Latent Dirichlet Allocation for Object Segmentation

Korawat Tanwisuth

May 4, 2019

Abstract

In this paper, we apply Latent Dirichlet Allocation, a model typically used to model hidden topics in a document corpus, to solve object segmentation problem. We illustrate how to formulate a vision problem to fit the framework of LDA. We then benchmark the results against K-means algorithm.

1 Introduction

Latent Dirichlet Allocation (LDA) has been widely used in the domain of language modeling to discover latent topics given a document corpus [1]. In recent years, it has been applied to solve computer vision problems. As an illustration, LDA is used to discover objects from a collection of images [3], to classify image into different scene categories [4], to classify human actions [2], and to model atomic activities in visual surveillance [5]. Motivated by these applications of LDA in computer vision, we apply LDA to segment objects in images into different object classes.

2 LDA

LDA is a generative probabilistic model for modeling a collection of documents in a corpus [1]. It is a three-level hierarchical model that assumes a bag of word assumption, meaning that the position of each word in a document is disregarded and only their frequencies are taken into account. The three levels consist of corpus level, the document level, and the individual word level. For each document, the algorithm first samples a distribution over a collection of topics, which follows a Dirichlet distribution with parameter α. It then selects one of the topics from this distribution. Then, the algorithm draws a word from a multinomial distribution over terms specific to the sampled topics z_i, which again follows a Dirichlet distribution with parameter β. The generative process can be seen in the graphical model shown in Figure 1.

1. For each topic $k = 1, 2, \ldots, K$
   $\phi_k \sim Dirichlet(\beta)$
2. For each document $d \in D$:
   (a) $\theta_d \sim Dirichlet(\alpha)$
   (b) For each word w_i in d:
      i. $z_i \sim Multinomial(\theta_d)$

ii. $w_i \sim Multinomial(\phi^{(z_i)})$

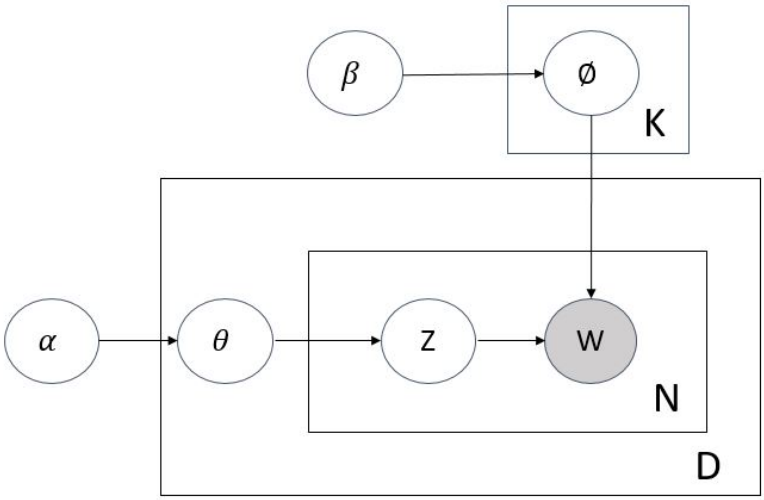

Figure 1: Graphical Model for Latent Dirichlet Allocation. K denotes the size of topics. N denotes the size of corpus. D denotes the number of documents. The shaded node means the variable is observed.

3 Gibbs Sampling

While the original version of LDA uses variational inference to fit the model, we decide to use Gibbs sampling to obtain parameters of our model. In particular, we use collapsed Gibbs sampling to fit our model, a variation of Gibbs sampling which integrates out the parameters $\Theta = (\theta_1, \ldots, \theta_D)$ and $\Phi = (\phi_1, \ldots, \phi_K)$. To derive the Gibbs sampling algorithm, we first need to specify the joint distribution over all the parameters and the data. Using the generative process described in the previous section, the joint distribution after integrating out the parameters can be factored as [6]:

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

We will now find each expression on the right hand side.

