Unsupervised Image Clustering and Topic Modeling for Accelerated Annotation

Crystal Wang, Yaateh Richardson, Ryan Sander
Massachusetts Institute of Technology
77 Massachusetts Avenue, Cambridge MA 02139
cwang506@mit.edu, yaatehr@mit.edu, rmsander@mit.edu

Abstract

Annotation of large imagery datasets remains a significant obstacle for training supervised machine learning models. Image retrieval in large, unlabeled datasets also remains a pertinent, difficult application that can be applied to image annotation and other problems in artificial intelligence. With unsupervised machine learning techniques, namely Scale-Invariant Feature Transform (SIFT), k-means Clustering, (pre-trained) Convolutional Neural Network features, and Latent Dirichlet Allocation (LDA), this work aims to organize images in an unsupervised manner using latent features. We contend it is possible for this unsupervised image organization technique to accelerate image annotation and retrieval processes without sacrificing labeling accuracy.

1. Introduction

Nearly all state-of-the-art supervised learning techniques require training on annotated and curated datasets. Annotation of these datasets, particularly on spatial/image data, can be quite costly and significantly hinder machine learning research and performance. One salient example of this is the Coco Imagery Dataset: annotating the 2.5 million instances in this dataset required 20k worker hours for category labeling at image-level, 10k hours for instance spotting, and a staggering 55k hours for instance segmentation [2]. We argue that this annotation and data curating process can be streamlined, both in terms of time and computational resources, through the use of unsupervised image clustering and topic modeling. While our focus is on unsupervised clustering of images as a pre-processing step, the implications include improvement in annotation efficiency and image classification tasks. With appropriate post-processing, the image annotation process can be streamlined for other computer vision tasks as well, such as object detection, classification, and image/scene segmentation.

Image retrieval in unstructured, unlabeled datasets is another key task in machine learning and computer vision that we aim to improve through this work. Image retrieval has applications ranging from related image querying (e.g. "find the 5 most similar images to this input image") to generating similar training examples for supervised computer vision tasks.

In this paper, we outline a framework for unsupervised image clustering and topic modeling to streamline the image annotation and querying process, even in the absence of ground truth image labels and structure.

At a high level, our pipeline consists of three stages. A feature extraction stage which uses unsupervised computer vision techniques such as Scale-Invariant Feature Transform (SIFT) [6] or pre-trained Convolutional Neural Network activation channels to extract relevant features from an image. A clustering stage, where we cluster the the extracted features into into histograms for a discretized feature representation. Using these clusters, we are then able to apply Latent Dirichlet Allocation (LDA) [1] for topic modeling on our images. Finally, the topic modeling stage of our pipeline provides a distribution over latent topics for each image in our dataset. With an optimal choice of model parameters, these topic distributions will enable us to potentially group similar images to one another.

2. Related Work

Our work rests upon combining unsupervised machine learning algorithms to construct our machine learning pipeline.

2.1. Spatial LDA with Gaussians

The foundation of our framework is built upon the Spatial LDA paper by Xiaogang Wang et. al. [8]. Wang et. al. implemented Spatial LDA which used a visual "bag of words" that have visual encodings for spatial information that a generic "bag of words" misses. Though this paper provided us with substantial first steps with our project, our approaches differ in that [8] leverages Gaussian filters for use as input to their topic modeling system, while we use SIFT and pre-trained convolutional neural network feature

extraction techniques cascaded with a k-means clustering step for input into our topic modeling system.

Image clustering can be broken down into two steps: the process used to create a feature embedding, and the clustering method itself. Feature embeddings range from low level forms of feature extraction such as edge and corner detection algorithms, PCA, and image processing techniques like SIFT, to higher level forms such as Convolutional Neural Networks and Autoencoders. We took advantage of the dearth of previous work in image feature extraction and based our pipeline on SIFT and CNNs, as well as our evaluation baselines.

On the topic modeling side, a key element of our application was Latent Dirichlet Allocation (LDA). This is a topic modeling algorithm in which we assign latent topics to each of our "documents" (in our case images), using the generative process outlined through the block diagram below.

