# A Comparison of SIFT and Dense SIFT on Content-based Image Retrieval

Ismail Hakki Kocdemir, *Computer Engineering Department, METU*

**Abstract**—In this report, I compare regular and Dense version of SIFT (Scale-invariant Feature Transform) , a well-known feature detection and description algorithm for images. I have experimented with different settings of parameters on both versions to see the effect individually and relative to each other. Performance criterion is the mAP( mean average precision) scores on a content-based image retrieval task conducted on our dataset.

**Index Terms**—computer vision, SIFT, bag of words, dense SIFT, CBIR, feature extraction

## 1 Introduction

BEING able to detect descriptive features in an image has many applications in computer vision including object detection, tracking and 3D modeling. SIFT (Scale-invariant Feature Transform) is found to be one of the best techniques in matching accuracies, distinctiveness and invariance to scale and transformations. In this report, I analyze the performance of SIFT in image retrieval task and compare it to an altered version, namely Dense SIFT. Throughout the article, I will be explaining the principles behind the algorithms and how they differ, describing the steps in the image retrieval pipeline and presenting my results from several experiments I conducted and reason about them.

## 2 SIFT

SIFT was proposed in [1] by David Lowe in 1999 for the first time. The algorithm detects and describes interest points on an image. Points that could be helpful for describing the image or a section of the image is detected by a what we can call a filter that can detect significant changes in the intensity (gradient) in the image, or namely edges/corners. In SIFT, this is done by convolving the image with Gaussian function with different values for variance, and then taking differences of these convolved images. The difference of Gaussians yields a scale space where we can look for a maximum in response to the filter indicating an edge/corner. After detection/localization of key points, a descriptor for each of them is formed by summarizing the gradients on a fixed sized window centered around the corresponding key point into a histogram.

## 3 DENSE SIFT

In SIFT, interest points are detected in a sparse manner with Difference of Gaussians method. Conversely, in Dense SIFT [5], image is densely sampled in periodic locations and no detection is performed prior to creating descriptors for each sample. Dense SIFT becomes advantageous in object detection tasks since more information is extracted from the image unlike the original algorithm providing descriptors on a more sparsely extracted feature point set. However, it should be noted that dense SIFT is not invariant to scale since descriptors is evaluated on fixed locations with fixed size, hence, different tunings on frequency of the sampling and the window size can produce different results.

## 4 Methodology

Here, I will explain the steps I followed to test and compare the algorithms on image retrieval performance with given dataset. Roughly, I first extracted the features with SIFT, created a database of visual words by clustering those features, represented each image with a histogram formed with extracted visual words, and made queries based on the similarities of these representations.

### 4.1 Representing Images with Bag of Features

I used a cPython [3] wrapper of *vlfeat* library [4], *cyvlfeat*, which was originally written in C, to extract the interest points in each image on the dataset both with SIFT and dense SIFT.

For dense SIFT, I used default windows size of 3 and a step size of 10. I have not tried different window sizes, but step size of 10 is what I found to be the optimum value for the performance on our dataset within the limits of my patience for the duration of extraction process, and what my piece of hardware could pull off in that duration.

To group key points for constructing a visual word dataset for all images, I made use of Kmeans clustering algorithm [2] as implemented in *sklearn* library in Python. In order to see the effect of the number of clusters and difference between dense and original SIFT quickly, initially, I used a small random sample of the dataset with $10000$ images. With this sample, I have tried following values for clusters with both original and dense descriptors: $k = 32, 64, 128, 256$.

After the point where I could see the pattern, I have run the clustering on bigger samples such as $25\%$ and $5\%$ of the whole data. I could not use all images due to insufficient memory and excessive amount of time

required for the computation. With $5\%$ of the images, I have tried $k = 32, 64, 128$ for both original and dense SIFT descriptors. For $k = 256$, I only clustered dense version. With the sample formed with $25\%$ of images, I only aimed for the best performers which were $k = 128, 256$ for both versions of descriptors. The close mAP scores on corresponding number of clusterings in smallest sample and the larger samples assured that the smallest sample is able to demonstrate the effect of $k$ and descriptor type (dense or original), hence, I did not repeat clustering for smaller $k$ values.

I used cluster centers to encode each image in the dataset with the distribution of its features on the clustered visual words in a histogram with the number of cluster centers. Since all images have possibly different number of descriptors, a similarity metric is likely to fail when a pair of images compared differ in number of descriptors by a significant margin. Thus, I applied $l1$ normalization to each histogram.

