Dimensionality Reduction and Clustering of Text Documents

Freddy Chong Tat Chua
School of Information Systems
Singapore Management University
80 Stamford Road
Singapore 178902
freddy.chua.2009@smu.edu.sg

ABSTRACT

The human language is inherently unstructured. Human authors never follow the strict rules of grammar because their human readers possess cognitive abilities to interpret the semantic meaning of unstructured human written text. Since unstructured text possess ambiguities and uncertainties, using probabilities to model human language is a natural choice. The investigation of a widely used probabilistic model is the motivation of my survey here. This probabilistic model known as Latent Dirichlet Allocation (LDA), has seen widespread adoption in the field of information retrieval. As prelude to the introduction of LDA, I shall review the Latent Semantic Analysis (LSA) and Probabilistic Latent Semantic Analysis (PLSA) for historical interests. Finally, I briefly discuss Hierarchical Latent Dirichlet Allocation.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Clustering

General Terms

Probabilities and Statistics, Algorithms and Experimentation

Keywords

Bayesian Models, Gibbs Sampling, Singular Value Decomposition

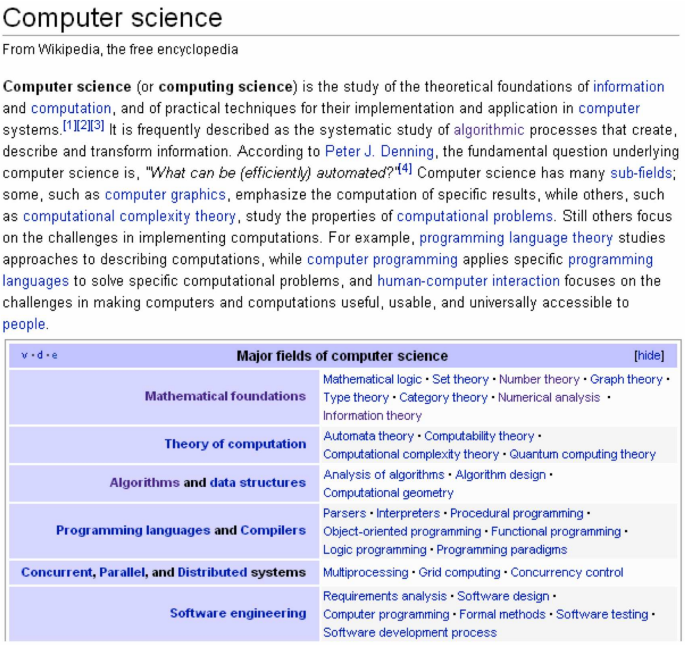
1. INTRODUCTION

Information Retrieval (IR) as a field emerged in computer science when authors started storing their documents in digital format. IR importance increased after wide spread usage of internet. Since there are millions of webpages on the internet, users need a quick and simple way of finding web pages that are relevant to their information need. To fulfill the information need, internet search engines emerged to provide search service for users. Traditionally, these search engines were built using deterministic mathematical approaches such as set theory, logic, linear algebra and combinatorics. However, unlike computer programming languages which follows strict grammars and syntax, the common human language is unstructured with great amount of ambiguity. The unstructured nature of human language makes it difficult for search engines to understand the semantic meaning of web pages. To overcome the difficulty of understanding human languages, an IR related field, Natural Language Processing (NLP) emerged. Despite NLP progress, their scalability for understanding the huge amount of text on the web is still intractable. To make computation tractable for understanding text documents, IR researchers have treated documents using a simple approach known as bag-of-words. Using the bag-of-words approach, IR researchers have been able to cluster documents based on similarities between them.

A notable technique which uses the bag-of-words assumption for document classification of a text corpus is Latent Dirichlet Allocation (LDA) proposed by Blei et al [2]. The success of LDA in the research community extends far beyond the field of Information Retrieval. LDA has also seen wide application in related fields such as data mining and multimedia classification. My survey here is motivated by the importance of LDA and the bayesian principles used in LDA. To gain a thorough understanding of LDA, it is necessary to appreciate the predecessors of LDA, namely Latent Semantic Analysis (LSA) [4] and Probabilistic Latent Semantic Analysis (PLSA) [9].

Despite the success of these clustering techniques, it is not clear how many clusters should be assigned to for a text corpus. The naive approach performs the clustering repeatedly by varying the number of clusters and observing the scores of precision, recall and perplexity. To address the issue of variable clusters, Blei et al proposed the Hierarchical Latent Dirichlet Allocation (hLDA) based on principles of nonparametric bayesian techniques [1]. Besides being able to vary the number of clusters, hLDA also model subclusters which meant that documents belonging to a cluster can be further classified into specialized clusters. A close look at the the Computer Science article in Figure 1 also reflects this phenomenon.

The paper is organized as follows, we give a derivation of LSA and show a basic folding-in technique for answering IR queries in Section 2. Section 3 introduces PLSA for modeling



Computer science

From Wikipedia, the free encyclopedia

Computer science (or **computing science**) is the study of the theoretical foundations of information and computation, and of practical techniques for their implementation and application in computer systems.[1][2][3] It is frequently described as the systematic study of algorithmic processes that create, describe and transform information. According to Peter J. Denning, the fundamental question underlying computer science is, *"What can be (efficiently) automated?"*[4] Computer science has many sub-fields; some, such as computer graphics, emphasize the computation of specific results, while others, such as computational complexity theory, study the properties of computational problems. Still others focus on the challenges in implementing computations. For example, programming language theory studies approaches to describing computations, while computer programming applies specific programming languages to solve specific computational problems, and human-computer interaction focuses on the challenges in making computers and computations useful, usable, and universally accessible to people.

v • d • e	Major fields of computer science [hide]
Mathematical foundations	Mathematical logic • Set theory • Number theory • Graph theory • Type theory • Category theory • Numerical analysis • Information theory
Theory of computation	Automata theory • Computability theory • Computational complexity theory • Quantum computing theory
Algorithms and data structures	Analysis of algorithms • Algorithm design • Computational geometry
Programming languages and Compilers	Parsers • Interpreters • Procedural programming • Object-oriented programming • Functional programming • Logic programming • Programming paradigms
Concurrent, Parallel, and Distributed systems	Multiprocessing • Grid computing • Concurrency control
Software engineering	Requirements analysis • Software design • Computer programming • Formal methods • Software testing • Software development process

Figure 1: Snippet of Computer Science article in Wikipedia

text documents. After reading PLSA, readers should be able to appreciate the extension of PLSA to LDA in Section 4.2. Section 5 will talk briefly about hLDA. Finally, we end the paper with a conclusion in Section 6 and discuss what I have learned in this survey.

