# Latent Dirichlet Allocation for Text, Images, and Music

Diane J. Hu
Department of Computer Science
University of California, San Diego
dhu@cs.ucsd.edu

**Abstract**

Latent Dirichlet Allocation (LDA) is an unsupervised, statistical approach to document modeling that discovers latent semantic topics in large collections of text documents. LDA posits that words carry strong semantic information, and documents discussing similar topics will use a similar group of words. Latent topics are thus discovered by identifying groups of words in the corpus that frequently occur together within documents. In this way, LDA models documents as a random mixture over latent topics, with each topic being characterized by its own particular distribution over words. In this report, we show that LDA is not only useful in the text domain, but also in the image and music domain. In particular, we discuss algorithms that extend LDA to accomplish tasks like document classification for text, object localization for images, and automatic harmonic analysis for music. For each domain, we also emphasize approaches that go beyond LDA's traditional bag-of-words representation to achieve more realistic models that incorporate order information.

## 1 Introduction

The massive amount of digital multimedia content available to us today necessitates some good methods for retrieval, organization, and management. For example, imagine that we are given an extremely large collection of text documents. It would be desirable for each document to have some sort of "short description" that could quickly tell us what it is about. Going further, it would also be useful for these short descriptions to have representations that are consistent across the entire collection. This way, we may use it to determine how closely related one document is to another. We may even use it to visualize where a particular document stands relative to all other documents in the corpus in terms of content similarity. One can see that having these short descriptions would be invaluable for searching and indexing a large collection of text data.

Latent Dirichlet Alllocation (LDA) [3] is an algorithm that specifically aims to find these short descriptions for members in a data collection. Originally proposed in the context of text document modeling, LDA posits that one way of summarizing the content of a document quickly is to look at the set of words it uses. Because words carry very strong semantic information, documents that contain similar content will most likely use a similar set of words. As such, mining an entire corpus of text documents can expose sets of words that frequently co-occur within documents. These sets of words may be intuitively interpreted as *topics* and act as the building blocks of the short descriptions.

More formally, LDA is a probabilistic, generative model for discovering latent semantic topics in large collections of text data. Each discovered topic is characterized by its own particular distribution over words. Each document is then characterized as a random mixture of topics indicating the proportion of time the document spends on each topic. This random mixture of topics is essentially our "short description": It not only expresses the semantic content of a document in a concise manner, but also gives us a principled

approach for describing documents quantitatively. We can now compare how similar one document is to another by looking at how similar the corresponding topic mixtures are.

Though originally proposed for text documents, LDA exists as a very general topic discovering framework. In recent years, the model has been extended to numerous applications in other domains, including: object recognition [11, 30, 28, 6, 35], natural language processing [31, 5], video analysis [16, 36], collaborative filtering [22], spam filtering [2], web-mining [23], authorship disambiguation [27], and dialogue segmentation [25].

In this report, we give an overview of how LDA is applied specifically in the areas of text, images, and music, with an emphasis on how extended models attempt to incorporate order information. We begin in section 2 with a review of the original LDA model. From thereon, we discuss how LDA can be applied to different kinds of data. Section 3 focuses on the application of original LDA to text, as well as extensions of LDA that go beyond the bag-of-words representation. Section 4 shows how LDA has been applied to the tasks of object categorization and localization in images. We show how the surveyed works also break out of the bag-of-features representation by progressively incorporating spatial information into their LDA model. Finally, section 5 introduces some of our own work in applying LDA to symbolic music files for automatic harmonic analysis.

# 2 Latent Dirichlet Allocation (LDA)

In the original Latent Dirichlet Allocation (LDA) model [3], an unsupervised, statistical approach is proposed for modeling text corpora by discovering latent semantic topics in large collections of text documents. The key insight into LDA is the premise that words contain strong semantic information about the document. Therefore, it is reasonable to assume that documents on roughly similar topics will use the same group of words. Latent topics are thus discovered by identifying groups of words in the corpus that frequently occur together within documents. Learning in this model is unsupervised because the input data is incomplete: the corpus provides only the words within documents; there is no training set with topic or subject annotations.

We will explore the LDA model in the remaining sections. In section 2.3, we describe the generative procedure on which LDA is based, and the joint distribution that is specified by this procedure. In section 2.4, we see how the values of the latent random variables (topics) are inferred by conditioning on the observed random variables (words) using Bayes rule.

## 2.1 Background

Before LDA, a number of methods, both probabilistic and non-probabilistic, have been proposed for the task of text document modeling. The goal of these methods remain the same across both categories: to find a way to characterize each document using short descriptions that can preserve essential statistical relationships. This is necessary for basic tasks such as document classification, summarization, and similarity and relevance judgment.

Non-probabilistic methods mainly use word counts as features. In the popular *td-idf* scheme [29], each document in the corpus is reduced to a fixed-length list of numbers, roughly corresponding to the frequency of appearance for a basic set of words within that document. Later, the *LSI* model [9] uses the singular value decomposition (SVD) of the word-by-document matrix from *td-idf* to identify a subset of the feature space that captures the most variance.

Probabilistic models improved on these previous models. They mainly fall under the category of latent variable models, and introduce the idea of topics. The *mixture of unigrams model* [24] assumes that all

words from a document are drawn independently from a single topic, where each topic is a random variable that has its own particular distribution over words. *Probabilistic latent semantic indexing* (pLSI) [14], which allows each document to exhibit multiple topics, followed naturally. In pLSI, each word of a *training* document is generated from a different, randomly chosen topic, where topics are drawn from a document-specific distribution over latent topics shared by the whole corpus. This document-specific distribution, however, had many drawbacks as it allowed no natural way to assign probabilities to previously unseen documents.

## 2.2 Notation and Terminology

Before moving forward, we first introduce the terminology and notation that will be used throughout the rest of this section. Later sections may introduce new, but parallel, terminology to better represent different kinds of data other than text.

1. A *word* $w \in \{1, \ldots, V\}$ is the most basic unit of discrete data. For cleaner notation, $w$ is a $V$-dimensional unit-based vector. If $w$ takes on the $i$th element in the vocabulary, then $w^i = 1$ and $w^j = 0$ for all $j \neq i$.
2. A *document* is a sequence of $N$ words denoted by $\mathbf{w} = (w_1, w_2, \ldots, w_N)$, where $w_n$ is the $n$th word in the sequence.
3. A *corpus* is a collection of $M$ documents denoted by $\mathcal{D} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_M\}$.
4. A *topic* $z \in \{1, \ldots, K\}$ is a probability distribution over the vocabulary of $V$ words. Topics model particular groups of words that frequently occur together in documents, and thus can be interpreted as "subjects." As we will discuss in section 2.3, the generative process imagines that each word within a document is generated by its own topic, and so $\mathbf{z} = (z_1, z_2, \ldots, z_N)$ denotes the sequence of topics across all words in a document.