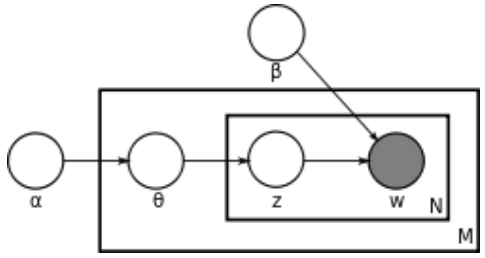

Figure 1: Plated block diagram of the graphical process capturing the Latent Dirichlet Allocation Algorithm, which we use for topic modeling of our images using clustered features [1]

The important difference to keep in mind between our methods and the techniques mentioned here, is that our work is meant to be unsupervised, and as such should be able to cluster without prior training. It would have been possible to fine tune a CNN and classify labels, but that would be supervised or semi-supervised learning. Our goal was to cluster and evaluate performance without any notion of labels, which has the benefit of discovering salient features/topics that researchers and users may otherwise not discover.

2.2. Image Clustering Baselines

To evaluate the effectiveness of our contributions, we compared against other forms of clustering and feature embedding, but not necessarily ones that are completely unsupervised.

2.2.1 PCA K-Means

Our first baseline is against standard PCA and k-means clustering. This is a basic implementation that serves as a sanity check. The clustering process was as follows:

1. Fit the PCA model to all training data for a given number of components, which is tuned as a hyperparameter
2. Project training data with PCA
3. Fit k-means to projected data with the number of clusters equal to number of labels
4. Project and predict clusters for a validation set to evaluate

Some adaptations were necessary to make it work. Images are rich feature spaces and PCA must fit every sample (or at least a representative subset) to compute the principal components. 2,000 images at (300×300) resolution were too large to fit in memory for normalization, PCA and kmeans, so we abandoned our implementation in favor of the Incremental PCA implementation from scikit-learn [7]. We also had to perform various pre-processing steps, and set image sizes and number of PCA components as hyperparameters.

2.2.2 Variational Auto-Encoders

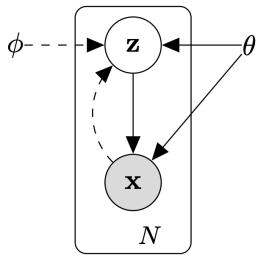

Figure 2: The plate model for VAE. the dotted lines represent approximations and the solid lines are true distributions [4].

Another method we hoped to generate feature embeddings with was the Variational Auto-Encoder (VAE) [4]. We then hoped to cluster these features as a baseline counterpart. The advantage of using a deep generative model is that our representations can be in different kernels, as opposed to PCA which projects down to the most representative linear subspace. In particular, VAE uses a parameterized variational approximation $q(z|x)$ and Evidence Based Lower Bound (ELBO) to approximate the distribution of latent variables given observations and their labels, as in Figure 2. A doubly stochastic variant of the Expectation Maximization algorithm parameterizes the q distribution instead of computing it discretely, yielding $q_\phi(z|x) \approx p_\Theta(z|x)$ when computing the ELBO. This is possible because Neural Nets don't need to assume closed form solutions with mean field approximation.

The end result is that the encoding step tunes a latent representation, and we can evaluate our representation with the decoder as seen in Figure 3. This is a supervised encoding method, but could allow one to quickly visually evaluate their latent variable representation, perhaps before comparing the effectiveness of clustering on those features. However, we found the method to be too expensive and unstable to reproduce images from a labeled distribution.

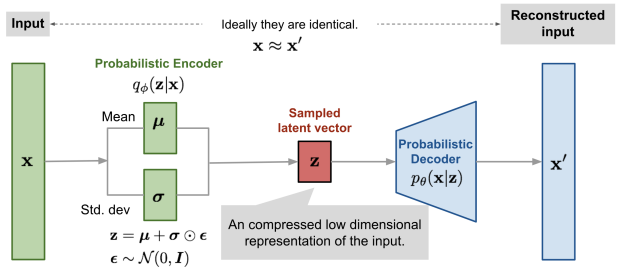

Figure 3: An Illustration of a Variational Autoencoder (VAE). Note that our data is not shown. Image from [9].

3. Approach

Our project can be broken down into three parts: feature extraction using either a CNN or SIFT, k-means clustering on the features, and LDA clustering to generate topic distributions for each image. After the features are extracted, we use k-means to cluster the feature to make a "bag-of-words" representation for each image, where each cluster outputted from k-means is a "word". For a given image, the feature Finally, we use Latent Dirichlet Allocation (LDA) to cluster the "bag-of-words" feature representations into topics. Once the images are sorted in the clusters, we evaluate the performance of the LDA model.

