Ensemble Diffusion for Retrieval

Song Bai[1]*, Zhichao Zhou[1]*, Jingdong Wang[2], Xiang Bai[1]†, Longin Jan Latecki[3], Qi Tian[4]
[1]Huazhong University of Science and Technology, [2]Microsoft Research Asia
[3]Temple University, [4]University of Texas at San Antonio
{songbai,zzc,xbai}@hust.edu.cn,jingdw@microsoft.com,latecki@temple.edu,qi.tian@utsa.edu

Abstract

*As a postprocessing procedure, diffusion process has demonstrated its ability of substantially improving the performance of various visual retrieval systems. Whereas, great efforts are also devoted to similarity (or metric) fusion, seeing that only one individual type of similarity cannot fully reveal the intrinsic relationship between objects. This stimulates a great research interest of considering similarity fusion in the framework of diffusion process (*i.e.*, fusion with diffusion) for robust retrieval.*

In this paper, we firstly revisit representative methods about fusion with diffusion, and provide new insights which are ignored by previous researchers. Then, observing that existing algorithms are susceptible to noisy similarities, the proposed Regularized Ensemble Diffusion (RED) is bundled with an automatic weight learning paradigm, so that the negative impacts of noisy similarities are suppressed. At last, we integrate several recently-proposed similarities with the proposed framework. The experimental results suggest that we can achieve new state-of-the-art performances on various retrieval tasks, including 3D shape retrieval on ModelNet dataset, and image retrieval on Holidays and Ukbench dataset.

1. Introduction

Object retrieval is a fundamental yet hot topic in computer vision, which has attracted much attention for decades. Given a query instance, its target is to find objects sharing similar visual appearances with the query in a large database. For a long time, it is crucial to design discriminative representations, so that the metric defined on the representations can be robust to common deformations, such as rotation, occlusion, illumination, *etc*. Conventionally, Bag-of-Words (BoW) is usually employed thanks to the design of local descriptors (*e.g.*, [27] for images, [26, 37] for shapes, [42, 43, 15, 14] for 3D models). In recent years, the rapid development of deep learning algorithms and GPU computing platforms has shifted the research attention to deep-learned features [2, 31, 11, 47, 24], which yield a remarkable performance boost against conventional hand-crafted features.

Nevertheless, the underlying manifold structure is neglected when directly computing the pairwise similarity in the metric space. To this end, a re-ranking component called diffusion process (see [10] for a survey) is usually plugged as a postprocessing step to refine the search results. Diffusion process models the relationship between objects on graph-based manifold, wherein similarity values are diffused along the geodesic path in an iterative manner.

Meanwhile, different visual similarities generally focus on different aspects of objects. Thus, it probably occurs that two objects quite distant in one similarity space may be close in another. Therefore, enormous efforts are also devoted to similarity fusion for the sake of leveraging complementary nature among those distinct feature modalities. To inherit the property of manifold-preserving from diffusion process, fusion is usually reconsidered within the framework of diffusion process, which leads to a new methodology called *Fusion with Diffusion*.

Existing fusion with diffusion methods either utilize a naive solution by simply combining the edge weights of multiple affinity graphs (Sec. 3.1), or consider a homogeneous fusion (Sec. 3.2). However, most of them are susceptible to noisy similarities owing to the lack of weight learning mechanism. To remedy this, we propose a novel fusion with diffusion method called Regularized Ensemble Diffusion (RED) in Sec. 3.3. Compared with existing diffusion processes, the contributions of RED are two folds: i) RED is a theoretically sound and flexible approach to integrate multiple (more than 2) similarities and learn their weights in the diffusion framework; ii) the weights of similarities are learned in a totally unsupervised setting, whose essence is to seek for an optimal weight configuration to maximize the smoothness of multiple tensor-order graph manifolds.

The validity of those fusion with diffusion methods (both

*indicates equal contributions.
†corresponding author.

existing and newly proposed ones) is evaluated on various retrieval tasks. Benefiting from the progressive capacity of deep-based visual representations and handcrafted features, we are able to provide new state-of-the-art performances with 3D model retrieval and natural image retrieval.

2. Revisiting Diffusion Process

We begin with preliminary facts about graph diffusion process in this section. Assume $\mathcal{G} = (X, W)$ is a weighted graph, where $X = \{x_1, x_2, \dots, x_N\}$ is the set of vertices representing the data points, and the edge between x_i and x_j has weight $w_{ij} \in W$. Diffusion process learns a more faithful similarity $A \in \mathbb{R}^{N \times N}$ via iteration. According to the taxonomy given in [10], variants of diffusion process primarily differ in the definition of transition matrix and updating scheme. Below we review some representative variants which can be deemed to operate on tensor product graph, which is testified superior to other variants.

Setting D as a diagonal matrix with elements $D_{ii} = \sum_{j'=1}^{N} W_{ij'}$, the transition matrix can be defined as $P = D^{-1}W$. Afterwards, Locally Constrained Diffusion Process (LCDP) [50] propagates the similarities via

$$A^{(t+1)} = PA^{(t)}P^{\mathrm{T}}, \tag{1}$$

where superscript t is the number of iterations. Since LCDP cannot guarantee the convergence, the iteration has to be stopped manually.

Apart from LCDP, Tensor Product Graph (TPG) diffusion [51] is proven to reach convergence after a sufficient number iterations with the updating scheme

$$A^{(t+1)} = PA^{(t)}P^{\mathrm{T}} + I, \tag{2}$$

where $I \in \mathbb{R}^{N \times N}$ is the identity matrix.

Meanwhile, there are some other kinds of diffusion process, such as Manifold Ranking [56], Graph Transduction (GT) [7], Self Diffusion (SD) [45], Self-smoothing Operator (SSO) [21], Contextual Dissimilarity Measure (CD-M) [20], Rank Diffusion [34], RL-Sim Re-ranking [33], Reciprocal kNN Graph Learning [32], *etc*.

3. Fusion with Diffusion

Different from diffusion process that works with only one affinity graph, fusion with diffusion can tackle $M \geq 2$ affinity graphs $\mathcal{G}^v = (X, W^v)_{v=1}^{M}$ simultaneously. As above, our target is to learn the new similarity $A \in \mathbb{R}^{N \times N}$ which 1) captures the geometry of the underlying manifolds, and 2) leverages the complementarity among multiple visual features.

