First Index-Free Manifold Ranking-based Image Retrieval with Output Bound

Dandan Lin*, Victor Junqiu Wei†, Raymond Chi-Wing Wong*
* The Hong Kong University of Science and Technology
{dlinaf, raywong}@cse.ust.hk
†Noah's Ark Lab of Huawei
wei.junqiu1@huawei.com

***Abstract*—Image retrieval keeps attracting a lot of attention from both academic and industry over past years due to its variety of useful applications. Due to the rapid growth of deep learning approaches, more better feature vectors of images could be discovered for improving image retrieval. However, most (if not all) existing deep learning approaches consider the similarity between 2 images *locally* without considering the similarity among a *group* of similar images *globally*, and thus could not return accurate results. In this paper, we study the image retrieval with manifold ranking (MR) which considers both the local similarity and the global similarity, which could give more accurate results. However, existing best-known algorithms have one of the following issues: (1) They require a bulky index, (2) some of them do not have any theoretical bound on the output, and (3) some of them are time-consuming. Motivated by this, we propose an algorithm, namely *Monte Carlo-based MR* (*MCMR*) for image retrieval, which does not have the above issues. We are the first one to propose an index-free manifold ranking-based image retrieval with the output theoretical bound. Lastly, our experiments show that *MCMR* outperforms existing algorithms by up to 4 orders of magnitude in terms of query time.**

I. Introduction

Image retrieval [2] keeps attracting a lot of attention from both academic and industry over past years due to its variety of useful applications. In academic, before the growth of deep learning studies, most researchers [2, 3, 35] studied how to "engineer" good image features (manually) such that any two given images should have a high "pre-defined" similarity measure if they look similar to human. Recently, due to the growth of deep learning approaches, researchers focused on how to use deep learning models [32] like convolutional neural networks (CNNs) to capture or find the "embedded" features to get rid of (manual) feature engineering. Besides, it is found [7, 16, 19, 21, 32] that the embedded features capture a lot of important and good ingredients in the image, resulting a good performance of some tasks (e.g., similarity search) in image retrieval. In industry, giant technology companies in the world have large research teams for image retrieval due to their attractive applications. One example is Taobao, the biggest mobile e-commerce platform in China, with its famous application of "similar item search" which returns a list of items which are similar to the photo of an item that a customer would like to buy [31]. Another example is Google, the world leading search company hosted in US, with its image search engine function which returns a list of images similar to an upload image.

A. Feature Extraction & Similarity Search

Specifically, image retrieval involves 2 phases [3, 6, 7, 10, 16, 17, 21, 23, 32], namely the feature extraction and the similarity search. The feature extraction is to find a representation of each image called a *feature vector* which is in the form of a d-dimensional vector. The similarity search is to find a list of images which are "similar" to a given image.

Feature extraction is a very fundamental and important phase for image retrieval. With good feature extraction, the similarity search (in the second phase) could be done more accurately and more effectively. Due to the successful development of deep learning approaches, the features found by these approaches could capture image ingredients well [7, 16, 17, 19, 32]. These deep learning approaches employ the CNN frameworks to find the features. Some representative approaches are the basic CNN approaches [7, 16, 19] and the advanced CNN approach called *MAC* [16, 17] which exploits a feature of focusing essential parts of images and could be regarded as the state-of-the-art in the literature. In [32], it was found that the features could help to improve the accuracy of the similarity search by up to 51.3%.

Although feature extraction is well-studied among deep learning approaches, since most studies [7, 16, 17] directly adopt a "traditional" $\mathcal{L}^p$-norm-based measure (e.g., the Euclidean Distance) of evaluating the similarity of two images, the performance of similarity search is not that good because it is found in [33] that this traditional measure only captures the similarity between 2 images *locally* without considering the similarity among a *group* of similar images *globally*.

B. Manifold Ranking: A Better Similarity Search Approach

However, Manifold ranking (MR) [6, 33, 34], one similarity measure for similarity search, is found to be an effective measure of capturing the similarity both locally and globally. Due to this diverse ability, in recent years, MR is applied for not only image retrieval but also other problems such as *person re-identification* [1, 12], *document similarity search* [22], *identifying quantitative chemical relationship* [18], *saliency detection* [9, 14, 20, 25, 28] and *object co-segmentation* [15].

