mixing proportions $\{\theta_k\}$ directly depends on evolution of Equation 11.

$$\theta_k = \frac{c^k_d + \alpha_k}{\sum_{j=1}^{K} (c^j_d + \alpha_j)} \tag{12}$$

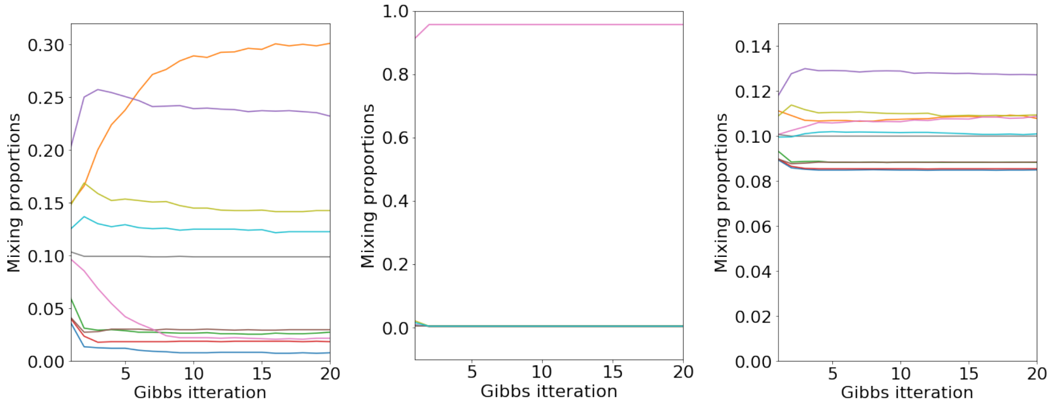
Probability of $z_d = k$ depends on two factors. **Language similarity** factor measures how similar document d is to other documents assigned to category k. **Category presence** factor takes high values for categories which have many documents assigned to them.

Every Gibbs iteration, we attempt to improve clustering of similar documents together, prioritising their assignment into categories which have already high presence among all documents. However, mixing proportions will converge to different distributions depending on the initial category assignments for documents (rich get richer phenomenon), Figure 5. Choice of $\boldsymbol{\gamma}$ and $\boldsymbol{\alpha}$ will also determine the final shape of the posterior, Figure 6.



(a) Initialisation 1 (b) Initialisation 2 (c) Initialisation 3

Figure 5: Initialisation of document assignment categories determines the direction in which topic-category k will evolve, including its final semantic definition as well as the likelihood of having assigned more documents in the future iterations.



(a) $\alpha = 10$, $\gamma = 0.1$ (b) $\alpha = 10$, $\gamma = 100$ (c) $\alpha = 1000$, $\gamma = 0.1$

Figure 6: Choice of α and γ affects posterior mixing proportions. Increasing $\boldsymbol{\gamma}$ makes documents' topics appear more uniform and they are assigned the same category. Increasing $\boldsymbol{\alpha}$ makes documents less dependent on categories of other documents, thus model maintains more topic-categories.

Convergence of mixing proportions to different local optima means different semantic definitions for final topic-clusters. However the quality of language model, quantified by perplexity, is similar in each case, Figure 7. While more Gibbs iterations could enable convergence to a true posterior, the quality of models after a few iterations is sufficient for a useful practical approximation.

