Re-ranking by Multi-feature Fusion with Diffusion for Image Retrieval

Fan Yang[†], Bogdan Matei[§] and Larry S. Davis[†]
[†]University of Maryland College Park
[§]SRI International
{fyang,lsd}@umiacs.umd.edu, bogdan.matei@sri.com

Abstract

We present a re-ranking algorithm for image retrieval by fusing multi-feature information. We utilize pairwise similarity scores between images to exploit the underlying relationships among images. The initial ranked list for a query from each feature is represented as an undirected graph, where edge strength comes from feature-specific image similarity. Graphs from multiple features are combined by a mixture Markov model. In addition, we utilize a probabilistic model based on the statistics of similarity scores of similar and dissimilar image pairs to determine the weight for each graph. The weight for a feature is query-specific, where the ranked lists of different queries receive different weights. Our approach for calculating weights is data-driven and does not require any learning. A diffusion process is then applied to the fused graph to reduce noise and achieve better retrieval performance. Experiments demonstrate that our approach significantly improves performance over baseline methods and outperforms many state-of-the-art retrieval methods.

1. Introduction

Content-based image retrieval has been studied for decades due to its importance in web and image search. The bag-of-words (BOW) representation based on local features, such as the SIFT descriptor [19], is widely used in retrieval systems. Numerous improvements with respect to the performance and scalability of the original BOW representation [27] have been proposed [21, 32, 23, 7, 6]. To reduce the dimensionality of the standard BOW representation, which requires millions of visual words,

This work was supported by NSF EAGER grant: IIS1359900, Scalable Video Retrieval. Bogdan Matei was partly supported by AFRL with contract FA8750-12-C-0103. This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. Distribution statement "A" (Approved for Public Release, Distribution Unlimited).

Figure 1. An example of retrieved images by four features and our fusion method on Holidays dataset [15]. The left-most image is the query. Retrieved images are ranked higher if they have high similarity scores with the query. Images with red bounding boxes are correct matches.

Jégou *et al*. [17] introduced the Vector of Locally Aggregate Descriptors (VLAD) to achieve a trade-off between memory footprint and retrieval performance. Despite their power in capturing local patterns of an object, local features such as SIFT and VLAD descriptors may not be suitable for describing the global characteristics of an image, which are well captured by global features. Therefore, a retrieval system may benefit from fusing multiple complementary features that reflect different aspects of an image's characteristics. Recently, Zhang *et al*. [35] explicitly investigated query specific fusion by performing a link analysis on a graph constructed by merging multiple ordered ranked lists retrieved for a query. Deng *et al*. [8] proposed a weakly supervised multi-graph learning approach for fusion by using image attributes. Zhang *et al*. [36] presented a co-indexing approach that combines local invariant features with semantic attributes for constructing inverted files.

We will demonstrate that pairwise similarity scores between images using multiple features preserve rich information, leading to accurate retrieval results. We present an unsupervised, data-driven approach to fuse multiple features based on a graph representation. For each feature, given the query and initially retrieved images, we construct an undirected graph whose vertices represent these images and in which edge strength is the pairwise similarity score between

images. We employ a mixture Markov model to combine multiple graphs into one. In contrast to [35], where graphs are equally weighted, we introduce a probabilistic model to compute the importance of each feature under a naive Bayesian formulation, which depends only on the statistics of image similarity scores. Despite its simplicity, the proposed probabilistic model consistently improves retrieval performance after re-ranking. Instead of re-ranking the retrieved images directly from the fused graph, we employ an iterative diffusion algorithm, which propagates similarity scores throughout the graph to alleviate the effect of noise. This further improves the retrieval performance. In particular, we apply the locally constrained diffusion process (LCDP) [33] on the localized K-NN graph to obtain the refined similarity scores. Experiments on four datasets verify that our fusion algorithm consistently and significantly improves retrieval performance by individual features. In Figure 1, we present an example of retrieved images from the Holidays [15] dataset using individual features and our fusion. Similar images which were not ranked highly by any individual feature are ranked higher after fusion.

Our contributions are summarized as follows: First, we combine image similarity graphs computed from multiple features for a query in a principled way using the mixture Markov model, where initially retrieved images are interconnected and their relationships are exploited. Second, we design a probabilistic model to adaptively determine query-specific weights for features rather than heuristically summing graphs up. Third, we apply a diffusion process mostly used in shape retrieval to natural image retrieval to further improve the retrieval performance. Finally, due to its efficiency and simplicity, our fusion algorithm can be easily applied to most retrieval systems to refine initially retrieved results without using original features. Our algorithm has achieved the best reported results on Holidays [15] (**88.3**% mAP) and UKbench [21] (**3.86** N-S score) datasets. Note that we focus on the improvement of retrieval performance by re-ranking rather than designing a superior retrieval system based on single feature. To the best of our knowledge, we are the first to study diffusion on multiple graphs for retrieval and show state-of-the-art results. We address two new problems that were not addressed in [9]: switching between graphs and inferring fusion weights.

