CENG483 HW1

Constructing a CBIR System

Murat Topak
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
murat.topak@metu.edu.tr

***Abstract*—This document explains the work carried out and presents results about constructing a Content Based Image Retrieval (CBIR) system based on Bag of Features (BoF) representation of SIFT and Dense-SIFT local descriptors.**

Keywords—Content Based Image Retrieval; Interest Points, Local Feature Descriptors; SIFT; Dense-SIFT; Clustering; Bag of Features Representation

I. INTRODUCTION

When building up a CBIR system with a BoF approach, each image in the dataset is said to have some predefined visual words. These entities help comparing images with respect to their content. Basically, a normalized histogram of words contained in each image can be used to infer similarity. This is possible through calculating the Euclidean distance between the histograms. The lesser the distance between two images, the more similar they are.

In the experiments, the visual words are defined from the dataset by clustering the descriptors extracted from the images. K-Means, SIFT and Dense-SIFT are used to cluster and extract the local descriptors. Also, to see the effect of the different number of unique words, the results are presented with varying number of clusters.

II. IMPLEMENTATION DETAILS

OpenCV's SIFT related features have been used throughout the experiments. This includes finding key points and computing the local descriptors. While SIFT implementation uses both, Dense-SIFT implementation extracts key points from predefined regions, so it only uses OpenCV to compute local descriptors from these.

For clustering, Scikit-learn's MiniBatchKMeans has been used. It was not the traditional K-Means because the test machine was having memory problems with that. As compared in [1] the two produce highly similar results.

When finding key points with the SIFT, some images were problematic. More specifically, the images that seemed to have only noise were hard to find any key points since SIFT's key point detector uses difference of Gaussian (DoG) and this eliminates the noise by blurring such an image, so nothing is left to detect. An example of this is in Fig. 1.

Dense-SIFT has been used to find key points in these rarely occurring cases so that each image in the dataset is guaranteed to have visual words.

When finding key points with Dense-SIFT, memory of the machine was the main concern. This allowed defining patches at regular intervals of 15 pixels and nothing smaller. For example, Dense-SIFT finds the key points in Fig. 2 for the image in Fig. 1.

Fig. 1: SIFT finds no key points in ‘uiSypscOYh.jpg’.

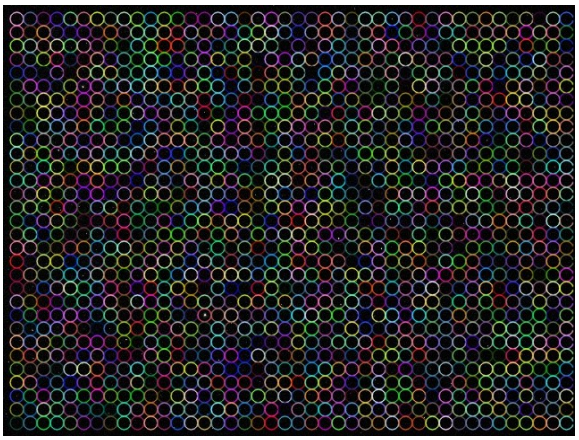

Fig. 2: Dense-SIFT key points at every 15 pixels.

III. Results

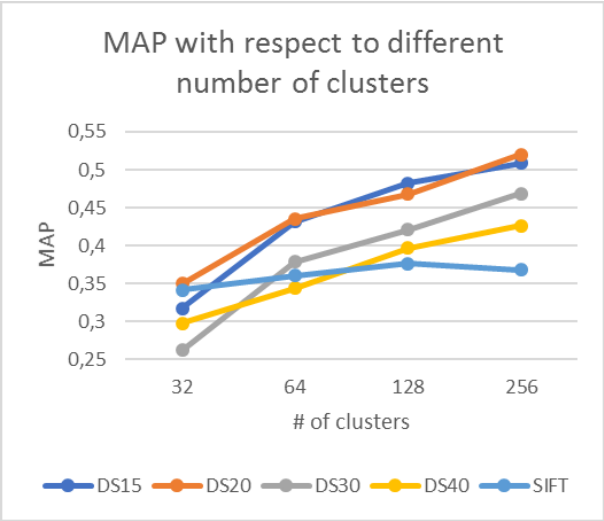

Fig. 3: MAP values with different methods

SIFT vs. DSIFT

In Fig. 3, there are four lines belonging to Dense-SIFT. Each one of them has a different step size (15, 20, 30, 40). For example, it was 15 in Fig. 2. The smaller the number, the denser the key points in the image.

Having explained that, it is clearly seen any DSIFT approach eventually gets better than the plain SIFT as the number of clusters increases. Another observation is SIFT does not improve much with different number of clusters while DSIFT follows a linear fashion in the same case. Therefore, it wouldn't be wrong to say DSIFT performed better than SIFT in the experiments. This is no surprise. In [2], it is mentioned SIFT applied at dense grids (Dense-SIFT) have been shown to lead to better performance for tasks such as object-categorization and texture classification.

A basic explanation for this is that a larger set of local image descriptors computed over a dense grid usually provide more information than corresponding descriptors evaluated at a much sparser set of image points. However, it is not all good with Dense-SIFT. The computational complexity of it will always be a problem. In the experiments, the machine was not able to go smaller than DSIFT15.

Number of Clusters

It can be inferred from the figure that better MAP is achieved as the number of clusters increases. This is related to the fact that a dictionary with more unique words can describe an image more precisely.

The only exception to this is SIFT. The reason might be that extracted key points are of not that much use compared to DSIFT. Hence, this implies DoG key point detector is not as good as DSIFT's regularly quantized patches at describing an image.

Even though it was not specified, when run with 512 clusters, the performance was still getting better in some cases.

IV. Query Results

The best MAP in the experiments has been obtained by DSIFT20 with 256 clusters. The following figures demonstrate the rights and wrongs of the model.

For example, in Fig. 4 and Fig. 5, the output image is the zoomed in version of the input image. These matches make sense since SIFT is known to resistant to scale changes.

On the other hand, the incorrect match can be explained with the fact that SIFT descriptors work with the gradients. For example, in Fig. 6, the blue-red canoes and the blue red regions in the carpet might have similar image gradients.

Fig. 4: A correct match

Fig. 5: Another correct match

Fig. 6: An incorrect match

REFERENCES

[1] "Comparison of the K-Means and MiniBatchKMeans clustering algorithms — scikit-learn 0.19.1 documentation", Scikit-learn.org, 2018. Available: http://scikit-learn.org/stable/auto_examples/cluster/plot_mini_batch_kmeans.html. [Accessed: 12- Apr- 2018].

[2] T. Lindeberg, *Scale invariant feature transform*, vol. 7, no. 5. 2012, p. 10491.