## 2.3 Generative Process

LDA is an unsupervised, generative model that proposes a stochastic procedure by which words in documents are generated. Given a corpus of unlabeled text documents, the model discovered hidden topics as distributions over the words in the vocabulary. Here, words are modeled as observed random variables, while topics are observed as latent random variables. Once the generative procedure is established, we may define its joint distribution and then use statistical inference to compute the probability distribution over the latent variables, conditioned on the observed variables.

We begin by describing, at a high-level, the process for generating a document in the corpus. First, we draw a topic weight vector (modeled by a Dirichlet random variable) that determines which topics are most likely to appear in a document. For each word that is to appear in the document, we choose a single topic from the topic weight vector. To actually generate the word, we then draw from the probability distribution conditioned on the chosen topic. From this procedure, we see that each word in a document is generated by a different, randomly chosen topic. The available choice of topics for each document are, in turn, drawn from a smooth distribution over the space of all possible topics. This process can be described more formally below.

For each document indexed by $m \in \{1 \ldots M\}$ in a corpus:

1. Choose a $K$-dimensional topic weight vector $\theta_m$ from the distribution $p(\theta|\alpha) = \text{Dirichlet}(\alpha)$.

2. For each word indexed by $n \in \{1 \dots N\}$ in a document:

   (a) Choose a topic $z_n \in \{1 \dots K\}$ from the multinomial distribution $p(z_n = k|\theta_m) = \theta_m^k$.

   (b) Given the chosen topic $z_n$, draw a word $w_n$ from the probability distribution $p(w_n = i|z_n = j, \beta) = \beta_{ij}$.

Since the generative process imagines each word to be generated by a different topic, the LDA model allows documents to exhibit multiple topics to different degrees. This overcomes the limitations of the mixture of unigrams model which only allows a single topic per document. We also note that the LDA model does not attempt to model the order of words within a document. It is precisely this "bag-of-words" concept that is central to the efficiency of the LDA model.

The parameter $\alpha$ is a $K$-dimensional parameter that stays constant over all of the documents within a corpus. By drawing our topic mixture weights from $\alpha$, an additional level of indirection is introduced that allows us to analyze an unseen document in relation to the rest of the existing documents in the corpus. This overcomes the limitations of the pLSI model, which treats the topic mixture weights as a large set of individual parameters that are explicitly linked to documents in the training set. The parameter $\phi$ is a $V \times K$ matrix that is also estimated from data. It encodes each of the $K$ topics as a distribution over $V$ words. In some instances, an additional Dirichlet prior $\eta$ is placed over the multinomial parameter $\beta$ to account for sparsity in the case of a large vocabulary. We do not include discussion of the learning procedure for this hyperparameter, but the interested reader can refer to the original LDA paper for details.

The Dirichlet distribution is given by:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \prod_i \theta^{\alpha_i - 1} \tag{1}$$

The Dirichlet is used because it has a number of nice properties: it is in the exponential family, has finite dimensional sufficient statistics, and is a conjugate prior to the multinomial distribution. These properties will make the inference and parameter estimation algorithms for LDA simpler, as we will see in appendix A.1. As a Dirichlet random variable, the topic weight vector $\theta$ has the property such that $\theta^i \geq 0$ and $\sum_{i=1}^K \theta^i = 1$, and thus lies in the $(K-1)$-simplex. Basically, each document is characterized by its own topic weight vector, which indicates the amount of contribution of each of the $K$ topics in that document.

The generative process given above defines a joint distribution for each document $\mathbf{w}_m$. Assuming for now that parameters $\alpha$ and $\beta$ are given to us, the joint distribution over the topic mixtures $\theta$ and the set of $N$ topics $\mathbf{z}$ is:

$$p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n, \beta) \tag{2}$$

Figure 1(a) shows the graphical representation of the joint distribution for the entire corpus. Plate notation is used to more concisely represent multiple independently, identically distributed random variables within the model. All variables that lie within a box (or plate) indicates that it is replicated $X$ number of times, where $X$ is the number in the bottom right hand corner of the box.

## 2.4 Inference and Learning

The model given in eq. (2) is completely specified by the parameters $\beta$ and $\alpha$. Now, the central task for using LDA is to determine the posterior distribution of the latent topic variables conditioned on the words

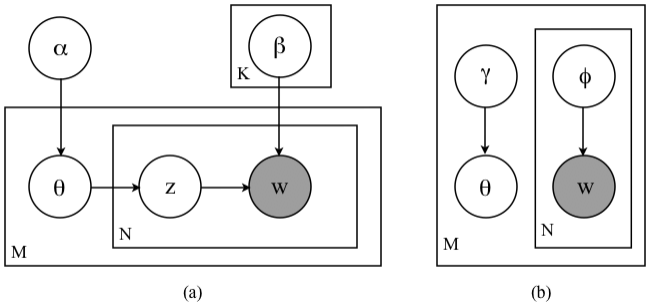


Figure 1: (a) Graphical representation of our model and (b) the variational approximation (right) for the posterior distribution in eq. (3). See Appendix A for details.

that we observe in each document. Bayes rule is used:

$$p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(\mathbf{w}|\alpha, \beta)}$$

The normalizing factor in the denominator of eq. (3) is the marginal distribution, or likelihood, of a document:

$$\begin{aligned} p(\mathbf{w}|\alpha, \beta) &= \int p(\theta|\alpha) \left( \prod_{n=1}^{N} \sum_{z_n \in Z} p(z_n|\theta) p(w_n|z_n, \beta) \right) d\theta \\ &= \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left( \prod_{i=1}^{K} \theta_i^{\alpha_i - 1} \right) \left( \prod_{n=1}^{N} \sum_{i=1}^{K} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) d\theta \end{aligned} \tag{3}$$

The optimal values of $\alpha$ and $\beta$ are chosen to maximize the log-likelihood of all of the documents in the corpus,

$$\mathcal{L}(\alpha, \beta) = \sum_{m=1}^{M} \log p(\mathbf{w}|\alpha, \beta) \tag{4}$$

A maximum likelihood estimation approach is used to find the optimal parameter values. However, because the model includes latent variables $\mathbf{z}$ and $\theta$, the data is considered incomplete, thus preventing the minimization from being able to be computed directly. As with most latent variable models, the Expectation-Maximization (EM) algorithm is used instead. The EM algorithm iteratively updates the parameters by computing expected values of the latent variables under the posterior distribution in eq. (3).