$$\begin{aligned}
p(\mathbf{w}|\mathbf{z}, \beta) &= \int p(\mathbf{w}|\mathbf{z}, \Theta, \beta)p(\Theta|\beta)d\Theta \\
&= \prod_{k=1}^{K} \int p(\mathbf{w}|\mathbf{z}, \theta_k, \beta)p(\theta_k|\beta)d\theta_k \\
&= \prod_{k=1}^{K} \frac{1}{B(\beta)} \int \prod_{v=1}^{V} \phi_{k,v}^{\Psi_{k,v}} \times \prod_{v=1}^{V} \phi_{k,v}^{\beta-1} d\phi_k \qquad (1) \\
&= \prod_{k=1}^{K} \frac{1}{B(\beta)} \int \prod_{v=1}^{V} \phi_{k,v}^{\Psi_{k,v}+\beta-1} d\phi_k \\
&= \prod_{k=1}^{K} \frac{B(\Psi_k + \beta)}{B(\beta)}
\end{aligned}$$

where $\Psi_{k,v} = \sum_{i=1}^{N} I(w_i = v, z_i = k)$ denotes the number of times topic k is assigned to word v. Here, Ψ_k denotes kth row of the count matrix Ψ and N denotes the size of the corpus. The integral follows from conjugacy of Dirichlet and multinomial sampling model.

Using the same argument, we can show that

$$p(\mathbf{z}|\alpha) = \int p(\mathbf{z}|\Theta, \alpha)p(\Theta|\alpha)d\Theta = \prod_{d=1}^{D} \frac{B(\Omega_d + \alpha)}{B(\alpha)}$$

Similarly, $\Omega_{d,k}$ denotes the number of times words in document d is assigned to topic k and Ω_d denotes the dth row of the count matrix Ω.

Now, to perform collapsed Gibbs sampling, we will need to find the following full conditional:

$$\begin{aligned} p(z_i = k|\mathbf{z}_{-i}, \mathbf{w}, \alpha, \beta) &\propto \frac{p(\mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(\mathbf{z}_{-i}, \mathbf{w}_{-i}|\alpha, \beta)} \\ &= \frac{B(\Psi_k + \beta)}{B(\Psi_k^{-i} + \beta)} \times \frac{B(\Omega_d + \alpha)}{B(\Omega_d^{-i} + \alpha)} \\ &= \frac{\Psi_{k,v} + \beta_1 - 1}{\sum_{v=1}^{V} \Psi_{k,v} + V\beta_1 - 1} \times \frac{\Omega_{d,k} + \alpha_1 - 1}{\sum_{k=1}^{K} \Omega_{d,k} + K\alpha_1 - 1} \\ &\propto \frac{\Psi_{k,v} + \beta_1 - 1}{\sum_{v=1}^{V} \Psi_{k,v} + V\beta_1 - 1} \times \Omega_{d,k} + \alpha_1 - 1 \end{aligned} \tag{2}$$

where $\mathbf{z}_{-i}$ indicates the latent variable $\mathbf{z}$ without the ith component.

The full conditional here is easily updated since we only need to decrement the count matrix Ψ and Ω before computing the probability that z_i takes a particular value k and then sample $z_i = k$ based on the updated probability given in (2)

4 Problem Formulation

The data set we use is the MSRC image data set, which consists of 240 images and 14 object classes. Each image comes with a ground truth segmentation labeled by humans 2. Our objective here is to apply an unsupervised learning algorithm to segment these images into 14 object classes. In order to apply LDA, a model typically used in natural language processing, to a problem in computer vision, we need to formulate our problem in such a way that the requirements of LDA are satisfied. To model our data using LDA, we need to specify what documents and words mean in our context. It might not be immediately clear what should be considered as documents and words in this problem; however, one option is to let each image be a document and each word be some local feature in an image. To capture local features in an image, we decide to use K-means algorithm to cluster local patches and use the centroids given by the K-means algorithm as our vocabulary. Thus, the size of our vocabulary is determined by the number of clusters we specify for K-means.