2. LATENT SEMANTIC ANALYSIS

Latent Semantic Indexing is a linear algebraic technique that uses Singular Value Decomposition(SVD) to factorize a given matrix of term-document occurrence [4]. I will describe the factorization process of SVD, then show an example of how SVD can be used to improve retrieval results.

2.1 Singular Value Decomposition

Definition 1. Suppose we have t number of terms and d number of documents in our text corpus. We can form a term by document matrix X of size $t \times d$. SVD decomposes X into the product of three matrices $X = TSD^T$, where T is a $t \times d$ matrix, S is a $d \times d$ matrix and D^T is a $d \times d$ matrix. T and D^T are orthogonal matrices while S is a diagonal matrix.

Lemma 1. Multiplying matrix X by its own transpose, always yields a square matrix. XX^T is a square matrix of size $t \times t$. X^TX is a square matrix of size $d \times d$.

THEOREM 1. *The square matrices XX^T and X^TX can always be factorized into their corresponding eigenvectors and eigenvalues. We will not discuss the proof in detail here but interested readers might want to refer to [11] for a detailed explanation.*

We now show how to derive the matrices T, S and D^T.

$$\begin{aligned} X &= TSD^T \\ X^T &= (TSD^T)^T \\ &= DS^TT^T \\ XX^T &= (TSD^T)(DS^TT^T) \\ &= TS(D^TD)S^TT^T \end{aligned}$$

If we constrain D to be orthogonal, then $D^T = D^{-1}$. Hence,

$$\begin{aligned} XX^T &= TS(D^TD)S^TT^T \\ &= TS(D^{-1}D)S^TT^T \\ &= TSS^TT^T \end{aligned}$$

If we constrain S to be a diagonal matrix, then $S = S^T$. Hence,

$$\begin{aligned} XX^T &= TSS^TT^T \\ &= TS^2T^T \end{aligned}$$

Doing the same manipulations for X^TX and constraining T to be orthogonal allow us to obtain

$$\begin{aligned} X^TX &= DS^TT^TTSD^T \\ &= DS^2D^T \end{aligned}$$

From Theorem 1, obtaining the eigenvectors of XX^T and X^TX gives us T and D^T respectively. Since LSA have been mentioned in class, I will skip the examples of LSA.

2.2 Folding-in

Suppose we have a query q and we want to retrieve the relevant documents in the reduced semantic space. First we have to map q into the reduced semantic space d_q.

$$\begin{aligned} q &= T_rS_rd_q^T \\ T_r^Tq &= T_r^TT_rS_rd_q^T \\ S_r^{-1}T_r^Tq &= S_r^{-1}S_rd_q^T \\ S_r^{-1}T_r^Tq &= d_q^T \\ d_q &= (S_r^{-1}T_r^Tq)^T \\ &= q^TT_rS_r^{-1} \end{aligned}$$

We can also apply the same method to update the matrices without recomputing SVD. Suppose we have a updated term document matrix $X^{(t+m)\times(d+q)}$ where q represents number of new documents and m represents number of new terms in these documents. We can first compute the updated matrices $D_{(d+q)\times r}$ using the current $T_{t\times r}$ and $S_{r\times r}$ matrices where r is the dimension of the reduced space.

$$\begin{aligned} X_{t\times q} &= T_{t\times r}S_{r\times r}D_{r\times q}^T \\ D_{q\times r} &= X_{q\times t}^TT_{t\times r}S_{r\times r}^{-1} \end{aligned}$$

Using the computed $D_{q\times r}$, we append it to the current $D_{d\times r}$ matrix to obtain a $D_{(d+q)\times r}$ matrix. Then we compute the new $T_{(t+m)\times r}$ matrix.

$$\begin{aligned} X_{m\times q} &= T_{m\times r}S_{r\times r}D_{r\times q}^T \\ X_{m\times q}D_{q\times r}S_{r\times r}^{-1} &= T_{m\times r} \\ T_{m\times r} &= X_{m\times q}D_{q\times r}S_{r\times r}^{-1} \end{aligned}$$

We also append the computed $T_{m\times r}$ to $T_{t\times r}$ and obtain a $T^{(t+m)\times r}$ matrix. Finally we have shown that the updated

term document matrix $X^{(t+m)\times(d+q)}$ can be factorized in this manner,

$$X^{(t+m)\times(d+q)} = T^{(t+m)\times r} S_{r\times r} D^T_{r\times(d+q)}$$

3. PROBABILISTIC LATENT SEMANTIC ANALYSIS

Definition 2. Suppose we have M number of documents and V number of words in the vocabulary. Let d_m be the index for each document and w_v be the occurence of word v in the document d_m. We define a joint probability model $P(d_m, w_v)$ as the probability of observing word v in the document d_m. As we have shown earlier in LSA, it is possible to represent words and documents in latent space. PLSA achieves the same effect by introducing a latent variable z_k that denotes the topic of each word in a document, where K is the number of topics in the corpus and $1 \leq k \leq K$. PLSA makes several assumptions to define the model.