Unfortunately, this posterior distribution is intractable due to the coupling of the $\theta$ and $\beta$ parameters. Therefore, a strategy for approximate probabilistic inference must be used instead. At a high-level, the approximation consists of two steps. First, we choose a tractable family of distributions, whose statistics are easy to compute. We use this as a surrogate for the true posterior distribution:

$$q(\theta, \mathbf{z}|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n) \tag{5}$$

Figure 1(b) illustrates the corresponding graphical model for this approximating family of tractable distributions; it was obtained by dropping edges in the graph, and thus removing dependencies between variables that caused intractability. Next, we attempt to select the particular member of this family that best approximates the true posterior distribution. This is done by computing the optimal variational parameters, $\gamma$ and $\phi$, that minimizes the difference between the variational distribution and true posterior distribution. Appendix A contains the details of inference and learning in this approximation.

# 3 Applications of LDA to Document Modeling

We continue from section 2 in our exploration of LDA and text. In this section, we show how LDA is applied to real text data. In recent years, LDA has been adapted to tasks in many different text-based domains including web mining [23], collaborative filtering [22], and spam detection [2]. In this report, however, we limit our discussion to focus on different adaptations of LDA specifically in the context of document modeling. This task is general enough that it can give the interested reader an idea of how the LDA model may be applied to other text-related domains. Ideas associated with document modeling will also help us transition into non-text domains, such as images and music, which we will explore later in sections 4 and 5.

Document modeling can be an ambiguous notion, and evaluating its performance can be even more difficult to quantify. In this report, we define the task of document modeling as any effort made to simplify the representation of a text corpus by capturing meaningful semantic structure. Here, we present three methods of evaluation that are commonly used in the literature:

1. **Perplexity.** One of the ways we will evaluate a document modeling framework is to compute its *perplexity* on a held-out test set. Perplexity is a standard measure of performance for statistical models of natural language, and is defined to be

$$\text{perplexity}(\mathcal{D}_{\text{test}}) = \exp\left\{-\frac{\sum_{m=1}^{M}\log p(d_m)}{\sum_{m=1}^{M} N_m}\right\} \tag{6}$$

   where $N_m$ is the number of words in the $m$th document of corpus $\mathcal{D}_{\text{test}}$. Intuitively, perplexity indicates the uncertainty of predicting a single word. Lower values are more desirable, indicating better generalization performance. Chance performance would result in a perplexity that is equal to the size of the vocabulary.

2. **Analysis of Inferred Parameters.** A more intuitive way of evaluating the quality of the model is to analyze the learned parameters, namely $\theta$, the topic weight vector for individual documents, and $\beta$, the inferred topics for the entire corpus. In the topic weight vector, larger weights should be given to topics that are discussed more heavily in the document; smaller or zero weight should be given to topics that are not covered at all. Similarly, each topic's distribution over elements in the vocabulary should give higher probability to words associated with a certain topic.

3. **Document Classification.** This method evaluates the model by quantifying how accurately semantic content is captured by a document's topic weight vector. The topic weight vector acts as a low-dimensional representation of the document and can be used as input into any discriminative classification algorithm. The goal is to correctly assign each document to a class label (e.g. magazine title, newspaper section) in this reduced dimensional space.

In the remainder of this section, we use these three methods to evaluate different approaches to document modeling.

## 3.1 Original LDA Model

In the original LDA paper [3], a large test set was fitted to LDA, mixture of unigrams, and pLSI models and the perplexity under each model was compared. It was reported that both the pLSI and mixture of unigrams models suffered from higher perplexity scores across all choices of $K$, a consequence of severe overfitting. Next, the top words from the distribution of four of the topics were displayed; each group of top words was clearly identified as revolving around a specific topic, such as "arts," "budgets," "children," "education,"

etc. Analysis was also done on the word level for individual documents by looking at the generating topic $z_n$ for each word $w_n$. Finally, binary classification was attempted on each document in the corpus by training a support vector machine (SVM) on the low-dimensional representations. Though the feature space was reduced by 99.6 %, classification performance decreased only slightly compared to tests that used the whole set of word features.

## 3.2 Beyond Bag-of-Words

In this section, we briefly sample work that has extended the original LDA model in a way that goes beyond the "bag-of-words" assumption by modeling some form of sequence order within documents. We see that there are many different possible dependencies to model, including word class dependencies, word order dependencies, and topic order dependencies. It will also be interesting to compare dependency modeling in text with analogous methods in other non-text domains.

### 3.2.1 Modeling Word Class Dependencies

First, we describe a model [31] that improves document modeling accuracy by distinguishing between words that have short-range dependencies (provide syntactic function) and words that have long-range dependences (provide semantic content). To model both types of words, a composite model is created by linking syntactic and semantic components through a hidden Markov model (HMM). Sentences are generated by following a path through the HMM. Landing in a syntactic state generates a syntactic word from a simple multinomial distribution. Landing in a semantic state generates a word from a topic model (LDA). Thus, the choice of states within the HMM determines the syntax of the sentence.

Now, instead of treating all words within a document equally, additional effort is made to 1) distinguish between different word classes, and 2) predict the order in which these word classes occur. Results show that this approach improves document classification over the original LDA model because of the decrease in "noise": Before, the original LDA model forced all words into topics, even common nouns that carry little semantic information. Now, only semantic words (usually nouns) are handled by the topic model and organized into topics. Everything else is regarded as having syntactic function, and thus broken down into different part-of-speech categories using a separate model.

### 3.2.2 Modeling Word Dependencies

Wallach [33] proposes the Bigram Topic Model (BTM), which models dependencies at the word level. This is done by incorporating aspects of the hierarchical Dirichlet bigram language model developed in [21] into the original LDA model. In this way, both short-range word dependencies (using bigrams) and long-range word dependencies (using topics) are modeled.

The BTM takes on a similar generative procedure as LDA. In fact, the procedure for topic generation (up to step 2(b) in the generative process outlined in section 2.3) in the BTM remains unchanged. However, word generation is now defined by a conditional distribution $p(w_n = i|w_{n-1} = j, z_n = k)$, that is based not only on the chosen topic, but also the preceding word. The BTM must now keep track of two additional things: the number of times each word has occurred in the corpus and a matrix of word transition probabilities for each topic. Instead of $\beta$, we have a matrix $\phi$ with $VK$ rows. Each row is a distribution over $V$ words for topic $k$ and preceding word $j$. Finally, a Dirichlet prior is placed over all rows of the matrix. Two versions are tested: First, a single Dirichlet prior is shared across all topics. Next, different Dirichlet priors are used for rows corresponding to different topics.

The model is evaluated by measuring the perplexity on an unseen data set. Results show that the BTM with different Dirichlet priors for different topics is able to achieve a lower perplexity score than that of the original LDA model. Interestingly, the topics inferred by the BTM shows similar results as those of [31], discussed in section 3.2.1. The BTM also separates out function words (e.g. "the", "in", and "to") from the rest by collecting them into a single topic. The rest of the topics are largely comprised of words with strong semantic content.