Figure 2: Original image(left column). Ground truth labeled by human (right column)

5 Implementation

As laid out in problem formulation, we need to assign each local patch a label based on topic assignments provided by LDA. For each image, we first sample coordinates densely on a grid of pixels. For each pixel, we create a square box of size 12×12 to capture local features inside the image. We then sample 5000 of these patches to ensure that we cover the majority of the image. The selection of the patch size and the number of patches is based on several experiments we perform to ensure we keep a balance between complexity and accuracy. After sampling these local patches for each image, we combine these local patches into a collection of N feature vectors of size $p_h \times p_w \times 3$, where N denotes the total number of images in the data set, p_h denotes patch height, and p_w denotes patch width. This procedure is performed so that we can use K-means algorithm to create a codebook of size V, which becomes the size of our vocabulary. Specifically, we use mini batch K-means, an algorithm based on K-means that allows learning of cluster assignments by using mini batches of data. We use the mini batch version because we want a relatively large number of vocabularies in order to allow LDA to further classify these groups into final cluster assignments.

After performing the K-means algorithm, each of these feature vectors of local patches then receives a label based on the cluster assignment from K-means algorithm. The final cluster assignments which become our codewords in our vocabulary are shown in 3. We then perform LDA on a document-term matrix created by counting how many of the V clusters given by K-means are in each image. Thus, a row represents a document or an image and a column represents a word in the vocabulary or one of the centroids. LDA then proceeds to cluster each of the V groups into 14 classes, the total number of object classes in the data set. The choice of hyper-parameters, α and β, is based on the experiment we perform in the result section.

To learn the parameters $\Theta = (\theta_1, \ldots, \theta_D)$ and $\Phi = (\phi_1, \ldots, \phi_K)$, we use collapsed Gibbs sampling as explained the earlier section. We create two count matrices Ω and Ψ to count the number of times each topic is assigned to a word in each document and the number of times each word in the corpus is assigned to a particular topic respectively. For each document m and word n, we decrement the entries $\Omega_{w_{m,n},z_{m,n}}$ and $\Psi_{z_{m,n},w_{m,n}}$, where $w_{m,n}$ denotes word n in document m and $z_{m,n}$ denotes latent variable corresponding to the current word in that document before sample the latent variable z_i. This operation is reflected in the update of the full conditional (2). We then sample the updated label with probability according to (2) and update the count matrices. We then keep updating

the latent variable z_i based on the rule described until we reach convergence. In this case, our convergence criteria is simply the likelihood function. Now, we estimate Θ and Φ using the following equations:

$$\hat{\phi}_{k,v} = \frac{\Psi_{k,v} + \beta}{\sum_{v'=1}^{V} \Psi_{k,v'} + V\beta}$$

,

$$\hat{\theta}_{m,k} = \frac{\Omega_{d,k} + \alpha}{\sum_{k'=1}^{K} \Omega_{d,k'} + K\alpha}$$

We use Φ, the topic-word distribution, to label each of the centroid based on the 14 object classes. For each word v in the vocabulary, we simply take the most probable topic assignment according to topic-word distribution. Based on this cluster assignment, all the local patches that are in the same cluster are assigned the same labels. Because we sample densely from a grid for each image, there will be overlap between local patches. Hence, to segment the object by assigning color to each pixel based on topic assignment given by LDA, we decide to take the most occurring values of topic assignments for each pixel. If there is a tie or there is no patch that cover that pixel, we look for neighboring values to decide the majority vote. We show the results we obtain from LDA and those from regular K-means algorithm in the next section.