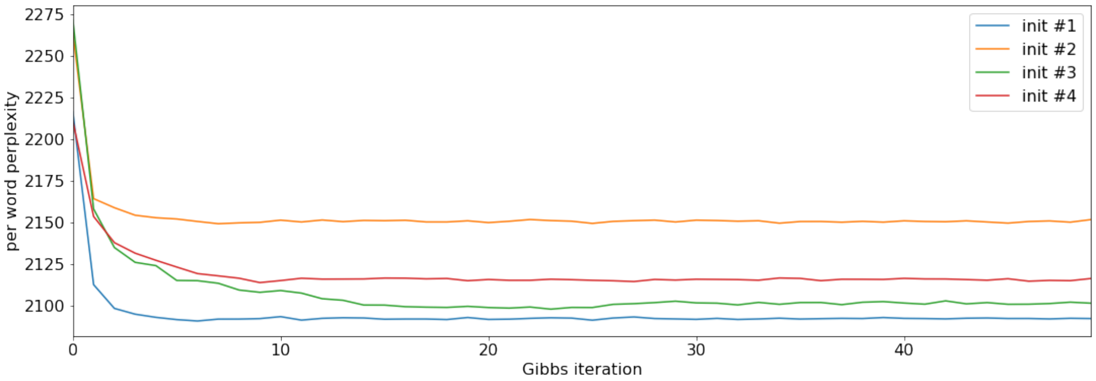


Figure 7: Gibbs sampler with different initial document assignments (different random seed) reach similar Per word Perplexity.

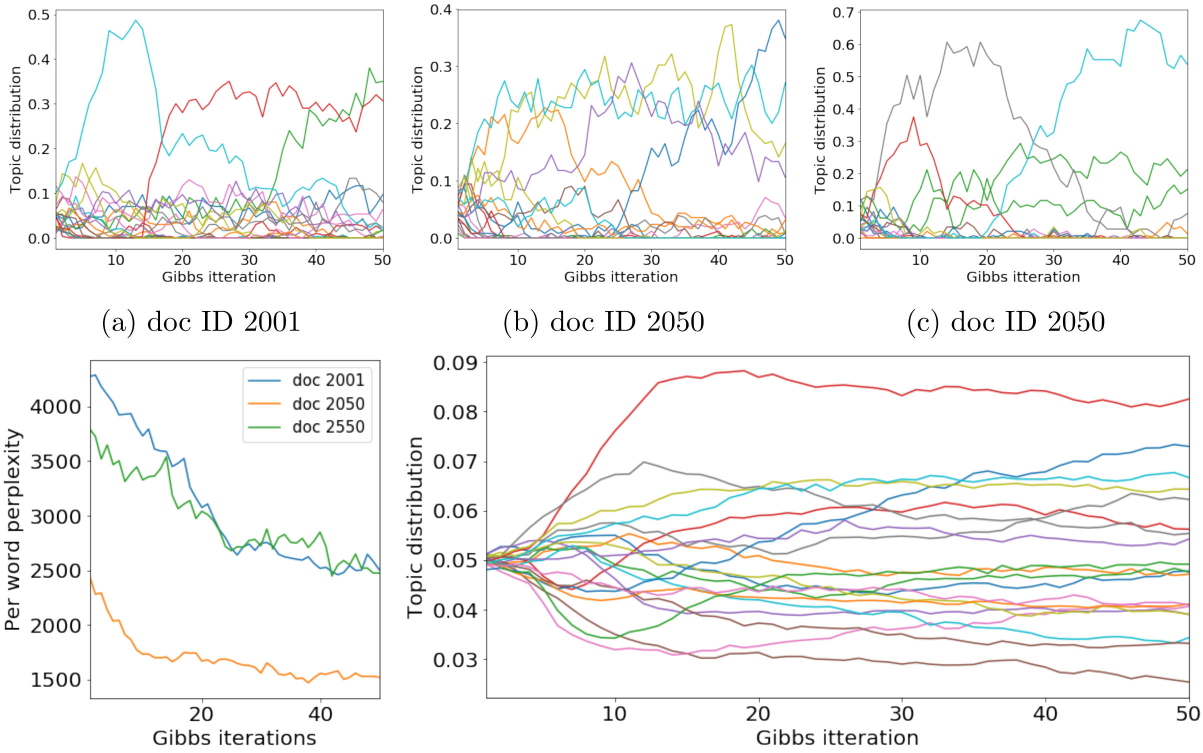
5 Part E

Documents may use language across mixture of topics, thus to allow soft assignment over topic-categories, LDA expands dimensions of latent parameters and mixing proportions.

