Image Retrieval using Eigen Queries

Nisarg Raval, Rashmi Tonge and C. V. Jawahar

Center for Visual Information Technology
IIIT Hyderabad, India – 500 032

Abstract. Category based image search, where the goal is to retrieve images of a specific category from a large database, is becoming increasingly popular. In such a setting, the query is often a classifier. However, the complexity of the classifiers (often SVMs) used for this purpose hinders the use of such a solution in practice. Problem becomes paramount when the database is huge and/or the dimensionality of the feature representation is also very large. In this paper, we address this issue by proposing a novel method which decomposes the query classifier into set of known *eigen queries*. We use their precomputed results (or scores) for computing the ranked list corresponding to novel queries. We also propose an approximate algorithm which accesses only a fraction of the data to perform fast retrieval. Experiments on various datasets show that our method reports high accuracy and efficiency. Apart from retrieval, the proposed method can also be used to discover interesting new concepts from the given dataset.

1 Introduction

Traditional methods for image retrieval rely on the nearest neighbour methods for similarity search. There exist many efficient indexing schemes [1–5] for this purpose. These methods work well when the goal is to retrieve images similar to the query image from the image database. Often the users intent is more complex. User may be interested in a broader concept that can only be retrieved with powerful classifiers like Support Vector Machines (SVM). This needs scalability in query by classifier setting.

With the popularity of the mobile applications such as "Google Goggles", use of instance search has also increased. This category of solutions retrieves information about a specific object/instance [6]. Many indexing schemes have been successfully used for speeding up the instance search in large databases [7, 8]. However, instead of a specific object, user might be interested in the set of images which contain a specific category. In such cases, methods based on classification outperform traditional near neighbour search approaches in terms of retrieval accuracy [6, 9–11]. This class of retrieval is the primary focus of this work. To improve the accuracy of image retrieval, the trend is to use state of the art classifiers like Support Vector Machines (SVM) with sophisticated features like Bag of Words (BoW) [7] histogram with dense pyramidal SIFT (PHOW) [12].

The idea is to train a classifier with enough positive and negative examples, and query the database with this classifier. Based on the classification scores, the database images are ranked, and the top ranked images are returned to the user. Even though this approach improves the retrieval quality significantly, the feasibility of such a system is yet doubtful. It takes considerable amount of time to classify and rank the large number of images present in the database.

We address this problem by expressing the novel queries in terms of some fixed set of distinguished queries. We efficiently compute the classification scores corresponding to novel queries for all the database images and retrieve the top-k relevant results. More precisely, our method of scoring is independent of the dimensionality of the feature representation, which is usually very high for images. We also give an extension of our method which computes the scores for only a fraction of the database, thus gaining the speedup for large databases. The proposed method is based on the concept of *Eigen Queries* which are eigen representations computed from multiple queries. They capture variations present in the queries.

Intuitively, we propose a method to find a set of prominent queries (eigen queries) for a database. This computation is done offline. The respective classification scores (eigen scores) for entire image database can be precomputed during a preprocessing step. At the query time, the novel query classifier is expressed as a linear combination of the eigen queries. These coefficients and the precomputed eigen scores are then used to efficiently compute the final classification scores for the entire database, and top ranked results are retrieved. The entire method can be summarized into three steps.

1. **Preprocessing:** Constructing eigen queries from multiple queries and computing their respective scores (i.e., eigen scores).
2. **Encoding:** Representing query classifier as a linear combination of eigen queries and computing corresponding coefficients.
3. **Retrieval:** Retrieving top-k results using previously computed eigen scores and coefficients.

We present two algorithms for image retrieval using eigen queries. The first approach is called *Eigen Search* (ES), which computes the scores for entire database and retrieves the corresponding top-k results. The second approach called *Subset Search* (SS) intelligently selects a small subset of images which have higher probability of occurring in top-k results, and processes only those images to retrieve the results.

The organization of the paper is as follows. Section 2 gives the brief review of previous related works. In Section 3, we give motivation for *Eigen Queries* and outline our approach. Section 4 and Section 5 describe the proposed algorithms in detail. In Section 6, we show experimental results to evaluate our method on some of the standard datasets. Finally, we conclude in Section 7.