Next, we would like to give 2 case studies showing how manifold ranking has better performance compared with some best-known models. We included 3 deep learning approaches (which uses the $\mathcal{L}^p$-norm-based measure for similarity search) for comparison, namely *VGG16* [19], *ResNet101* [7], and

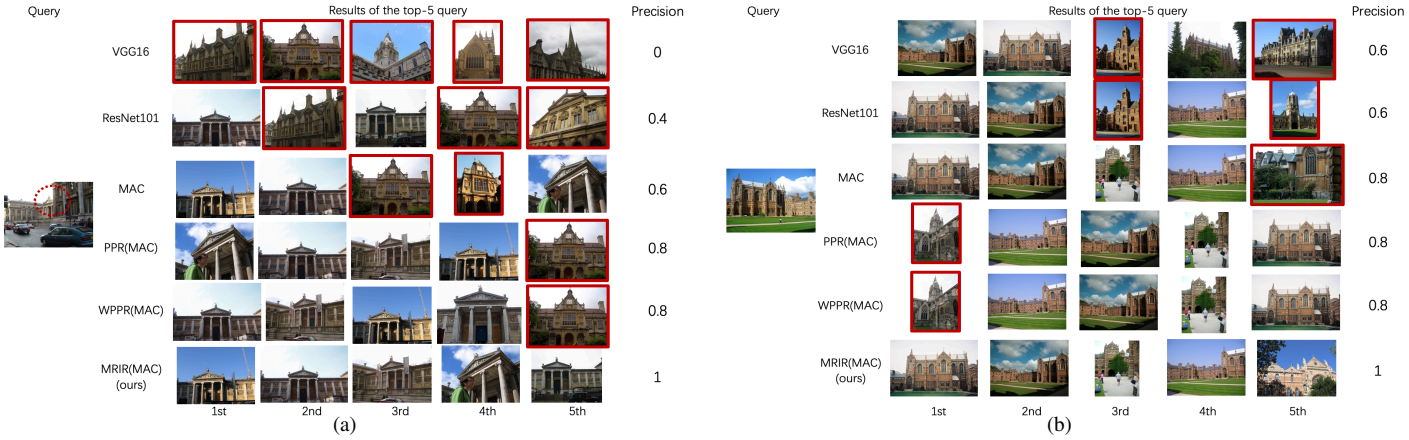

Fig. 1: Top-5 Query Results on Dataset *Roxford5k* returned by different models.

MAC [16, 17], where *MAC* is considered as the state-of-the-art in the literature. We also included 2 popular similarity approaches for comparison, namely *Personalized PageRank* (PPR) [27] and *Weighted Personalized PageRank* (WPPR). We call the manifold ranking framework as the manifold ranking image retrieval (*MRIR*). Note that since the 2 popular similarity approaches and the manifold ranking framework requires the feature space, we also adopt the feature space learnt by the state-of-the-art deep learning approach, *MAC*, for this purpose. We denote these 3 models with *PPR(MAC)*, *WPPR(MAC)* and *MRIR(MAC)* where the feature space is denoted in the bracket of the notation of the model names. The implementation details of these case studies could be found in our technical report [11].

Fig. 1 shows the results of the 2 case studies on a benchmark dataset *Roxford5k*. In Fig. 1(a), the query image is given on the left hand side where the object in the red dotted circle denotes the "special" part/characteristics of the building which should be found in the "correct" output. On the right hand side, we could see the result of the top-5 query returned by each mentioned model. For each model, the result has 2 parts where the first part contains the 5 images with an ordering from the most similar image (on the left side) to the least similar image (on the right side) in the output of this model and the second part is the precision of this model based on these 5 images. Besides, in the first part, if the image in the result of each model is incorrect, it is enclosed with a red border boundary. In this case study, we found that only the manifold ranking framework, *MRIR(MAC)*, could obtain 100% precision but all others returned incorrect images in the result. In Fig. 1(b), similar results are obtained for another query image.