$$(BMM) \quad \underset{Dx1}{\{z_d\}}, \quad \underset{Kx1}{\{\theta_d\}} \quad \longrightarrow \quad (LDA) \quad \underset{DxK}{\{z_d\}}, \quad \underset{KxD}{\{\theta_d\}} \tag{13}$$

5.1 Topic distribution

In BMM we saw that depending on initialisation (and α, γ), the model clustered documents into only a few topic-categories. Figure 8.e shows that LDA retains presence of all K=20 categories. This is because soft category assignment allows LDA to cluster documents across many dimensions. Document ID 2001, Figure 8.a, can be clustered close to documents with high **red** topic-component and/or high **green** topic-component, rather than having to choose only one, as in BMM.



(a) doc ID 2001

(b) doc ID 2050

(c) doc ID 2050

(d) perplexity

(e) all documents in B

Figure 8: Topic distribution evolution with Gibbs iterations. A, B, C are topic distribution for single documents. D shows how topic-specification reduces their perplexity. E is topic distribution computed for all documents in B.

5.2 Perplexity

LDA has achieved best perplexity, because the soft assignment of documents, enables it to capture finer characteristics of language in each document. This in turn allows for more accurate document clustering, thus improving quality of language model.

	no latent variables	BMM	LDA
per-word perplexity	2683	2098	1631

Table 2: Comparison of per word perplexity values for each model, evaluated for all documents in B.

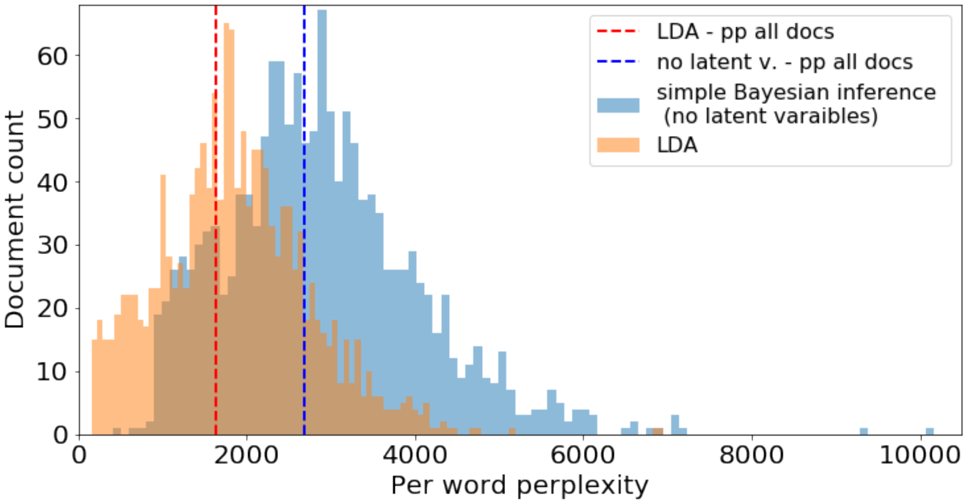


Figure 9: Comparison of perplexity distributions, evaluated for each document in test set B, for simple Bayesian inference model and LDA.

Figure 10 shows that after 50 Gibbs iterations perplexity reaches proximity of asymptote, suggesting that more iterations wouldn't significantly improve quality of the model.