2 Related Work

The concept of learning eigen functions (principal components) has been used in many computer vision tasks in past. Turk and Pentland [13] gave a simple and fast algorithm for face recognition using the concept of “eigen faces” which are essentially the eigen vectors of the covariance matrix computed from a set of different face images. Any new image can then be represented as a linear combination of these eigen faces. Similar ideas are then used in many specific problems like object detection [14], object matching and tracking [15] and visual surveillance [16]. The idea of expanding query in terms of some other queries is also popular in Concept Based Multimedia Retrieval (CBMR). However, there are certain fundamental differences between our work and CBMR. To the best of our knowledge, the proposed concept of *Eigen Queries* is novel and never used before in multimedia retrieval. We use eigen decomposition to automatically generate various meaningful concept detectors. As opposed to that, most of the CBMR approaches index the database using predefined concept detectors [17, 18]. In CBMR, query is represented by set of keywords (text) which are mapped to predefined concept detectors by various methods like statistical corpus analysis [19], using Ontology [20, 21] etc. However, our method specifically focuses on retrieving relevant images without computing scores for entire database. In a way, the rich semantic processing carried out in the text domain is replaced by a linear expansion in the classifier space (W vectors in SVMs).

Recently, Rastegari *et. al.* [22] proposed methods based on pruning and quantization to solve the problem of concept based retrieval where one is interested only in the top-k results. However, their method is restricted to binary features, and does not apply to the commonly used real features. Our method is fundamentally different from theirs in a sense that we concentrate more on the nature of the queries in a practical setting. We learn eigen queries and express novel queries in terms of eigen queries.

3 Overview

Even though billions of queries are addressed by today’s search engines on a daily basis, most of them come from a small set of unique concepts[1]. Therefore in practice, the results of these unique concepts can be precomputed on large database. Now, given a new query, it can be matched and approximated with these precomputed results.

3.1 Eigen Query

In our case, query (SVM hyperplane W) represents a category in which user is interested. The number of such distinct categories is very small compared to

[1] It has been shown that among 1 billion search queries more than 75% of the queries can be approximated by the rest 25% [23]

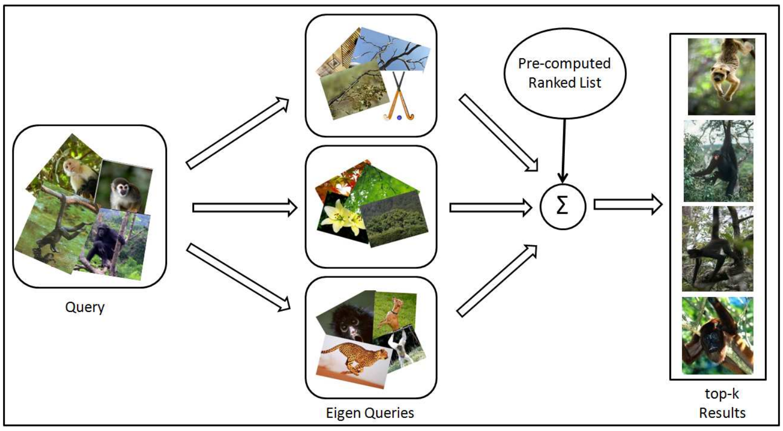

Fig. 1: Conceptual view of the proposed approach. First, the query category is expressed in terms of eigen queries (categories). Then, the scoring is performed on the precomputed result set of corresponding eigen queries. Finally, based on the scores top-k relevant results are retrieved.

the database size. Therefore, results of these queries can be precomputed even on a large database. Since we are approximating the given query in terms of these precomputed results, we need a method to come up with a robust set of queries which can efficiently represent any new query. Also, this set of precomputed results should effectively handle the wide range of new queries spanning across various categories. Hence, to capture the variations among different categories we use the technique of learning principal components of the distribution of queries. We first compute the covariance matrix of the query log (set of SVM hyperplanes) comprising various categories. Then, we perform eigen decomposition on this covariance matrix to compute its eigen vectors and eigen values. Finally, we construct *Eigen Queries* which are nothing but the eigen vectors corresponding to high eigen values. These eigen queries and their corresponding scores on the image database are generated as an offline process. During online phase of querying database, a new query is approximated by these eigen queries and corresponding precomputed scores are used to retrieve top-k results efficiently. The conceptual view of the proposed approach is outlined in Fig 1.