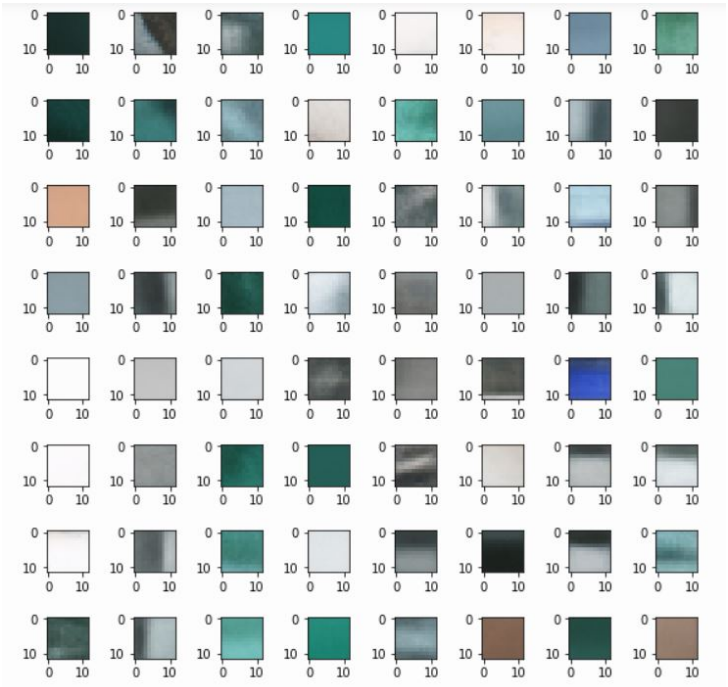

Figure 3: Image patches densely samples from our image from our data set are clustered into 1000 clusters. 64 of them are randomly chosen and shown here. Each cluster center becomes a word in our vocabulary. Notice that each centroid has different visual property , implying that our vocabulary has good coverage for the data set

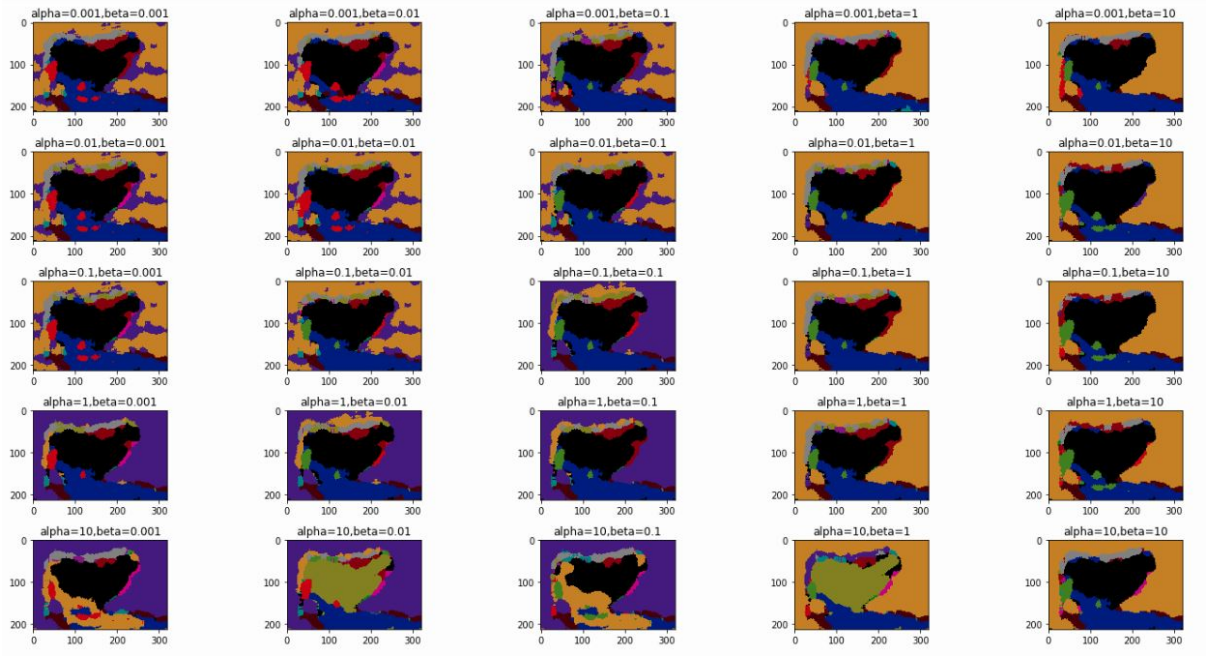

Figure 4: The effect of varying hyper-parameters, α and β.

