Programming Assignment 2

Locality Sensitive Hashing

Atishay Jain (210050026), Chaitanya Garg (210050039)

15th October 2024

1 Datasets

3 datasets were used -

- CIFAR-10: Consists of 60,000 32x32 color images across 10 classes, with 50,000 training images and 10,000 test images
- CIFAR-100: Similar to CIFAR-10 but with 100 classes, making the retrieval task more challenging due to more diversed number of classes
- ImageNet-1K: large-scale dataset containing over 1 million images across 1000 classes, providing even more complex and diverse set of images for retrieval. Due to Memory Limits on Kaggle (Did create embeddings for all images, but training on all gave CPU out of memory error), we have used 120,000 training and 20,000 testing images.

1.1 Data Processing and Feature Extraction

Resized all images to a uniform size suitable for feature extraction and normalized the pixel values. Then, we used **ResNet-50** model to extract feature embeddings from the images. ResNet-50 is a pretrained CNN which is widely used for feature extraction because of its ability to capture rich visual representations. Hence, we obtained the high-dimensional feature vector which were later used for clustering and hashing techniques.
For ImageNet, we have downloaded the images and saved their ResNet Embeddings on the go using $streaming = True$, due to the large size of dataset and limited memory of kaggle. Then for each task we load the embedding from this locally created and saved dataset.

2 Training and Inference Methods

2.1 KMeans

- We have used *kmeans* from $faiss$ library as it utilises GPU and hence is faster than *kmeans* from *sklearn*.
- Used Cosine Similarity to rank images within the cluster.

2.2 Random Projection Hashing

- Normal Vectors of Hyperplanes are samples from Normal distribution to ensure they are uniformaly random in a hypersphere.
- Used cosine Similarity to rank images with same hash codes.

2.3 Neural LSH

- Used $H = 16$, $L = 10$, $J = 8$.
- Sampled 10 Negatives for each datapoint, from batches of size 125.
- Used cosine similarity to rank images with same hash code as query image in atleast 1 bucket.

3 Evaluation Metrics

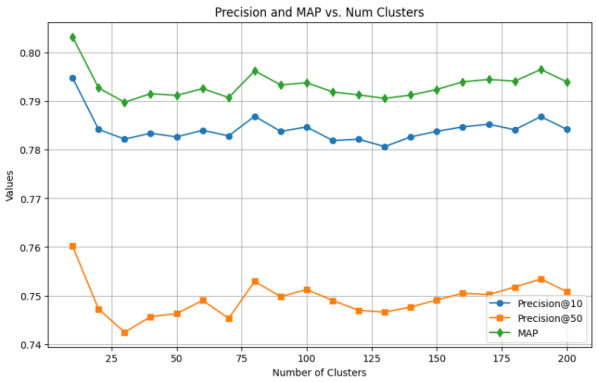
We obtained results for the following metrics -