2. Related work

There is rich literature on image retrieval. With BOW representations, Sivic *et al*. [27] applied standard term frequency-inverse document frequency (tf-idf) method to image retrieval. A hierarchical clustering algorithm [21] was then proposed to construct a vocabulary tree which reduces the computational cost and is scalable to large scale datasets. Various improvements on [21] have also been proposed. Query expansion [1, 7, 6] refines the initial ranked list using retrieved images as additional queries. Spatial verification [23] is applied to original descriptors to re-rank retrieved images; Hamming embedding with weak geometry consistency [15] also improves performance. Nearest neighbor structure in the image space is also exploited [25]. Some works focus on solving the burstiness problem of descriptors by multiple match removal [16] or spatially reweighting nodes in the vocabulary tree [32]. Focusing on features, Jégou *et al*. [17] proposed the VLAD descriptor by aggregating local SIFT descriptors into compact features. Improvements on VLAD have been presented, including PCA and whitening [14], signed square root (SSR) on VLAD vectors [18] and Multi-VLAD descriptors [2]. RootSIFT [1] uses the Hellinger kernel on the original SIFT and shows promising improvements.

Due to their noise reduction qualities, diffusion processes have also been used for retrieval. Most apply diffusion processes to shape retrieval, such as locally constrained diffusion process (LDCP) [33], and to clustering such as authority shift clustering (ASC) [4]. A review of diffusion algorithms can be found in [9], where the LDCP algorithm was shown to have superior performance compared to other diffusion processes.

For image re-ranking, [13] proposed a click boosting method using the user click data to help re-rank initially retrieved images by textual and visual features, which may not be applicable when click data is missing. [29] proposed to mining frequent closed patterns as image representations, and designed a scoring function to re-rank images using mined patterns. [34] adopted a hypergraph-based sparse coding algorithm to predict clicks using multiple visual features. An initial ranked list is re-ranked based on predicted clicks of retrieved images. Multi-feature fusion is also widely used in image retrieval. Wang *et al*. [31] designed a graph-based learning algorithm for inferring weights of features, which requires a large number of queries beforehand to estimate relevance scores of initially retrieved images. Similarly, [3] utilized a Markov random field and manual relevance feedback to combine retrieval results by visual and textual features. A graph-based algorithm [35] used k-reciprocal nearest neighbors to fuse multiple ranked lists obtained by local and global features. However, only a graph metric (Jaccard similarity) is used and graphs are equally weighted. Deng *et al*. [8] proposed a weakly supervised multi-graph learning approach for fusion using image attributes that may not be always available. Zhang *et al*. [36] presented a co-indexing approach that combines local invariant features with semantic attributes learned from a large scale dataset for constructing inverted files. Our work differs from [35] in that we utilize the pairwise similarity scores and combine multiple graphs in a principled probabilistic framework. In contrast to [8, 36], our approach is fully unsupervised and does not require attributes.

3. Our approach

We first construct a graph to represent the relationships of initially retrieved images and the query using pairwise similarity scores between images in each feature. The query-specific weights to combine graphs are computed from the similarity score statistics, which is purely data-driven and does not require any learning. With the weight for each graph, we apply the mixture Markov model to fuse them. A diffusion process is applied to the fused graph to reduce noise and further improve retrieval performance.

3.1. Graph construction

Given a query image, an initial retrieval algorithm is performed to rank images from a dataset according to the similarity scores between the query image and dataset images. Suppose we have r features, each of which is a type of feature focusing on a specific aspect of an image. For each feature $\mathcal{M}_m$, the similarity between images $\mathcal{I}_i$ and $\mathcal{I}_j$, denoted as $s^m_{i,j}$, where $0 \leq s^m_{i,j} \leq 1$, is obtained by comparing two feature vectors. Generally, the initial rankings produced from different features will not agree; our hope is that by appropriately fusing them we will obtain an overall more accurate ranking.

From initial retrieval results of all r features, we obtain n unique images totally, including the initial query. The pairwise relationships with respect to feature $\mathcal{M}_m$ among these images is represented by a graph $G_m = (V_m, E_m, e_m)$ where vertices V_m are images connected by edges E_m with edge strength e_m. The e_m is the similarity between two images under feature $\mathcal{M}_m$. The original dataset may contain millions of images, resulting in a very long ranked list of retrieved images for each query and thus a huge graph. Therefore, based on an estimation or prior knowledge of the possible number of similar images in the database, we only choose the top L retrieved images for each feature to construct a tractable graph. The ranked list of top L retrieved images is referred as a short list. We denote the union of nodes from all graphs as V. For each graph G_m, we add vertices which are from V but not initially retrieved by feature $\mathcal{M}_m$ into the graph. Edges connecting a previously missing vertex and initially retrieved vertices in the graph are also added. In this way, we complete each graph with missing vertices, so that each graph has the same set of vertices V. Even if short lists from multiple features are disjoint, by completing graphs, we include pairwise relationships between vertices in these short lists and may still improve performance.

Each graph can be represented as a symmetric matrix $\mathbf{S}_m \in \mathbb{R}^{n \times n}$ with diagonal elements $s^m_{i,i} = 1$, known as an affinity matrix. Each element in the affinity matrix $\mathbf{S}_m$ represents the edge strength between nodes v_i and v_j in the graph. The i-th row in the affinity matrix $\mathbf{S}_m$ contains similarity scores between image $\mathcal{I}_i$ and all other images (including $\mathcal{I}_i$ itself). For r features, we have a set of r graphs $\mathcal{G} = \{G_1, G_2, ..., G_r\}$ corresponding to a set of affinity matrices $\mathcal{S} = \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_r\}$ of the same size. The similarity score $s^m_{i,j}$ between a query $\mathcal{I}_i$ and a dataset image $\mathcal{I}_j$ that was not retrieved by feature $\mathcal{M}_m$ is simply set to 0.