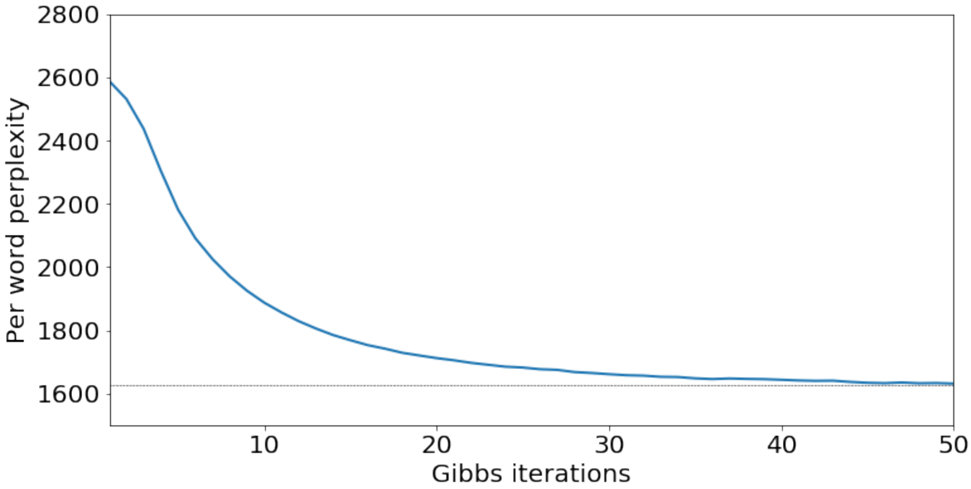


Figure 10: Per word perplexity for LDA decreases with number of Gibbs iterations.

5.3 Entropy

Entropy is a measure of average uncertainty about next word or alternatively the amount of surprise. Here computed in units nats/word.

$$H(\beta_k) = -\sum_{i=1}^{M} \beta_{k,i} log(\beta_{k,i}) \tag{14}$$

Figure 11 shows that entropy has decreased for all topics. Every iteration each topic-category becomes more semantically defined. Documents with similar language are assigned to same topic-categories and therefore the word distribution for each category becomes better defined (more uniform).

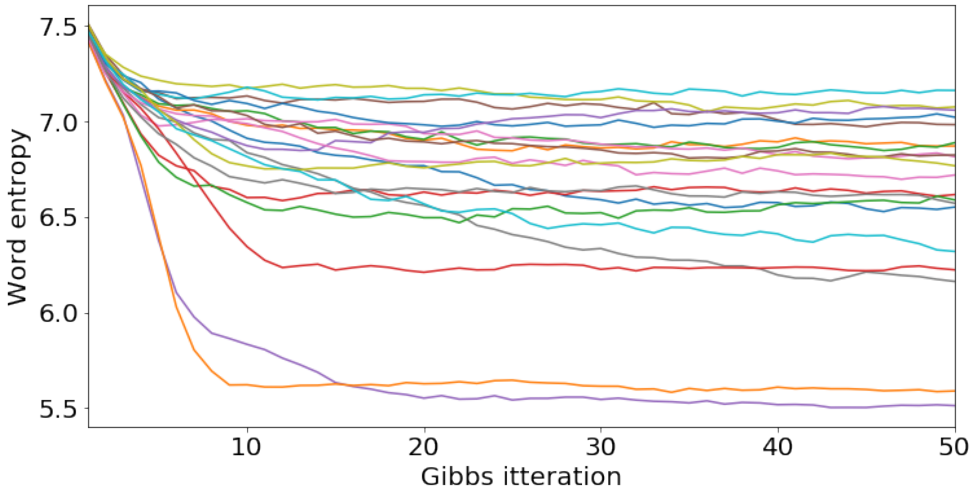


Figure 11: Entropy for each topic-category decreases with number of Gibbs iterations.

References

[1] MURAWAKI Yugo, Categorical Distributions in Natural Language Processing, http://murawaki.org/misc/cat.pdf

[2] Murphy, K.P. and Massachusetts Institute Of Technology (2012). Machine learning: a probabilistic perspective. Cambridge (Ma): Mit Press.

[3] Eric P. Xing, Lecture 19: Bayesian Nonparametrics: Dirichlet Processes, 10-708: Probabilistic Graphical Models 10-708, Spring 2014
www.cs.cmu.edu/ epxing/Class/10708-14/scribe_notes/scribe_note_lecture19.pdf