Effective Image Database Search via Dimensionality Reduction

Dahl, Anders Bjorholm; Aanæs, Henrik

Published in:
2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops

Link to article, DOI:
10.1109/CVPRW.2008.4562957

Publication date:
2008

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Dahl, A. B., & Aanæs, H. (2008). Effective Image Database Search via Dimensionality Reduction. In 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (pp. 1-6). Anchorage, Alaska: IEEE. DOI: 10.1109/CVPRW.2008.4562957

Effective Image Database Search via Dimensionality Reduction

Anders Bjorholm Dahl[†‡] and Henrik Aanæs[†]
Informatics DTU, The Technical University of Denmark[†]
Dralle A/S[‡]
abd@imm.dtu.dk haa@imm.dtu.dk

Abstract

Image search using the bag-of-words image representation is investigated further in this paper. This approach has shown promising results for large scale image collections making it relevant for Internet applications. The steps involved in the bag-of-words approach are feature extraction, vocabulary building, and searching with a query image. It is important to keep the computational cost low through all steps. In this paper we focus on the efficiency of the technique. To do that we substantially reduce the dimensionality of the features by the use of PCA and addition of color. Building of the visual vocabulary is typically done using k-means. We investigate a clustering algorithm based on the leader follower principle (LF-clustering), in which the number of clusters is not fixed. The adaptive nature of LF-clustering is shown to improve the quality of the visual vocabulary using this. In the query step, features from the query image are assigned to the visual vocabulary. The dimensionality reduction enables us to do exact feature labeling using kD-tree, instead of approximate approaches normally used. Despite the dimensionality reduction to between 6 and 15 dimensions we obtain improved results compared to the traditional bag-of-words approach based on 128 dimensional SIFT feature and k-means clustering.

1. Introduction

Effective search algorithms are important to overcome the challenges of image search on the Internet. The bag-of-words approach have shown promising results for large scale image search [8, 16, 17, 18, 20] and for image categorization [6, 22, 23].

There are three main steps in the bag-of-words representation: (*i*) Feature extraction from the database images, (*ii*) building the bag-of-words representation, (*iii*) and searching with a query image.

Image features (*i*) The image features are descriptions of local image patterns, see e.g. [7, 11, 13, 14]. In the bag-of-words representation they are treated as an independent collection of data points characterizing the depicted scene. Features are typically n-dimensional vectors of unit length, thus points on an n-dimensional hypersphere.

Representing features as 128 dimensional SIFT vectors have shown to be very effective for object recognition problems [11, 15]. Despite the discriminative power of the SIFT features it is computational expensive to represent image features with 128 dimensions. PCA have been applied to SIFT features of different dimensionality [8, 9, 15]. The performance of these descriptors was comparable to the original descriptor, with a reduction to the range of 20 to 36 dimensions.

In this paper we show it to be efficient to reduce the dimensionality of the feature descriptor much further. We use PCA on SIFT features with a reduction in the range of 6 to 15 dimensions including 3 color dimensions. This is a very compact representation compared to the 128 dimensional SIFT features.

Visual vocabulary (*ii*) Features have been used for object recognition where similarity between images is found by comparing features directly, see e.g. [10, 11, 19]. For large image collections this is not feasible, which is the motivation for the bag-of-words representation.

The bag-of-words approach is based on representing features by canonical representative instead of features themselves. Canonical feature representatives can be viewed as visual words. Each feature in an image is labeled with a reference to a visual word. This way the image is described as a histogram of visual words - a frequency vector, which is a much more compact representation than the individual features.

It was proposed by Sivic and Zisserman [20] to build a image search method based on he idea of text retrieval in large document collections. They demonstrated an efficient algorithm based on inverted files and feature weighing. Feature weights are found from the distribution of visual words

978-1-4244-2340-8/08/$25.00 ©2008 IEEE

in the database images.

Visual words are typically found by clustering of features from a database of images. Often k-means clustering is used for building the visual vocabulary. In k-means the number of clusters is fixed, which can lead to undesirable partitioning of the data. This way natural clusters have the risk of being merged or split, leading to mislabeling of features and loss in discriminative power. Philbin *et al.* [17] did not find other clustering methods than k-means an option, because of the high number and high dimensionality of the features.

The dimensionality reduction applied in this paper, enables us to use a clustering approach inspired by the sequential leader-follower clustering (LF-clustering) [5, 12]. This way we are able to find clusters in a very effective manner, without knowledge about the number of clusters before the clustering takes place.