1. Given a document d_m in the corpus, we are able to infer the topic z_k of the document, $P(z_k|d_m)$.

2. Given a topic, we are able to infer which word v is likely to appear, $P(w_v|z_k)$.

With the introduction of this latent topic variable z, we can define the joint probability of observing a word in the document as $P(d_m, w_v, z_k)$, where

$$P(d_m, w_v, z_k) = P(w_v|z_k)P(z_k|d_m)P(d_m)$$

Since z_k is latent and we can only observe d_m and w_v, then we need to marginalize z_k out of the equation by summing over all possible values that z_k can take.

$$\begin{aligned} P(d_m, w_v) &= \sum_{k=1}^{K} P(d_m, w_v, z_k) \\ &= \sum_{k=1}^{K} P(w_v|z_k)P(z_k|d_m)P(d_m) \\ &= \sum_{k=1}^{K} P(w_v|z_k)P(z_k)P(d_m|z_k) \end{aligned}$$

We will like to define the probability of the entire corpus $P(d, w)$ where

$$\begin{aligned} P(d, w) &= \prod_{m=1}^{M}\prod_{v=1}^{V} P(d_m, w_v) \\ &= \prod_{m=1}^{M}\prod_{v=1}^{V}\sum_{k=1}^{K} P(w_v|z_k)P(z_k)P(d_m|z_k) \end{aligned}$$

Before we proceed on, we will like to show its relation to LSA. Suppose we assume that in our corpus in Table 1, our documents consists of two main topics, then what we want to find is the conditional probability tables here as shown in Table 2. These conditional probabilities bear similarities to the LSA formulation for a semantic space of two dimension. These values should be filled in to maximize the likelihood value $P(d, w)$. In statistics, it is computationally more tractable to maximize the log equivalent of likelihood

Table 1: An example for Bayesian Networks

d_1:	Learning Bayesian Networks
d_2:	Probabilistic Graphical Models
d_3:	Learning in Graphical Models
d_4:	Modern Information Retrieval
d_5:	Introduction to Data Mining
d_6:	Data Management and Information Retrieval

Terms Documents Matrix, X

Terms	Documents					
	d_1	d_2	d_3	d_4	d_5	d_6
networks	1	0	0	0	0	0
learning	1	0	1	0	0	0
bayesian	1	0	0	0	0	0
graphical	0	1	1	0	0	0
models	0	1	1	0	0	0
information	0	0	0	1	0	1
retrieval	0	0	0	1	0	1
data	0	0	0	0	1	1
mining	0	0	0	0	1	0

Table 2: The Conditional Probability Tables

$P(d_i\|z_k)$	d_1	d_2	d_3	d_4	d_5	d_6
z_1	?	?	?	?	?	?
z_2	?	?	?	?	?	?

$P(w_j\|z_k)$	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
z_1	?	?	?	?	?	?	?	?	?
z_2	?	?	?	?	?	?	?	?	?

$P(z_1)$	?
$P(z_2)$	?

values, hence, the log likelihood $\mathcal{L}$ is often expressed in this manner,

$$\begin{aligned} \mathcal{L} &= \log P(d, w) \\ &= \log \prod_{m=1}^{M}\prod_{v=1}^{V}\sum_{k=1}^{K} P(w_v|z_k)P(z_k)P(d_m|z_k) \\ &= \sum_{m=1}^{M}\sum_{v=1}^{V} n(d_m, w_v) \log \sum_{k=1}^{K} P(w_v|z_k)P(z_k)P(d_m|z_k) \end{aligned}$$

where $n(d_m, w_v)$ is the number of times word w_v occur in document d_m. Maximizing the likelihood here by filling in the conditional probability tables is essentially an optimization problem. For maximizing likelihood, the Expectation Maximization algorithm often used for computing these value [5]. We shall skip the full derivation here but interested readers may refer to [10]. We shall follow the equations as described in [9] to compute these values. In the E step of the algorithm, we first compute $P(z_k|d_m, w_v)$.

$$P(z_k|d_m, w_v) = \frac{P(z_k)P(d_m|z_k)P(w_v|z_k)}{\sum_{l=1}^{K} P(z_l)P(d_m|z_l)P(w_v|z_l)} \quad (1)$$

In the M step of the algorithm, we compute the quantities $P(w_v|z_k), P(d_m|z_k), P(z_k)$.

$$P(w_v|z_k) = \frac{\sum_{m=1}^{M} n(d_m, w_v)P(z_k|d_m, w_v)}{\sum_{m=1}^{M}\sum_{n=1}^{N} n(d_v, w_n)P(z_k|d_v, w_n)} \quad (2)$$

$$P(d_m|z_k) = \frac{\sum_{v=1}^{V} n(d_m, w_v)P(z_k|d_m, w_v)}{\sum_{n=1}^{N}\sum_{v=1}^{V} n(d_n, w_v)P(z_k|d_n, w_v)} \quad (3)$$

$$P(z_k) = \frac{\sum_{m=1}^{M}\sum_{v=1}^{V} n(d_m, w_v)P(z_k|d_m, w_v)}{\sum_{m=1}^{M}\sum_{v=1}^{V} n(d_m, w_v)} \quad (4)$$

After running the EM algorithm, we get such values in Table 3.

Table 3: The Conditional Probability Tables

$P(d_i\|z_k)$	d_1	d_2	d_3	d_4	d_5	d_6
z_1	0	0	0	0.29	0.29	0.43
z_2	0.38	0.25	0.38	0	0	0

$P(w_j\|z_k)$	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
z_1	0	0	0	0	0	0.29	0.29	0.29	0.14
z_2	0.13	0.25	0.13	0.25	0.25	0	0	0	0

$P(z_1)$	0.47
$P(z_2)$	0.53

Since PLSA is a probabilistic technique, the results of small samples may not always be consistent. Therefore, we ran experiments on a corpus of a larger size. Using text corpus from wikipedia articles for Computer science, Information theory, Cryptography, Cryptanalysis, History of cryptography, Database, Data mining and Information retrieval. We run PLSA on these articles. Then we rank the words and obtain the following sets of words for each topic. Refer to Table 4 for the results.