6 Results and Benchmark

We will now benchmark our method against regular K-means to see if LDA can improve the clustering procedure. While we can visually inspect the result of an object segmentation algorithm, we decide to use a numerical metric to measure the quality of the segmentation given by an algorithm. To compare different methods numerically, we create a matrix of all 0s and allocate a 1 in the location of a pixel when there is a change of boundary in the ground truth image labeled by humans. We then apply the same method to the image output by the algorithm. Now, let x_{ij} denote the value of the matrix at row i and column j in the image colored by the algorithm, and y_{ij} denote the value in the matrix corresponding to the ground truth image. We use the following metric to compare the result of an object segmentation algorithm.

$$\mathcal{L} = \sum_j \sum_i |x_{ij} - y_{ij}|$$

To select the hyper-parameters, we perform an experiment by creating a grid of α and β values ranging from 0.001 to 10 and select the hyper-parameters that minimizes the loss function we chose. We can see the effect of each hyper-parameter in 4. When we fix each β and vary the value of α (looking across the row), the images become less noisy as the value α increases. α is the intensity parameter which controls the weight of each topic on each document. A high value of α means we put relatively equal weights among many topics. Likewise, when we fix each α and vary the value of β, the the image becomes less noisy as β controls the intensity of word for each topic. A low value means we will have a few dominant words for each topic. As discussed in the implementation section, we take the maximum of the probability of a word belonging to each topic. Based on the value of the loss function, we chose $\alpha = 0.01$ and $\beta = 10$.

Table 1: Errors by categories for each clustering algorithm

	Grasses	**Trees**	**Houses**	**Airplanes**	**Cows**	**Faces**	**Cars**	**Bicycles**
LDA	2353	3906	3864	2919	3019	4247	4508	5639
K-means	2740	4453	4420	3567	3588	4657	4727	6442
Random	16081	16257	16186	16116	16163.3	15918	15976	16739

Table 2: Average errors across 240 images for each algorithm

LDA	**K-means**	**Random Assignment**
3807	4324	16180

While there are 14 object classes, some images might contain many object classes. This, in turn, reduces the total number of categories of images to 8 categories. We can see in 1 that, for each category, LDA outperforms K-means and that the improvement from random assignment is dramatic. We illustrate the results for each image category below in 5 and 6. We can see that, on average, the algorithm performs particularly well; however, on such category as bicycles and airplanes, the images look noisy as the objects do not stand out like those in other categories.