Since most fusion with diffusion methods stem from certain variants of diffusion process, we will use

$$A^{(t+1)} = \alpha S A^{(t)} S^{\mathrm{T}} + (1 - \alpha) I \tag{3}$$

proposed in [4] to keep consistency, where $S = D^{-1/2} W D^{-1/2}$ and $\alpha \in (0, 1)$ is a constant.

3.1. Naive Fusion

The most straightforward solution is to use a linear combination of multiple similarities. Naive fusion simply averages the input similarities. Specifically, the transition matrix used in Eq. (3) is computed as

$$S = \frac{1}{M} \sum_{v=1}^{M} S^v, \tag{4}$$

where S^v is the transition matrix of the v-th affinity graph. Subsequently, a standard diffusion process is applied. This fusion strategy is extensively investigated in Locally Constrained Mixed Diffusion (LCMD) [28], Graph Fusion [53, 53], Yang *et al.* [49], *etc*.

Though simple to implement, naive solution totally ignores the correlations among different similarities. Moreover, it is sensitive to noisy similarities, since it cannot adaptively decrease the weights of noise. Note that there exist some heuristic ways of weight learning here [49, 55].

3.2. Tensor Product Fusion

Another typical fusion strategy is tensor product fusion. Zhou *et al.* [57, 58] generalize TPG diffusion process [51] to deal with two similarities, where the high-order graph is built by computing the tensor product of two distinct affinity graphs.

By slightly adjusting the transition matrix in Eq. (3), we formulate the tensor product graph fusion as

$$A^{(t+1)} = \alpha S^{(2)} A^{(t)} {S^{(1)}}^{\mathrm{T}} + (1 - \alpha) I, \tag{5}$$

where $S^{(1)}$ and $S^{(2)}$ are the transition matrices associated with two similarities, respectively.

As proven in Proposition 1[1], after sufficient iterations, Eq. (5) converges to

$$A = \lim_{t \to \infty} A^{(t)} = (1 - \alpha) vec^{-1} \left((I - \alpha \mathbb{S})^{-1} \tilde{I} \right), \tag{6}$$

where $\mathbb{S} \in \mathbb{R}^{N^2 \times N^2}$ is the Kronecker product of $S^{(1)}$ and $S^{(2)}$, *i.e.*, $\mathbb{S} = S^{(1)} \otimes S^{(2)}$, and I is the identity matrix of the appropriate size. $vec(\cdot)$ is the vectorization of the input matrix by stacking its columns one by one, and its inverse function is $vec^{-1}(\cdot)$. To simplify the notation, we define $\tilde{Y} = vec(Y)$ for any input matrix Y.

Though the iterative formulation of diffusion is intensively exploited, it lacks theoretical explanations of how this

[1]Due to the space limitation, all the propositions used in the main paper are in the **supplementary material**.

kind of similarity fusion better captures the manifold structure. In this paper, we prove in Proposition 2 that the limit of $A^{(t)}$ presented in Eq. (6) can be derived by solving

$$\min_{A} \frac{1}{2} \sum_{i,j,k,l=1}^{N} W_{ij}^{(1)} W_{kl}^{(2)} \left(\frac{A_{ki}}{\sqrt{D_{ii}^{(1)} D_{kk}^{(2)}}} - \frac{A_{lj}}{\sqrt{D_{jj}^{(1)} D_{ll}^{(2)}}} \right)^2 + \mu \sum_{k,i=1}^{N} (A_{ki} - I_{ki})^2, \tag{7}$$

where $\mu = \frac{1}{\alpha} - 1 \in (0, +\infty)$ is a constant regularizer. Eq. (7) consists of two terms. The first term measures the smoothness of the tensor product graph, indicating that if x_i is similar to x_j in the 1st similarity space, *i.e.*, large $W_{ij}^{(1)}$, and x_k is similar to x_l in the 2nd similarity space, *i.e.*, large $W_{kl}^{(2)}$, then the learned similarities A_{ki} and A_{lj} should have a small difference. The second term suggests that the self-similarity I_{kk} should be preserved to a certain extent.

Here, we make a key observation that is essential in this paper. Though formulated in an iterative model, tensor product fusion can be theoretically explained using an optimization framework. Its essence is to seek for an optimal configuration A that minimizes the objective value of Eq. (7), and consequently, A maximizes the smoothness of the joint graph manifold. While the iterative formulation in Eq. (5) can only integrate two similarity matrices, it is possible to ensemble any number of such matrices in an optimization framework. More importantly, as we will show below, it is possible to learn weights for these similarities so that the smoothness of those manifolds is maximized.

Tensor product fusion considers the complementary structures of two different affinity graphs. However, it evades the weight learning issue, so that its performances are easily deteriorated once one of the two affinity graphs involves some noisy edges. Furthermore, in general, $S^{(1)} \otimes S^{(2)} \neq S^{(2)} \otimes S^{(1)}$. That is to say, when fusing two similarities, the order of computing the Kronecker product makes differences.

3.3. Regularized Ensemble Diffusion

Insofar as we can conclude, learning the weights of multiple similarities has not been treated seriously by most existing fusion with diffusion methods. Since retrieval task usually does not have labeled training data, we expect weight learning can be done in an unsupervised manner, possibly with fewer additional parameters. Inspired by [6] where a weight learning paradigm can be exerted on affinity graphs to assist neighborhood structure mining, the proposed Regularized Ensemble Diffusion (RED) makes viable the automatic weight learning for fusion with diffusion. However, the key novelty of RED lies in three facts. 1) Instead of using an exponential weight learner as [6], RED adopts a more robust weight learning paradigm with regularization. 2) Although formulated in an optimization problem, RED can be efficiently solved in the spirit of the standard iteration-based diffusion process. 3) RED inherits the ability of capturing high-order relationships from tensor product diffusion [10], so that it can learn a more discriminative similarity.