The performance of such a system will depend on how well the given query can be represented by the known eigen queries. To analyze the representation capability of eigen queries, we construct many eigen queries using images of Caltech256 [24] dataset. These eigen queries are computed from a number classifiers designed with various subsets of positive and negative images. For visualization purpose, we use HOG features [25] to compute eigen queries. (For our experiments in Section 6, we use visual Bag of Words representation built over dense SIFT descriptor.) We also run these eigen queries on the dataset and find the top images in the ranked list which are shown in Fig 2. As we can see, using eigen queries we obtain new categories (sub-categories) which are not explicitly available in Caltech256. We get interesting new eigen queries which classify objects with specific poses, orientations, parts etc. Thus we get meaningful eigen queries which can be thought as building blocks for new categories. When the query is very similar to known eigen queries, the retrieval accuracy will be very high and the retrieval time will be low as we can represent the query precisely with very few eigen queries. But when the query is entirely new, the performance of the system will go down. This issue can be addressed by recomputing the set of eigen queries over the period by adding frequently used queries and removing non frequent queries.

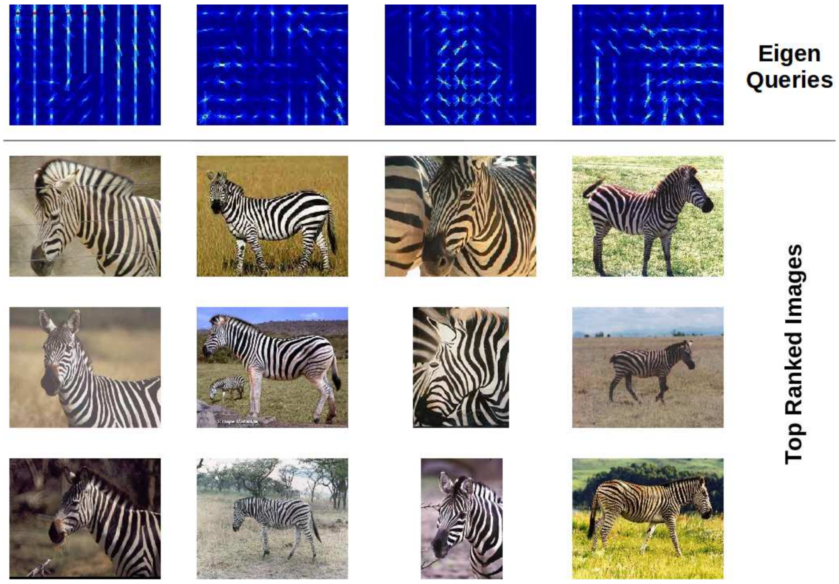

Fig. 2: Visualization of selected Eigen Queries learned on Caltech256 dataset. These eigen queries capture different meaningful concepts. First row represents the SVM weights with HOG features. The next three rows depicts top examples for each of these queries.

3.2 Retrieval with Eigen Queries

Before giving details of our approach, we first formalize the problem. As explained in Section 1, the problem is to retrieve images of a specific category from a large collection of image database. We represent the database with n images as $D = \{X_i | X_i$ is the feature representation of the i^{th} image and $i = 1 \rightarrow n\}$, where each $X_i \in \Re^d$. In order to retrieve images of a specific query category, we need to categorize database images into two categories based on whether it belongs to the query category or not. Since we need a binary categorization, we use

SVM in binary classification setting with linear kernel, which is efficient to learn and evaluate compared to nonlinear ones. Also with sophisticated features [26, 27] the accuracy of linear kernel is comparable with the accuracy of non linear kernels. Once the hyperplane W for the query q is given, the classification score for any sample X is computed as follows.

$$score(W, X) = W.X = \sum_{j=1}^{d} w_j.x_j \tag{1}$$

We rank all the elements in D based on their classification scores using above formula and retrieve the top ranked images as a result. In next two sections, we give details of our methods to solve the problem of efficiently retrieving the top-k results of a given query.