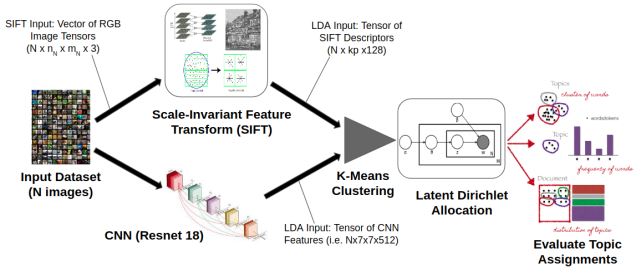

Figure 4: An overview of the feature extraction, clustering, and LDA pipeline.

3.1. Feature Extraction

3.1.1 Scale Invariant Feature Transformation (SIFT)

For each image, a Scale Invariant Feature Transform (SIFT) is applied to extract key features from the image. Since the photos in the dataset could have different orientations, lighting conditions, or scale, it is important that the features extracted do not depend on these arbitrary positionings. For example, we would want the same features extracted of the same photo of a room, even if one of the pictures were flipped upside down. An important advantage of SIFT is that it is robust to exactly these variations in images when extracting features. The SIFT algorithm first finds keypoints, which are places where there may be features in the image. The keypoints are found where there are intensity changes in a certain neighborhood size. Then, for each keypoint, a descriptor is found, which is a 128×1 vector containing the gradient magnitudes and orientations of a small window around the keypoint [6]. We use these SIFT descriptors as features since they are very robust to spatial rotation, translation, and scale. A hyperparameter to tune here would be the number of keypoints that the SIFT algorithm finds, since we use the same number of keypoints for every image.

3.1.2 CNN

Another form of feature extraction we evaluated were Convolutional Neural Networks (CNNs). We used a variety of models including ResNet 18, 34, and 50, GoogleNet, and AlexNet. All of these nets were pre-trained from the PyTorch hub [8]. To extract features from them we removed varying numbers of layers from these pre-trained models. The idea behind this is that CNNs generally diverge as they create features, and then converge upon a prediction. For example, in the Alexnet implementation the last three layers are dense/pooling layers which are used for classification, while in resnet 50 only the last two are pooling and dense layers. Thus we interpreted the activations of inner kernels for each image as a feature encoding, and flattened/stacked them for use as feature embeddings.

3.2. K-Means Clustering

LDA is common for clustering documents, and a "bag-of-words" feature representation is commonly used for word documents. To get something similar for an image, we first have to define the vocabulary, and that was made using k-means clustering. The k-means model was fitted to all the features that were extracted from the images and then clustered into k clusters, which we will tune as a hyperparameter. To make our bag of words feature representation for each image, we predict which cluster each feature falls under and then create a histogram of how many clusters each image has. Then, the histogram becomes our "bag-of-words" representation for each image.

3.3. LDA

Once the "bag-of-words" features were created for each image, the features from every image were concatenated and the LDA model was fitted to the features. Then, for each image we used the LDA model to predict a probability distribution over clusters that its feature should be in. We took the cluster with the maximum probability as the cluster the image should be in, but we keep track of the overall probability distribution over clusters for each image for evaluation purposes.

4. Dataset

We chose the ADE20k segmented dataset from MIT CSAIL [10]. The train set consisted of 21000 images, with segments describing the makeup of each image. Images were categorized by the context in which they were taken (we will refer to these contexts as labels) and there were 758 contexts. The purpose of this was to go beyond label prediction metrics and evaluate how well our clustering method was really performs on underlying features. Using segmented data gives us an extra level of granularity for evaluation. The images were in the RGB colorspace and of variable size.

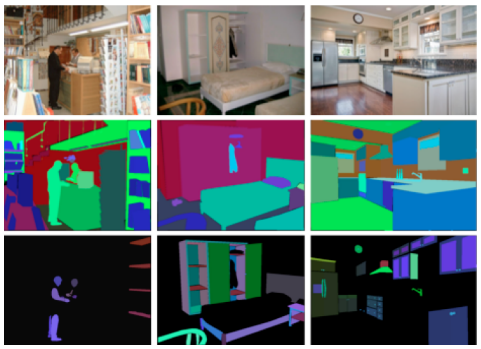

Figure 5: A subset of scenes from the ADE20k dataset, with RGB (top row), class scene segmentation (middle), and class part segmentation (bottom).