Let $\beta = \{\beta_1, \beta_2, \ldots, \beta_M\}$, with β_v $(1 \leq v \leq M)$ being the weight of the v-th affinity graph. We formulate the weight learning for β and the affinity learning for A in a unified framework as

$$\min_{\beta, A} \sum_{v=1}^{M} \beta_v H^v + \mu \sum_{k,i=1}^{N} (A_{ki} - I_{ki})^2 + \frac{1}{2} \lambda \|\beta\|_2^2, \quad \textit{s.t.}\ \ 0 \leq \beta_v \leq 1,\ \sum_{v=1}^{M} \beta_v = 1, \tag{8}$$

where

$$H^v = \frac{1}{2} \sum_{i,j,k,l=1}^{N} W_{ij}^{v} W_{kl}^{v} \left(\frac{A_{ki}}{\sqrt{D_{ii}^{v} D_{kk}^{v}}} - \frac{A_{lj}}{\sqrt{D_{jj}^{v} D_{ll}^{v}}} \right)^2, \tag{9}$$

and $\lambda > 0$ is the weight **R**egularizer, controlling the distribution of the learned weights. As Eq. (8) shows, if the v-th graph is non-smooth (large H^v), it will be assigned small weight β_v such that the objective value will decrease.

At first glance, one may doubt that RED is relevant to the iterative diffusion processes [10] expatiated above. However, as we present below, its optimization heavily relates to an iterative solver. The optimization of Eq. (8) can be decomposed into two subproblems:

3.3.1 Update A, fix β

In this situation, $\frac{1}{2}\lambda\|\beta\|_2^2$ is a constant which can be omitted directly. With similar algebraic operations used in Proposition 2, Eq. (8) can be transformed into

$$\min_{\tilde{A}} \sum_{v=1}^{M} \beta_v \tilde{A}^{\mathrm{T}} (I - \mathbb{S}^v) \tilde{A} + \mu \|\tilde{A} - \tilde{I}\|^2, \tag{10}$$

where $\mathbb{S}^v = S^v \otimes S^v \in \mathbb{R}^{N^2 \times N^2}$ is the Kronecker product of the v-th transition matrix with itself.

Closed-form Solution. The partial derivative of Eq. (10) with respect to $\tilde{A}$ is

$$2 \sum_{v=1}^{M} \beta_v (I - \mathbb{S}^v) \tilde{A} + 2\mu(\tilde{A} - \tilde{I}). \tag{11}$$

Since Eq. (10) is convex with respect to $\tilde{A}$, one can directly obtain its closed-form solution by setting Eq. (11) to zero. Consequently, we get

$$\tilde{A} = (1 - \sum_{v=1}^{M} \alpha_v)(I - \sum_{v=1}^{M} \alpha_v \mathbb{S}^v)^{-1} \tilde{I}, \tag{12}$$

where

$$\alpha_v = \frac{\beta_v}{\mu + \sum_{v'=1}^{M} \beta_{v'}}. \tag{13}$$

Finally, the optimal solution of this subproblem can be obtained as $A = vec^{-1}(\tilde{A})$.

Iteration-based Solver. However, the closed-form solution in Eq. (12) is computationally prohibited even for small graphs. As we have to calculate the inverse of matrix of size $N^2 \times N^2$, the required time complexity is $O(N^6)$! This naturally motivates us to seek for an efficient iteration-based solver like the standard diffusion processes.

As presented in Proposition 3, the solution in Eq. (12) can be recovered by running

$$A^{(t+1)} = \sum_{v=1}^{M} \alpha_v S^v A^{(t)} S^{v\mathrm{T}} + (1 - \sum_{v=1}^{M} \alpha_v) I \tag{14}$$

for a sufficient number of iterations with an arbitrary initialization of $A^{(1)}$. Consequently, the time complexity of updating A is reduced from $O(N^6)$ to $O(N^3)$.

3.3.2 Update β, fix A

When A is fixed, Eq. (8) can be simplified as the following problem

$$\min_{\beta} \sum_{v=1}^{M} \beta_v H^v + \frac{1}{2}\lambda \|\beta\|_2^2, \ s.t.\ 0 \le \beta_v \le 1, \ \sum_{v=1}^{M} \beta_v = 1. \tag{15}$$

It can be efficiently solved using coordinate descent.

In each iteration of coordinate descent, two elements β_i and β_j are selected to be updated, while the others are fixed. Taking into account the Lagrange function for the constraint $\sum_{v=1}^{M} \beta_v = 1$, we have the following updating scheme

$$\begin{cases} \beta_i^* = \frac{\lambda(\beta_i + \beta_j) + (H^j - H^i)}{2\lambda}, \\ \beta_j^* = \beta_i + \beta_j - \beta_i^*. \end{cases} \tag{16}$$

The obtained β_i^* (or β_j^*) may violate the constraint $\beta_v \ge 0$. Hence, we set $\beta_i^* = 0$ if $\lambda(\beta_i + \beta_j) + (H^j - H^i) < 0$, and vice versa for β_j^*.

The above optimization procedure is guaranteed to converge. In each subproblem, we obtain its optimal solution. By solving two subproblems alternatively, the objective value of Eq. (8) keeps decreasing monotonically. In addition, the objective function is lower bounded by zero. Thus, the convergence can be verified.

Compared with naive fusion and tensor product fusion, RED is robust to noisy features by adaptively tuning the weights β. More importantly, zero weights are allowed such that irrelevant graphs can be totally filtered out, which is impossible for the approach in [6]. The pseudocode of the derived ensemble diffusion is presented in Alg. 1. Note that as no prior knowledge is available to judge the discriminative power of different similarities in unsupervised retrieval, a natural initialization for the weight β_v is $\frac{1}{M}$.

Algorithm 1: Regularized Ensemble Diffusion.

```
Input:
$M$ similarity matrices $\{W^{(v)}\}_{v=1}^{M} \in \mathbb{R}^{N\times N}$, $\lambda$, $\mu$
Output:
The learned similarity $A$;
begin
    Initialize the weight $\beta^{(v)} = \frac{1}{M}$;
    repeat
        Update $A$ using Eq. (14);
        Update $\beta$ using Eq. (16);
    until convergence
    return $A$
```

3.3.3 Further Explanation of the Loss Function

We consider here a pathological case of input similarity $W^{(\triangle)}$ that makes the loss function of RED uninformative. This happens when $W^{(\triangle)} = I$ is defined as

$$W_{ij}^{(\triangle)} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \ne j, \end{cases}$$

which only contains self-similarity of each object.