#### 3.2.3 Modeling Topic Transitions

The Hidden Topic Markov Model (HTMM) [13] is an approach that more realistically models transitions between topics. In this model, proposed that in a typical document, words in the same sentence, as well as successive sentences, are more likely to be drawn from the same topic rather than different ones. This creates dependencies between the hidden topic variables and is modeled by a hidden Markov model (HMMs). More specifically, the topics in a document form a Markov chain with a transition probability that depends on the topic weight vector $\theta$ and a topic transition variable $\psi_n$, which takes on either 0 or 1. When $\psi_n = 1$, a new topic is drawn from $\theta$ for the $n$th word. Otherwise, the topic is identical to that of the $n-1$th word. Topic transitions are constrained to occur only between sentences.

The perplexity of both the HTMM and LDA models were evaluated, each using identical parameters. The perplexity is significantly lower for the HTMM when the length of each document is small (i.e. $N \leq 64$). Analyzing inferred topics show that the HTMM tends to find topics that consist of words that are consecutive in the document. This leads to a sense of word-sense disambiguation. For example, when the model was applied to a collection of 1740 NIPS documents, the word "support" in the context of a support vector machine was assigned to a topic different than that of the same word used in an acknowledgment section. The same experiment, done using LDA, assigned both instances to the same topic.

### 3.3 Discussion

We have seen that all three beyond bag-of-words models have shown improvement over the original LDA model in at least one of the three evaluation methods discussed above. In general, we see that the greatest contribution of incorporating word order into the LDA framework is its ability to resolve polysemy – words that have multiple meanings in different contexts. By incorporating both topics and word dependencies, the meaning of a word may be more accurately determined, leading to lower perplexity scores and higher document classification performance.

## 4 Applications of LDA to Object Categorization

In this section, we extend LDA to model collections of images and show how it has been used to make progress on problems in the computer vision field, specifically on the tasks of object categorization and localization. We first briefly describe each problem:

- **Object categorization** usually applies to images that contain only a single object. The goal is to categorize each image by the (single) object that it contains. It is also common for a "background" category to be considered.
- **Object localization** usually applies to images that contain several different objects. The goal is to segment the image so that the location of each object within each image can be identified.

In the following, we present three models that seem to best sample the field that crosses object recognition with LDA-based methods. As with text, we see a gradual progression from bag-of-features representations to ones that incorporate spatial information (analogous to incorporating word order information in section 3. We conclude the section with a general analysis of LDA's strengths and weaknesses, and how it fits in with other bag-of-features models in vision.

## 4.1 Notation and Terminology

We first define some terms and notation used commonly in the literature that parallel those defined in section 2.2:

- A *codeword* $x \in \{1, \ldots, V\}$ is the most basic unit of discrete data in our model. Essentially, it is a "visual word."
- A *patch* is the most basic unit of an input image. It is important to distinguish between a patch and a codeword: A *patch* is a small region of the raw image that can be converted to a codeword by applying a feature detector (e.g. SIFT), and then finding the closest matching codeword in the codebook.
- A *codebook* is a collection of $V$ learned codewords. This codebook is formed by extracting patches from all images in a corpus, converting them into codewords, clustering these codewords into $V$ clusters, and then selecting the centroid of each cluster as a representative for the $v$th codeword in the codebook.
- An *image* is a collection of $N$ patches denoted by $\mathbf{x} = (x_1, x_2, \ldots, x_N)$, where $x_n$ is the $n$th codeword in the collection.
- A *corpus* is a collection of $M$ images denoted by $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, where $\mathbf{x}_m$ is the $m$th image in the corpus.
- A *topic* $z \in \{1, \ldots, K\}$ is a probability distribution over the vocabulary of $V$ codewords. Topics model particular groups of codewords that frequently occur together in images, and thus may usually be interpreted as "objects".

## 4.2 Modeling Images as Documents

We begin with Sivic's model [30] that, arguably, applies LDA in the most straight-forward way. In this paper, images play the role of documents, codewords play the role of words, and object categories play the role of topics. An image is modeled as a mixture of hidden topics, or object categories, that are present in the image.

To generate an image under this model, a topic weight vector is drawn from a Dirichlet distribution. This topic weight vector determines which objects are most likely to appear in the image. For each patch in the image, we draw an object category from the topic weight vector. Given this object category, we generate a codeword by drawing from that object category's distribution. This process can be described more formally as follows.

For each image indexed by $m \in \{1, \ldots, M\}$ in a corpus:

1. Choose a $K$-dimensional topic weight vector $\theta_m$ from the distribution $p(\theta|\alpha)$ =Dirichlet($\alpha$).
2. For each patch indexed by $n \in \{1, \ldots, N\}$ in the image:

(a) Choose an object category $z_n \in \{1, \ldots, K\}$ from the multinomial distribution $p(z_n = k|\theta_m) = \theta_m^k$.

(b) Given the chosen object category $z_n$, draw a codeword $x_n$ from the probability distribution $p(x_n = i|z_n = j, \beta) = \beta_{ij}$.

Codewords are learned by using vector quantized SIFT descriptors computed on affine covariant regions, which are represented by ellipses. These ellipses are computed at twice the originally detected region size in order for the image appearance to be more discriminating. Affine covariance is used as it is robust to viewpoint changes, while SIFT descriptors, which are based on histograms and local orientation, gives tolerance to illumination change.

For object categorization, $K$ is set to the total number of different objects in the image collection. Recall that $\theta$ is a $K$-dimensional topic weight vector that indicates the probability of each topic being present in that image. Each image $m$, then, is classified as containing object $k^*$ if

$$k^* = \underset{1 \leq k \leq K}{\operatorname{argmax}} \, \theta_m^k \tag{7}$$

For object localization, $K$ is again set to the total number of different objects in the image collection. For each patch in the image, we look at the probability $p(z_n|w_n)$ of the object category that it was generated by. Patches for which $p(z_n^k|w_n) > 80\%$ are identified as containing object $k$ with high probability. The location for which object $k$ is present in an image is identified by the union of patches having high posterior probability for that particular object category.

This model was compared to a landmark paper of that time by Fergus et. al. [12]. To achieve object categorization, Fergus modeled image objects as a random constellation of parts; advantages of this model include the ability to account for shape variations, occlusion, and image clutter. However, 400 hand-labeled images were required for each training class. In comparison, Sivic's unsupervised, LDA-based model showed competitive object categorization performance on the Caltech 4 dataset [10] without the need for any labeled data. Its approach can be paralleled to that of the Fergus model: instead of detecting regions of interest that make up the constellation parts, it finds codewords that commonly occur together within an image, thus also accounting for occlusion and image clutter. Additionally, this LDA-based model can be used for segmentation purposes, which the Fergus model does not address.

### 4.3 Modeling Image Segments as Documents

The concepts of the previous model are further expanded in a model by Russell et. al. [28]. Hypothesizing that a flat collection of codewords was too global of a representation for object categorization, the authors proposed a model that inserts an initial segmentation step. In computer vision, image segmentation is a seminal problem that seeks to simplify the representation of an image by grouping together pixels that correspond to the same real-world object. A correct segmentation is quite subjective, even to humans. In the context of this report we will define a "good" segment to be one that contains an object, to the best of our interpretation.