Image query (*iii*) The features in a query image is labeled with a reference to the nearest visual words in the vocabulary. The complexity of this assignment is dependent on the size of the visual vocabulary. It is empirically shown that visual vocabularies have to be relative large to be effective. I.e. in the range from 5K to 1M visual words typically of 128 dimensional SIFT descriptors [8, 16, 17, 18, 22, 23]. Finding the exact nearest neighbor in high dimensions is hard to do in less than $O(n^2)$ complexity, so approximations are used.

Nistér and Stewénius [16] came up with the idea of using a tree structure to simultaneously speed up the clustering and the feature labeling. This *Vocabulary Tree* is made by a hierarchical k-means clustering approach. The tree can be used for an approximate feature search which reduces the complexity from $O(n^2)$ to $O(n \log n)$. The approximate feature search is improved in [18] based on the ideas of [2].

Despite the efforts of improving the feature assignment for the query images, the computation is still approximate. The dimensionality reduction of image features that we use enables us to effectively use a kD-tree for feature labeling. This way we obtain an exact nearest feature assignment.

These modifications simplifies the computation of the bag-of-words representation, in both building, storing and searching the image database. This is even done with an improvement in recognition quality. Effective computation is essential for reaching Internet scale image search.

2. Methods

The methods used for improving the efficiency of the bag-of-words approach is described in the following. The improvements are related to reducing the size of the feature descriptors, improving the clustering approach, and improving the feature assignment. The steps in building the bag-of-words model is illustrated in figure 1.

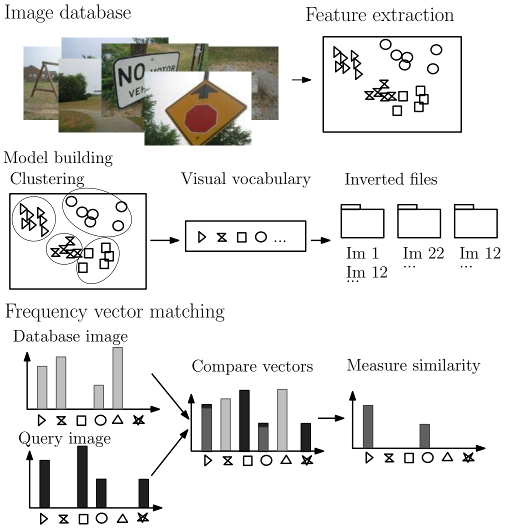

Figure 1. Illustration of the bag-of-words model. Features are extracted from the image database. The visual vocabulary is build using clustering, and inverted files is made based on the feature labeling of the database images. Finally feature vectors are used for matching images, with the frequency vector overlap giving the matching score.

Feature representation PCA is applied to reduce the dimensionality of the feature vectors. We calculate the eigenvectors for the PCA from all features in our training set. The reduction of the SIFT descriptor is from 128 to between 3 and 12 dimensions.

After dimension reduction we add color to our features. The color is simply the mean RGB value in a 10×10 pixels patch around the localization of each feature. The color patch is concatenated to the PCA reduced SIFT vector:

$$s = [\alpha s_{PCA}, (1-\alpha) s_{RGB}] \qquad (1)$$

where s_{PCA} is the PCA reduced SIFT feature, s_{RGB} is the mean RGB values, and α is a weighing parameter. In our experiments we found $\alpha = 0.5$ to be a good choice. So the final feature gets a size of 6 to 15 dimensions.

We use histogram equalization of the whole image for the R, G, and B bands separately, before we extract the color features. This is done to minimize the effect of change in light between two images of the same scene. Both the PCA-SIFT and color feature are normalized to unit length before concatenation, and normalized again after concatenation. This way we try to avoid non intended bias because of difference in size of the two features.

Clustering We investigate the use of LF-clustering [5, 12] as an alternative to k-means. The motivation for this method is to avoid the risk of undesirable partitioning of data caused by a wrong number of clusters. In the LF-clustering algorithm, data is treated sequentially without any iterative steps. This gives the method a potential for being less computational expensive than k-means.

LF-clustering builds on the idea of defining a minimum dissimilarity between clusters. This is similar to the Mean-Shift clustering algorithm [3], where clusters are found according to a certain bandwidth. But the computational cost of LF-clustering is far less than Mean-Shift.