4.1. Dealing with Class Imbalance

Furthermore, the dataset was very unbalanced, with 60% of the labels having 10 samples or less. See Figure 6 for the label context breakdown. Since the class imbalances were too great for importance re-weighting, we built a tuneable dataset loader to pick subsets of images from each label such that the number of samples per label were equal. For most procedures we picked the top 25 most populated labels, which resulted in 25 sets of 79 samples per class. Having such small sample size was not detrimental to our clustering pipeline because it is unsupervised, but for supervised training techniques that could have been used as baselines - fine tuning CNNs, training VAEs, etc - this was certainly a limiting factor. To compensate, some of our baselines use the 5 most populated labels with 671 samples per label.

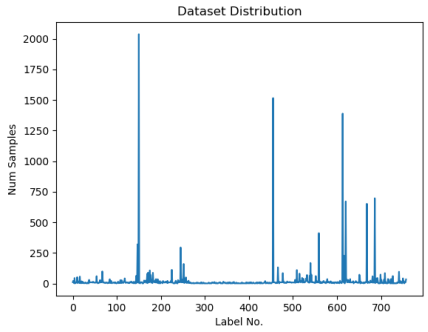

Figure 6: A histogram of the context (class) sample distribution in ADE20k

5. Evaluation

We used both quantitative and qualitative methods to evaluate our image clustering pipeline. This evaluation was only possible because we had ground truth labels for our images, which is why we elected to use a labeled dataset. We apply the evaluation methods discussed below to analyze and better understand our results.

5.1. Qualitative Evaluation Techniques

Topic modeling algorithms yield results that can provide substantial insight into the performance of the topic modeling algorithm, namely, we can qualitatively verify the efficacy of our clustering and topic modeling pipeline by looking at how similar images assigned to the same latent topic are to one another.

5.2. Quantitative Evaluation Techniques

We also sought to quantify these results for tuning our hyperparameters and developing further insights into the performance of our system.

5.2.1 Measuring Feature Distribution Differences

One method through which we evaluated our model and its associated hyperparameters was by measuring the distance in latent feature space between images of the same underlying class. This evaluation technique was applied on the k-means clustering stage, using the histogram of features associated with each ground truth image. It was also applied at the topic modeling stage, using the difference in probability distributions over topics for each image. For evaluating the k-means "bag of words," we qualitatively looked at the labels' "bag of words" representations with the hope that the

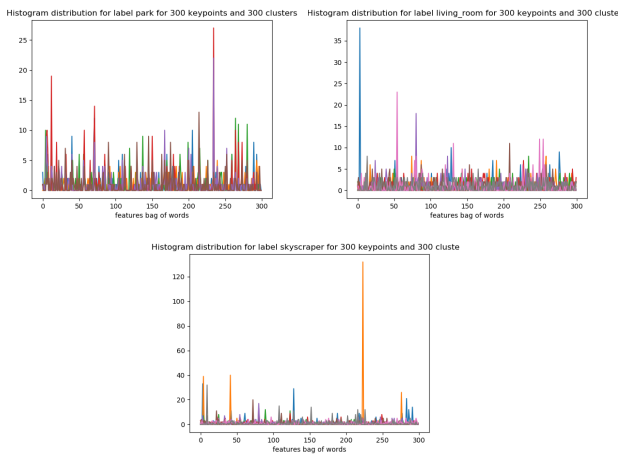

Figure 7: The "bag of words" distribution for the "park", "living_room", and "skyscraper" labels for a model trained with 300 keypoints and 300 clusters. The distributions between each image are pretty similar, suggesting that the k-means model tuned with 300 keypoints and 300 cluster does well.

distributions would be similar, for varying numbers of SIFT keypoints and k-means clusters, as seen in Figure 7. Then, for topic modeling, we would want each label's probability distributions to be similar, so for each label, we computed the average of pairwise Symmetric KL Divergence between all images, as described in Equation 1. This evaluation enabled us to find approximately optimal sets of hyperparameter by using a grid search over the number of keypoints, clusters, and topics.