A classic saying tells us that "all good segments look alike, but each bad segment is different (bad) in its own way." This insight is exactly the motivating philosophy behind this work. This model proposes to generate not only one, but multiple segmentations of each image by varying the parameters of the segmentation algorithm. The size of the segment is chosen to be about the size of an object. This produces a large "soup" of overlapping image segments for a collection of images. The idea, then, is that segments corresponding to real objects will be exactly the ones represented by coherent object categories (topics), whereas segments overlapping object boundaries will need to be explained by a mixture of several groups (topics). The full algorithm is below.

For each image indexed by $m \in \{1, \ldots, M\}$ in a corpus:

1. Compute multiple candidate segmentations using Normalized Cuts.
2. Segment each image into $R$ segments, and for each segment, compute a histogram of codewords
3. Perform LDA on the set of all segments in the image collection, treating each segment as a document. The generative process and joint distribution of an image is identical to what is described in 2.3; the only difference is that documents/images are swapped out for image segments.
4. For each discovered topic, sort all segments by how well they are explained by this topic. The KL divergence between $z_{rm}$, the $V$-dimensional multinomial parameter describing the codeword distribution within a segment, and $\beta_k$, the learned multinomial weights for a given topic $k$, is used to evaluate how well a segment $r$ in image $m$ is explained by topic $k$.

A "good segment" (i.e. one that more tightly bounds an object) is defined to have a similar codeword distribution as that of one of the discovered topics. Object localization, thus, is achieved by identifying segments within an image that are best explained by a certain topic, and labeling it as that. Note that this methods works for images containing single or multiple objects.

Compared to the previous model by Sivic et. al [30] discussed in section 4.2, this method limits the "search" for an object to local regions within the image. In doing so, this method implicitly incorporates some spatial information as object categorization is now done at the segment level instead of the codeword level – all codewords within a segment (which are obviously a part of the same local neighborhood) are forced to have identical labels. This helps in addressing the problem of visual polysemy: the model now has the ability to differentiate the same codeword that is a part of two different image objects. The merit of this approach is reflected by a higher segmentation accuracy. Furthermore, this high segmentation accuracy is emphasized in image categories that have varying backgrounds since, in this case, the initial segmentation step is more useful in separating the target object from its surroundings.

## 4.4 Modeling Spatially Coherent Codewords

It became clear to the community that visual codewords in images contain less semantic information than words in a text document. Analogous to the movement for proceeding beyond bag-of-word methods in text, models began incorporating spatial information in hopes of strengthening the semantic contribution of each codeword in an image.

The spatially coherent latent topic model, or Spatial-LTM, by Cao et. al. [7] is one method for object localization and categorization that incorporates spatial information into the LDA framework. It constrains pixels with similar appearance in neighboring regions to share the same latent topic (or object category) assignment. Compared to the original LDA model, this model includes an additional level of latent variables: an image is treated as a "bag of segments" and each segment is treated as a "bag of codewords". The segment, therefore, is a latent variable that models groups of codewords that are in close spatial proximity.

Like LDA, the algorithm first chooses a topic weight vector, drawn from a Dirichlet distribution, that determines which objects are most likely to be present in the image. The image is then over-segmented such that segments end up being smaller than potential objects. For each region, a topic is chosen from the topic weight vector that represents the most likely object that is present at that location. An *appearance feature* which captures color and texture is also calculated from a distribution for that region. All codewords contained in this region are then generated from a multinomial distribution conditioned on the chosen topic as well as the appearance feature. This generative process is defined formally below.

For each image indexed by $m \in \{1, \ldots, M\}$ in a corpus:

1. Choose a $K$-dimensional topic weight vector $\theta_m$ from the distribution $p(\theta|\alpha) = \text{Dirichlet}(\alpha)$.
2. Segment each image into $R$ segments.
3. For each segment indexed by $r \in \{1, \ldots, R\}$ in an image:
   (a) Choose an object category $z_r \in \{1, \ldots, K\}$ from the multinomial distribution $p(z_r = k|\theta_m) = \theta_m^k$.
   (b) Compute appearance feature vector $a_r$ from the distribution $p(a_r|z_r, \lambda)$.
   (c) For each codeword indexed by $n \in \{1, \ldots, N\}$ contained in the region:
      i. Given the chosen topic $z_r$, draw a codeword $x_{nr}$ from the probability distribution $p(x_{nr} = i|z_r = j, \beta) = \beta_{i,j}$.

Codewords are obtained by using scale invariant saliency detectors to find interest points within the each segment. Each interest point is then described by SIFT. Care is also taken for the segmentation algorithm to over-segment instead of under-segment. While the segments in the model by Russell et. al. [28] were treated as potential objects in it its entirety, segments here act as building blocks for an object. This allows the algorithm to recognize objects that are heavily occluded in ways that [28] could not.

For images that contain only a single object, object categorization is done in the same way as in the model by Sivic et. al. [30], described in section 4.2, using eq. (7). For the task of object localization, each of the $r$ segment within an image is labeled with an object category $k^*$ such that the posterior probability

$$k^* = \underset{k}{\text{argmax}}\ p(a_r, \mathbf{x}_r|z_r = k) \tag{8}$$

All segments assigned to the same object category are grouped together and collectively labeled with that object category.

## 4.5 Discussion

To evaluate the usefulness and practicality of LDA-based methods compared to other bag-of-word methods in the field, we survey a number of recent papers that report image categorization results on similar datasets [12, 8, 11, 20, 37, 34]. For the problem of image categorization, LDA-based methods seem to achieve, at best, comparable performance with other bag-of-word approaches. Criticism of LDA-based methods in vision has been expressed by [20], who claims that LDA has the disadvantage of being an unsupervised and dimensionality reducing algorithm. However, while these LDA-based methods fall short of state-of-the-art performance, we see continual effort in this area [11, 26, 34, 35]. As such, we will review some strengths of LDA-based methods, and discuss why this continues to be an interesting area of research:

1. **Low-Dimensional Representation.** LDA concisely represents each image as a random mixture of $K$ topics, while other bag-of-words methods commonly deal with histograms that contain as many elements as there are codewords (which could run up to a couple thousand).
2. **Unsupervised.** LDA requires no labeled data. This may not be an issue for standard vision datasets, since labels are provided. However, real-world datasets will rarely have available labels, and hand-labeling may not be an option due to sheer volume.
3. **Intuitive Topic Representation.** LDA discovers a set of latent topics from the entire image collection that is expressed as a distribution over codewords. These topics have an intuitive make-up and most often correspond to actual objects in the training set. This makes it easy to understand what the model has learned, as well as summarize the contents of the image collection as a whole.