Table 4: List of words by topics using PLSA and LDA

PLSA Topic 1	LDA Topic 1	PLSA Topic 2	LDA Topic 2
data	data	cipher	comput
databas	databas	kei	inform
mine	comput	cryptographi	cipher
pattern	inform	inform	kei
relat	mine	us	cryptographi
us	system	encrypt	us
model	model	attack	system
object	scienc	algorithm	algorithm
system	retriev	secur	theori
inform	pattern	cryptanalysi	attack
applic	us	cryptograph	encrypt
set	document	code	secur
transact	object	messag	cryptanalysi
lock	relat	channel	cryptograph
manag	program	comput	code
3	applic	system	messag
program	queri	theori	channel
exampl	3	commun	scienc
oper	transact	entropi	commun
document	set	develop	entropi

4. LATENT DIRICHLET ALLOCATION

Because of the heavy use of probabilities and symbolic notations here, I will first like to go through the basics of probabilities. I introduce these notations here instead of the previous section because PLSA is significantly simpler than LDA. I also provide a toy example using Dice Toss to illustrate the basic principles of bayesian parameter learning. It is important to understand this section before moving on to the section on LDA model.

4.1 Probabilities and Identities

$$\begin{aligned}
\Gamma(x) &= (x-1)! \\
\int_0^1 f^{a-1}(1-f)^{b-1}\, df &= \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} \\
\int_0^1 \prod_{i=1}^{N} f_i^{a_i-1}\, df &= \frac{\prod_{i=1}^{N}\Gamma(a_i)}{\Gamma(\sum_{i=1}^{N} a_i)} \\
B(a) &= \frac{\prod_{i=1}^{N}\Gamma(a_i)}{\Gamma(\sum_{i=1}^{N} a_i)}
\end{aligned}$$

4.1.1 Dice Toss Toy Example

Suppose I have a dice of I sides. I toss the dice and the probability of landing on side i is $p(s=i|f) = f_i$. I throw the dice N number of times and obtain a set of results s where $s = (s_1, s_2, s_3, \ldots, s_n, \ldots, s_N)$. We can specify the probability of observing this set of results in this manner,

$$\begin{aligned}
p(s|f) &= \prod_{n=1}^{N} p(s_n|f) \\
&= f_1^{n_1} f_2^{n_2} f_3^{n_3} \ldots f_i^{n_i} \ldots f_I^{n_I} \\
&= \prod_{i=1}^{I} f_i^{n_i} \quad (5)
\end{aligned}$$

where n_i denotes the number of times side i appear. Notice that the $\prod_{n=1}^{N}$ has been changed to $\prod_{i=1}^{I}$. The $\prod_{n=1}^{trials}$ has been expressed as $\prod_{i=1}^{states}$.

Suppose that f is a dirichlet distribution with a as hyperparameters. Then we express the probability distribution of f as,

$$p(f|a) = \frac{\Gamma(\sum_{i=1}^{I} a_i)}{\prod_{i=1}^{I}\Gamma(a_i)} \prod_{i=1}^{I} f_i^{a_i-1}$$

If we want to update the parameter f base on the observation of s, then we can express f in this manner.

$$\begin{aligned}
p(f|s,a) &= \frac{p(s|f,a)p(f|a)}{\int_0^1 p(s|f,a)p(f|a)\, df} \\
&= \frac{\prod_{i=1}^{I} f_i^{n_i} \frac{\Gamma(\sum_{i=1}^{I} a_i)}{\prod_{i=1}^{I}\Gamma(a_i)} \prod_{i=1}^{I} f_i^{a_i-1}}{\int_0^1 \prod_{i=1}^{I} f_i^{n_i} \frac{\Gamma(\sum_{i=1}^{I} a_i)}{\prod_{i=1}^{I}\Gamma(a_i)} \prod_{i=1}^{I} f_i^{a_i-1}\, df} \\
&= \frac{\frac{\Gamma(\sum_{i=1}^{I} a_i)}{\prod_{i=1}^{I}\Gamma(a_i)} \prod_{i=1}^{I} f_i^{n_i+a_i-1}}{\frac{\Gamma(\sum_{i=1}^{I} a_i)}{\prod_{i=1}^{I}\Gamma(a_i)} \int_0^1 \prod_{i=1}^{I} f_i^{n_i+a_i-1}\, df} \\
&= \frac{\Gamma(\sum_{i=1}^{I} n_i + a_i)}{\prod_{i=1}^{I}\Gamma(n_i+a_i)} \prod_{i=1}^{I} f_i^{n_i+a_i-1}
\end{aligned}$$

Notice that after updating f base on s observations, f is still a dirichlet distribution. This property is known as conjugate priors. Base on this property, estimating the parameters f_i after observing n trials is a simple counting procedure. Suppose I want to obtain f_x from f =

$(f_1, f_2, \ldots, f_x, \ldots, f_i, \ldots, f_I)$,

$$
\begin{aligned}
&E(f_x|s,a) \\
&= \int_0^1 f_x p(f|s,a)\ df \\
&= \int_0^1 f_x p(f|s,a)\ df \\
&= \int_0^1 f_x \frac{\Gamma(\sum_{i=1}^I n_i + a_i)}{\prod_{i=1}^I \Gamma(n_i + a_i)} \prod_{i=1}^I f_i^{n_i+a_i-1}\ df \\
&= \frac{\Gamma(\sum_{i=1}^I n_i + a_i)}{\prod_{i=1}^I \Gamma(n_i + a_i)} \int_0^1 f_x^{n_x+a_x} \prod_{i=1, i\neq x}^I f_i^{n_i+a_i-1}\ df \\
&= \frac{\Gamma(\sum_{i=1}^I n_i + a_i)}{\prod_{i=1}^I \Gamma(n_i + a_i)} \frac{\Gamma(n_x + a_x + 1) \prod_{i=1, i\neq x}^I \Gamma(n_i + a_i)}{\Gamma(n_x + a_x + 1 + \sum_{i=1, i\neq x}^I n_i + a_i)} \\
&= \frac{\Gamma(\sum_{i=1}^I n_i + a_i)\Gamma(n_x + a_x + 1)}{\Gamma(n_x + a_x)\Gamma(1 + \sum_{i=1}^I n_i + a_i)} \\
&= \frac{n_x + a_x}{\sum_{i=1}^I n_i + a_i} \qquad (6)
\end{aligned}
$$