3.2. Multi-feature graph fusion

After obtaining affinity matrices in $\mathcal{S}$ from Sec. 3.1, our goal is to fuse graphs in $\mathcal{G}$ using these matrices. Affinity matrices should be complementary and not too sparse, so that our approach can better utilize and propagate relationships of dataset images to achieve large improvement. Due to different scaling of similarity scores from different features, it is difficult to directly determine weights for the affinity matrices. We instead adopt a probabilistic approach based on the mixture Markov model inspired by [37]. The model is essentially a random walk on multiple graphs. [12] adopts random walk cluster spatial data, but on a single graph rather than multiple graphs. Suppose a walker is at vertex $v_i \in V$ in graph G_m. In the next step, it has 1) a certain probability $p_m(v_i)$ of staying in the same graph G_m and then walks to another vertex v_j in this graph with transition probability $p_m(v_j|v_i)$, or 2) probability $p_{m'}(v_i)$ of switching to graph $G_{m'}$ and then walks from v_i to v_j in graph $G_{m'}$ with transition probability $p_{m'}(v_j|v_i)$. Intuitively, sitting at a vertex, the walker first decides which graph to land in, jumps to that graph (or stays in the same graph), and then decides which neighboring vertex to go to according to the graph's affinity matrix. Mathematically, this procedure of walking from v_i to v_j across all graphs can be represented as

$$p(v_j|v_i) = \sum\nolimits_m p_m(v_j|v_i)p_m(v_i), \tag{1}$$

where $p(v_j|v_i)$ is the transition probability of walking from v_i to v_j in the fused graph. $p_m(v_i)$ is the probability of switching to (or staying in) graph G_m when the walker is at vertex v_i. It is the probability of switching between graphs.

Our next task is to compute the transition probability $p(v_j|v_i)$. Intuitively, $p(v_j|v_i)$ should be related to the edges between v_i and its neighbors. We resort to "degree of a vertex" and "volume of a graph" to explain our approach of computing $p(v_j|v_i)$. The degree of v_i in G_m is the sum of edge strength of all vertices connected to v_i, $d_m(v_i) = \sum_j e_m(v_i, v_j)$. The volume of graph G_m is the sum of all edge strength in the graph, $vol_m V = \sum_{v_i, v_j \in V} e_m(v_i, v_j) = \sum_{v_i \in V} d_m(v_i)$. The transition probability is then written as

$$p_m(v_j|v_i) = e_m(v_i, v_j)/d_m(v_i). \tag{2}$$

After a number of steps, the random walk model will reach a stationary state where the stationary probability at vertex v_i is defined as

$$\pi_m(v_i) = d_m(v_i)/vol_m V. \tag{3}$$

Suppose the stationary probability of the fused graph is a linear combination of stationary probabilities of all graphs, $\pi(v_i) = \sum_m w_m(v_i)\pi_m(v_i)$, where $w_m(v_i)$ is the weight for vertex $v_i \in V$ in graph G_m, $w_m(v_i) \leq 1$ and $\sum_m w_m(v_i) = 1$. For a node in a graph, higher stationary probability implies higher probability of switching to (or staying in) this graph, so that $p_m(v_i) \propto \pi_m(v_i)$. Without other prior knowledge, we can estimate the probability $p_m(v_i)$ by linearly weighting the ratio of the stationary probability of an individual graph to that of the fused graph as

$$p_m(v_i) = w_m(v_i)\frac{\pi_m(v_i)}{\pi(v_i)}. \tag{4}$$

Plugging (2), (3) and (4) into (1), we obtain

$$p(v_j|v_i) = \frac{1}{\pi(v_i)}\sum\nolimits_m w_m(v_i)\frac{e_m(v_i,v_j)}{vol_m V}. \tag{5}$$

We introduce the edge strength between vertices v_i and v_j in the fused graph as

$$e(v_i,v_j) = \sum\nolimits_m w_m(v_i)\frac{e_m(v_i,v_j)}{vol_m V} \tag{6}$$

and obtain $p(v_j|v_i) = e(v_i,v_j)/\pi(v_i)$. The volume of the fused graph is 1. The affinity matrix of the fused graph is not symmetric due to the use of transition probability (the transition probabilities from v_i to v_j and v_j to v_i may not be the same). So $e(v_i,v_j)$ can be regarded as the weight of a directed edge. The mixture Markov model on the undirected graphs reduces to a convex combination of normalized affinity matrices. Therefore, we normalize all affinity matrices $\mathbf{S}_m$ to $\mathbf{T}_m$ by $\mathbf{T}_m = \mathbf{S}_m/vol_m V$, and discuss how to determine the weight $w_m(v_i)$ for each v_i in graph G_m in the next section.

3.3. Feature weight calculation

To obtain the weight $w_m(v_i)$, we describe a probabilistic model to determine the query-specific weights which measure the importance of a feature for a particular query. Our model is based only on the statistics of data and does not require any learning.

For a query image $\mathcal{I}_i$, we let $\mathcal{P}$ be the set of images similar to $\mathcal{I}_i$, and let $\mathcal{Q}$ be the set of images which are dissimilar from $\mathcal{I}_i$. Given a similarity score $s^m_{i,j}$ of feature $\mathcal{M}_m$ (graph G_m), the likelihood of a retrieved image $\mathcal{I}_j$ belonging to $\mathcal{P}$ or $\mathcal{Q}$ is denoted as $p(\mathcal{I}_j \in \mathcal{P}|s_{i,j})$ and $p(\mathcal{I}_j \in \mathcal{Q}|s_{i,j})$. By Bayes' theorem, we have $p(\mathcal{I}_j \in \mathcal{P}|s^m_{i,j}) = p(s^m_{i,j}|\mathcal{I}_j \in \mathcal{P})p(\mathcal{I}_j \in \mathcal{P})/p(s^m_{i,j})$ and $p(\mathcal{I}_j \in \mathcal{Q}|s^m_{i,j}) = p(s^m_{i,j}|\mathcal{I}_j \in \mathcal{Q})p(\mathcal{I}_j \in \mathcal{Q})/p(s^m_{i,j})$. We define $\rho_m(i,j)$ as the ratio of $p(\mathcal{I}_j \in \mathcal{P}|s^m_{i,j})$ to $p(\mathcal{I}_j \in \mathcal{Q}|s^m_{i,j})$