4 Eigen Search

The method for retrieval is divided into two phases which are outlined in Algorithm 1 and Algorithm 2 respectively. The offline phase of preprocessing consists of construction of eigen queries and respective eigen scores. The online phase of querying consists of encoding and retrieval of top-k results of a given query.

Algorithm 1 Preprocessing

```
Require: Image database $D = \{X_1, X_2, ..., X_n\}$ where $X_i \in \Re^d$; $i = 1 \rightarrow n$, positive
    and negative training examples of $m$ categories, number of eigen queries $p$ s.t.
    $p \leq m$
Ensure: Set of eigen queries $V$ and eigen scores $E$
Construct a set of classifiers $B = \{W_1, W_2, ..., W_m\}$ using linear SVM and given
    training examples of $m$ categories.
Compute the eigen values and eigen vectors of the covariance matrix of elements
    in $B$.
Generate a set of eigen queries $V = \{V_1, V_2, ..., V_p\}$ which consists of $p$ eigen vectors
    corresponding to top $p$ eigen values.
for $i = 1 \rightarrow n$ do
    for $j = 1 \rightarrow p$ do
        $E_{ij} = V_j.X_i$
    end for
end for
Return $V$ and $E$
```

4.1 Preprocessing

The construction of eigen classifiers is a one time process and happens offline as a preprocessing step. We start with a set of queries which may be available from the past experience or can be computed for many already known categories. Together the set of queries (SVM hyperplanes) of all the categories (taken from query logs) is represented by a set of classifiers $B = \{W_1, W_2, ..., W_m\}$, where m is the number of categories. To adequately capture the variations among different categories, we generate eigen queries from these m classifiers. We compute the $d \times d$ covariance matrix of the m vectors in B. Then, we compute the eigen values and eigen vectors of the obtained covariance matrix using eigen decomposition. The eigen queries are the p eigen vectors corresponding to top p eigen values and represented by $V = \{V_1, V_2, ..., V_p\}$. Once the eigen queries are generated, the scores for all the images in database D are computed as shown in Equation 1. All of these eigen scores are stored in a matrix E where E_{ij} denotes the eigen score of i^{th} image using j^{th} eigen query, and are used at the query time to efficiently retrieve top-k results as explained in the next section.

Algorithm 2 Querying - Eigen Search (ES)

```
Require: Image database $D = \{X_1, X_2, ..., X_n\}$ where $X_i \in \Re^d$; $i = 1 \rightarrow n$, set of
    eigen queries $V$ and eigen scores $E$, query classifier $Q$
Ensure: Result set $R$ for the query $Q$
Find coefficients ($\alpha$) such that $Q = \sum_{j=1}^{p} \alpha_j.V_j$
$S[1 \rightarrow n] = 0$
for $j = 1 \rightarrow p$ do
    if $\alpha_j \neq 0$ then
        for $i = 1 \rightarrow n$ do
            $S_i = S_i + \alpha_j.E_{ij}$
        end for
    end if
end for
$R = \{$ images correspond to top-k scores in $S$ $\}$
Return $R$
```

4.2 Querying with Eigen Search

In order to query for a category c, user gives a corresponding query classifier Q. Then, Q is represented by a linear combination of all the eigen queries $V_i \in V$ as shown below.

$$Q = \sum_{j=1}^{p} \alpha_j.V_j \tag{2}$$

The solution to this problem can be found by optimization of $||Q - \sum_{j=1}^{p} \alpha_j.V_j||$. We used standard technique of Least Squared Error (LSE) to solve this optimization problem. The reconstruction error Er is defined as follows,

$$Er = (Q - \sum_{j=1}^{p} \alpha_j.V_j)^T (Q - \sum_{j=1}^{p} \alpha_j.V_j) \tag{3}$$

Now, the classification score for every sample X_i with respect to the query classifier Q can be calculated using Equation 1 and Equation 2 as follows:

$$score(Q, X_i) = Q.X_i = \sum_{j=1}^{p} \alpha_j.V_j.X_i = \sum_{j=1}^{p} \alpha_j.E_{ij} \tag{4}$$