Now I show the Collapsed Gibbs Sampling for this toy example. Suppose we want to obtain f but do not have the counts of occurence n_i. We can generate the samples s_n using f by deriving a markov chain, $p(s_n|s_{-n}, a)$, where s_{-n} represents all samples except s_n. Using the above results,

$$
\begin{aligned}
p(s_n = i|s_{-n}, a) &= E(f_i|s_{-n}, a) \\
&= \frac{n_i + a_i}{(\sum_{i=1}^I n_i + a_i)} \qquad (7)
\end{aligned}
$$

I will show later in Algorithm 1 how we adjust the parameters by excluding the results of s_n in s_{-n}.

4.2 LDA Model

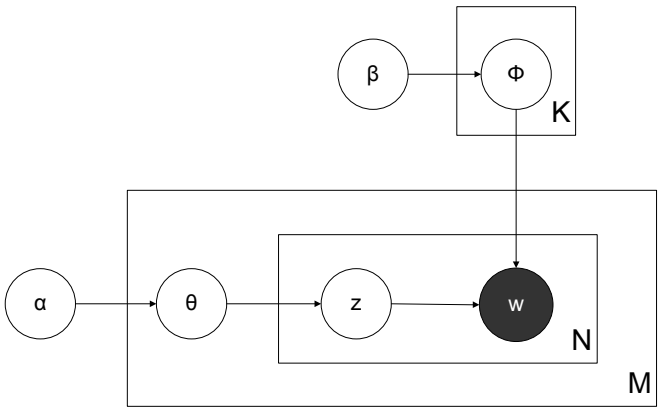


Figure 2: Latent Dirichlet Allocation

Refer to Figure 2 for a graphical description of LDA in plate notation form.

Definition 3. Suppose we have M documents and N words in document m. Let θ_m represent the dirichlet distribution of topics in each document m and α are the hyper-parameters of the dirichlet distribution. Let $w_{m,n}$ be the index of the word n in document m and $z_{m,n}$ be the topic k of each word $w_{m,n}$. Let ϕ_k represent the dirichlet distribution of words over topic k with β as hyper-parameters.

Formally, the LDA model can be described mathematically in this manner,

$$
\begin{aligned}
\alpha &= (\alpha_1, \ldots, \alpha_k, \ldots, \alpha_K) \\
p(\theta_m|\alpha) &= \frac{\Gamma(\sum_{k=1}^K \alpha_k)}{\prod_{k=1}^K \Gamma(\alpha_k)} \theta_{m,1}^{\alpha_1-1} \ldots \theta_{m,K}^{\alpha_K-1} \\
&= \frac{\Gamma(\sum_{k=1}^K \alpha_k)}{\prod_{k=1}^K \Gamma(\alpha_k)} \prod_{k=1}^K \theta_{m,k}^{\alpha_k-1} \\
\beta &= (\alpha_1, \ldots, \alpha_v, \ldots, \alpha_V) \\
p(\phi_k|\beta) &= \frac{\Gamma(\sum_{v=1}^V \beta_v)}{\prod_{v=1}^V \Gamma(\beta_v)} \phi_{k,1}^{\beta_1-1} \ldots \phi_{k,V}^{\beta_V-1} \\
&= \frac{\Gamma(\sum_{v=1}^V \beta_v)}{\prod_{v=1}^V \Gamma(\beta_v)} \prod_{v=1}^V \phi_{k,v}^{\beta_v-1}
\end{aligned}
$$

The discrete values $z_{m,n}$ depends on the topic distribution of each document m.

$$p(z_{m,n} = k|\theta_m) = \theta_{m,k}$$

The probability of observing each word $w_{m,n}$ in document m depends on the discrete topic $z_{m,n}$ and word distribution ϕ.

$$p(w_{m,n} = v|z_{m,n} = k, \phi) = \phi_{k,v}$$

The joint probability of observing a word is,

$$
\begin{aligned}
&p(w_{m,n}, z_{m,n}, \phi, \theta_m|\alpha, \beta) \\
&= p(w_{m,n}|z_{m,n}, \phi) p(z_{m,n}|\theta_m) p(\phi|\beta) p(\theta_m|\alpha)
\end{aligned}
$$

The joint probability of observing a corpus is

$$
\begin{aligned}
&p(w, z, \phi, \theta|\alpha, \beta) \\
&= \prod_{m=1}^M \prod_{n=1}^N p(w_{m,n}, z_{m,n}, \phi, \theta_m|\alpha, \beta) \\
&= \prod_{m=1}^M p(\theta_m|\alpha) \prod_{n=1}^N p(z_{m,n}|\theta_m) p(w_{m,n}|z_{m,n}, \phi)
\end{aligned}
$$

What we want to solve for are the parameters θ and ϕ. There are many ways of solving this model. I present a collapsed Gibbs Sampling approach to find these parameters [7, 8].

4.3 Gibbs Sampling

Gibbs Sampling is commonly used when it is hard to derive an analytical expression of a single variable from the joint probability distribution. Our main objective is to obtain,

$$
\begin{aligned}
\theta &= (\theta_{1,1}, \theta_{1,2}, \ldots, \theta_{m,k}, \ldots, \theta_{M,K}) \\
\theta_{m,k} &= \frac{n(m,k) + \alpha_k}{\sum_{k=1}^K n(m,k) + \alpha_k} \qquad (8) \\
\phi &= (\phi_{1,1}, \phi_{1,2}, \ldots, \phi_{k,v}, \ldots, \phi_{K,V}) \\
\phi_{k,v} &= \frac{n(k,v) + \beta_v}{\sum_{v=1}^V n(k,v) + \beta_v} \qquad (9)
\end{aligned}
$$

where $n_{k,v}$ is the number of times word v has been assigned topic k and here $n_{m,k}$ is the number of times topic k appear in document m. But n(m,k) and n(k,v) are unknown to us now.