Indeed, in this case, $H^{(\triangle)}$ is equal to 0 according to Eq. (9). Afterwards, RED will only favor the similarity $W^{(\triangle)}$ by setting $\beta_{(\triangle)} = 1$. In other words, all the other similarities are discarded. However, we emphasize that the goal of this paper is to learn a more faithful similarity from multiple pairwise similarities. Hence, the basic requirement is that we need to know exactly a meaningful relationship between objects. Unfortunately, $W^{(\triangle)}$ is meaningless in this sense, since it cannot depict the relationship between two objects. In this case, neither the proposed methods nor existing affinity learning algorithms can learn a meaningful similarity for retrieval.

In summary, although the proposed method can handle any types of similarities in terms of its basic theory, we do not encourage the users to deliberately force $W^{(\triangle)}$ to be the input, since it is an ill-defined similarity.

3.4. Discussions

We summarize the inherent differences between those fusion with diffusion methods in Table 1. As the table suggests, the biggest defect of tensor product fusion is that it can only deal with $M = 2$ similarities, limiting its promotion where multiple similarities ($M \ge 2$) are accessible. Secondly, RED is the most robust to noise similarities owing to the weight learning mechanism, followed by naive fusion and tensor product fusion (see the experiments). The robustness of naive fusion comes from a statistical assumption that if two objects are similar in most similarity spaces, they are true matching pairs.

The time complexity of naive fusion is the lowest. It only requires one diffusion step in Eq. (3), leading to a complexity of $O(t_d N^3)$, where t_d is the number of iterations in diffusion process. Tensor product fusion has to select 2 similarities each time to fulfill the diffusion step. Thus, its entire time complexity is $O\left(t_d M(M-1)N^3\right)$. The complexity of RED has two parts. The first is updating A in $O(t_d M N^3)$ via Eq. (14), while the second part is updating β. It seems that we need $O(N^4)$ to compute Eq. (9). However, as suggested in [10, 50], diffusion process is usually localized by propagating similarities only on k-nearest neighbor graphs, leading to $O(k^4)$ in this step. Then, it requires $O(t_a M^2)$ (t_a is the number of iterations in coordinate descent) via Eq. (16) in RED. Considering the outer iteration number T in alternating optimization, the final cost of RED is $O\left(T(t_d M N^3 + k^4 + t_a M^2)\right)$. Since $t_d, t_a, k, M \ll N$, we can conclude that the time complexity of all the above fusion with diffusion methods is dominated by $O(N^3)$, which is equivalent to the standard diffusion process [10].

4. Experiments

In this section, we give a thorough evaluation of those fusion with diffusion methods on various retrieval tasks.

4.1. 3D Model Retrieval

The proposed framework is firstly evaluated on 3D model retrieval on the ModelNet dataset, which is a large-scale 3D shape repository, currently consisting of $151,128$ 3D CAD models in 662 object categories. Following [46], two subsets are used for evaluation, *i.e.*, ModelNet40, containing $12,311$ shapes divided into 40 object categories, and ModelNet10, containing $4,899$ shapes divided into 10 object categories. As for the experimental setup, we use the same training-testing split as [41, 5], and employ Area Under precision-recall Curve (AUC) and mean Average Precision (mAP) as evaluation metrics. The parameters are given as $k = 16$ in kNN graph, $\mu = 0.3$ for fastening self-similarity, and $\lambda = 19$ for both two datasets.

Baselines. To obtain multiple similarities, we implemented 4 representative baselines. They are 1) Volumetric CNN [35]: It is combined with multi-view Convolutional Neural Network (CNN), and uses 3DCNN with multi-oriented pooling to obtain shape representations. 2) GIFT [5]: It is an elaborative search engine focusing on the scalability of 3D shape retrieval. We follow its pipeline by training an 8-layer CNN as the view feature extractor, and apply Hausdorff matching to activations of the $7th$ fully-connected layer; 3) ResNet [16]: As a residual learning framework capable of training ultra deep networks, ResNet has shown outstanding performances on image classification and object detection. We introduce, for the first time, ResNet for 3D shape analysis. Here we finetune a 50-layer ResNet to extract view features, and utilize Hausdorff distance as [5] for shape matching; 4) PANORAMA [30]: it is a classical shape descriptor, which is comprised of Discrete Fourier Transform and Discrete Wavelet Transform calculated on panoramic views. The performances of those baselines are listed in Table 2.

Methods	ModelNet40		ModelNet10	
	AUC	mAP	AUC	mAP
Vol. CNN	80.39	79.53	91.24	89.97
GIFT	77.19	76.52	88.97	87.98
ResNet	80.12	79.41	89.02	88.17
PANO.	45.10	44.52	62.37	61.47

Table 2. The performances (%) of baselines on ModelNet40 and ModelNet10 dataset, respectively.

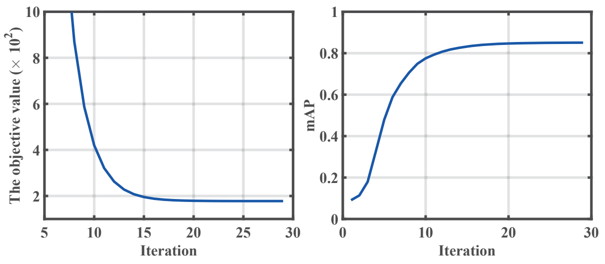

Figure 1. The objective value and mAP (%) of tensor product fusion at each iteration on ModelNet40 dataset. The two similarities fused are Volumetric CNN and GIFT.

Analysis of Tensor Product Fusion. In Fig. 1, we plot the objective value of Eq. (7) and the retrieval performance at each iteration of diffusion process. As can be clearly seen, when similarities are propagated iteratively, the objective value keeps decreasing and the retrieval performance keeps increasing until reaching the equilibrium. Such an observation validates our new perspective about tensor product fusion in Sec. 3.2, *i.e.*, the essence of the iterative solver of tensor product fusion is to recover a closed-form solution of an optimization problem, which measures the smoothness of the joint graph manifold.