### 4.2 CBIR Pipeline

Content-based Image Retrieval is basically returning a set of images for a given query based on a similarity metric performed on features extracted from the images. For our case, we have bag of feature representations of all images prepared beforehand. Given a query, I just pair the query image's representation with the rest of the images and return a score for each pair, which is simply calculated by euclidean distance of the feature vectors.

## 5 Experiments and Results

As can be seen on Table 1, 2 and 3, I have complete result set for the all cluster on the sample with size $10000$. This smallest set is sampled from the clusters/visual words extracted from SIFT descriptors, and Dense SIFT descriptors where parameters were set as follows: $step = 10, window = 3$.

Results on Table 1 suggests that, densely extracted features consistently produces better scores event when the number of descriptors are the same and randomly sampled from entire descriptor sets. The reason I think for that is , with such small sample, randomization turns out to produce better representations when descriptors are sampled from uniformly spaced locations (dense SIFT) on the image than descriptors that SIFT itself located, which are not quite that uniform (This might be causing bias towards certain features that are larger in number). In Table 2 and 3, where I used larger samples, the pattern does not change and dense SIFT performs better. However, with a different dataset, we might not get a similar results because of the step and window size in dense SIFT, which might require a different tuning. Hence, I doubt if the images required to be similar in the dataset are require considerable invariance to scale.

In figure 1, we see that there is always an increase in mAP scores with the increase in number of clusters. However, unlike SIFT, dense SIFT does not demonstrate

TABLE 1
Scores for sample with size $10000$

| Clusters | SIFT mAP | Dense SIFT mAP |
|---|---|---|
| 32 | 0.316328 | 0.36593 |
| 64 | 0.38397 | 0.39549 |
| 128 | 0.40907 | 0.46185 |
| 256 | 0.43852 | 0.46756 |

TABLE 2
$\%5$ of the descriptors: SIFT: $84876$, Dense SIFT: $227878$

| Clusters | SIFT mAP | Dense SIFT mAP |
|---|---|---|
| 32 | 0.33175 | 0.36197 |
| 64 | 0.39064 | 0.41310 |
| 128 | 0.41584 | 0.44522 |
| 256 | - | 0.47098 |

significant improvement over from $128$ to $256$ clusters. This also the case with larger samples. Hence, we can say that for dense SIFT descriptors, $128$ can be enough for representing the dataset we have. Nonetheless, higher number of clusters might yield better scores for SIFT features.

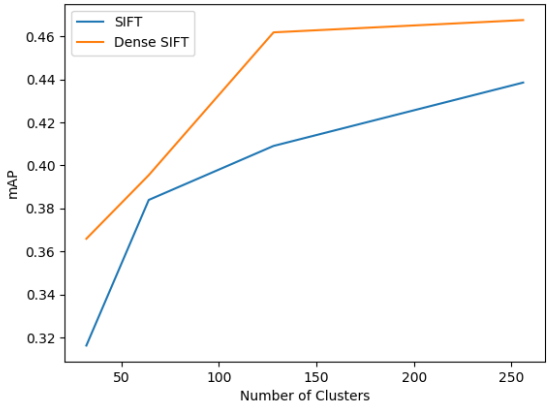


Fig. 1. mAP scores with different number of clusters. $10000$ descriptors are sampled.

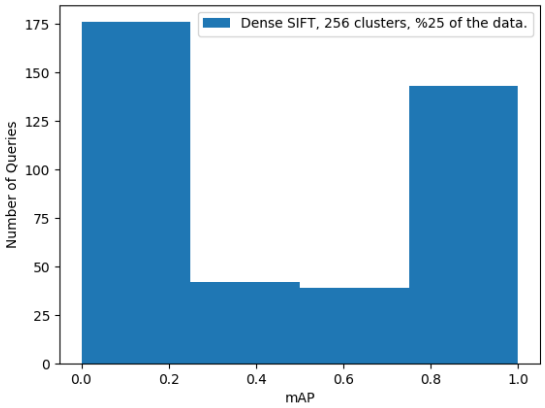


Fig. 2. Histogram of Dense SIFT mAP scores.

TABLE 3
%25 of the descriptors: SIFT: 424380, Dense SIFT: 1139392

| Clusters | SIFT mAP | Dense SIFT mAP |
|---|---|---|
| 128 | 0.41592 | 0.43969 |
| 256 | 0.43825 | 0.46049 |

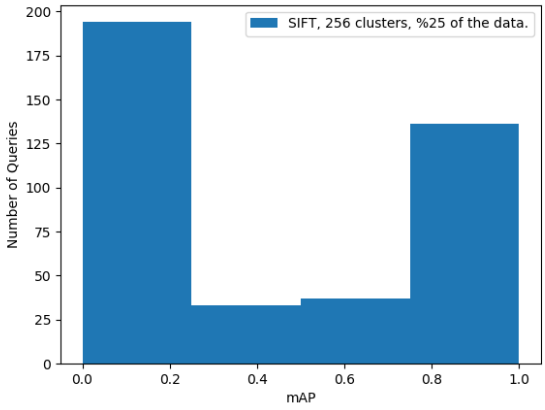


Fig. 3. Histogram of SIFT mAP scores.