The Gibbs Sampling algorithm allows us to reduce the parameter estimation problem to a simple counting and sampling process. We will like to derive a sampling, counting

and update procedure for estimating the parameters and latent variables.

4.3.1 Collapsed Gibbs Sampling

Following the approach in [7, 8], we integrate the ϕ and θ parameters to obtain the following,

$$p(z, w|\alpha, \beta) = p(w|z, \beta)p(z|\alpha)$$

$$\begin{aligned}
p(w|z,\beta) &= \int p(w,\phi|z,\beta)\ d\phi \\
&= \int p(w|z,\phi,\beta)p(\phi|z,\beta)\ d\phi \\
&= \int p(w|z,\phi)p(\phi|\beta)\ d\phi \\
&= \int \prod_{k=1}^{K}\left(\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}}\right)\frac{\Gamma(\sum_{v=1}^{V}\beta_v)}{\prod_{v=1}^{V}\Gamma(\beta_v)}\left(\prod_{v=1}^{V}\phi_{k,v}^{\beta_v-1}\right)d\phi \\
&= \int \prod_{k=1}^{K}\frac{\Gamma(\sum_{v=1}^{V}\beta_v)}{\prod_{v=1}^{V}\Gamma(\beta_v)}\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}+\beta_v-1}\ d\phi \\
&= \prod_{k=1}^{K}\frac{\Gamma(\sum_{v=1}^{V}\beta_v)}{\prod_{v=1}^{V}\Gamma(\beta_v)}\int\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}+\beta_v-1}\ d\phi \\
&= \prod_{k=1}^{K}\frac{\Gamma(\sum_{v=1}^{V}\beta_v)}{\prod_{v=1}^{V}\Gamma(\beta_v)}\frac{\prod_{v=1}^{V}\Gamma(\beta_v+n_{k,v})}{\Gamma(\sum_{v=1}^{V}\beta_v+n_{k,v})}
\end{aligned}$$

$$\begin{aligned}
p(z|\alpha) &= \int p(z,\theta|\alpha)\ d\theta \\
&= \int p(z|\theta,\alpha)p(\theta|\alpha)\ d\theta \\
&= \int p(z|\theta)p(\theta|\alpha)\ d\theta \\
&= \int \prod_{m=1}^{M}\left(\prod_{k=1}^{K}\theta_{m,k}^{n_{m,k}}\right)\frac{\Gamma(\sum_{k=1}^{K}\alpha_k)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\left(\prod_{k=1}^{K}\theta_{m,k}^{\alpha_k-1}\right)d\theta \\
&= \int \prod_{m=1}^{M}\frac{\Gamma(\sum_{k=1}^{K}\alpha_k)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\prod_{k=1}^{K}\theta_{m,k}^{n_{m,k}+\alpha_k-1}\ d\theta \\
&= \prod_{m=1}^{M}\frac{\Gamma(\sum_{k=1}^{K}\alpha_k)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\int\prod_{k=1}^{K}\theta_{m,k}^{n_{m,k}+\alpha_k-1}\ d\theta \\
&= \prod_{m=1}^{M}\frac{\Gamma(\sum_{k=1}^{K}\alpha_k)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\frac{\prod_{k=1}^{K}\Gamma(\alpha_k+n_{m,k})}{\Gamma(\sum_{k=1}^{K}\alpha_k+n_{m,k})}
\end{aligned}$$

At this point, we can observe that the form of $p(z, w|\alpha, \beta)$ is similar to the Dice Toss toy example discussed earlier. The derivation of the markov chain updates will be summarized and simplified since we have already seen the dice toss example. Hence, deriving the chain updates.

$$\begin{aligned}
p(z_i = k|z_{-i}, w, \alpha, \beta) &= \frac{p(w, z|\alpha, \beta)}{p(z_{-i}, w|\alpha, \beta)} \\
&= \frac{p(w, z|\alpha, \beta)}{p(z_{-i}, w|\alpha, \beta)} \\
&= \frac{p(w|z, \beta)p(z|\alpha)}{p(z_{-i}, w_{-i}|\alpha, \beta)p(w_i|\alpha, \beta)} \\
&= \frac{p(w|z, \beta)p(z|\alpha)}{p(w_{-i}|z_{-i}, \beta)p(z_{-i}|\alpha)p(w_i|\alpha, \beta)} \\
&\propto \frac{p(w|z, \beta)p(z|\alpha)}{p(w_{-i}|z_{-i}, \beta)p(z_{-i}|\alpha)}
\end{aligned}$$

$$\propto \frac{\beta_v + n_{k,v}}{\sum_{v=1}^{V}\beta_v + n_{k,v}} \cdot \frac{\alpha_k + n_{m,k}}{\sum_{k=1}^{K}\alpha_k + n_{m,k}} \tag{10}$$

Equation 10 follows Equation 6 and 7 earlier in the dice toss example. Now with Equation 8, 9 and 10, we are ready to look at Algorithm 1. Algorithm 1 outputs the parameters that tells us the topic distribution in documents and topic distribution in words. Refer to Table 4 for a summarize comparison with PLSA.

5. HIERARCHICAL LATENT DIRICHLET ALLOCATION

The LDA presented in previous section fails to address one issue: how many topics to use for a text corpus. Blei proposed a Hierarchical Latent Dirichlet Allocation (hLDA) based on nonparametric Bayesian statistics [1]. In nonparametric Bayesian statistics, the probability of observing a word in a document follows the probability distribution of stochastic processes instead of dirichlet distributions. The stochastic processes proposed in hLDA is a combination of two classical stochastic processes. The first stochastic process known as the Chinese restaurant process allows the number of topics to grow infinitely as many as required. The second stochastic process known as the stick breaking process allows for the division of topics into specialized topics to form a hierarchy of topics. Putting this two stochastic processes together, Blei call it the nested Chinese restaurant process [1]. Figure 3 shows an example of the words associated with the hierarchical topics of hLDA. I took this example from [3].