4. **Simultaneous Categorization and Segmentation.** LDA produces a topic mixture representation for each image as well as each image patch within an image. This allows categorization to be done at the image level, and multi-object segmentation to be done at the image patch level.

5. **Accounts for Occlusion and Image Clutter.** LDA operates on the codeword level; this makes it invariant to occlusion and image clutter as codewords act as building blocks toward whole image objects.

6. **Accounts for Visual Polysemy.** LDA-based methods that group codewords in spatial proximity address the problem of visual polysemy; the same codeword found in two different contexts will be differentiated.

# 5 Applications of LDA to Automatic Harmonic Analysis

Surprisingly little work has been done in applying LDA to the music domain. The few related works include [4], which attempted to discover latent structure in MFCC feature data using Hierarchical Dirichlet Processes (HDP) [32], and [1], which used probabilistic latent semantic indexing (pLSI) [14] to project audio songs into low-dimensional space for similarity tasks. This may imply that applying LDA to music and audio is less intuitive than for other domains such as text and vision. For example, it is not obvious what the basic "unit" of music data would be (analogous to the role of words in text), and how one would interpret the discovered latent topics. We believe, however, that a compelling representation of musical components in terms of these LDA parameters may lead to some useful models in the music retrieval field. In this section, we present our own work [15] in fitting the LDA model to symbolic music data for the application of automatic harmonic analysis.

Harmonic analysis can be broadly defined as any attempt to analyze the underlying harmonic structure of a musical piece. In western tonal music, two of the most important concepts in such an analysis are the *key* and the *tonic* of a musical piece. The key of a musical piece identifies a principal set of pitches that the composer will use to build melodies and harmonies. The key also defines the tonic, or the most stable pitch, and its relationship to all of the other pitches in the pitch set. Each musical piece will be characterized by one main key, but a compositional technique called *modulation* is often used to shift the key within a piece for more variation. While the key is in principle determined by elements of music theory, individual pieces and passages can exhibit complex variations on these elements. In practice, considerable expertise is required to resolve ambiguities. This has sparked much work in the area of automatic harmonic analysis which include the tasks of automatic key-finding and modulation detection.

A common approach for automatic key-finding proposed by Krumhansl and Schmukler [18] uses "key-profiles." A key-profile is a twelve-dimensional vector in which each element indicates the stability and importance of each neutral pitch-class relative to the given key. There are 24 key-profiles in total, one for each major and minor key. To find the key of each piece, each key-profile is correlated with an input vector that represent the total duration of each pitch-class in the musical piece. The key-profile to produce the highest correlation is determined to be the main key. We note that these key-profiles were compiled by Krumhansl and Kessler [19] from a set of probe tone studies that aimed to explore cognitive perceptions of stability within a tonal hierarchy. In the following, we show how LDA can be used to learn similar musical key-profiles automatically, from the statistics of large music collections.

## 5.1 Notation and Terminology

Again, we flesh out the notation that will be used in subsequent models. We try to parallel these terms with the original LDA notation as closely as possible.

1. A *note* $u \in \{A, A\sharp, B, \ldots, G\sharp\}$ is the most basic unit of discrete data. It is an element from the set of neutral pitch-classes. These are mapped to integer values $\{1, \ldots, 12\}$ for easy reference, where we have a vocabulary size of $V = 12$.

2. A *segment* is a basic unit of time in a song (e.g., a measure). We denote the $n$th segment as containing the set of notes $\mathbf{u}_n = \{u_{n1}, \ldots, u_n nL\}$. A segment can also be described by the number of time each note occurs. We use $x_n$ to denote the $V$-dimensional vector whose $j$th element $x_n^j$ counts the number of times that the $j$th note appears in the $n$th segment.

3. A *song* s is a sequence of $N$ segments. It can either be denoted as a sequence of segments: $s = \{\mathbf{u}_1, \ldots, \mathbf{u}_N\}$, or as a sequence of count vectors: $X = (x_1, \ldots, x_N)$.

4. A music *corpus* is a collection of $M$ songs denoted by $\mathcal{D} = \{s_1, \ldots, s_M\}$.

5. A *topic* $z$ is a probability distribution over the vocabulary of $V = 12$ pitch-classes. Topics model particular groups of notes that frequently occur together in musical segments. In practice, these groupings should contain the principle set of pitches for a particular musical key. Thus, we interpret each topic's distribution over the twelve pitch-classes as the key-profile for a musical key. In our implementation, we set $K = 24$ in order to learn key-profiles for the 12 major and minor scales in western tonal music.

## 5.2 Modeling Key-Profiles

In our LDA-based model [15], we automatically infer the 12 major and minor musical keys of western tonal music as latent topics from a large musical corpus that contain no key annotations. In our formulation, songs play the role of documents, musical notes play the role of words, and musical keys play the role of topics. These topics (or musical keys) are expressed as distributions over the twelve neutral pitch-classes, $A, A\sharp, B, \ldots, G\sharp$ etc. As such, they form 12-element distributions that act as key-profiles. These representations are used to accomplish automatic key-finding as well as similarity ranking in classical musical pieces.

We describe, at a high level, the generative process by which each song is generated. We begin by drawing a topic weight vector, from a Dirichlet distribution, that will determine which keys are present in the song. For each segment in the song, we choose a single key from the topic weight vector that will govern the choice of notes. Finally, we repeatedly draw notes from a probability distribution conditioned on the chosen key until we have generated all the notes in the given segment. This generative process is defined formally below.

For each song indexed by $m \in \{1, \ldots, M\}$ in a corpus:

1. Choose a $K$-dimensional topic weight vector $\theta_m$ from the distribution $p(\theta|\alpha) = \text{Dirichlet}(\alpha)$.

2. For each segment indexed by $n \in \{1, \ldots, N\}$ in a song:

   (a) Choose a key $z_n \in \{1, \ldots, K\}$ from the multinomial distribution $p(z_n = k|\theta_m) = \theta_m^k$.

   (b) For each note indexed by $\ell \in \{1, \ldots, L\}$ in the $n$th segment:

      (i) Given the chosen key $z_n$, choose a note $u_{n\ell}$ from the probability distribution $p(u_{n\ell} = i|z_n = j, \beta) = \beta_{ij}$.

We can determine the main key of a song by identifying the largest weight of the topic weight vector $\theta$ that maximizes the posterior distribution $p(\theta, \mathbf{z}|s, \alpha, \beta)$. Results show that this symbolic key-finding algorithm achieves a 6%-12% improvement over existing algorithms. Likewise, can determine the key of particular segments from the most probable values of the topic latent variables $z_n$. To track key modulations within

a piece, we examine all $K = 24$ topic weights. These weights indicate the proportion of time that the song spends in each key. They also provide a low-dimensional description of each song's harmonic structure. A symmetrized Kullback-Leibler (KL) divergence can be used as a measure of dissimilarity between songs based on their topic weights.