$$d(x_1, x_2) = \frac{1}{2} D_{KL}(x_1||x_2) + \frac{1}{2} D_{KL}(x_2||x_1) \quad (1)$$
$$= \frac{1}{2}\left(\sum_{i=1}^{K} x_{1i} \log\left(\frac{x_{1i}}{x_{2i}}\right) + x_{2i} \log\left(\frac{x_{2i}}{x_{1i}}\right)\right)$$

6. Results

6.1. Baselines

We compared our results to standard image clustering techniques, however because all of these clustering algorithms are unsupervised, there isn't necessarily a one to one correspondence between clusters and existing labels. So to get an idea of how our algorithm measures up, we looked for unique peaks in "predicted" clusters across all techniques. For baseline techniques, we limited our train and validation sets to the 5 most populated labels in ADE20k, for a total of 3355 samples distributed evenly amongst the labels.

6.1.1 K-Means on PCA Baseline

Because of memory limitations, we created a pipeline to evaluate PCA k-means at different resolutions. Since our images were variable sized, this meant flattening data and then padding it to equal dimensions. We converted all RGB image data to grayscale.

For each transformed dataset we fit an Incremental PCA implementation from sci-kit learn to the entire dataset [7], then use Minibatch k-means to cluster the transformed feature embeddings into 5 clusters (comparably to our LDA topic predictions).

We structured our baseline implementation using a paper by Ding et. al. [3]. Although this paper did not cluster images, we saw that they had the most success in low dimensional feature spaces, and focused our search around low dimensional PCA embeddings.

Finally, we used our trained PCA and k-means models on our validation set, which yielded intra-label validation metrics similar to our own model. See the chart below for clustering results.

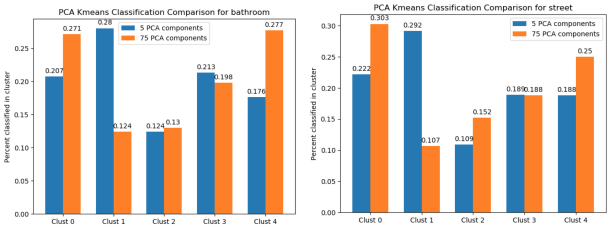

Figure 8: Distribution of predictions in the street and label context for k-means clustering on 5 pca components (blue) and 75 (orange). Note that the clustering of samples from our bathroom (left) and street (right) ground truth contexts are close to random (20%), and almost identical despite vastly different of feature representations. This was performed at (380×380) resize dimensions

Label	Clust. 0	Clust. 1	Clust. 2	Clust. 3	Clust. 4
street	0.222	0.292	0.109	0.189	0.188
bathroom	0.207	0.28	0.124	0.213	0.176
bedroom	0.209	0.271	0.125	0.222	0.173
living_room	0.231	0.292	0.124	0.173	0.18
skyscraper	0.209	0.295	0.137	0.206	0.154

Table 1: Entries are distribution over label predictions for PCA k-means clustering with 5 pca components. Images were resized and padded to a max dimension of (380,380)

As in Figure 8, the k-means clusters based on PCA do not take on clear representations of each label. In fact, we can posit that there are certain clusters that correspond to dominant image features since they remain high across all

clustered labels. For example, in Table 1, Cluster 1 shows higher classification rates than any other across all ground truth labels. This was a common trend in most feature embeddings that we tested (see the results section on CNN features).

6.1.2 K-Means on VAE Baseline (attempt)

We based our attempt at a Variational Autoencoder embedding on a blog by Augustinus Kristiadi which auto-encoded the MNIST dataset in Keras [5]. This model made quite a few simplifying assumptions. First, we assumed that the underlying latent variable distribution $p(z)$ was a normal Gaussian for simplicity of sampling and computing KL Divergence. Second, we assumed our target distribution $Q(z|X)$ to be a Gaussian, which allowed us to compute KL divergence with the following closed form equation.

$$D_{KL}[N(\mu(X), \Sigma(X)) \| N(0,1)] = \frac{1}{2} \sum_{k} \left(\exp(\Sigma(X)) + \mu^2(X) - 1 - \Sigma(X) \right) \quad (2)$$