$$\rho_m(i,j) = \frac{p(\mathcal{I}_j \in \mathcal{P}|s^m_{i,j})}{p(\mathcal{I}_j \in \mathcal{Q}|s^m_{i,j})} = \frac{p(s^m_{i,j}|\mathcal{I}_j \in \mathcal{P})p(\mathcal{I}_j \in \mathcal{P})}{p(s^m_{i,j}|\mathcal{I}_j \in \mathcal{Q})p(\mathcal{I}_j \in \mathcal{Q})} \tag{7}$$

where $p(\mathcal{I}_j \in \mathcal{P})$ and $p(\mathcal{I}_j \in \mathcal{Q})$ represent the marginal probability of image $\mathcal{I}_j$ being a similar image or a dissimilar image given a query image. The marginal probabilities can be obtained by prior knowledge or an estimation of the portion of similar images that should be returned given a specific query. For examples, if we know there are 10% similar images given a query, we set $p(\mathcal{I}_j \in \mathcal{P}) = 0.1$ and $p(\mathcal{I}_j \in \mathcal{Q}) = 0.9$.

To obtain $p(s^m_{i,j}|\mathcal{I}_j \in \mathcal{P})$ and $p(s^m_{i,j}|\mathcal{I}_j \in \mathcal{Q})$, we make the assumption that the similarity scores between two similar images and those between two dissimilar images come from different distributions. To proceed, we manually annotate a set of pairs of similar images from the dataset offline to obtain the similarity scores of similar images. Additionally, we compute similarity scores between dataset images and images from an unrelated dataset (selected from the Caltech-101 dataset [11]) to obtain the similarity scores between dissimilar images. We approximate the distributions of the two sets of similarity scores as Gaussian distributions, $\mathcal{N}_\mathcal{P} \sim (\mu_\mathcal{P}, \sigma^2_\mathcal{P})$ and $\mathcal{N}_\mathcal{Q} \sim (\mu_\mathcal{Q}, \sigma^2_\mathcal{Q})$. Note that we use a Gaussian assumption for simplicity and efficiency, and will show that it works well in our experiments. Other data fitting algorithms can be applied to better capture the underlying distributions at the cost of efficiency. In this way, (7) can be rewritten as

$$\rho_m(i,j) = \gamma\frac{p(s^m_{i,j}|\mathcal{N}_\mathcal{P})}{p(s^m_{i,j}|\mathcal{N}_\mathcal{Q})} = \gamma\frac{\sigma_\mathcal{Q}\mathcal{K}_\mathcal{P}(s^m_{i,j})}{\sigma_\mathcal{P}\mathcal{K}_\mathcal{Q}(s^m_{i,j})}, \tag{8}$$

where $\gamma = \frac{p(\mathcal{I}_j \in \mathcal{P})}{p(\mathcal{I}_j \in \mathcal{Q})}$, $\mathcal{K}_\mathcal{P}(s^m_{i,j}) = exp(-(s^m_{i,j} - \mu_\mathcal{P})^2/\sigma^2_\mathcal{P})$ and $\mathcal{K}_\mathcal{Q}(s^m_{i,j}) = exp(-(s^m_{i,j} - \mu_\mathcal{Q})^2/\sigma^2_\mathcal{Q})$. In practice, we do not compute $\rho_m(i,j)$ for every retrieved image $\mathcal{I}_j$. Instead, for a query image $\mathcal{I}_i$, we compute the mean of the K largest similarity scores as $\bar{s}^m_i$, which indicates how reliable this ranked list is regarding the query image $\mathcal{I}_i$. By substituting $s^m_{i,j}$ with $\bar{s}^m_i$ in (8), we have a query-specific confidence score $\rho_m(i)$ by (8), which is denoted as $\rho_m(v_i)$ with the graph representation. The query-specific weight of a query v_i in graph G_m is computed by $w_m(v_i) = \rho_m(v_i)/\sum \rho_m(v_i)$. In our work, the query-specific weight is only assigned to the query node in a graph. However, it is also applicable to non-query nodes, although there is no need to adjust fusion weights for non-query nodes as they are excluded during evaluation. For a non-query image v_j in graph G_m, we simply use equal weight $w_m(v_j) = 1/r$ for r features. Therefore, we obtain a weight vector

$$\mathbf{w}_m = (w_m(v_1), w_m(v_2), ..., w_m(v_n))^\top \tag{9}$$

computed from all vertices for each graph G_m. The normalized affinity matrix of the fused graph $\mathbf{T}$ is subsequently calculated as

$$\mathbf{T} = \sum\nolimits_m \mathbf{diag}(\mathbf{w}_m) \cdot \mathbf{T}_m \tag{10}$$

where the i-th diagonal element in $\mathbf{diag}(\mathbf{w}_m) \in \mathbb{R}^{n\times n}$ corresponds to $w_m(v_i)$. This process is equivalent to assigning different weights for a row from different features when combining affinity matrices. Our approach does not assign a single weight for each feature, thereby capturing more query-dependent information from the similarity scores.