Instead of computing dot product of two large vectors, precomputed eigen scores are multiplied with the learned coefficients, and summed up across all the eigen queries. The complexity of naive score calculation is $O(nd)$ while our algorithm has the complexity of $O(np)$, where n is the number of database images, d is the dimensionality of feature vector and p is the number of eigen queries. This gives a speedup over naive score calculation because, usually dimensions of feature vectors of images are very high, while number of eigen queries are generally very few i.e. $p << d$. Once the classification scores for all the images in D are computed, they are sorted and images corresponding to top-k scores are returned as a result. Even though in many cases $p << d$, there exist some compact codes $[1, 28, 29]$ which represent an image with very few dimensions. In such scenarios, the given exhaustive score calculation will incur overhead of p. To overcome this overhead we also propose an approximate solution which calculates the scores for only fraction of the database by intelligently pruning the non essential images.

5 Subset Search

Given a query classifier Q, the goal is to retrieve the image with feature vector X_i such that $score(Q, X_i) = \sum_{j=1}^{p} \alpha_j.E_{ij}$ (Equation 4) is maximum across all the database images. Since, the classification score is a sum of products of coefficients and eigen scores, it will be maximum for such X_i which has high eigen scores corresponding to positive coefficients and low eigen scores corresponding to negative coefficients. Therefore, instead of calculating scores for entire database we can calculate the scores for only those feature vectors which satisfy the above mentioned criteria. The preprocessing step is similar to the one described in Section 4.1 except the additional step of sorting. In order to select feature vector efficiently, we sort them based on their eigen scores for every eigen classifier during preprocessing step, that is each column of the array E will be sorted along its rows. Next, we explain the querying mechanism of subset search algorithm.

5.1 Querying with Subset Search

At query time, the coefficients are learned as described in Section 4.2. For each coefficient α_i, we consider feature vectors whose corresponding eigen scores fall into top t or bottom t scores of sorted list E_i based on weather α_i is positive or negative respectively. Since, we are taking t feature vectors for every eigen query and there are p such eigen queries, total number of feature vectors to be considered for calculating score is tp. The classification scores for these tp feature vectors are computed and images having top-k highest scores are returned as a result. Note, that in case of previous method of eigen search, we are considering all n feature vectors for all p eigen queries which makes the computation cost of $O(np)$. While in this case we are considering only t feature vectors, which makes the computation cost of $O(tp)$. If we take t much smaller than n then we can get speedup of $O(n/t)$. Reducing t will increase the speedup but will also decrease the accuracy because we may miss some feature vectors with high classification scores. If we have $t = n$, then the method is same as the previous algorithm. Thus, t is the parameter which control the trade off between accuracy and speedup.

Algorithm 3 Querying - Subset Search (SS)

```
Require: Image database $D = \{X_1, X_2, ..., X_n\}$ where $X_i \in \Re^d$; $i = 1 \to n$, set of
    eigen queries $V$ and eigen scores $E$, query classifier $Q$, subset size $t$
Ensure: Result set $R$ for the query $Q$
Find coefficients $(\alpha)$ such that $Q = \sum_{j=1}^{p} \alpha_j.V_j$
Process $\alpha$ in non decreasing order of their absolute value i.e. $abs(\alpha_j) \geq abs(\alpha_{j+1})$
$C = \{\}$ // initialize candidate set
for $j = 1 \to p$ do
    $t_j = t.abs(\alpha_j/\alpha_{max})$
    if $\alpha_j > 0$ then
        $C = C \cup \{$ top $t_j$ feature indexes corresponding to $E[1 \to t_j, j]$ $\}$
    end if
    if $\alpha_j < 0$ then
        $C = C \cup \{$ bottom $t_j$ feature indexes corresponding to $E[n - t_j + 1 \to n, j]$ $\}$
    end if
end for
$n' = |C|$
$S[1 \to n'] = 0$ // compute scores for all the images in candidate set
for $j = 1 \to p$ do
    if $\alpha_j \neq 0$ then
        for $i = 1 \to n'$ do
            $S_i = S_i + \alpha_j.E_{C_i j}$
        end for
    end if
end for
$R = \{$ images correspond to top-k scores in $S$ $\}$
Return $R$
```