Algorithm 1 Hierarchical Leader Follower clustering (LF-clustering)

```
Set data in one cluster: $d_{cl}$
Initialize number of levels: $n$, bandwidth start: $b_s$, bandwidth end: $b_e$
Set number of clusters: $n_{cl} = 1$
Bandwidth step $b_{st} = (b_S - b_e)/(n + 1)$
for $i = 1$ to $n$ do
    $b_{now} = b_e + (n - i + 1)b_s + b_e$
    for $j = 1$ to $n_{cl}$ do
        cluster($d_{cl}, b_{now}$)
        update $n_{cl}, d_{cl}$
        save clusters
    end for
end for
```

The hierarchical LF-clustering algorithm is summarized in algorithm 1. The procedure of the cluster step is to treat the features sequentially one at a time. The first feature makes up the first cluster. If the next feature is closer to the existing cluster than the defined bandwidth, it will be assigned to this cluster. Otherwise, it is will make a new cluster. When a feature is assigned to an existing cluster, the center of the cluster is updated. We use euclidean distance, so the cluster center is updated by:

$$c_n = \frac{c_o n_o + f_a}{n_n} \tag{2}$$

where c_n is the new cluster center, c_o is the old cluster center, n_o is the number of features in the cluster before the feature f_a is added, and the number of features is updated with one: $n_n = n_o + 1$.

To avoid clusters coming too close together we merge clusters being closer than the bandwidth, and we also set a minimum limit to the number features in a cluster. Clusters with too few point will be merged with the nearest cluster. In both steps the center of the cluster is updated according to equation (2).

The speed of this algorithm is dependent on the number of clusters. The expected complexity is $O(kn)$, where k is the number of clusters and n is the number of points. Small bandwidths and high dimensions result in many clusters and slows down the algorithm. To compensate for that we do the clustering hierarchically, starting with a large bandwidth and shrink it through the clustering process. We start clustering the whole point set. The obtained clusters are subsequently clustered into new clusters in the following steps. This way we obtain a substantial increase in speed. To avoid clusters getting to close the hierarchical clustering is followed by a merging step. All clusters closer than the bandwidth are removed.

We compare the performance of the LF-clustering to k-means. k-means is also applied hierarchically to obtain increased speed. We also define a minimum number of clusters for k-means clustering. Further clustering is stopped if a cluster has less than a minimum number of points. This can lead to very small clusters, but in our experiments the number of clusters containing very few points is negligible.

With hierarchical clustering we get clusters at each level in the hierarchy. But we only use the clusters found at the last level for the visual vocabulary.

Feature assignment Similarity of images are found by comparing frequency vectors of a query image to images in the database. Frequency vectors are made from the frequency of visual words in an image weighed with an entropy weight. The entropy weight is based on the distribution visual words in the database images. We use the same weight as used in [16], which is defined as:

$$w_i = log(\frac{N}{n_i}) \tag{3}$$

where w_i is the weight of word i, N is the total number of images in the database, and n_i is the number of images where word i occurs.

Frequency vectors for database images are given from clustering. But for a query image we need to label the features with visual words, so we need to find the visual words closest to the features in the query image. The dimensionality reduction of the features enables us to effectively use a kD-tree instead. This makes the feature assignment exact [4]. For small dimensions the expected complexity of the kD-tree is $O(n \log n)$, whereas for high dimensions the complexity becomes $O(n^2)$. Therefore, the kD-tree is only applicable with substantial dimensionality reductions.

Image matching Frequency vectors are compared using the L_1 norm, which is found to be superior to the euclidean distance just as observed in e.g. [16]. Our explanation for the L_1 norm being superior to the L_2 norm is the nature of the problem we are solving. We expect the same features to occur in images from the same scene. So the frequency vector overlap is a good measure for similarity between images. The L_1 norm gives equal weight to the overlapping and non overlapping parts, whereas the L_2 norm gives more weight to the non overlapping parts.

Before comparing the frequency vectors are normalized to unit length using the L_1 norm.

Inverted files are used for fast image retrieval. An inverted file is kept for each visual word in the vocabulary. In the inverted file is a reference to the images in the database containing that word. The frequency vector value of each reference image is stored together with the reference. Image retrieval is obtained by first calculating the the frequency vector for the query image using the weights in equation (3). With the frequency vector value for the reference images stored in the inverted files, we can compute the L_1 norm without looking up the entire frequency vectors of the reference images. This can be done because the L_1 norm can be calculated from the frequency vector overlap:

$$L_1 = 2 - 2O \tag{4}$$

where L_1 is the L_1 norm and O is the frequency vector overlap:

$$O = 2 - 2 \sum_{i|q_i \neq 0 \wedge d_i \neq 0} \min(q_i, d_i) \tag{5}$$

where q_i and d_i is the frequency values of query and database images respectively. Instead of ranking images by smallest L_1 norm we rank by largest overlap. Storing a value for each image together with the pointer in the inverted file has a memory cost, which should be viewed in relation to looking up frequency vectors for relevant images.