### 5.3 Discussion

Though not apparent, we see that this harmonic analysis application of LDA also goes beyond the bag-of-words representation, as musical notes that occur within the same segment are assumed to share the same topic. It is interesting to note that the underlying graphical model used in the harmonic analysis application is strikingly similar to that of Spatial-LTM, discussed in section 4.4. Both models introduces a new level of latent variables: the "bag of segments," where each segment is treated as a "bag of codewords" (in Spatial-LTM) or a "bag of notes" (in the harmonic analysis application). This representation is vital in the harmonic analysis application for differentiating between the different musical contexts that each note is a part of, especially since the entire vocabulary consists of only 12 different notes.

## 6 Summmary & Future Work

LDA is a versatile, generative model that can be used to discover underlying semantic topics from large data collections, be it of texts, images, or even music notes. In this report, we have given an introductory overview over the original LDA model for and then shown how LDA can be applied to numerous other domains and applications. In particular, we have focused on how LDA has been used for advanced document modeling, object categorization and localization, and automatic key-finding and harmonic analysis.

Because LDA discards word order, all models in each domain eventually face the challenge of going beyond the bag-of-words representation and incorporating ordering information into their LDA framework. In most models, this was expressed by the assumption that proximity leads to shared topics. In the Hidden Topic Markov Model, this meant that words within sentences shared the same discussion topic. In the Spatial-LDA model, this meant that neighboring pixels might form a coherent object. In our music model, this meant that notes within the same measure should be in the same key. We see that in each case, incorporating order information helped address the problem of polysemy. For document modeling, it means that the same word used in two different contexts would be differentiated. For object categorization, it means that the same codeword appearing in two different image objects would be differentiated. For automatic harmonic analysis, this was essential for a vocabulary of only 12 pitches; grouping together notes within segments was vital in distinguishing the scale that a single note was apart of.

In terms of future work, we see that the most developed models that include ordering information is undoubtedly in the document modeling area. For example, we see that the ideas introduced in sections 3.2.1 and 3.2.2 have been unexplored in these other domains. In section 3.2.1, the idea of separating out words from [31] that carry strong semantic information (e.g. proper nouns) versus those that do not (e.g. common nouns) may be quite interesting in the vision area. This method may be used to distinguish between visual features that are specific to certain images versus those that appear in many images across the entire dataset. Similarly, from section 3.2.2, directly incorporating statistics of word transitions in the Bigram Topic Model [33] can be very useful in our model for harmonic analysis. Music theory suggests that some key transitions are much more common than others; modeling these transition probabilities can probably boost recognition rates.

# A Variational Approximation

This appendix sketches out the variational approximation for inference and learning from section 2.4, closely following [17]. Section A.1 describes the procedure for approximating the intractable posterior in eq. (3) to infer the hidden variables. Section A.2 shows how parameters are estimated under this variational framework using a variational EM algorithm.

## A.1 Variational Inference

In our model, we substitute a surrogate variational distribution for the intractable posterior distribution $p(\theta, \mathbf{z}, U|\alpha, \beta)$, shown in eq. (3). This involves two steps, which we will describe in turn.

First, we must find a tractable *variational distribution*, a family of distributions parameterized by a set of *variational parameters*. This variational distribution acts as an approximation to the true distribution, but whose statistics are much easier to compute. One way to obtain a tractable family of distributions is to remove dependencies between variables that cause intractability in the true distribution. In the graphical illustration, this is equivalent to dropping some edges between nodes. Figure 1(b) shows the new graphical illustration reflecting the simplified model. Dropping the document index $m$, the resulting variational distribution is as follows:

$$q(\theta, \mathbf{z}|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n) \tag{9}$$

Here, the distribution $q(\theta|\gamma)$ is Dirichlet with variational parameter $\gamma$. This variational parameter exists at the document level, showing the distribution of topics in each document. The distributions $q(z_n|\phi_n)$ are multinomial with variational parameters $\phi_n$. This variational parameter exists at the word level, where $\phi_{ni}$ is the probability that $n$the word is generated by the $i$th latent topic.

The second step is to select optimal parameters for the family of distributions such that the variational distributions best matches the true distribution. More specifically, for each document $\mathbf{w}_m$, we wish to choose the variational parameters $\gamma_m$ and $\phi_m$ such that $(\theta, \mathbf{z}|\gamma_m, \phi_m)$ is made to most closely resemble the true posterior $p(\theta, \mathbf{z}|\mathbf{w}_m, \alpha, \beta)$.

To measure the quality of resemblance, we begin by computing a lower bound on the log-likelihood of each document $\log p(\mathbf{w}|\alpha, \beta)$ using Jensen's inequality:

$$\begin{aligned}
\log p(\mathbf{w}|\alpha, \beta) &= \log \int \sum_{\mathbf{z} \in Z} \frac{p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) q(\theta, \mathbf{z})}{q(\theta|\mathbf{z})} \, d\theta \\
&\geq \int \sum_{\mathbf{z}} q(\theta, \mathbf{z}) \log p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) \, d\theta - \int \sum_{\mathbf{z}} q(\theta, \mathbf{z}|\gamma, \phi) \log q(\theta, \mathbf{z}) \, d\theta \\
&= L(\gamma, \phi; \alpha, \beta)
\end{aligned} \tag{10}$$

It can be seen that the difference between the log likelihood $\log p(\mathbf{w}|\alpha, \beta)$ and the lower bound $\mathcal{L}(\gamma, \phi; \alpha, \beta)$ is actually the KL divergence (KL) between the variational posterior probability and the true posterior probability:

$$\mathrm{KL}(q, p) = \sum_{\mathbf{z}} \int q(\theta, \mathbf{z}|\gamma, \phi) \log \frac{q(\theta, \mathbf{z}|\gamma, \phi)}{p(\theta, \mathbf{z}|s, \alpha, \beta)} \, d\theta \tag{11}$$

Thus, the KL divergence can be used to measure the quality of the variational approximation. More specifically, the best approximation is one that minimizes the KL divergence in eq. (11) with respect to the variational parameters $\gamma$ and $\phi_n$ for each document. To derive update rules for these parameters, we simply

differentiate the KL divergence and set its partial derivatives equal to zero. The resulting update rules are as follows:

$$\phi_{ni} \propto \prod_{j=1}^{V} \beta_{ij}^{x_n^j} \exp\left(\Phi(\gamma_i) - \Phi(\textstyle\sum_{j=1}^{K} \gamma_j)\right) \tag{12}$$

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni} \tag{13}$$

We use $\Phi(\cdot)$ to denote the digamma function (the first derivative of the log Gamma function). Also, recall from section 2.2 that $x_n$ represents the $n$th word in the document as a $V$-dimensional unit-based vector. Thus, there will be exactly one value of $j$ for which $x_n^j = 1$, and that is when $j$ is equivalent to the vocabulary index of the $n$th word in the document.