C. *Our Proposed Method:* MCMR

Given a querying image, manifold ranking makes use of both the local similarity and the global similarity to compute a score called the *MR* (manifold ranking) score for each image. In a top-k query, the images with the greatest MR scores are returned as an output.

Definition 1 (Top-k MR Search)**.** *Given an image database, a query image q and a constant k, Top-k search finds the top-k images with the highest Manifold Ranking scores w.r.t q.*

Method	*Mogul-E* [5] (the fastest exact one)	*Mogul* [5] (the fastest approximate one)	*MCMR* (Ours)
Index-free?	No	No	Yes
Output Bound?	Yes	No	Yes
Efficiency?	No	Yes	Yes

TABLE I: Summary of existing methods for MR.

As shown above, manifold ranking is good for accurate image retrieval. We consider the following 3 requirements for evaluating a method M for manifold ranking, namely index-free, output bound and efficiency.

- *Index-free:* M has no index for computing MR scores.
- *Output bound:* The MR scores of images returned by M should have a theoretical quality guarantee.
- *Efficiency:* M is computationally cheap.

For the first index-free requirement, unfortunately, most (if not all) existing algorithms about manifold ranking requires to build an index, which is quite bulky and hinders the flexibility of any image database update. For example, in the Taobao platform involving at least 2 billion items (or images) [24], the existing algorithms are not scalable in this large dataset due to the bulky index. For the second output bound requirement, all existing algorithms returning the approximate MR scores returns answers without any theoretical guarantee. Thus, the output of the top-k MR search results returned by these existing algorithms is inaccurate. Although only existing exact algorithms could return answers with theoretical guarantee (i.e., the exact answer), they suffer from the bulky index size problem. For the third efficiency requirement, all exact algorithms are time-consuming. Though existing approximate algorithms could return answers in a short time, they still have no theoretical guarantee on the output.

However, our proposed approach called *MCMR* satisfies all these requirements. Table I summarizes the results about these 3 requirements of existing algorithms and our *MCMR*.

The following shows our contributions. Firstly, we propose an algorithm, namely *MCMR*, which adopts the random walk sampling strategy without pre-computing any index. This algorithm satisfies the 3 requirements and none of the existing algorithms satisfies these 3 requirement simultaneously. Secondly, to the best of our knowledge, we are the first to pro-

pose index-free algorithms for manifold ranking. All existing algorithms requires to build an index for efficient computation. Thirdly, the time complexity of *MCMR* is $O(n \log n)$ where n is the total number of images in the database. This is the first best-known time complexity result in the literature of returning the exact top-k results with quality guarantee. The existing best-known time complexity in this literature [5] is $O(n^2)$. Fourthly, we conducted experiments on 4 real-world image datasets. The experimental results show that *MCMR* outperforms existing algorithms by up to 4 order of magnitudes in terms of query time.

The remainder of this paper is organized as follows. We give the preliminaries of manifold ranking in Section II. Next, Section III reviews the related work. Then, Section IV presents our *MCMR* algorithm. Section V shows the results of our experiments. Finally, Section VI gives our conclusion.

II. Preliminaries

In this section, we introduce the background of MRIR.

In MRIR, an image database is modelled as a k-NN graph where each node represents an image. Let $G(V, E)$ denote an undirected k-NN graph where V and E are the set of nodes and edges, respectively. Given two nodes u and v, $e(u, v) \in E$ exists only when (i) u is one of the k-nearest neighbours of v or (ii) v is one of the k-nearest neighbours of u, according to [5, 6]. Thus, the k-NN graph is undirected. Note that the k-NN graph is regarded as an off-the-shelf component in MRIR since it is easy to construct and dynamically maintained [4].