Furthermore, the computation of $O(tp)$ can be reduced by using the fact that the eigen queries with high absolute values of coefficients (α) have higher impact on the classification score. So instead of taking t feature vectors from the eigen score set E of every eigen query, we can take feature vectors in proportion to the corresponding coefficients. This can be achieved by processing coefficients (and in turn eigen queries) in the decreasing order of their absolute values. Let α_i be the i^{th} coefficient in that order, α_{max} is the coefficient with maximum absolute value, then number of feature vectors t_i for the i^{th} coefficient is given by, $t_i = t.abs(\frac{\alpha_i}{\alpha_{max}})$. Thus, weighted subset of feature vectors corresponding to every coefficient is considered and classification scores are computed using only those feature vectors. Finally, as mentioned above they are ranked based on the score and the images corresponding to top-k results are returned. The entire procedure is outlined in Algorithm 3. In experimental section, we show that by using this weighted subset technique with very small t we can achieve considerable speedup without losing much accuracy.

6 Experiments

In this section, we show the results of various experiments and discuss the performance of our methods on various data sets. We perform experiments on three different image datasets – PASCAL Visual Object Classes Challenge 2011 [30], Caltech256 [24] and Large Scale Visual Recognition Challenge 2010 (ImageNet) [31]. For PASCAL and Caltech256, the images are represented as a $8K$ dimensional histogram of Bag of Words (BoW) with dense pyramidal SIFT (PHOW) [12] features extracted using the VLFeat library [32]. In case of ImageNet, the features are $1K$ dimensional histogram of BOW with dense SIFT [33] features, available from [31].

6.1 Performance Evaluation

We analyze the performance of our methods – *Eigen Search* (ES) and *Subset Search* (SS) on PASCAL data set which contains 20 categories with 28952 images in total, with 50% for training and 50% for testing. In real world scenarios, query logs having rich history are used to generate eigen queries. So we simulate the query logs by constructing large number of queries from the dataset and use it for eigen queries. We create 10000 queries by randomly sampling 20 positive examples and 80 negative examples from the training set using a linear SVM [34]. From this set of queries (SVM weight vectors), we create the eigen queries as explained in Section 4.1. The test set in the PASCAL acts as the database on which we perform the retrieval.

We consider multiple novel test queries for every category present in the PASCAL. These classifiers (queries) are evaluated on the test set (database) exhaustively for creating the *ground truth*. Performance is evaluated on the top-k samples, with $k = 10$, that is we retrieve top 10 ranked images from the database corresponding to a given query. The retrieval quality is measured in terms of recall which is calculated as a percentage of images from ground truth set that are correctly retrieved in top-k results. The recall for the entire experiment

Table 1: Retrieval Quality on PASCAL.

Eigen Queries	Error Er[%]	Recall [%]	
		Eigen Search	Subset Search
100	1.3	89	79
300	0.7	97	90
500	0.6	100	91
1500	0.1	100	94

Table 2: Subset Search Performance.

Subset Size	Recall [%]	Speedup
10	31	906
200	91	85
400	99	52
500	100	45

is calculated by averaging the recall of all queries. During evaluation, every test query is expressed in terms of eigen queries using Equation 2 as explained in Section 4.2. Note that this step is independent of the dataset (and number of images in the database).

Retrieval Quality: We first show the effect of number of eigen queries required to obtain various levels of approximations. From the 10000 queries constructed using training data, we build an eigen queries of size $100 - 1500$. The Results of both the methods (ES and SS) are summarized in Table 1. It is clear from the table that as we increase the number of eigen queries, the recall in both the methods increases. This is because with more number of eigen queries, query classifier can be approximated more accurately, which is reflected in the error column. As explained in Section 3.1, each eigen query represents an abstract category which works as a building block for the new query. Thus, more number of eigen queries can better capture the variations among categories and give better results. The results also shows that, we obtain very low reconstruction error Er (Equation 3), and almost the same top-k images with as low as 300 eigen queries. Our method ES provides almost the same set of top-k in all cases, while SS yields an acceptable level of recall for most queries, with $t = 200$.