Algorithm 1 Multi-feature Re-ranking with Diffusion

Input: r affinity matrices $\mathcal{S} = \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_r\}$ representing r graphs $\mathcal{G}$, the query image $\mathcal{I}_i$
Output: Re-ranked results for $\mathcal{I}_i$
for $m = 1$ **to** r
  Normalize $\mathbf{S}_m$ to $\mathbf{T}_m$ (Sec. 3.2);
  Compute the mean of the K largest similarity scores from $\mathbf{T}_m$ for $\mathcal{I}_i$ as $\bar{s}_i^m$;
  Compute query-specific confidence $\rho_m(v_i)$ by (8);
  Compute the weight vector $\mathbf{w}_m$ in (9), where $w_m(v_i) = \rho_m(v_i)/\sum \rho_m(v_i)$ for the query node and $w_m(v_i) = 1/r$ for non-query nodes.
end for
Obtain the affinity matrix $\mathbf{T}$ of the fused graph by (10);
Apply diffusion process to $\mathbf{T}$;
Infer new ranks from $\mathbf{T}$ for the query $\mathcal{I}_i$ by sorting similarity scores of the row associated with query node.

3.4. Diffusion process

From the new affinity matrix $\mathbf{T}$ obtained in (10), we can directly infer a new ranking. Nevertheless, the results can be improved by applying a diffusion process to $\mathbf{T}$ to reduce noise. The basic idea is to propagate the similarity score of a vertex to its neighboring vertices until a stationary state is reached. Here we employ an iterative diffusion process for efficiency. Given $\mathbf{T}$, the transition matrix is defined as $\mathbf{P} = \mathbf{D}^{-1}\mathbf{T}$, where $\mathbf{D}$ is a diagonal matrix whose i-th diagonal element $d(i,i) = d(v_i)$, where $d(v_i)$ is the degree of vertex v_i in the fused graph. We build a matrix $\mathbf{W}^t = (\mathbf{f}_1^t \quad \mathbf{f}_2^t \quad \cdots \quad \mathbf{f}_n^t)^\top$, where $\mathbf{f}_i^t$ is a column vector indicating the probability of being at a vertex starting from vertex v_i after t steps. We employ the LCDP algorithm [33], which iteratively updates $\mathbf{W}^t$ by $\mathbf{W}^{t+1} = \mathbf{P}_K\mathbf{W}^t\mathbf{P}_K^\top$, where $\mathbf{P}_K$ is the transition matrix for the K-NN graph G_K built by only keeping similarity scores of each node and its K nearest neighbors. The edge strength $e(v_i, v_j) = 0$ if vertex v_j does not belong to the K-NNs of v_i, and $\mathbf{W}^0 = \mathbf{P}_K$. Details can be found in [33]. The diffusion terminates after a pre-defined number of iterations or if $\mathbf{W}$ does not change. The diffused matrix is used to re-rank retrieved images to obtain the final results by sorting diffused similarity values of the row associated with the query node. The entire procedure of our fusion approach is presented in Algorithm 1.

4. Experiments

4.1. Implementation details

Datasets. We test our approach on the Holidays [15], UKbench [21], Oxford5k [23] and Paris6k [24] datasets. The Holidays dataset consists of 1491 image from various scenes, labeled as 500 categories. The first image in each category is used as query. For each query, the remaining 1490 images are dataset images. The UKbench dataset contains 10200 images from 2550 objects or scenes with 4 images for each object or scene, taken under different viewpoints and lighting conditions. The Oxford5k dataset consists of 5062 photos of famous Oxford landmarks. Groundtruth is provided for 11 different landmarks, each of which has 5 queries, resulting in 55 queries. Similarly, the Paris6k dataset contains 6412 photos of buildings in Paris, of which 55 photos serve as queries. The query region of Oxford5k and Paris6k is only part of an image, which is different from Holidays and UKbench, where the entire image is used as a query. Moreover, the viewpoint significantly changes across images, while images in Holidays and UKbench are mostly near-duplicate.

Features. We use 2 local features and 2 global features. For local features, we use Hessian affine feature point extractor and the 128-dimension SIFT descriptor [25] to compute BOW features. We use the visual words provided by [25] except on Holidays dataset where we train a 1M vocabulary by approximate k-means (AKM) [23]. Single assignment and tf-idf weighting are applied to construct BOW vectors. We adopt the 8192-dimension VLAD descriptor with signed square root (SSR), computed with 64 clusters provided by [18], For global features, we use GIST [22] and HSV color descriptors. The GIST descriptor is 1192-dimension while the color descriptor is 4000-dimension with 40 bins for H and 10 bins for S and V components.

Parameter settings. We compute cosine similarity between two BOW vectors. For other features, we compute the Euclidean distance x_d between two feature vectors and convert it to a similarity score by $exp(-x_d/\sigma)$. Our algorithm is not sensitive to σ, as we will show in the experiments. For simplicity, we set $\sigma_\mathcal{P} = \sigma_\mathcal{Q} = 1$ and fix them throughout all experiments. The parameter K, denoting the number of neighboring vertices in the K-NN graph and the number of top largest similarity scores of a query, is set to 6 for Holidays and UKbench, and 40 for Oxford5k and Paris6k. The length of the short list of retrieved images L is set to 700 for Holidays and UKbench, and 5000 for Oxford5k and Paris6k. Similarity scores between dataset images are computed offline, while the scores between queries and dataset images are computed online during retrieval. Graphs $\mathcal{G}$ are constructed during re-ranking using computed similarity scores.