MRIR can be formulated as follows: given a query node $q \in V$, it computes the MR scores of all nodes in the graph w.r.t q. Let $\mathbf{A} \subset \mathcal{R}^{n \times n}$ be the adjacency matrix of the graph G where n is the number of nodes. Normally, the edge weight can be defined by the heat kernel [5]: $A_{ij} = \exp\{-\mathbf{d}^2(v_i, v_j)/2\sigma^2\}$ if there is an edge linking node v_i with v_j; otherwise, $A_{ij} = 0$, where function $\mathbf{d}(v_i, v_j)$ is the Euclidean distance between v_i and v_j. Node v_i and node v_j are defined in the $\mathcal{L}^p$-space (obtained by the feature extraction method), and σ is a hyper-parameter. Note that the sum of each row/column in $\mathbf{A}$ can be smaller or larger than 1. Let $\mathbf{q}$ be an $n \times 1$ column vector of zeros except that the q-th entry is set to 1, i.e., $q(q) = 1$. Let $\mathbf{x}_q^*$ denote an $n \times 1$ column exact MR scores vector w.r.t q where $x_q^*(v)$ denotes the exact MR score of node v w.r.t q. Recall that $x_q^*(v)$ measures how similar v is to q.

Intuitively, MR can be understood from the perspective of information spreading from the query node in the whole k-NN graph. Initially, the query node q owns a fixed number of information which will be propagated along the edges in the graph. Then, each node in the graph constantly receives the information from its neighbours until the total information held by each node remains unchanged, reaching a stationary state (or convergence). After the spreading process, the intrinsic global structure can be revealed in the form of the final information hold by each node, which is exactly the MR score of each node w.r.t q. The larger the MR score of node v is, the more node v is similar to q. Next, we introduce how to compute MR scores by the information spreading process. Firstly, it symmetrically normalizes the adjacency matrix $\mathbf{A}$ and constructs a new matrix $\mathbf{W}$ such that $\mathbf{W} = \mathbf{C}^{-1/2}\mathbf{A}\mathbf{C}^{-1/2}$ where $\mathbf{C}$ is the diagonal matrix of $\mathbf{A}$ such that $C_{ii} = \sum_{j=1}^{n} A_{ij}$. To be distinguished from $\mathbf{A}$, $\mathbf{W}$ is called the symmetrically normalized matrix. This normalization is necessary for the convergence of information spreading later. Next, MR scores are computed by iteratively using the following equation until the convergence is reached:

$$\mathbf{x}^{(t+1)} = \alpha \mathbf{W}\mathbf{x}^{(t)} + (1 - \alpha)\mathbf{q} \tag{1}$$

where $\mathbf{x}^{(t)}$ is the MR score vector obtained in the t-th iteration and $\mathbf{x}^{(0)} = \mathbf{0}$. In each iteration, each node receives the information from its neighbours (the first term), and also retains its initial information (the second term). The parameter α specifies the relative amount of the information from its neighbours and itself. This iterative process is denoted as *Power*. When it converges, the exact MR scores (i.e., $\mathbf{x}_q^*$) are obtained. Besides, it has been shown in [5, 6, 33] that after convergence, the vector $\mathbf{x}_q^*$ holds the following equation:

$$\mathbf{x}_q^* = (1 - \alpha)(\mathbf{I} - \alpha \mathbf{W})^{-1}\mathbf{q}. \tag{2}$$

So, one can obtain $\mathbf{x}_q^*$ by computing the matrix inversion.

A greedy method to solve the top-k MRIR search is to firstly compute the MR scores of all nodes w.r.t q by Equation (2), and then return the top-k nodes by sorting the MR scores in decreasing order. However, this greedy method takes $O(n^{2.373})$ time, which is expensive when the graphs (i.e., image databases) are large. Thus, a fast solution is essential.

III. Related Work

MRIR. Recently, *FMR* [8], *EMR* [29], *Mogul* [5] and *Mogul-E* [5] were proposed to improve the efficiency of MRIR. However, they all do not satisfy the 3 requirements simultaneously mentioned in Section I. Among them, *Mogul-E* is *exact* method which compute the exact MR scores, and others are *approximate* methods which compute approximate MR scores. All the methods focus on quickly solving Equation 2 by using different types of matrix decomposition and by constructing several pre-computed indices in the preprocessing phase. Since all types of matrix decomposition used by these methods except *Mogul-E* are approximate, the MR scores computed by them are with errors. None of these approximate methods satisfy the requirement of "output bound". Besides, all the methods including *Mogul-E* need to construct several special-purpose indexes in the prepossessing phase, and thus, they are not index-free. It is worth mentioning that *Mogul* and *Mogul-E* are the state-of-the-art approximate and exact algorithm, respectively. However, the query time complexities of *Mogul* and *Mogul-E* are $O(n)$ and $O(n^2)$, respectively. Although the time complexity of *Mogul* is small, it does not return accurate results due to its inability of returning an output with an output bound. Refer to [11] for more details.