Subset Search: As explained in Section 5.1 parameter t controls the trade off between accuracy and speedup. To study the effect of t we measure the performance of SS by varying t as shown in Table 2. For this experiment, we use only 300 eigen queries. As shown in the table, recall increases with increase in t, which is natural because by increasing t we are considering more and more images from the dataset to calculate the score. Though our original representation of the images were with $8K$ dimensional histograms, we are able to obtain almost 100% recall by processing on an average only 2% images per eigen query, thus gaining the speedup of 52. This shows that our approximate algorithm SS achieves high accuracy by accessing only a fraction of the data.

6.2 Scalability

We validate our method on datasets which have larger number of categories and huge number of images. We consider Caltech256 and ImageNet datasets for this experiment. For ImageNet, we use the well accepted train-test split as in the case of PASCAL. For Caltech256, we use 40% for training (query construction and evaluation) and the rest for testing.

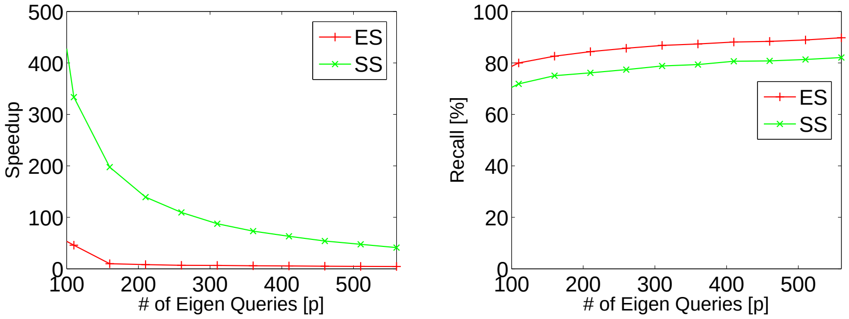

Fig. 3: Performance on Caltech256 dataset in terms of speedup and recall for different number of eigen queries.

From Fig 3 and Fig 4, we can see that the recall and speedup for these two datasets are also similar to PASCAL. Our method is not seriously getting affected by the scale of the dataset. For example, we can obtain a speedup of 333 for Caltech256 with a recall of 71% using *Subset Search*. In case of ImageNet we are getting speedup of 36 with the recall of 69%. The speedup in case of ImageNet is low compared to other tow datasets because the dimensionality of ImageNet is only $1K$ as compared to $8K$ of others which reduces the gap between naive scoring and our method.

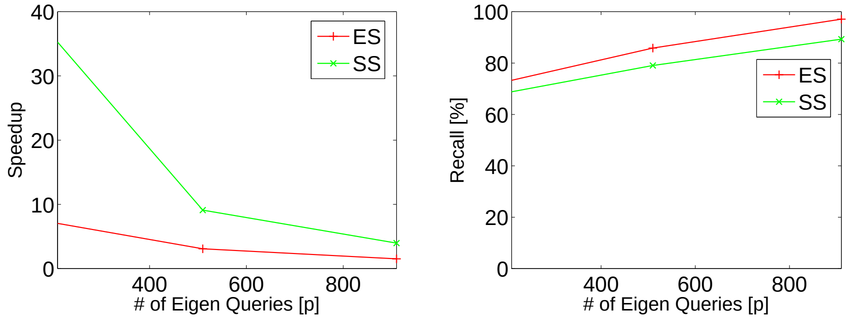

Fig. 4: Performance on ImageNet dataset in terms of speedup and recall for different number of eigen queries.

7 Conclusions

We showed the effective utilization of eigen queries for category based image retrieval system. We also gave an efficient pruning technique which retrieves results by accessing only fraction of the data. By evaluation on popular image datasets, we showed that the proposed methods achieve high speedup with comparable accuracy in terms of recall when compared to naive scoring. We also performed empirical analysis on various parameters like the number of eigen queries, data access, etc. and discussed their effects on retrieval system. We showed that, the proposed system is scalable by evaluating it on datasets consisting large number of images of varied categories. The proposed concept of *eigen queries* adequately captures the variations among different categories and hence, can be useful for other computer vision tasks as well.