In Equation 2 the first KL divergence parameter is $Q(z|X)$, the second is $p(z)$. Even with these simplifying assumptions and the key innovation for a VAE - reparameterizing $Q(z|X)$ as a neural network (encoder) - the VAE model was extremely computationally expensive. We were limited to relatively low latent variable dimensionalities ranging from 50 to 250. 250 latent variables took over 24 hours to train in Pytorch. We attempted to evaluate performance by visually inspecting encoded and decoded samples, but from visual inspection our most complex model only yielded noise. We did not attempt to cluster over these features. It could be that our model assumptions were incorrect and distributions over latent image features cannot be accurately represented by gaussians. Upon further reflection, GANs could have worked, but training a GAN for each label class would have been impractical (both supervised and computationally expensive) as an image clustering technique.

Figure 9: Samples from our VAE model trained for 500 epochs on 3500 images distributed across 5 labels

-				Class Label		
-				"living_room"	"skyscraper"	"park"
Keypoints	K-Means Clusters	Topics	Global KL Div.	KL Div.	KL Div.	KL Div.
150	150	100	6.758	6.624	6.734	4.818
150	150	20	4.570	4.317	4.295	3.416
200	200	20	4.567	4.291	4.363	3.202
300	300	20	4.556	4.195	4.165	3.017
350	300	20	4.727	4.254	4.403	3.141
400	300	20	4.649	4.099	4.309	3.142
500	400	20	4.664	4.173	4.320	3.320

Table 2: Symmetric KL Divergence of three labels: "living room," "skyscraper," and "park" over varying hyperparameters for the **Training Set**.

6.2. LDA on K-Means

6.2.1 SIFT features

LDA also produces a probability distribution over the potential topics that an image can be in, so we use similar evaluation techniques as in the baseline. Within each ground truth label, the probability distribution over topics for constituent clustered features (words) should to be similar. So we used a metric to compare predicted distributions for each sample image. In each ground truth label, we took the average of pairwise Symmetric KL Divergence between all images. We also computed the pairwise Symmetric KL Divergence between all images in all labels to provide a baseline for the dataset, which is denoted "Global KL Div." Table 2 shows the average distances for varying keypoints, k-means clusters, and topics used.

For the training dataset, it seems as the hyperparameter combination of 300 keypoints, 300 clusters, and 20 topics and 400 keypoints, 300 clusters, and 20 topics do the best in terms of Symmetric KL Divergence. Here are some label distributions over each topic for the model trained with 300 keypoints, 300 clusters, and 20 topics. Figure 10 shows some images taken from Topics 3, 4, 5, and 9 in the respective rows. For all choices of hyperparameters, the baseline Symmetric KL Divergence between all images is higher than the Symmetric KL Divergence between labels, suggesting that LDA is finding probability distributions over topic that are aligned with each ground truth label.

- **Topic 4:** 51 snowy mountains, 39 mountain, 26 alley, 25 coast, 15 building facade, 15 skyscraper
- **Topic 3:** 7 skyscraper, 1 bathroom, 1 dining room, 1 hotel room, 1 waiting room, 1 living room
- **Topic 5:** 7 bathroom, 5 bedroom, 3 office, 3 kitchen, 2

home office, 2 art studio,

- **Topic 9:** 19 park, 17 street, 14 alley, 13 airport terminal

Here are some label distributions for the model trained with 400 keypoints, 300 clusters, and 20 topics.

- **Topic 1:** 42 park, 30 street
- **Topic 0:** 24 snowy mountains, 21 mountain, 18 building facade, 17 alley, 13 coast, 13 skyscraper
- **Topic 5:** 8 home office, 7 living room, 4 bedroom, 4 dining room, 4 game room, 3 office, 3 waiting room, 2 bathroom

For validation, we kept the trained LDA models and ran the evaluation metrics on unseen validation data. Table 3 shows the Symmetric KL Divergences for the same labels, "living_room", "skyscraper", and "park" as well as the "Global Symmetric KL Divergence" which was computed over all image probability distributions. The model with the lowest validation Symmetric KL Divergence is 200 keypoints, 200 clusters, and 20 topics. The hyperparameters are

Figure 10: Images taken from Topics 3, 4, 5, and 9. Each row has samples from the respective topic. This model was trained with 300 keypoints, 300 k-means clusters, and 20 topics.