Single Image Super-Resolution (SISR) problem. SISR is a classic computer vision problem, which aims to recover a high-resolution (HR) image from a low-resolution (LR) image [26]. Recently, a lot of deep-learning-based approaches have been proposed to solve SISR problem [26]. However, this problem is very different from MRIR studied in this paper

from two perspectives. Specifically, the first perspective lies in their different goals where SISR wants to re-construct the HR image from a LR image while MRIR aims to return top-k images similar to the query image from the image database. The second one is about the input format where SISR requires a set of pairs of LR images and HR images which could not be used directly for our problem.

IV. Proposed Algorithm: MCMR

To address the deficiencies of existing algorithms, we propose an algorithm called the *Monte Carlo-based Manifold Ranking* (*MCMR*) algorithm, which is efficient without index and can return the *true* top-k nodes with accuracy guarantee. The idea behind *MCMR* is to utilize the *Graph Sampling approaches* [30]. Among various graph sampling approaches, Random Walk Sampling is the mainstream one due to its scalability and simplicity of implementation. The general idea of Simple Random Walk Sampling strategy is as follows. A single random walk starts at a given node, then repeatedly jumps to another node by choosing from the current node's neighbours uniformly at random. After many steps, the probability of a node being visited tends to reach a stationary probability distribution [30]. In addition, we observed that the MR score of a node v w.r.t the query node q (i.e., $x_q^*(v)$) can be regarded as a stationary probability of v being visited by a walk starting from q multiplied by a coefficient (which is decided by the symmetrically normalized matrix $\mathbf{W}$ and the parameter α). It is because $x_q^*(v)$ is obtained by repeatedly receiving the information from its neighbours. Thus, Random Walk Sampling can be a possible solution for our problem.

Challenge. To perform unbiased general Random Walk Sampling on the graph, the whole process can be divided into two phases: (1) *the simulation phase* (which simulates enough number of samples, i.e., random walks), and (2) *the estimation phase* (which performs an *unbiased estimation* on simulated samples). However, it is *non-trivial* and *challenging* to apply Random Walk Sampling to our problem since the scheme of information propagation in MR is based on the symmetrically normalized matrix $\mathbf{W}$, which is not a stochastic matrix. Thus, we cannot simulate the random walks as the general Random Walk Sampling strategy does, indicating that new simulation and estimation phases are required for our problem.

A. Overview

The major idea behind *MCMR* is to simulate *weighted random walks* instead of the simple random walks. Given a query node q, *MCMR* consists of two phases: the simulation phase and the estimation phase. In particular, in the simulation phase, it simulates a specified number of *weighted random walks* from q. In the estimation phase, it estimates the MR scores of all nodes in the graph w.r.t q based on the simulated weighted random walks.

B. Implementation Details of MCMR

The simulation phase. First of all, we formally introduce our weighted random walk in Definition 2.

Definition 2 (Weighted Random Walk)**.** *Given a parameter α, and the query node q, a single weighted random walk is simulated as follows: it starts from node q; and at each step, it chooses one of the following two options: (i) terminate with $(1-\alpha)$ probability; (ii) with α probability, moves to a neighbour of the current node according to our* transition policy *(to be introduced later).*

From Definition 2, the expected length of our weighted random walk is $O(\frac{1}{1-\alpha})$. Next, we introduce our *transition policy*. Suppose that the walk is currently at node v. The idea behind the policy is to choose a neighbour of v with *a weighted probability* instead of uniformly at random. Definition 3 gives the details of our transition policy.