- Mean Average Precision (MAP)
- Precision@10
- Precision@50

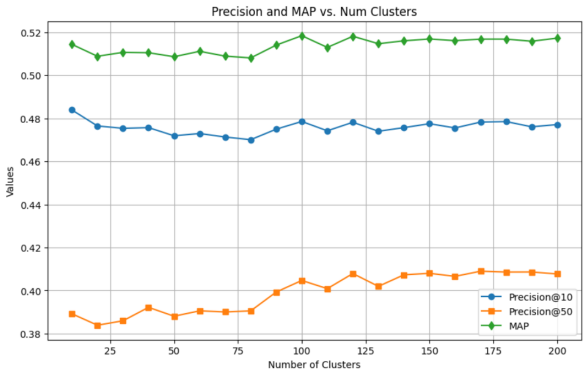
4 Results

4.1 K-Means

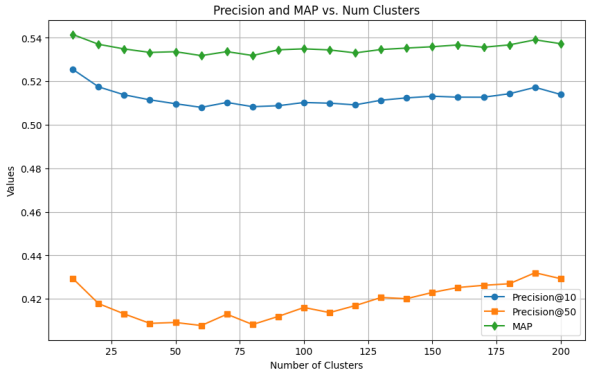
4.1.1 Variation with K



(a) CIFAR-10



(b) CIFAR-100

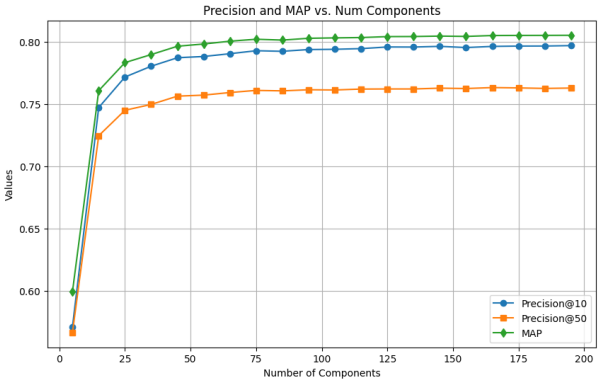


(c) ImageNet-1K

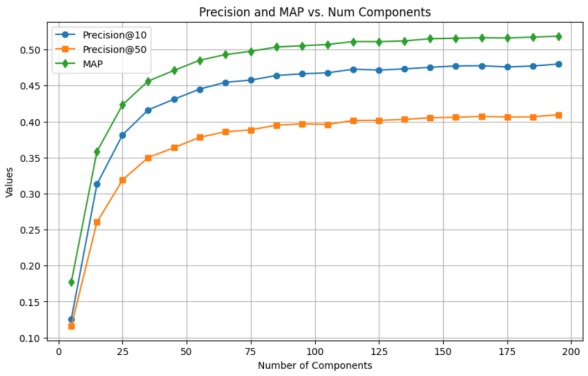
Figure 1: Varying number of clusters (K)

4.1.2 Variation with PCA

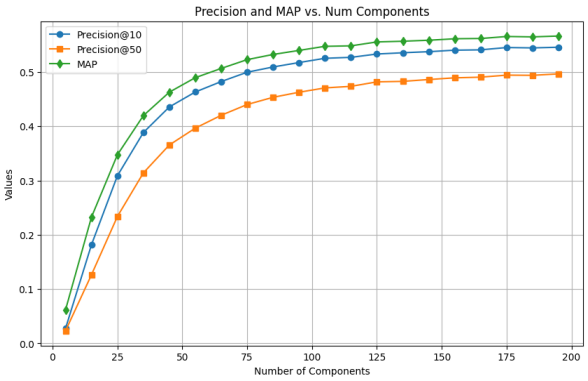
Here we applied PCA first which reduced the dimensions of the feature vectors to *num_components* used in PCA. Then, performed K-Means clustering on these features.