-				Class Label		
-				"living_room"	"skyscraper"	"park"
Keypoints	K-Means clusters	Topics	Global KL Div.	KL Div.	KL Div.	KL Div.
150	150	100	6.265	6.338	5.694	3.436
150	150	20	4.235	4.119	3.675	1.989
200	200	20	3.933	3.786	3.139	2.013
300	300	20	4.268	4.211	3.159	2.253
350	300	20	4.419	4.318	3.473	2.247
400	300	20	4.528	3.897	3.342	2.135
500	400	20	4.623	4.106	4.092	2.207

Table 3: Symmetric KL Divergence of three labels: "living room," "skyscraper," and "park" over varying hyperparameters for the **Validation Set**.

lower than the training model, which suggests some sort of overfitting at the training level. Again, the "Global Symmetric KL Divergence" is higher than the Symmetric KL Divergences within each label, suggesting that LDA is finding probability distributions that align within each label.

Some distributions over labels for topics are included below from the validation dataset.

- **Topic 6:** 5 mountain, 4 coast, 3 mountain snowy, 2 skyscraper
- **Topic 8:** 2 kitchen, 2 waiting room, 1 bathroom, 1 bedroom, 1 dining room, 1 home office
- **Topic 13:** 4 kitchen, 3 building facade, 3 dining room, 3 attic, 2 office, 2 bedroom, 2 waiting room, 1 home office, 1 hotel room, 1 living room, 1 game room
- **Topic 9:** 6 park, 2 office, 2 street

6.2.2 CNN Features

Similarly to the other feature embeddings used, CNNs required cropping and feature normalization. To resize images without disturbing aspect ratio decided on random crop and padding. We used pre-trained nets (and appropriate preprocessing transforms) on both RGB and grayscale versions of our training and validation sets. The transforms normalized our images for mean and variance within channels. We did not fine-tune the variants of nets we used, as our approach was meant to be purely unsupervised.

We found our results with CNN features to be largely similar to those of the PCA baseline, in that they weren't encoding information that was particularly suitable to clustering. Unlike sift where our feature embeddings targeted high spatial frequency features and were invariant to color,

our CNN feature embeddings yielded nearly identical prediction distributions for our intra-label validation. We have hypothesized a couple of explanations as to why:

- Our CNN feature emeddings were very high dimensional, and gradient based training is necessary to filter our relevant information. Our k-means approach clustered and used all of the features in that space.
- The CNN kernel activations were dominated by low spatial frequency features so they aren't suitable for picking out salient latent features (closely related to the first idea).

The nets were trained on RGB images, but we evaluated the features on both grayscale and RGB images to the same effect. See our intra-label validation charts in Figure 11.

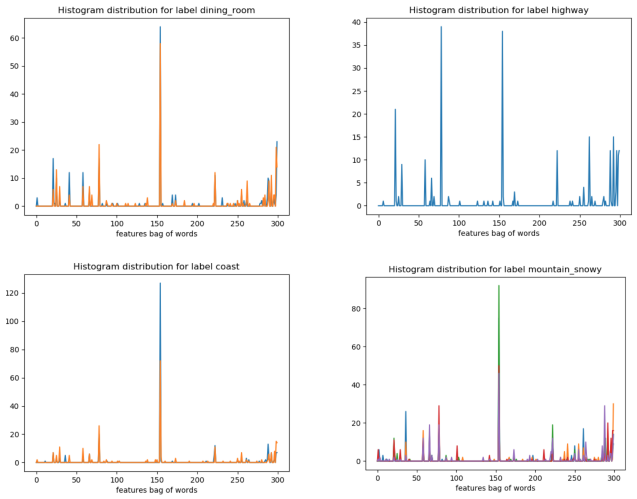

Figure 11: Topic predictions over CNN features for mountains, coasts, highways, and dining rooms look largely the same for Resnet34 with the last two layers removed. This was true across all labels, although the distribution took on a different form for each CNN variant

7. Conclusion

Our results demonstrate that our unsupervised image clustering pipeline is capable of effectively grouping topics of similar ground truth labels together via latent topic embeddings. However performance greatly hinges on the feature embeddings. From our results, we posit that globalized feature embeddings are not robust enough to encode embeddings for a wide variety of image contexts. Highly localized gradient maximizing features such as SIFT were more successful. This may be because SIFT keypoints find and encoding information about high spatial frequency image features instead of attempting to make embeddings globally.