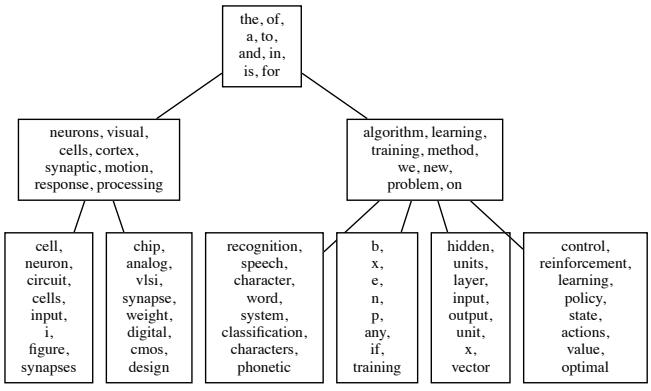


Figure 3: Hierarchical Latent Dirichlet Allocation

Algorithm 1 LDA

Initialize $z[M][N]$, $nkv[K,V]$ and $nmk[M,K]$ arrays to 0
Initialize $\beta[V]$ and $\alpha[K]$ to 1
$docs[M][N] \leftarrow$ read in documents
for $m \leftarrow 1$ to M **do**
  for $n \leftarrow 1$ to N **do**
    $k \leftarrow$ a random number between 1 and K
    $v \leftarrow docs[m][n]$
    $nkv[k,v] \leftarrow nkv[k,v] + 1$
    $nmk[m,k] \leftarrow nmk[m,k] + 1$
    $z[m][n] \leftarrow k$
  end for
end for
while run for 1000 iterations **do**
  for $m \leftarrow 1$ to M **do**
    for $n \leftarrow 1$ to N **do**
      $k \leftarrow z[m][n]$
      $v \leftarrow docs[m][n]$
      {The subtraction here corresponds to z_{-i}}
      $nkv[k,v] \leftarrow nkv[k,v] - 1$
      $nmk[m,k] \leftarrow nmk[m,k] - 1$
      for $k \leftarrow 1$ to K **do**
        $p[k] \leftarrow \frac{\beta[v]*nkv[k,v]}{\sum_{v=1}^{V} \beta[v]+nkv[k,v]} * \frac{\alpha[k]+nmk[m,k]}{\sum_{k=1}^{K} \alpha[k]+nmk[m,k]}$
      end for
      for $k \leftarrow 2$ to K **do**
        $p[k] \leftarrow p[k] + p[k-1]$
      end for
      $u \leftarrow$ a random number between 0 and 1
      $u \leftarrow u * p[K]$
      for $k \leftarrow 1$ to K **do**
        if p[k] > u **then**
          break and use current k
        end if
      end for
      $nkv[k,v] \leftarrow nkv[k,v] + 1$
      $nmk[m,k] \leftarrow nmk[m,k] + 1$
      $z[m][n] \leftarrow k$
    end for
  end for
end while
Initialize $\phi[K,V]$ and $\theta[M,K]$
for $k \leftarrow 1$ to K **do**
  for $v \leftarrow 1$ to V **do**
    $\phi[k,v] \leftarrow \frac{nkv[k,v]+\beta[v]}{\sum_{v=1}^{V} nkv[k,v]+\beta[v]}$
  end for
  for $m \leftarrow 1$ to M **do**
    $\theta[m,k] \leftarrow \frac{nmk[m,k]+\alpha[k]}{\sum_{k=1}^{K} nmk[m,k]+\alpha[k]}$
  end for
end for
return $\phi[K,V]$ and $\theta[M,K]$.

6. CONCLUSION

I have presented in detail, the formulation of LDA and the derivation of the Gibbs Sampling technique for solving LDA. The derivation is necessary because the math equations are hard to understand without working through it ourselves. I believe my derivation of LDA Gibbs Sampling is much more thorough than [7] and [8]. This will be useful for other researchers who are relatively new to Bayesian parameter learning.

In earlier section, LSA and PLSA was presented because of their incremental advances leading to the invention of LDA. Then I have briefly discussed hLDA. Although the mathematical formulation of LSA, PLSA and LDA are different, they are fundamentally performing the same task, grouping similar terms and documents together. Whether is it algebraic method or probabilistic method, all mathematical techniques aim to achieve the similar objective of representing unstructured data into structured and well organized knowledge. The semantics of linear algebra is significantly much simpler than PLSA and LDA. PLSA also is much simpler than LDA. However, probabilistic representation is much simpler to understand when we want to interpret the results. Having prior knowledge of the text corpus and using the bayesian formalism in PLSA and LDA allow us to process the text corpus quickly. One notable difference between LSA and LDA is that LDA allows the number of topics to exceed the number of documents which expands the semantic space rather than reducing it. However, the results of expanding semantic space requires further validation.

One question remains in my mind. Since the mathematics of LDA is so much harder, what makes it different from PLSA? The difference is not obvious since both techniques uses probabilities for defining the topics of documents and words. Girolami addressed this issue in [6] and explained that LDA introduces dirichlet priors which avoids an overfitting problem in the text corpus. Another issue that I have not addressed is how does such document classification improve IR results? Wei and Croft have shown that LDA does improve IR results [12]. Finally, I have not give an in-depth explanation of hLDA. hLDA builds on the field of nonparametric Bayesian statistics which requires a thorough theoretical foundation in this area. hLDA was proposed a year later after LDA but the research community have readily adopted LDA instead of hLDA. This is because the research community have not fully understood how to interpret the topics discovered by LDA, hence, the usefulness of hLDA is still limited by our understanding of text.