Table 3 lists the retrieval performances of tensor product fusion by combining two of M similarities. Firstly, different orders of fusing the same 2 similarities yield different performances. Second, tensor product fusion totally fails when one of the two to-be-fused similarities is not discriminative enough. For instance, the fusion of Volumetric CNN and PANORAMA achieves mAP 67.16, significantly lower than the baseline mAP 79.53 of Volumetric CNN.

Comparison of Fusion Methods. Table 4 shows the comparison of different fusion methods. We can see that tensor

Methods	M	#Weight	#Noise	#Complexity	#Parameter	#Performance
Naive fusion	≥ 2	$\times$	Good	$O(t_d N^3)$	0	Bad
Tensor product fusion	$=2$	$\times$	Bad	$O\left(t_d M(M-1)N^3\right)$	0	Good
RED	≥ 2	$\surd$	Excellent	$O\left(T(t_d M N^3 + k^4 + t_a M^2)\right)$	1	Excellent

Table 1. The summary of fusion with diffusion methods. M denotes the number of similarities that can be handled. #Weight denotes whether weight learning exists. #Noise denotes the robustness to noise. #Complexity denotes the time complexity. #Parameter denotes the number of additional parameters over the standard diffusion process. #Performance denotes the experimental performances.

	Vol. CNN	GIFT	ResNet	PANO.
Vol. CNN	-	83.23	83.77	**67.16**
GIFT	83.15	-	85.08	69.15
ResNet	83.86	**85.12**	-	69.56
PANO.	69.29	71.08	72.32	-

Table 3. The mAPs (%) of tensor product fusion on ModelNet40 dataset. The best and the worst fusing results are marked in red and blue respectively.

Methods	AUC	mAP
Naive Fusion	85.26	84.55
Tensor Product Fusion	68.56$\sim$86.00	67.16$\sim$85.12
RED	**87.03**	**86.30**

Table 4. The performance comparison (%) of different fusion methods on ModelNet40 dataset. As tensor product fusion can only deal with two similarities, its performances are given in an interval.

Dataset	Vol. CNN	GIFT	ResNet	PANO.
ModelNet10	0.281	0.332	0.387	0
ModelNet40	0.296	0.356	0.348	0

Table 5. The learned weights β by RED.

Methods	ModelNet40		ModelNet10	
	AUC	mAP	AUC	mAP
SPH [23]	34.47	33.26	45.97	44.05
LFD [8]	42.04	40.91	51.70	49.82
PANORAMA [30]	45.00	46.13	60.72	60.32
ShapeNets [46]	49.94	49.23	69.28	68.26
DeepPano [39]	77.63	76.81	85.45	84.18
MVCNN [41]	-	78.90	-	-
GIFT [5]	83.10	81.94	92.35	91.12
RED (ours)	**87.03**	**86.30**	**93.20**	**92.15**

Table 6. The performance comparison (%) with state-of-the-art on ModelNet40 and ModelNet10 dataset.

product fusion (its highest AUC and mAP are 86.00 and 85.12 respectively) outperforms naive fusion slightly, since it considers the complementary structures between two homogenous graphs.

Moreover, the proposed RED achieves the best performances due to the effective weight learning mechanism. One desirable expectation is that the weight of PANORAMA should be smaller, since it leads to much lower retrieval performances. Table 5 presents the weights learned by RED. As can be seen, RED sets it to 0 on both ModelNet40 dataset and ModelNet10 dataset. As RED allows for zeros weights, ultra non-smooth graphs (PANORAMA in our case) can be totally eliminated. Therefore, RED can be adapted to more diverse situations even if considerable noise is present.

These results also indicate that if more noisy similarities are fused, the performance difference between RED and other fusion methods will be more dramatic. We will further investigate this in Sec. 4.3.

Comparison with State-of-the-art. In Table 6, we give a comparison with all the representative methods which report retrieval performances on ModelNet dataset, available at `http://modelnet.cs.princeton.edu/`. The best-performing existing methods are Multi-view Convolutional Neutral Network (MVCNN) [41] and GIFT [5]. Specifically, MVCNN proposes a view aggregation layer to produce a compact 3D shape descriptor. By using metric learning additionally, it reports mAP 78.90 on ModelNet40 dataset. Meanwhile, GIFT introduces approximated Hausdorff distance for multi-view matching and Aggregated Contextual Activation for re-ranking. It reports mAP 81.94 on ModelNet40 dataset. By contrast, the proposed RED yields mAP 86.30 on ModelNet40 dataset, outperforming GIFT by 4.36 and MVCNN by 7.40 percent. Besides, RED provides a new best performance on ModelNet10 dataset (AUC 93.20 and mAP 92.15). We envision that better performances can be achieved if more complementary similarities (*e.g.*, [22, 14, 15, 11, 47, 48]) are fused by RED.

4.2. Natural Image Retrieval

We also test the proposed methods on two well-known benchmark datasets for image retrieval, *i.e.*, Holidays dataset [18] and Ukbench dataset [29].

Holidays dataset contains $1,491$ personal photos. 500 images are used as queries. Most queries only have 1-2 ground-truth images, which makes the dataset very challenging for diffusion-based re-ranking methods. The retrieval performance is measured by mean Average Precision (mAP) over all the queries. Ukbench dataset consists of $10,200$ images, grouped into $2,550$ categories. Each image is taken as the query in turn and the rest images serve as the database. The evaluation metric is called N-S score, which counts the average recall of the top-4 ranked images. Thus, the perfect N-S score is 4. Since the scale and the category distribution of the two datasets are quite different, we set $k=7$, $\lambda=28$ on Holidays dataset, and $k=3$, $\lambda=4K$ on

Datasets	NetVLAD	SPoC*	ResNet	HSV
Holidays	88.29	86.07	81.83	61.83
Ukbench	3.739	3.698	3.709	3.195

Table 7. The performances of baselines on Holidays dataset (mAP) and Ukbench dataset (N-S score), respectively. Note that SPoC originally reports mAP 80.2 on Holidays dataset and N-S score 3.65 on Ukbench dataset, which is different from SPoC*.