Definition 3 (Transition Policy)**.** *At each step, the weighted random walk jumps to a neighbour u of the current node v with probability $\frac{W_{v,u}}{D_{vv}}$, where $D_{vv} = \sum_{u \in \mathcal{N}(v)} W_{v,u}$ and $\mathcal{N}(v)$ is the set of neighbours of v.*

Here, we say that D_{vv} is the (v,v)-th entry of the diagonal matrix $\mathbf{D}$. In this way, it is ensured that sum of the probabilities that node v moves to one of its neighbours is equal to 1.

The estimation phase. Now, we present how to estimate the MR scores of all nodes based on our new simulation phase.

By using our transition policy, the information from a node v to a node u could not be captured *appropriately*. Specifically, during the information spreading process, at each iteration, the information that v transfers to its neighbour u should be $W_{v,u}$, instead of $\frac{W_{v,u}}{D_{vv}}$ in our transition policy. The basic idea of our estimation phase is to compute u's information *appropriately* at each step of a weighted random walk by using some derivations involving $W_{v,u}$ (instead of $\frac{W_{v,u}}{D_{vv}}$). In addition, we regard the MR score of a node v w.r.t q as the "expected" amount of information that node v obtains from node q (via the weighted random walks). The correctness of our estimation is proved in Theorem 1.

Let $\pi(q,v)$ be the estimated MR score of a node $v \in V$ obtained by our *MCMR* method. Let n_r denote the number of weighted random walks simulated. Suppose that the i-th weighted random walk $\{X_l\}_{1 \le l \le L}$ of length L starts from the query node q. First, we define the *scalar* of the i-th walk for each $v \in V$, denoted as $S_i(v)$, to be the amount of the information that the i-th walk should transfer to node v as follows:

$$S_i(v) = \begin{cases} \prod_{l=1}^{L-1} D_{X_l X_l}, \text{if the } i\text{-th walk ends at } v, \\ 0, \text{otherwise}, \end{cases} \quad (3)$$

where $i \in \{1, 2, ..., n_r\}$ and X_l is the l-th node in this walk. Next, we define the total scalars of all walks for each $v \in V$, denoted by $S(q,v)$, to be $\sum_{i=1}^{n_r} S_i(v)$. Finally, the estimated MR score $\pi(q,v)$ of node v is computed as follows: $\pi(q,v) = \frac{S(q,v)}{n_r}$.

The MCMR algorithm. Algorithm 1 gives the pseudocode of our *MCMR* algorithm. It firstly initializes the estimated MR score $\pi(q,v)$ and the sum of scalars $S(q,v)$ to 0 for each node $v \in V$ (Line 1). Then, it generates n_r random walks as follows: for the i-th walk, it initializes the scalar of this walk

Algorithm 1 MCMR

Input: Graph $G = (V, E)$, query node q, matrices $\mathbf{W}$ and $\mathbf{D}$, constant parameter α, number of random walks n_r
Output: Estimated MR score $\pi(q, v)$ for each $v \in V$
1: $\pi(q, v) \leftarrow 0$,$S(q, v) \leftarrow 0$ for each $v \in V$;
2: **for** i from 1 to n_r **do**
3: $S_i \leftarrow 1$; $v \leftarrow q$;
4: **while** $rand() > 1 - \alpha$ **do**
5: Pick one neighbor u of v with the probability $\frac{W_{v,u}}{D_{vv}}$;
6: $S_i \leftarrow S_i \cdot D_{vv}$; $v \leftarrow u$;
7: $S(q, v) \leftarrow S(q, v) + S_i$;
8: $\pi(q, v) \leftarrow S(q, v)/n_r$ for each $v \in V$;

S_i as 1 (Line 3); at each step, it uniformly generates a random value; if the value is greater than $1-\alpha$, it picks up an neighbour u of current node v with probability $\frac{W_{v,u}}{D_{vv}}$ and multiplies S_i by D_{vv} (Lines 4-6); otherwise, the walk terminates at v, it adds the scalar of this walk S_i to $S(q, v)$ (Line 7). After the simulation phase finishes, the estimated MR score $\pi(q, v)$ is computed as $\frac{S(q,v)}{n_r}$ for each node $v \in V$ (Line 8).