3. Experiments and Results

The results in this paper are primarily found through empirical studies described in the following section.

3.1. Data set

We use the first 1400 images from the Nistér and Stewénius data set [16, 21] in our experiment. This data set contains a series of 4 images of the same scene, so we have 350 different scenes. We use three of the images from one scene to train the model and the last for testing. The test result is the percentage of the correct images ranked in top 3. This data set is relatively small compared to other experiments, but we found it sufficient for illustrating the effects of our model choices. In future work this should be extended to a larger data set. We also use the preprocessed SIFT features supplied with this data set.

3.2. Experiments

To test the effect of using color added SIFT features and LF-clustering we have made experiments with and without color features and with ordinary k-means and LF-clustering. We have also built a model based on the 128 dimensional SIFT features with and without color to illustrate the performance of the ordinary SIFT features.

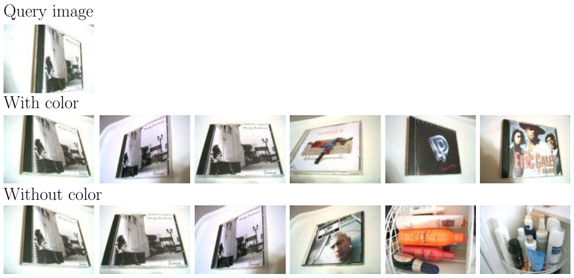

Figure 2. First 6 images retrieved using 11 dimensional vectors with and without color. With color the highest ranked images looks alike, whereas without color number 5 and 6 are quite different from the rest. This was observed as a general trend.

Color added PCA SIFT These features are made as described in section 2. We use 3, 8, and 12 dimensional PCA SIFT features, so the resulting color added features are 6, 11, and 15 dimensions. To compare to features without color we take SIFT features reduced with PCA to 6, 11 and 15 dimensions.

Clustering experiments For all test sets we have done clustering using k-means and LF-clustering. The number of features from the LF-clustering are in the range from 8,000 to 12,000 clusters, so we have chosen to let the k-means cluster hierarchically to 10 clusters in 4 levels resulting in 10,000 clusters.

3.3. Results

We have summarized the experimental results in table 1 and 2. The best classification results are obtained with LF-clustering with 15 dimensional color added features. LF-clustering is slightly better than k-means. But the most pronounced effect is the addition of color, which significantly improves the result. It should be noticed, that performance is improved in relation to the full SIFT feature, even with added color. In the full model we also use exact feature assignment for comparison, even though it would not be fast enough for a real application.

Addition of color also gave a ranking of the images that seemed more logical, which is shown in figure 2.

4. Discussion

Our experiments shows that it is possible to obtain a good recognition performance with the bag-of-words model with a substantial reduction in dimensionality of the features. A reduction to between 6 and 15 dimensions makes it

With color			
Method	6 dim	11 dim	15 dim
k-means	81.7	87.2	88.4
LF-clustering	84.0	87.5	**89.9**
Without color			
Method	6 dim	11 dim	15 dim
k-means	73.3	81.6	82.2
LF-clustering	76.3	81.9	83.9

Table 1. Model performance with different clustering methods, dimensions of the features, and with and without color. Notice the effect of adding color the SIFT features. The best performance is marked in bold.

Method	128 dim	131 dim (color)
k-means	85.7	89.6

Table 2. k-means clustering for the full SIFT features. The results for 128 dimensions is the normal SIFT feature and the 131 dimensions is with color added.

possible to use exact methods for feature assignment, where we use a kD-tree.

For our experiments the best results are obtained with a vocabulary based on 15 dimensional PCA color features using on LF-clustering. It even outperforms the full SIFT features including color, and it is substantially better than the normal 128 dimensional SIFT features. Even with 11 dimensions we obtain better results than with normal SIFT.

Especially the effect of adding color to the PCA reduced SIFT features is very important for the performance. The method for adding color to the features is extremely simple and yet very powerful, which indicates that there are much information in image color. The histogram equalization works well for this data set. But this might be overly simple for images with high variation in viewpoint. Other ways of combining the gradient information from the SIFT features and color information might be even more powerful, and should be investigated further, see e.g. [1].

Another very important benefit from the information gain from adding color, is the dimension reduction of the features. Low dimensional features makes the model computational less expensive. This is mostly in relation the clustering for building the visual vocabulary, which can be done off line. But on line feature assignment can also be computed faster and have a higher quality, because of the option of doing exact feature search using a kD-tree.