References

1. Norouzi, M., Fleet, D.: Minimal loss hashing for compact binary codes. In: ICML. (2011) 353–360
2. Je andgou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. TPAMI **33** (2011) 117–128
3. Grauman, K.: Efficiently searching for similar images. Commun. ACM **53** (2010) 84–94
4. Kulis, B., Grauman, K.: Kernelized locality-sensitive hashing. TPAMI **34** (2012) 1092–1104
5. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR. (2007) 1–8
6. Arandjelović, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: CVPR. (2012)
7. Sivic, J., Zisserman, A.: Video google: a text retrieval approach to object matching in videos. In: ICCV. Volume 2. (2003) 1470–1477
8. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: CVPR. Volume 2. (2006) 2161–2168
9. Chechik, G., Sharma, V., Shalit, U., Bengio, S.: Large scale online learning of image similarity through ranking. JMLR **11** (2010) 1109–1135
10. Biederman, I.: Recognition-by-components: A theory of human image understanding. Psychological Review **94** (1987) 115–147
11. Frome, A., Singer, Y., Sha, F., Malik, J.: Learning globally-consistent local distance functions for shape-based image retrieval and classification. In: ICCV. (2007) 1–8
12. Bosch, A., Zisserman, A., Muoz, X.: Image classification using random forests and ferns. In: ICCV. (2007) 1–8
13. Turk, M., Pentland, A.: Eigenfaces for recognition. Journal of Cognitive Neuroscience **3** (1991) 71–86
14. Moghaddam, B., Pentland, A.: Probabilistic visual learning for object detection. In: ICCV. (1995) 786–793
15. Black, M.J., Jepson, A.D.: Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. IJCV **26** (1998) 63–84
16. Oliver, N., Rosario, B., Pentland, A.: A bayesian computer vision system for modeling human interactions. TPAMI **22** (2000) 831–843

17. Snoek, C.G.M., Worring, M.: Concept-based video retrieval. In: Foundations and Trends in Information Retrieval. (2009)
18. Riad, A.M., Elminir, H.K., Abd-Elghany, S.: Article: A literature review of image retrieval based on semantic concept. IJCA **40** (2012) 12–19
19. Natsev, A.P., Haubold, A., Tešić, J., Xie, L., Yan, R.: Semantic concept-based query expansion and re-ranking for multimedia retrieval. In: MULTIMEDIA, ACM (2007) 991–1000
20. Wang, H., Liu, S., Chia, L.T.: Does ontology help in image retrieval?: a comparison between keyword, text ontology and multi-modality ontology approaches. In: MULTIMEDIA, ACM (2006) 109–112
21. Wang, H., Jiang, X., Chia, L.T., Tan, A.H.: Ontology enhanced web image retrieval: aided by wikipedia & spreading activation theory. In: MIR, ACM (2008) 195–201
22. Rastegari, M., Fang, C., Torresani, L.: Scalable object-class retrieval with approximate and top-k ranking. In: ICCV. (2011) 2659–2666
23. Silverstein, C., Marais, H., Henzinger, M., Moricz, M.: Analysis of a very large web search engine query log. SIGIR Forum **33** (1999) 6–12
24. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology (2007)
25. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. Volume 1. (2005) 886–893 vol. 1
26. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. CVPR (2010) 3360–3367
27. Maji, S., Berg, A., Malik, J.: Classification using intersection kernel support vector machines is efficient. CVPR (2008) 1–8
28. Torralba, A., Fergus, R., Weiss, Y.: Small codes and large image databases for recognition. In: CVPR. (2008)
29. Sanchez, J., Perronnin, F.: High-dimensional signature compression for large-scale image classification. In: CVPR. (2011) 1665–1672
30. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV **88** (2010) 303–338
31. A. Berg, J.D., Fei-Fei, L.: Large scale visual recognition challenge. http://www.image-net.org/challenges/LSVRC/2010/ (2010)
32. Vedaldi, A., Fulkerson, B.: Vlfeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/ (2008)
33. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
34. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2** (2011) 27:1–27:27 Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.