Figures 4 and 5 reveals that representations either produce either very correct or very bad retrieval scores. The jump from mAP score of 36% with $k = 32$, to 47% with $k = 256$ show itself mostly as exchange between very low scores and very high scores in queries. This also can be observed in Table 4, with the increase in the standard deviation which accords with mediocre scores remaining almost the same and extreme scores getting more equally separated to the edges. In figure 2 and 3, where I think I have obtained the most consistent scores due to highest number of clusters and largest sample size, dense SIFT has consistently more queries in higher quartile intervals (higher half) than original SIFT.

TABLE 4
%5 of the descriptors: SIFT: 84876, Dense SIFT: 227878

| | | Bins for mAP | | | |
|---|---|---|---|---|---|
| Clusters | Std. Dev. | .0-.25 | .25-0.50 | 0.50-.75 | .75-1. |
| SIFT | | | | | |
| 32 | 0.4062 | 239 | 41 | 20 | 100 |
| 64 | 0.4149 | 210 | 35 | 44 | 111 |
| 128 | 0.4228 | 198 | 38 | 37 | 127 |
| Dense SIFT | | | | | |
| 32 | 0.4062 | 220 | 38 | 44 | 98 |
| 64 | 0.4149 | 199 | 41 | 41 | 119 |
| 256 | 0.4228 | 178 | 38 | 34 | 150 |

## 6 Conclusion

With this work, I have explained and compared the properties of SIFT and dense SIFT, presented their application to Content-based Image Retrieval and performance differences. I did not see any case where original SIFT outperformed dense version. This is in line with what is stated in the literature: dense SIFT is a better option when pairing images/detecting objects thanks to density of the interest point extraction resulting in more informative representation of images.

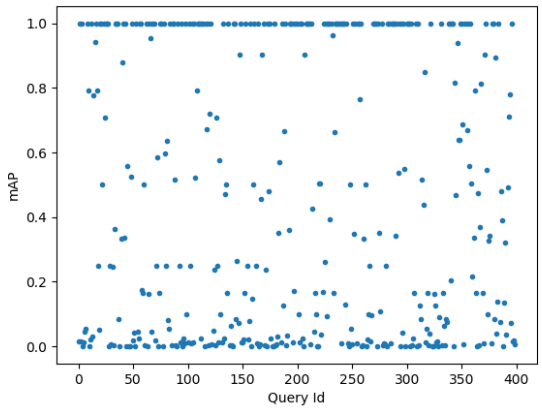


Fig. 4. Dense SIFT with 256 clusters. %5 of the descriptors are sampled.

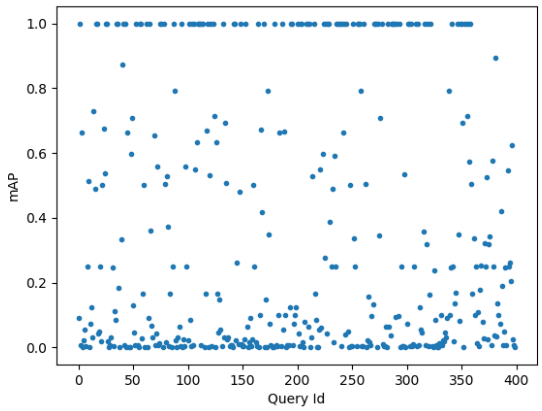


Fig. 5. Dense SIFT with 32 clusters. %5 of the descriptors are sampled.

## References

[1] Lowe, David G. (1999). Object recognition from local scale-invariant features. Proc. 7th International Conference on Computer Vision (ICCV'99) (Corfu, Greece): 1150-1157.

[2] Lloyd, Stuart P. "Least squares quantization in PCM." Information Theory, IEEE Transactions on 28.2 (1982): 129-137.

[3] Stefan Behnel, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn and Kurt Smith. Cython: The Best of Both Worlds, Computing in Science and Engineering, 13, 31-39 (2011), DOI:10.1109/MCSE.2010.118

[4] Vedaldi, Andrea and Brian Fulkerson. Vlfeat: an open and portable library of computer vision algorithms. ACM Multimedia (2010).

[5] Fei-Fei, Li, and Pietro Perona. "A bayesian hierarchical model for learning natural scene categories." Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 2. IEEE, 2005.