## A.2 Parameter Estimation

We wish to obtain estimates of the model parameters $\alpha$ and $\beta$ that maximize the log likelihood of the entire corpus:

$$\mathcal{L}(\alpha, \beta) = \sum_{d=1}^{M} \log p(\mathbf{w}|\alpha, \beta) \tag{14}$$

However, since $p(\mathbf{w}|\alpha, \beta)$ cannot be computed tractably, we refer to a variational EM algorithm that uses the variational lower bound $L(\gamma, \phi; \alpha, \beta)$ in eq. (10) as a surrogate for the intractable log likelihood instead. The variational EM algorithm then estimates the parameters $\alpha$ and $\beta$ to maximize this lower bound while fixing the variational parameters $\gamma$ and $\phi$ to values obtained from the update rules in eq. (13) and (12). The update rule for $\beta$ can be written out analytically:

$$\beta_{ij} \propto \sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi_{dni}^* w_{dn}^j \tag{15}$$

Maximizing the lower bound with respect to $\alpha$ is a little more complicated. Taking the derivative of $L(\gamma, \phi; \alpha, \beta)$ with respect to $\alpha_i$ gives:

$$\frac{\partial L}{\partial \alpha_i} = M\left(\Phi(\textstyle\sum_{j=1}^{K} \alpha_j) - \Phi(\alpha_i)\right) + \sum_{m=1}^{M} \left(\Phi(\gamma_{mi}) - \Phi(\textstyle\sum_{j=1}^{K} \gamma_{mj})\right) \tag{16}$$

Unfortunately, this derivative, with respect to $\alpha_i$, depends on $\alpha_j$, for all $j \neq i$, and so we must resolve to an iterative method to find the maximum value for $\alpha$. We thus invoke the linear-time Newton-Raphson algorithm in which the Hessian is inverted in linear time, to obtain $\alpha$.

Given these update rules, the full variational EM algorithm is as follows:

1. (E-step) Fix the current model parameters $\alpha$ and $\beta$ and compute the variational parameters $\gamma_m$ and $\phi_m$ for each document $\mathbf{w}_m$ by maximizing the KL divergence in eq. (11) using the update rules in eq. (13) and (12).
2. (M-step) Fix the current variational parameters $\gamma$ and $\phi$ across all songs from the E-step and compute the model parameters $\alpha$ and $\beta$ by maximizing the lower bound in eq. (10) using the update rules in eq. (15) and (16).

These two steps are repeated until the lower bound on the log likelihood converges.

# References


[1] J. Arenas-Garcia, A. Meng, K.B. Petersen, T. Lehn-Schioler, L. K. Hansen, and J. Larsen. Unveiling music structure via plsa similarity fusion. *IEEE International Workshop on Machine Learning for Signal Processing*, pages 419–424, 2007.

[2] I. Biro, D. Siklosi, J. Szabo, and A. A. Benczur. Linked latent dirichlet allocation in web spam filtering. *Proc. 5th Int. Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2009.

[3] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[4] M. Hoffmanand D. Blei and P. Cook. Content-based musical similarity computation using the hierarchical dirichlet process. *In ICMC*, 2008.

[5] J. Boyd-Graber and D. Blei. Syntactic topic models. *NIPS*, 2009.

[6] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. *In Proc. ICCV*, 2007.

[7] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. *In Proc. ICCV*, 2007.

[8] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cdric Bray. Visual categorization with bags of keypoints. *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[9] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[10] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. *In CVPR Workshop on Generative-Model Based Vision*, 2004.

[11] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. *In Proc. CVPR*, pages 524–531, 2005.

[12] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. *In CVPR*, 2:264–271, 2003.

[13] A. Gruber, M. Rosen-Zvi, and Y. Weiss. Hidden topic markov models. *AISTATS*, 2007.

[14] T. Hofmann. Probabilistic latent semantic indexing. *Proc. of 23rd Int. SIGIR Conference*, 1999.

[15] D. Hu and L. Saul. A probabilistic topic model for unsupervised learning of musical key-profiles. *Submitted to ISMIR*, 2009.

[16] H. Wang J.C. Niebles and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *In Proc. BMVC*, 2006.

[17] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

[18] C. Krumhansl. *Cognitive Foundations of Musical Pitch*. Oxford University Press, 1990.

[19] C. Krumhansl and E. J. Kessler. Tracing the dynamic changes in perceived tonal organization in a spatial representation of musical keys. *Psychological Review*, 89:334–368, 1983.

[20] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *In CVPR*, pages 2169–2178, 2006.

[21] D. J. C. MacKay and L.C. B. Peto. A hierarchical dirichlet language model. *Natural Language Engineering*, 1:289–307, 1995.

[22] B. Marlin. Modeling user rating profiles for collaborative filtering. *NIPS*, 2003.

[23] Q. Mei, C. Liu, H. Su, and C. Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. *Proc. 15th Int. World Wide Web Conference (WWW'06)*, 2006.

[24] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39(2/3):61–67, 1999.

[25] M. Purver, K. Kording, T. Griffiths, and J. Tenenbaum. Unsupervised topic modelling for multi-party spoken discourse. *In Proc. of COLING-ACL*, 2006.

[26] P. Quelhas, F. Monay, J. m. Odobez, D. Gatica-perez, T. Tuytelaars, and L. Van Gool. Modeling scenes with local descriptors and latent aspects. *In ICCV*, pages 883–890, 2005.

[27] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and document. *In UAI*, 2004.

[28] B. Russell, A. Efros, J. Sivic, W. Freeman, and A. Zisserman. Multiple segmentations to discover objects and their extent in image collections. *In Proc. CVPR*, 2006.

[29] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[30] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W.T. Freeman. Discovering objects and their locations in images. *In Proc. ICCV*, pages 370–377, 2005.

[31] D. Blei T. Griffiths, M. Steyvers and J. Tenenbaum. Integrating topics and syntax. *NIPS*, 2005.

[32] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101[476]:1566–1581, 2006.

[33] H. M. Wallach. Topic modeling: beyond bag-of-words. *Proc. of 23rd Int. ICML*, pages 977–984, 2006.

[34] Gang Wang, Ye Zhang, and Li Fei-Fei. Using dependent regions for object categorization in a generative framework. *In CVPR*, pages 1597–1604, 2006.

[35] X. Wang and E. Grimson. Spatial latent dirichlet allocation. *NIPS*, 2007.

[36] Y. Wang, P. Sabzmeydani, and G. Mori. Unsupervised activity perception by hierarchical bayesian model. *In Proc. CVPR*, 2007.

[37] Hao Zhang, Alexander C. Berg, Michael Maire, and Jitendra Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. *In CVPR*, pages 2126–2136, 2006.