Methods	Holidays	Ukbench
Naive Fusion	90.69	3.907
Tensor Product Fusion	85.12∼92.46	3.626∼3.884
QALF [55]	88.31	3.846
Graph Fusion [53]	90.65	3.918
SN [6]	91.72	3.919
RED (ours)	**93.32**	**3.938**

Table 8. The performance comparison of different fusion methods on Holidays and Ukbench dataset.

Ukbench dataset. μ is set to 0.08 on both datasets.

Baselines. Four baseline similarities are re-implemented: 1) NetVLAD [1]: An end-to-end trained network which has a new generalized Vector of Locally Aggregated Descriptors (VLAD) [19] layer; 2) SPoC [2]: A strategy about sum-pooling activations of convolutional layers of pretrained C-NNs; 3) ResNet [16]: The fully-connected layer of a pre-trained 50-layer ResNet is used to extract holistic features; 4) HSV color histogram: Following [53, 52, 54], we extract 1000-dimensional HSV color histograms (20×10×5 bins for H, S, V components). Note for deep features, the rotated Holidays dataset released in [3] is used.

Except NetVLAD, all the extracted features are firstly square-rooted [9], and then L_2 normalized. Especially for SPoC, with such a square-root normalization, we obtain much higher performance than the original one reported in [2]. Table 7 shows the performance of our implementation of the 4 baseline methods.

Comparison of Fusion Methods. Table 8 compares the results of different fusion methods. Besides naive fusion and tensor product fusion, we also include three newly-proposed image retrieval algorithms, including Graph Fusion [53], Query-adaptive Late Fusion (QALF) [55] and Smooth Neighborhood (SN) [6]. All the competitors are implemented using the same similarities as RED.

As a representative algorithm, Graph Fusion considers a naive fusion of multiple similarities with equal weights. To get multiple similarities directly comparable, the edge weights are expressed using the Jaccard coefficient of two neighborhood sets. Then, re-ranking is conducted on the fused graph with PageRank. Apart from affinity learning discussed in this paper, SN focuses on mining robust neighborhood structures on multiple affinity graphs. It imposes an exponential weight learner so that the weights of similarities are always larger than 0. Hence, it suffers from the fact that the negative effects of the noisy similarities cannot be entirely eliminated. QALF calculates the weights of similarities by studying the L shape of ranking list. The final similarity is obtained by a weighted combination without diffusion process.

In comparison, the superiority of RED firstly lies that we adopt a more robust weight learning paradigm with theoretical guarantee, than equal weights used by Graph Fusion, the exponential weight learner used by SN and the heuristic weight learning by QALF. More importantly, RED formulates the weight learning and the tensor-order affinity learning in a unified framework, which can efficiently output a more accurate search result. As can be drawn from Table 8, RED achieves better performances on both datasets.

Note that the performance gap is more dramatic with Holidays dataset. Our interpretation is that Ukbench dataset has a very balanced category distribution, *i.e.*, exactly 4 images per category, which makes it easier for algorithms to fit such a distribution.

Comparison with State-of-the-art. In Table 9, a comprehensive comparison to various state-the-of-arts is presented.

The selected methods can be coarsely divided into two kinds. As the focus of this paper, the first kind aims at feature fusion or diffusion, including LCMD [28], CDM [20], kNN Re-ranking [38], and Hello Neighbor [36]. Yang *et al.* [49] propose a data-driven approach to estimate weights of different similarities, and report mAP 88.3 on Holidays dataset and N-S score 3.86 on Ukbench dataset. By using different input similarities, Graph Fusion [53] originally achieves mAP 84.64, and QALF [55] achieves 88.0 on Holidays dataset. The second kind facilitates using deep learning for image retrieval, including Gordo *et al.* [13], Convolutional Kernel Network [31], MOP-CNN [12], SPoC [2] and Neural codes [3]. Since this kind of algorithms usually ignores the geometry structure parameterized by one or more similarities, it can be expected that they are compatible with RED for the sake of better retrieval performances.

Moreover, PGM [25] proposes to use spatial verification, and reports the highest mAP 89.2 on Holidays dataset to our best knowledge. Possibly benefiting from the usage of local descriptors, SN [6] originally achieves N-S score 3.98 on Ukbench dataset. By contrast, RED reports a very competitive performance, *i.e.*, the best mAP 93.3 on Holidays dataset and the second best N-S score 3.94 on Ukbench dataset.

4.3. Discussion

Discussions are primarily done with Holidays dataset.

Robustness to Noise. Most similarities used in our previous experiments are informative in a sense. To simulate the situation where less informative similarities exist, we manually generate 5 similarities by assigning each pair of objects a random value in the interval $(0, \sqrt{2})$ as their pairwise dis-

Datasets	RED	[25]	[6]	[13]	[49]	[40]	[55]	[53]	[2]	[12]	[31]	[3]	[38]	[28]	[20]	[36]
Holidays	**93.3**	89.2	-	89.1	88.3	88.1	88.0	84.6	80.2	80.2	79.3	79.3	76.2	-	-	-
Ukbench	3.94	-	**3.98**	-	3.86	-	3.84	3.83	3.65	-	3.76	3.56	3.52	3.70	3.68	3.67

Table 9. The comparison with state-of-the-art on Holidays dataset and Ukbench dataset.

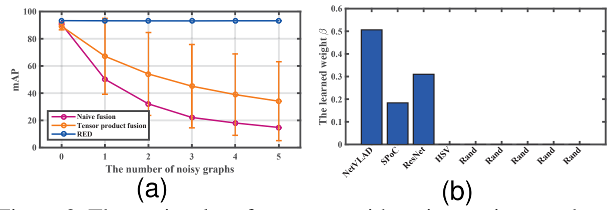

Figure 2. The retrieval performances with an increasing number of noisy similarities (a), and the learned weights β (b) when 5 noisy similarities are fused.

tance. The image retrieval performance of the 5 similarities is around mAP 0.40. We add the noisy similarities to the four baseline methods (Sec. 4.2), and plot the retrieval performances of different fusion with diffusion methods by varying the number of fused noisy similarities in Fig. 2(a). For illustration, the performance of tensor product fusion is given by the average value and the standard deviation of mAPs of all combinations of similarity pairs.