Evaluation protocol. We use N-S score [21] on UKbench, which measures the recall of the top 4 retrieved images, and mean average precision (mAP) on other 3 datasets.

Table 1. Comparisons with state-of-the-art approaches. We use N-S score on UKbench, and mAP (in %) on other datasets. "-" means the results are not reported. SV, MA, QE and WGC stand for spatial verification [23], multiple assignment [24], query expansion [6, 7, 1] and weakly geometric consistency [15].

	Baselines				Fusion			Baselines + SV/MA/QE/WGC							
Datasets	BOW [25]	VLAD [18]	GIST [10]	Color	**Ours**	[35]	[36]	[23]	[15]	[16]	[25]	[6]	[20]	[26]	[28]
Holidays	77.2	55.9	35.0	55.8	**88.3**	84.6	80.9	-	75.1	84.8	-	-	75.8	82.1	88.0
UKbench	3.50	3.22	1.96	3.09	**3.86**	3.77	3.60	3.45	-	3.64	3.67	-	-	-	-
Oxford5k	67.4	32.6	24.2	8.5	76.2	-	68.7	66.4	54.7	68.5	81.4	82.7	**84.9**	78.0	83.8
Paris6k	69.3	38.0	19.2	8.4	**83.3**	-	-	-	-	-	80.3	80.5	82.4	73.6	80.5

4.2. Retrieval performance

We compare our method with a few state-of-the-art approaches. The quantitative comparison is shown in Table 1. The baselines using individual features in our work are initial retrieval results from pairwise similarities without any other techniques, *i.e.*, spatial verification (SV), query expansion (QE), multiple assignment (MA) and weak geometric consistency (WGC), etc. However, most other approaches using a single feature rely on various additional improvements. In particular, we compare with [35] and [36] which also exploit multiple features to improve retrieval performance. We will show that our fusion algorithm greatly improves baselines' performance and outperforms state-of-the-art approaches even we only uses similarity scores. Note that we are not designing superior baselines, which is outside the scope of this work.

As shown in Table 1, the BOW representation achieves the best retrieval performance among all baselines across different datasets, while GIST and color features are not discriminative enough. Nevertheless, our multi-feature fusion algorithm significantly improves the final retrieval performance on all datasets and outperforms state-of-the-art algorithms. On Holidays and UKbench datasets, we obtain **88.3**% mAP and **3.86** N-S score respectively, which are the best reported results to our knowledge. Compared to the best baseline (BOW), our fusion improves the results by 14.4% on Holidays and 10.3% on UKbench with a simple probabilistic model. In contrast, the relative improvements by [35] that is also based on graph fusion are 9.2% (77.5% to 84.6%) on Holidays and 6.5% (3.54 to 3.77) on UKbench, while they are 9.6% (73.8% to 80.9%) and 5.4% (3.42 to 3.6) by [36]. Compared to other single feature based methods with sophisticated processing steps, our fusion depends only on similarity scores to calculate query-specific weights and perform diffusion process, and exploits more reliable information about the relationships among images, thus producing better retrieval results.

On Oxford5k and Paris6k datasets, the color feature only achieves 8.5% and 8.4% mAP due to large viewpoint changes, cluttered background and a constrained region of interest (ROI) for query. Additionally, the performance of GIST and VLAD features also drops. Different from [35], we do not specifically remove an inferior feature, but include all features in the fusion without any bias, even though the color feature performs much worse than others. It is clear that our fusion still greatly improves final retrieval performance. Our experiments clearly shows that our fusion is very robust and is not deteriorated by a single inferior feature (color). It improves the best baseline (BOW) by 13.1% and achieves 76.2% mAP on Oxford5k, which outperforms [36] and is comparable to other state-of-the-art approaches. On Paris6k, our fusion brings the mAP from 69.3% by the best baseline (BOW) up to 83.3% without spatial verification, query expansion and other techniques, which is a 20.1% relative improvement. The performance gain is larger than that on the near-duplicate datasets where individual features have already achieved good performance due to less variance, making the potential of fusion limited. In contrast, on Oxford5k and Paris6k, a single feature is often not powerful enough to distinguish different images and multiple features better complement each other. We believe that larger performance gains could be achieved if our fusion method were applied to better baselines with better image similarity metrics. Sample retrieval results are in the supplementary material. With respect to computational complexity, our fusion only takes around 1s on a 3.4GHz CPU to re-rank all retrieved images for a single query.

4.3. Discussion

Contributions of individual components. To evaluate the importance of individual components of the proposed method, we conduct additional experiments by adding or removing a component and measuring how accuracy changes. The configurations are detailed as follows. With the original affinity matrices from multiple features, the accuracy can be measured by selecting the maximal mAP among all baselines, denoted as B. The approaches by fusion with equal weights and query-specific weights are denoted as EW and QW, respectively, where results are directly inferred from the combined affinity matrix without diffusion. Both the EW and QW approaches use all dataset images. Two variants using a short list are denoted as SL+QW and SL+EW. Our entire framework is denoted as SL+QW+DP, while the variant using EW and SL for diffusion is denoted as SL+EW+DP. The comparisons on the test datasets are shown in Table 2. We can see that both QW and DP contribute to the improvements while using a proper SL also

Table 2. Retrieval performance by different variants of the proposed method. N-S score on UKbench, and mAP (in %) on other datasets.