Furthermore, both our qualitative and quantitative evaluation results demonstrate that our feature extraction hyperparameters substantially affect our results in both the clustering and topic modeling stages of the pipeline. Although our quantitative evaluation metric, Symmetric KL Divergence, was contrived to help us find hyperparameters the Global Averaged KL Divergence shows us that we can quantitatively evaluate clustering over a balanced dataset. Future work includes factoring in importance re-weighting, and testing on skewed datasets before this technique is used in the field as pre-processing for image clustering. Finally, studies should be conducted on the relative effectiveness of human annotation on preprocessed datasets. If it greatly improves the efficiency of annotators, then using LDA on k-means clustered SIFT features remains a viable option for data pre-processing.

8. Division of Labor

- **Yaateh Richardson:** Wrote a parameterizable dataset module to apply transforms and restrict training/validation to subsets over given parameters for labels. This dataset also applied transforms and preprocessing for CNN feature extraction and baselines. Adapted Crystal's pipeline to use this parameterizable dataset such that we could quickly grid search over different configurations of the dataset, and made it robust for reruns and re-evaluation. Wrote the CNN feature extraction code, and grid searched and evaluated CNN features over 15 different variants (both RGB and grayscale). Also wrote the baseline PCA k-means clustering metrics and required pre-processing (implemented standard pca and feature normalization, did not impelement incremental PCA), as well as our attempt at a VAE feature embedding for a second baseline (implemented based on Keras implementation). Finally, grid searched over our PCA k-means baseline and evaluated those results.
- **Crystal Wang:** Built a pipeline including SIFT feature extraction, k-means clustering on the SIFT features, LDA on the clusters, and evaluation of the outputted probability distributions over topics. Wrote evaluation functions to find the Global KL Divergence, KL Divergences between images in labels, and the distribution of labels over each topic. Also built the validation pipeline to extract features, cluster, predict topics, and evaluate for the validation images. Tested and analyzed results over varying hyperparameters for the SIFT portion of the pipeline. Evaluated the k-means clustering qualitatively for varying hyperparameters.
- **Ryan Sander:** Helped to write a module for integrating an LDA implementation into our pipeline, wrote evaluation functions for calculating distances between

distributions, wrote functions for calculating the distributions of different segmented ground truth labels over classes, wrote functions for cropping images according to their bounding box size, conducted tests on different combinations of topics, keypoints, and clusters, and set up an Anaconda environment for our pipeline.

9. Acknowledgements

We would like to sincerely thank all of the 6.867 teaching staff for guidance and help throughout the semester. We would especially like to thank Matthew McDermott and David Sontag for invaluable feedback and direction pertaining to this project.

10. References

References

[1] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[2] Philipe Ambrozio Dias et al. "FreeLabel: A Publicly Available Annotation Tool based on Freehand Traces". In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2019, pp. 21–30.

[3] Chris Ding and Xiaofeng He. "K-means clustering via principal component analysis". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 29.

[4] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2013. arXiv: 1312.6114 [stat.ML].

[5] Agustinus Kristiadi. *Variational Autoencoder: Intuition and Implementation*. 2019. URL: https://wiseodd.github.io/techblog/2016/12/10/variational-autoencoder/.

[6] David G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60.2 (Nov. 2004), pp. 91–110. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[7] David A Ross et al. "Incremental learning for robust visual tracking". In: *International journal of computer vision* 77.1-3 (2008), pp. 125–141.

[8] Xiaogang Wang and Eric Grimson. "Spatial Latent Dirichlet Allocation". In: *Advances in Neural Information Processing Systems 20*. Ed. by J. C. Platt et al. Curran Associates, Inc., 2008, pp. 1577–1584. URL: http://papers.nips.cc/paper/3278-spatial-latent-dirichlet-allocation.pdf.

[9] Lilian Weng. *From Autoencoder to Beta-VAE*. 2018. URL: https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html.

[10] Bolei Zhou et al. "Semantic understanding of scenes through the ade20k dataset". In: *arXiv preprint arXiv:1608.05442* (2016).