The LF-clustering for building the vocabulary showed a slight improvement in performance. The speed of the hierarchical LF-clustering was about the same as k-means in our implementations, but the LF-clustering has potential for being faster because each point is only treated once in each level in the clustering hierarchic. A requirement is, that the number of clusters does not become too large in one clustering operation.

We did not apply LF-clustering to the 128 dimensional SIFT features, because we found it performing very poorly. The data did not cluster, so we had either one cluster containing all features or all features in their own cluster depending on the bandwidth. For the 128 and 131 dimensional features we chose only to use the k-means clustering.

Our results are good compared to [8, 16] but this might partly be due to the fact that we have only conducted experiments on 1400 images. We also just look at one data set, so for future work the model should be tested on a larger set of data. It is worth noting that the improvement is also shown relative to the normal 128 dimensional SIFT feature, but this observation should also be shown to hold for a larger data set.

We have not included any form of blocking of visual words, as suggested in [20]. In [23] stop words are used in relation to different criteria, but without any clear improvement. We experience that the entropy weighing of the descriptors improves the results. Information about the entropy of the visual words could be included in already in building of the visual vocabulary, so the total entropy of the model would be maximized relative to the number of clusters, and we might be able to obtain good performance with a small vocabulary. This is to be done in future work.

A problem of the design of the bag-of-words model is it static nature. It is not designed for adding or removing images from the database, because it will require a new clustering of all the images in the database, which is very time consuming. In future work it will be relevant to investigate how the method could be designed for the dynamic nature of many databases, e.g. image databases on the Internet.

5. Conclusion

We have shown a way to substantially reduce the dimensionality of the SIFT features used in the bag-of-words model through PCA and addition of color to the features. This has enabled us to use a kD-tree for feature labeling, which is an exact method. When 128 dimensional SIFT features are used it is necessary to apply approximations. The reduction in feature dimensions enabled us to apply a clustering algorithm based on the leader follower principle (LF-clustering) where the number of clusters is a result of the clustering. The addition of color and use of LF-clustering are compared to normal PCA SIFT and k-means clustering, which is normally applied in the bag-of-words model. We obtain a clear improvement in performance on a test set containing 1400 images. Especially adding color to the features improves the performance, whereas the clustering algorithm gives a slight improvement. We also get a clear performance improvement compared to the bag-of-words model based on 128 dimensional SIFT features and k-means clustering.

6. Acknowledgments

We thank Dralle A/S for financial support. We also thank Henrik Stewénius for helpful references and for supplying benchmark data for the experiments.

References

[1] A. E. Abdel-Hakim and A. A. Farag. Csift: A sift descriptor with color invariant characteristics. *CVPR, 2006.*, 2:1978–1983, 2006.

[2] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. *Computer Vision and Pattern Recognition, 1997.*, pages 1000–1006, 1997.

[3] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[4] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, second edition, 2000.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[6] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. *ICCV 2005.*, 2:1816–1823, 2005.

[7] L. van Gool, T. Kadir, F. Schaffalitzky, J. Matas, A. Zisserman, C. Schmid, T. Tuytelaars, and K. Mikolajczyk. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1-2):43–72, 2005.

[8] H. Jegou, H. Harzallah, and C. Schmid. A contextual dissimilarity measure for accurate and efficient image search. In *IEEE Conference on Computer Vision & Pattern Recognition*, june 2007.

[9] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. *CVPR 2004.*, 02:506–513, 2004.

[10] D. G. Lowe. Object recognition from local scale-invariant features. *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, 2:1150–1157, 1999.

[11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[12] R. López de Màntaras and J. Aguilar-Martín. Self-learning pattern classification using a sequential clustering technique. *Pattern Recognition*, 18(3-4):271–277, 1985.

[13] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, volume 1, pages 384–393, London, 2002.

[14] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[15] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[16] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2161–2168, June 2006.

[17] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. *CVPR 2007.*, pages 1–8, 2007.

[18] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. *CVPR 2007.*, pages 1–7, 2007.

[19] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.

[20] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. *ICCV 2003.*, (vol.2):1470–7 vol.2, 2003.

[21] H. Stewénius and D. Nister. Recognition benchmark images (http://www.vis.uky.edu/s̃tewe/ukbench/), 2006.

[22] B. Triggs, F. Jurie, and E. Nowak. Sampling strategies for bag-of-features image classification. *Lecture Notes in Computer Science*, 3954:490–503, 2006.

[23] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo. Evaluating bag-of-visual-words representations in scene classification. *Proceedings of the ACM International Multimedia Conference and Exhibition*, pages 197–206, 2007.