	B	EW	QW	SL+EW			SL+QW			SL+EW+DP			SL+QW+DP		
SL length L	-	-	-	700	1500	5000	700	1500	5000	700	1500	5000	700	1500	5000
Holidays	77.2	81.1	84.0	82.1	-	-	83.6	-	-	86.4	-	-	**88.3**	-	-
Ukbench	3.50	3.72	3.76	3.76	3.76	3.75	3.77	3.77	3.77	3.84	3.84	3.84	**3.86**	3.86	3.85
Oxford5k	67.4	69.2	70.3	63.7	64.3	69.1	65.6	65.3	70.3	73.2	73.8	74.0	75.2	75.7	**76.2**
Paris6k	69.3	68.1	71.2	65.7	66.0	67.6	67.5	68.9	69.6	80.1	80.8	81.4	82.0	82.6	**83.3**

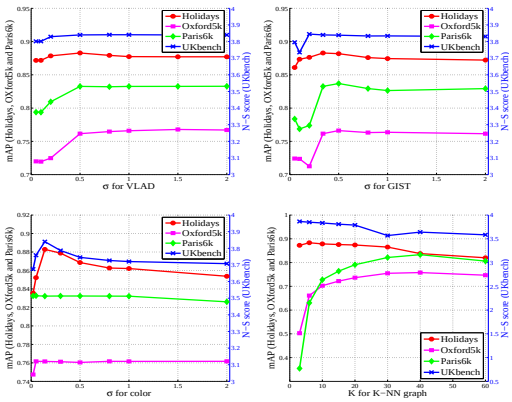

Figure 2. Performance under different σ for VLAD, GIST and color, and K for K-NN graph used in the diffusion process.

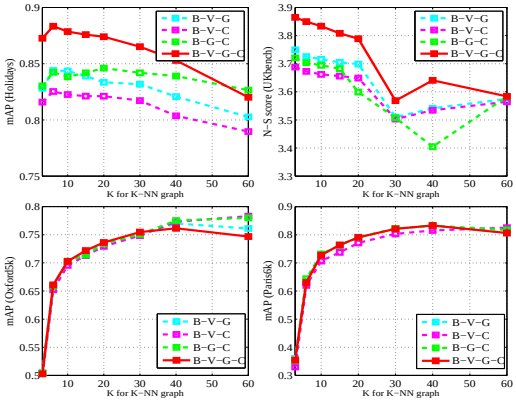

Figure 3. Performance of different feature combinations with respect to varying K. B, V, G and C stand for BOW, VLAD, GIST and color features.

increases accuracy. Specifically, in most cases, results by QW are better than those by EW, showing the effectiveness of our probabilistic model derived from statistics of similarity scores. Additionally, if there are a large number of relevant images to be retrieved for a query (Oxford5k and Paris6K), we need to include more images in the short list to obtain good results; otherwise the performance drops below the best baseline because many similar images are excluded from the fused graph. In contrast, a small short list is sufficient when there are only a few similar images to be retrieved. Therefore, we can control the length of short list to achieve a trade-off between computational complexity and accuracy.

Parameter evaluation. The proposed method has several parameters to set: the length of the short list L, the number of nearest neighbors K in K-NN graph and σ for converting the Euclidean distance to similarity score for VLAD, GIST and color features. To evaluate the sensitivity of our method to these parameters, we conduct experiments by changing one parameter at a time. The retrieval results regarding different L are shown in Table 2. Performance by changing other parameters are shown in Figure 2.

Our method is robust and not sensitive to these parameters as long as they are in a reasonable range. In particular, performance does not change much even when σ is 4 times of its optimal value, meaning that we can safely fix a larger σ for all datasets without sacrificing accuracy too much. In all experiments, σ is empirically set to 0.5, 0.34 and 0.14 for VLAD, GIST and color features. Additionally, on Oxford5k and Paris6k datasets which consist of a large number of similar images for each query, we need a large K to include them in the graph and highly rank them after re-ranking. In contrast, on Holidays and UKbench datasets which only contain a small number of similar images, a small K is sufficient to include most of them in the graph; otherwise similarity scores of those similar images will be contaminated by irrelevant images if K is too large.

Combinations of features. We conduct experiments using different feature combinations to further verify the effectiveness of our fusion algorithm. In Figure 3, we show the performance using 4 combinations of features which fuse 3 or 4 types of features. Since VLAD+GIST+color performs much worse than other combinations, we do not display its results in the figure for better visualization [1]. In most cases, fusing features from all 4 features achieves the best results, especially on Holidays and UKbench datasets, which verifies that our fusion algorithm is very robust and is not easily affected by an inferior feature (color in this case). Moveover, our fusion successfully exploits complementary information from multiple features, thereby greatly improving the performance compared to combinations of 3 features. Only when K becomes very large, the performance by fusing all 4 features is slightly worse than that by other combinations due to large amount of noise from multiple features. Note that our fusion does not set any restrictions

[1] The best results by VLAD+GIST+color are 52.4%, 2.91, 30.5% and 40.3% on Holidays, UKbench, Oxford5k and Paris6k datasets.

on the number or type of features to be fused.

Large scale experiments. To evaluate the scalability of our approach, we conduct additional experiments on a web image dataset NUS-WIDE [5] which contains 269,648 images associated with 81 concept tags, where images may have multiple labels. We use all six types of low level features provided by [5] [2] and follow the approach in [30] by randomly selecting 1K images as queries and using the remaining images as dataset images. Retrieved images sharing at least one semantic label with the query are regarded as correct matches. We set $L = 1000$ for each ranked list rather than using all images to construct the graph, and evaluate the performance by mAP (Table 3). Our approach still improves the performance and only takes around 5 seconds for re-ranking retrieved images for a query due to the use of short list. Details of experimental setting and more discussions are in the supplementary material.