7. REFERENCES

[1] D. Blei, T. Griffiths, and M. Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 2009.

[2] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine learning Research*, pages 993–1022, 2003.

[3] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, 2004.

[4] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by

latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[5] 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 (Methodological)*, 39(1):1–38, 1977.

[6] M. Girolami and A. Kabán. On an equivalence between plsi and lda. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference*, pages 433–434. ACM, 2003.

[7] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235, 2004.

[8] G. Heinrich. Parameter estimation for text analysis. Technical report, University of Leipzig, 2008.

[9] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference*, pages 50–57. ACM Press, 1999.

[10] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1):177–196, January 2001.

[11] G. Strang. *Introduction to Linear Algebra, Fourth Edition*. Wellesley Cambridge Press, 2009.

[12] X. Wei and B. W. Croft. Lda-based document models for ad-hoc retrieval. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference*, 2006.

APPENDIX

I wrote these programs myself.

A. RUNNING PROGRAMS

A.1 PLSA for Toy Example

1. Run Matlab.
2. Run the script EM from the directory \matlab\.
3. Run the script a few times to observe the random results generated.

This random results obtained motivated me to use a larger text corpus in the next section.

A.2 PLSA for wikipedia articles

1. Run Matlab.
2. Run the script PLSA from the directory \matlab\.
3. This will take some time.

A.3 LDA

I wrote LDA with reference to an existing C++ implementation of Gibbs Sampling from [8].

1. Run TopicModel\LDAGibbsSampling\bin\Release \LDAGibbsSampling.exe
2. This is quite fast.

A.4 LDA Gutenberg corpus

1. Run TopicModel\Gutenberg\bin\Release \Gutenberg.exe <no. of topics>
2. This is will take some time.
3. results store in results.<no. of topics>.topics.txt

The results are shown here.

Topic 1 (Nature)	Topic 2 (Western Lifestyle)	Topic 3 (Politics & History)	Topic 4 (Casual)	Topic 5 (Biology)
species	coffee	great	said	bone
water	co	king	man	disease
great	house	war	time	tissue
light	pounds	government	know	skin
found	de	states	see	blood
time	trade	century	good	treatment
see	states	made	come	infection
body	united	sidenote	mr	joint
work	illustration	time	day	patient
man	york	france	made	cases
illustration	tea	england	go	form
form	cafe	state	old	nerve
feet	roasting	church	say	met
part	years	years	where	fig
thus	made	people	thought	tissues
animals	water	french	eyes	glands
made	london	part	hand	bones
life	cup	work	room	pain
where	green	german	went	result
seen	time	general	came	wound

Title	Nature	Western Lifestyle	Politics & History	Casual	Biology
A Book of Natural History	87.91%	0.05%	0.05%	11.97%	0.02%
A Tale of Two Cities	0.02%	0.00%	0.04%	99.94%	0.00%
Adventures of Tom Sawyer	0.03%	0.01%	0.06%	99.90%	0.01%
Alice In Wonderland	0.11%	0.01%	0.07%	99.79%	0.02%
All About Coffee	3.83%	85.06%	5.46%	5.65%	0.00%
An Introduction to the History of Western Europe	0.00%	0.00%	99.93%	0.07%	0.00%
Current History, A Monthly Magazine	0.02%	0.01%	79.42%	20.55%	0.00%
Discourse on the method of rightly conducting the reason and seeking truth in the sciences	53.25%	0.02%	13.23%	32.75%	0.74%
Dracula	0.00%	0.01%	0.00%	99.98%	0.00%
Emma	0.00%	0.00%	0.00%	99.99%	0.00%
Encyclopaedia Britannica, 11th Edition	19.66%	0.23%	80.08%	0.00%	0.02%
Frankenstein	5.52%	0.01%	1.16%	93.29%	0.02%
General Science	98.53%	1.42%	0.04%	0.00%	0.01%
History of the United States	0.01%	0.00%	99.90%	0.09%	0.00%
Jane Eyre	0.21%	0.02%	0.00%	99.77%	0.00%
Les Miserables	0.65%	0.00%	1.69%	97.65%	0.01%
Manners, Custom and Dress During the Middle Ages and During the Renaissance Period	4.12%	0.00%	86.17%	9.71%	0.01%
Manual of Surgery	8.76%	0.00%	0.01%	0.01%	91.21%
Metamorphosis	0.04%	0.02%	0.19%	99.73%	0.02%
Music Notation and Terminology	99.88%	0.01%	0.05%	0.05%	0.01%
Natural History of the Mammalia of India and Ceylon	99.93%	0.01%	0.00%	0.06%	0.00%
On the origin of species	99.99%	0.00%	0.00%	0.01%	0.00%
Romeo and Juliet	0.03%	0.03%	0.01%	99.90%	0.02%
The Adventures of Sherlock Holmes	2.30%	0.20%	0.56%	96.93%	0.00%
The Art of War	9.85%	0.02%	76.20%	13.93%	0.01%
The Communist Manifesto	6.98%	0.07%	92.75%	0.14%	0.06%
The Kama Sutra of Vatsyayana	24.90%	0.01%	13.44%	61.62%	0.03%
The Last Supper	20.30%	0.11%	72.47%	7.02%	0.11%
The Notebooks of Leonardo Da Vinci	92.85%	0.00%	0.80%	6.35%	0.00%
The Outline of Science	98.24%	0.02%	0.04%	1.67%	0.03%
The Practice and Science Of Drawing	92.31%	0.00%	0.06%	7.61%	0.01%
The Prehistoric World	89.38%	0.00%	10.55%	0.06%	0.01%
The Romance of Lust	0.00%	0.00%	0.00%	99.99%	0.00%
The Tempest	1.79%	0.07%	10.96%	87.10%	0.08%
War and Peace	0.00%	0.00%	0.00%	99.99%	0.00%
Woman as Decoration	40.24%	0.01%	22.98%	36.74%	0.02%