It can be seen clearly that the performance of RED remains almost unchanged even if 5 noisy similarities are integrated. The reason is that the weights of those 5 similarities learned by RED are all zero, as shown in Fig. 2(b). In contrast, naive fusion and tensor product fusion encounter a sharp decrease in performance. When 5 noisy similarities are fused, naive fusion only achieves mAP 14.82. Therefore, one can clearly observe the significance of the weight learning part used in RED.

Sensitivity to Parameters. The most important parameter involved in RED is the weight regularizer λ. Fig. 3(a) shows that the retrieval performances of RED are not so sensitive to the parameter λ. In Fig. 3(b), the learned weights of NetVLAD and HSV are illustrated. Firstly, we can observe that RED is tolerable to the change of λ, as the curve changes gently. Secondly, when $\lambda \leq 12$, NetVLAD always has weight 1. When $\lambda \leq 100$, HSV has weight 0. It reveals that in a wide range of λ, RED can preserve the discriminative power of informative similarities, and eliminate the negative influence of non-informative similarities. At last, we can conclude that when $\lambda \to \infty$, equal weights will be obtained. It also reveals that the determination of λ relies on the degree of complementary nature among different similarities. If rich complementarity exists, a relatively larger λ is needed. The other parameters of RED also occur in the conventional diffusion process, including k and μ. We discuss them in the supplementary material.

Convergence Speed. The objective value of Eq. (8) and the retrieval performance of RED as the iteration increases are

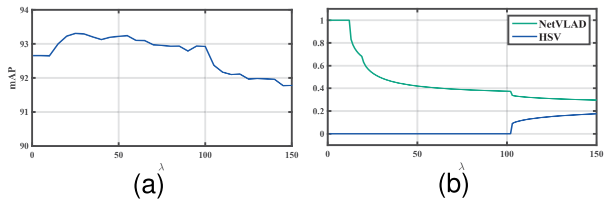

Figure 3. The retrieval performance (a) and the learned weights β (b) when varying λ.

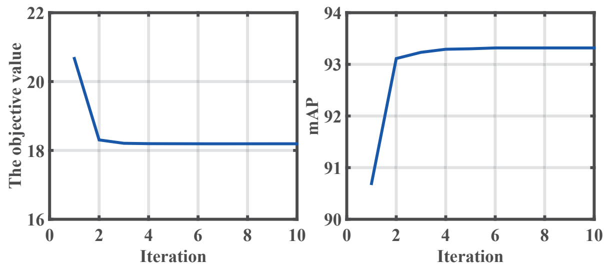

Figure 4. The objective value and mAP of RED at each iteration.

given in Fig. 4. As can be seen, RED converges very fast within less than 4 iterations.

5. Conclusion

In this paper, we focus on similarity fusion in the framework of diffusion process for retrieval. Considering that most existing works are sensitive to noisy similarities, we propose Regularized Ensemble Diffusion (RED), with weights positively related to the smoothness of the (tensor product) graph-based manifolds. Comprehensive experimental results on 3D model retrieval and image retrieval demonstrate the effectiveness of RED.

Although the iterative solver of RED significantly reduces its time complexity, it still requires $O(N^3)$ to finish the similarity propagation step as the conventional diffusion process [10]. Therefore, how to reduce the time complexity of diffusion process [44] to meet the requirement of real-time retrieval can be investigated further. Moreover, RED needs a strategy to efficiently handle out-of-dataset queries, as the iteration has to be done once a new query is added. One possible solution can be adapted from the newly-proposed Regional Diffusion [17], and we leave it as our future work. *The codes are available at* `https://sites.google.com/site/songbaihust/`.

Acknowledgements This work was supported by NSFC 61573160 and 61429201, NSF IIS-1302164, to Dr. Qi Tian by ARO grants W911NF-15-1-0290 and Faculty Research Gift Awards by NEC Laboratories of America and Blippar.

References

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *CVPR*, 2016. 7

[2] A. Babenko and V. Lempitsky. Aggregating local deep features for image retrieval. In *ICCV*, pages 1269–1277, 2015. 1, 7, 8

[3] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, pages 584–599, 2014. 7, 8

[4] S. Bai, X. Bai, Q. Tian, and L. J. Latecki. Regularized diffusion process for visual retrieval. In *AAAI*, pages 3967–3973, 2017. 2

[5] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. J. Latecki. Gift: A real-time and scalable 3d shape search engine. In *CVPR*, 2016. 5, 6

[6] S. Bai, S. Sun, X. Bai, Z. Zhang, and Q. Tian. Smooth neighborhood structure mining on multiple affinity graphs with applications to context-sensitive similarity. In *ECCV*, pages 592–608, 2016. 3, 4, 7, 8

[7] X. Bai, X. Yang, L. J. Latecki, W. Liu, and Z. Tu. Learning context-sensitive shape similarity by graph transduction. *TPAMI*, 32(5):861–874, 2010. 2

[8] D. Y. Chen, X. P. Tian, Y. T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. *Comput. Graph. Forum*, 22(3):223–232, 2003. 6

[9] J. Delhumeau, P.-H. Gosselin, H. Jégou, and P. Pérez. Revisiting the vlad image representation. In *ACM international conference on Multimedia*, pages 653–656, 2013. 7

[10] M. Donoser and H. Bischof. Diffusion processes for retrieval revisited. In *CVPR*, pages 1320–1327, 2013. 1, 2, 3, 5, 8

[11] Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu, and E. Wong. 3d deep shape descriptor. In *CVPR*, pages 2319–2328, 2015. 1, 6

[12] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *ECCV*, pages 392–407, 2014. 7, 8

[13] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *ECCV*, pages 241–257, 2016. 7, 8

[14] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok. A comprehensive performance evaluation of 3d local feature descriptors. *IJCV*, pages 1–24, 2015. 1, 6

[15] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan. Rotational projection statistics for 3d local surface description and object recognition. *IJCV*, 105(1):63–86, 2013. 1, 6

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5, 7

[17] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 8

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

[19] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *TPAMI*, 34(9):1704–1716, 2012. 7