Table 3. Performance in terms of mAP (%) evaluated on top 500 and top 1000 retrieved/reranked results on NUS-WIDE dataset.

	CH	CC	EDH	WT	CM	BoW	S+Q	S+Q+D
@500	1.85	2.00	1.97	2.20	2.02	1.98	2.18	**2.35**
@1000	2.43	2.66	2.60	2.25	2.84	2.70	2.89	**3.22**

5. Conclusions

We introduced a re-ranking algorithm by multi-feature fusion with diffusion for image retrieval. We exploit the pairwise similarity scores between images to infer their relationships. Initial ranks from one feature are represented as an undirected graph where edge strength is similarity score. Graphs are combined by a mixture Markov model where the query-specific weight is calculated by a probabilistic model utilizing the statistics of similarity scores. Diffusion is then applied to the fused graph to reduce noise. Our approach significantly and consistently improves the performance of baselines and is very robust to variations in its parameters.

References

[1] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, pages 2911–2918, 2012. 2, 6

[2] R. Arandjelović and A. Zisserman. All about VLAD. In *CVPR*, 2013. 2

[3] R. O. Chávez, H. J. Escalante, L. E. Sucar, et al. Multimodal markov random field for image reranking based on relevance feedback. *ISRN Machine Vision*, 2013, 2013. 2

[4] M. Cho and K. M. Lee. Authority-shift clustering: Hierarchical clustering by authority seeking on graphs. In *CVPR*, pages 3193–3200, 2010. 2

[5] T. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng. NUS-WIDE: a real-world web image database from national university of singapore. In *CIVR*, 2009. 8

[6] O. Chum, A. Mikulík, M. Perdoch, and J. Matas. Total recall II: Query expansion revisited. In *CVPR*, pages 889–896, 2011. 1, 2, 6

[7] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *ICCV*, pages 1–8, 2007. 1, 2, 6

[8] C. Deng, R. Ji, W. Liu, D. Tao, and X. Gao. Visual reranking through weakly supervised multi-graph learning. In *ICCV*, pages 2600–2607, 2013. 1, 2

[9] M. Donoser and H. Bischof. Diffusion processes for retrieval revisited. In *CVPR*, 2013. 2

[10] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *CIVR*, 2009. 6

[11] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007. 4

[12] D. Harel and Y. Koren. Clustering spatial data using random walks. In *KDD*, pages 281–286, 2001. 3

[13] V. Jain and M. Varma. Learning to re-rank: query-dependent image re-ranking using click data. In *WWW*, pages 277–286, 2011. 2

[14] H. Jégou and O. Chum. Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *ECCV*, pages 774–787, 2012. 2

[15] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, pages 304–317, 2008. 1, 2, 5, 6

[16] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, pages 1169–1176, 2009. 2, 6

[17] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, pages 3304–3311, 2010. 1, 2

[18] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(9):1704–1716, 2012. 2, 5, 6

[19] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999. 1

[20] A. Mikulík, M. Perdoch, O. Chum, and J. Matas. Learning a fine vocabulary. In *ECCV (3)*, pages 1–14, 2010. 6

[21] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, 2006. 1, 2, 5

[22] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001. 5

[23] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 1, 2, 5, 6

[24] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008. 5, 6

[25] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. J. V. Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, pages 777–784, 2011. 2, 5, 6

[26] D. Qin, C. Wengert, and L. V. Gool. Query adaptive similarity for large scale object retrieval. In *CVPR*, 2013. 6

[27] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, 2003. 1, 2

[28] G. Tolias, Y. S. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, pages 1401–1408, 2013. 6

[29] W. Voravuthikunchai, B. Crémilleux, and F. Jurie. Image re-ranking based on statistics of frequent patterns. In *ICMR*, page 129, 2014. 2

[30] J. Wang, S. Kumar, and S. Chang. Semi-supervised hashing for large-scale search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(12):2393–2406, 2012. 8

[31] M. Wang, H. Li, D. Tao, K. Lu, and X. Wu. Multimodal graph-based reranking for web image search. *IEEE Transactions on Image Processing*, 21(11):4649–4661, 2012. 2

[32] X. Wang, M. Yang, T. Cour, S. Zhu, K. Yu, and T. X. Han. Contextual weighting for vocabulary tree based image retrieval. In *ICCV*, pages 209–216, 2011. 1, 2

[33] X. Yang, S. Köknar-Tezel, and L. J. Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. In *CVPR*, pages 357–364, 2009. 2, 5

[34] J. Yu, Y. Rui, and D. Tao. Click prediction for web image reranking using multimodal sparse coding. *IEEE Transactions on Image Processing*, 23(5):2019–2032, 2014. 2

[35] S. Zhang, M. Yang, T. Cour, K. Yu, and D. N. Metaxas. Query specific fusion for image retrieval. In *ECCV*, pages 660–673, 2012. 1, 2, 6

[36] S. Zhang, M. Yang, X. Wang, Y. Lin, and Q. Tian. Semantic-aware co-indexing for image retrieval. In *ICCV*, pages 1673–1680, 2013. 1, 2, 6

[37] D. Zhou and C. J. C. Burges. Spectral clustering and transductive learning with multiple views. In *ICML*, pages 1159–1166, 2007. 3

[2] They are color histogram (CH), color correlogram (CC), edge direction histogram (EDH), wavelet texture (WT), color moments (CM) and bag of words (BoW).