(a) CIFAR-10



(b) CIFAR-100



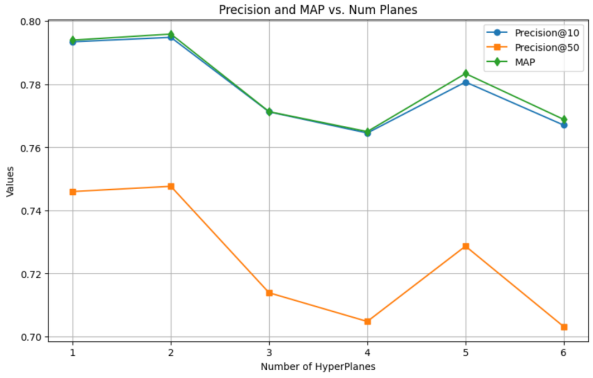
(c) ImageNet-1K

Figure 2: Varying number of Principal Components

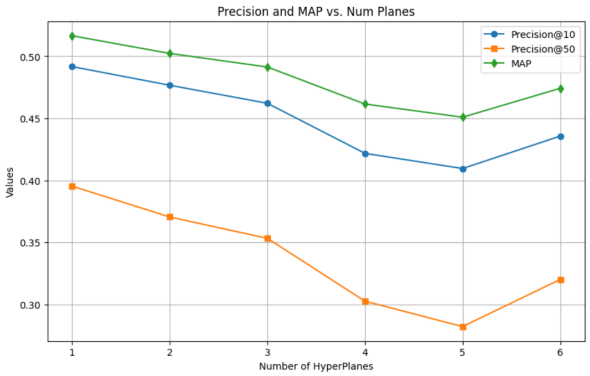
4.1.3 Observations

- From CIFAR datasets, we can deduce that Results are best (with minimal variation), when $K = num\_labels$.
- In the graphs of PCA, we can observe that the precision values remain almost constant when we decrease the number of components, upto a certain threshold, for example 50. After that, if we reduce more, precision takes serious hit. Hence we can safely reduce dimensionality from 2048 to 50, without any serious effect on precision, while significantly reducing computational complexity and time.
- Among CIFAR-10 and CIFAR-100, we observe that the trend of values remain similar, while the precision values decrease with increase in number of labels in the dataset. This effect can be attributed to the fact that because our similarity function and embedding may not exactly capture the reasoning behind the labeling, so the finer the labeling of the dataset, the more likely that our embeddings and similarity functions don't capture the nature of the labeling and hence deem non-similar images as similar.

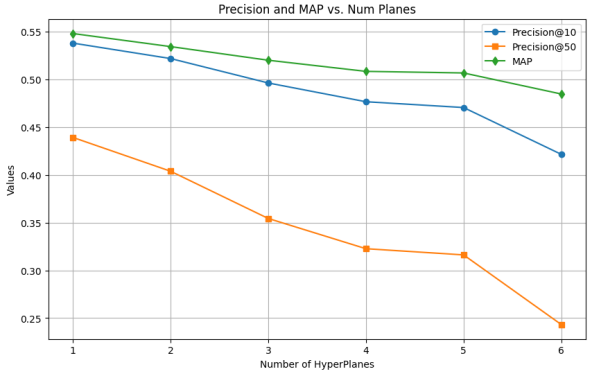
4.2 Random Projection Hashing



(a) CIFAR-10



(b) CIFAR-100



(c) ImageNet-1K

Figure 3: Varying number of Hyperplanes

4.2.1 Observations

- The precision appears to be slightly decreasing as number of hyperplanes increase. This is because we increase pruning of our search space with more hyperplanes, and hence it discards more of the relevant images as belonging to different hashes. Increasing number of planes decreases search space and hence decreasing compute time, but losses precision due to discarding relevant images.
- Among CIFAR-10 and CIFAR-100, the previous observation still holds, that the trend of values remain similar, while the precision values decrease with increase in number of labels in the dataset.

4.3 Neural LSH

	Precision@10	Precision@50	MAP
CIFAR-10	0.7783	0.7229	0.7788
CIFAR-100	0.4610	0.3499	0.4934
ImageNet-1K	0.5159	0.3735	0.5326

Table 1: Precision of Neural LSH vs Datasets

4.3.1 Observations

- The trend of precision among CIFAR-10 and CIFAR-100 as mentioned above still holds.

4.4 Observations - Across Methods

- We observe that the different hashing/clustering methods do not have any significant impact on the precision values. This, counter-intuitively, suggests that a simple KMeans Clustering is as good as Neural LSH.