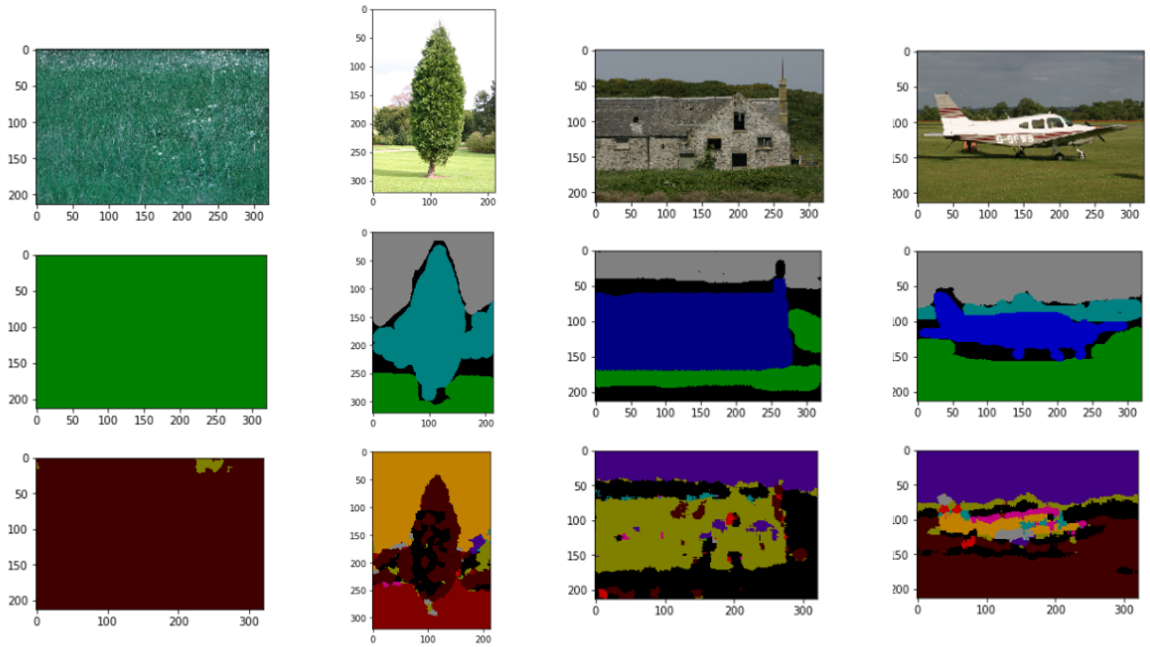

Figure 5: Original images are shown in the first row. The images in the second row are the labels given by humans. The third row shows the images produced by LDA.

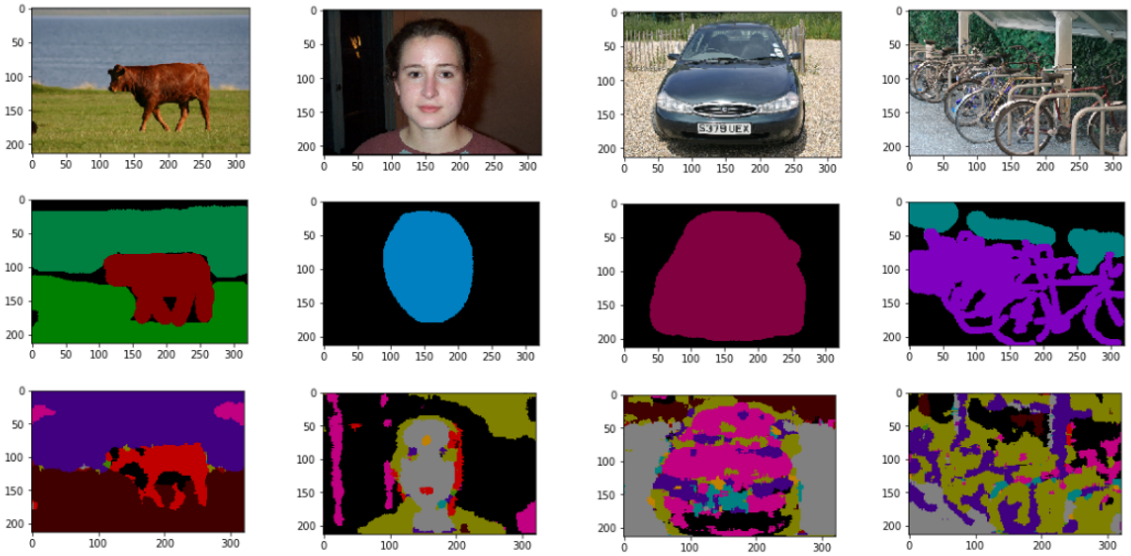

Figure 6: Original images are shown in the first row. The images in the second row are the labels given by humans. The third row shows the images produced by LDA.

7 Conclusion

In this paper, we are able to apply LDA to solve image segmentation problem. We illustrate how to formulate the problem to fit the LDA framework and show how to learn parameters in LDA using collapsed Gibbs sampling. Our results show that, compared to K-means, LDA improves the quality of the segmentation based on L_1 metric we chose. A limitation of our approach, however, is that we ignore spatial structure in the image by assuming the bag of word assumption. Future research effort might be worth investigating how to incorporate spatial structure into LDA model, allowing the model to better handle image data.

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. Journal of Machine Learning Research, 3:993–1022, 2003.

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

[3] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering object categories in image collections. In Proc. ICCV, 2005.

[4] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In Proc. CVPR, 2005.

[5] X. Wang, X. Ma, and E. Grimson. Unsupervised activity perception by hierarchical bayesian models. In Proc. CVPR, 2007.

[6] Y. Wang. Distributed Gibbs sampling of latent topic. 2008