[20] H. Jegou, C. Schmid, H. Harzallah, and J. Verbeek. Accurate image search using the contextual dissimilarity measure. *TPAMI*, 32(1):2–11, 2010. 2, 7, 8

[21] J. Jiang, B. Wang, and Z. Tu. Unsupervised metric learning by self-smoothing operator. In *ICCV*, pages 794–801, 2011. 2

[22] E. Johns, S. Leutenegger, and A. J. Davison. Pairwise decomposition of image sequences for active multi-view recognition. In *CVPR*, 2016. 6

[23] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *SGP*, pages 156–164, 2003. 6

[24] Q. Ke and Y. Li. Is rotation a nuisance in shape recognition? In *CVPR*, pages 4146–4153, 2014. 1

[25] X. Li, M. Larson, and A. Hanjalic. Pairwise geometric matching for large-scale object retrieval. In *CVPR*, pages 5153–5161, 2015. 7, 8

[26] H. Ling and D. W. Jacobs. Shape classification using the inner-distance. *TPAMI*, 29(2):286–299, 2007. 1

[27] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1

[28] L. Luo, C. Shen, C. Zhang, and A. van den Hengel. Shape similarity analysis by self-tuning locally constrained mixed-diffusion. *TMM*, 15(5):1174–1183, 2013. 2, 7, 8

[29] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, 2006. 6

[30] P. Papadakis, I. Pratikakis, T. Theoharis, and S. J. Perantonis. Panorama: A 3d shape descriptor based on panoramic views for unsupervised 3d object retrieval. *IJCV*, 89(2-3):177–192, 2010. 5, 6

[31] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronin, and C. Schmid. Local convolutional features with unsupervised training for image retrieval. In *ICCV*, pages 91–99, 2015. 1, 7, 8

[32] D. C. G. Pedronette, O. A. Penatti, and R. d. S. Torres. Unsupervised manifold learning using reciprocal knn graphs in image re-ranking and rank aggregation tasks. *Image and Vision Computing*, 32(2):120–130, 2014. 2

[33] D. C. G. Pedronette and R. D. S. Torres. Image re-ranking and rank aggregation based on similarity of ranked lists. *Pattern Recognition*, 46(8):2350–2360, 2013. 2

[34] D. C. G. Pedronette and R. d. S. Torres. Rank diffusion for context-based image retrieval. In *ICMR*, pages 321–325, 2016. 2

[35] C. R. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, 2016. 5

[36] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, pages 777–784, 2011. 7, 8

[37] B. Ramesh, C. Xiang, and T. H. Lee. Shape classification using invariant features and contextual information in the bag-of-words model. *Pattern Recognition*, 48(3):894–906, 2015. 1

[38] X. Shen, Z. Lin, J. Brandt, S. Avidan, and Y. Wu. Object retrieval and localization with spatially-constrained similarity

measure and k-nn re-ranking. In *CVPR*, pages 3013–3020, 2012. 7, 8

[39] B. Shi, S. Bai, Z. Zhou, and X. Bai. Deeppano: Deep panoramic representation for 3-d shape recognition. *IEEE Signal Processing Letters*, 22(12):2339–2343, 2015. 6

[40] M. Shi, Y. Avrithis, and H. Jégou. Early burst detection for memory-efficient image retrieval. In *CVPR*, pages 605–613, 2015. 8

[41] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, pages 945–953, 2015. 5, 6

[42] H. Tabia, M. Daoudi, J.-P. Vandeborre, and O. Colot. A new 3d-matching method of nonrigid and partially similar models using curve analysis. *TPAMI*, 33(4):852–858, 2011. 1

[43] H. Tabia, H. Laga, D. Picard, and P.-H. Gosselin. Covariance descriptors for 3d shape matching and retrieval. In *CVPR*, pages 4185–4192, 2014. 1

[44] H. Tong, C. Faloutsos, and J. Pan. Fast random walk with restart and its applications. In *ICDM*, pages 613–622, 2006. 8

[45] B. Wang and Z. Tu. Affinity learning via self-diffusion for image segmentation and clustering. In *CVPR*, pages 2312–2319, 2012. 2

[46] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shape modeling. In *CVPR*, 2015. 5, 6

[47] J. Xie, Y. Fang, F. Zhu, and E. Wong. Deepshape: Deep learned shape descriptor for 3d shape matching and retrieval. In *CVPR*, pages 1275–1283, 2015. 1, 6

[48] J. Xie, M. Wang, and Y. Fang. Learned binary spectral shape descriptor for 3d shape correspondence. In *CVPR*, pages 3309–3317, 2016. 6

[49] F. Yang, B. Matei, and L. S. Davis. Re-ranking by multi-feature fusion with diffusion for image retrieval. In *WACV*, pages 572–579, 2015. 2, 7, 8

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

[51] X. Yang, L. Prasad, and L. J. Latecki. Affinity learning with diffusion on tensor product graph. *TPAMI*, 35(1):28–38, 2013. 2

[52] S. Zhang, M. Yang, T. Cour, K. Yu, and D. N. Metaxas. Query specific fusion for image retrieval. In *ECCV*, pages 660–673, 2012. 7

[53] S. Zhang, M. Yang, T. Cour, K. Yu, and D. N. Metaxas. Query specific rank fusion for image retrieval. *TPAMI*, 37(4):803–815, 2015. 2, 7, 8

[54] L. Zheng, S. Wang, Z. Liu, and Q. Tian. Packing and padding: Coupled multi-index for accurate image retrieval. In *CVPR*, pages 1939–1946, 2014. 7

[55] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, and Q. Tian. Query-adaptive late fusion for image search and person re-identification. In *CVPR*, pages 1741–1750, 2015. 2, 7, 8

[56] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, pages 169–176, 2004. 2

[57] Y. Zhou, X. Bai, W. Liu, and L. J. Latecki. Fusion with diffusion for robust visual tracking. In *NIPS*, pages 2978–2986, 2012. 2

[58] Y. Zhou, X. Bai, W. Liu, and L. J. Latecki. Similarity fusion for visual tracking. *IJCV*, pages 1–27, 2016. 2