C. Theoretical analysis

In this section, we first prove that the estimated MR scores obtained from *MCMR* are unbiased as shown in Theorem 1 and then give a time complexity analysis of *MCMR*. Please refer to our technical report [11] for the step-by-step proofs of all theorems and lemmas in this paper.

Theorem 1. *Let $\pi(q, v)$ be the estimated MR score obtained by Algorithm 1. We have $E[\pi(q, v)] = x^*_q(v)$.*

Proof. The proof sketch is given here. Firstly, we have $E[\pi(q, v)] = E[\frac{S(q,v)}{n_r}] = \frac{E[\sum_{i=1}^{n_r} S_i(v)]}{n_r} = E[S_i(v)]$. Next, we prove that $E[S_i(v)]$ is the expected amount of the information of a weighted random walk from q propagating to node v by considering all possible walks from q, which is exactly $x^*_q(v)$. □

Time complexity. We give the time complexity of *MCMR* for solving the top-k search. The straightforward way is to use Algorithm 1 to compute the estimated MR scores $\pi(q, v)$ of each node v, and then returns the top-k nodes with the highest estimated scores. By using *Bernstein Inequality* [13], Theorem 2 shows that Algorithm 1 returns the exact top-k set with high probability. Theorem 2 indicates that *MCMR* satisfies the output-bound requirement.

Theorem 2. *Given a k-NN graph $G(V, E)$, a query node q, failure probability p_{fail} and $n_r = \frac{10 \ln(1/p_{fail})}{3 \cdot \sqrt{2k} \cdot (gap_k)^2}$, Algorithm 1 returns the top-k set V_k such that for any node u in V_k, if $x^*_q(u) - x^*_q(v_{k+1}) \geq gap_k$, with probability at least $(1 - p_{fail})$, the following holds: $\pi(q, u) - \pi(q, v_{k+1}) > 0$, where v_{k+1} is the node whose exact MR score is the $(k + 1)$-th largest and $gap_k = x^*_q(v_k) - x^*_q(v_{k+1})$.*

Since the expected length of a single walk is $\frac{1}{1-\alpha}$, we have the time complexity as shown in Theorem 3.

Theorem 3. *Let $n_r \geq \frac{10 \ln(1/p_{fail})}{3 \cdot \sqrt{2k} \cdot (gap_k)^2}$ and $p_{fail} = \frac{1}{n}$,* MCMR *returns the top-k set in $O(\frac{\ln(n)}{(1-\alpha)\sqrt{2k}(gap_k)^2})$ time on expectation.*

Dataset	Name	k-NN	n	m	T (*Power*)
COIL	C5	5	7.2K	58.5K	50
	C10	10		111.1K	
	C15	15		164.9K	
	C20	20		216.7K	
NUS-WIDE	P5	5	269.7K	2.2M	1000
	P10	10		4.3M	
	P15	15		6.4M	
	P20	20		8.5M	
Flickr	F5	5	503.5K	4.0M	1000
	F10	10		7.9M	
	F15	15		11.7M	
	F20	20		15.3M	

TABLE II: Datasets. (K = 10^3, M=10^6.)

Since gap_k is not pre-known, in our experiments, we set a parameter c for tuning the number of walks to be simulated, i.e., $n_r = c \times \frac{10 \ln(1/p_{fail})}{3}$. Obviously, the more random walks are simulated, the longer query time of *MCMR* and the higher the precision of top-k results. Besides, if Chernoff Inequality [13] is applied, it is easy to show that the time complexity of *MCMR* is $O(n \log n)$ which is independent of gap_k (see [11] for details). Thus, *MCMR* satisfies the efficiency requirement.

V. Experiment

(A) Experimental Setting

Machine setting. All experiments were conducted on a machine with Intel(R) Xeon(R) E5-2650 @ 2.2GHz CPU and 500GB memory. We implemented our algorithms in C++.

Methods. We compared *MCMR* with three algorithms: *Mogul* [5], *Mogul-E* [5] and *Power* [34]. Note that we do not include other approximate algorithms, i.e., *EMR* and *FMR*, since *Mogul* outperforms them in terms of both efficiency and accuracy as shown in [5]. Because the source codes of *Mogul* and *Mogul-E* are not publicly available, we implement them strictly following the pseudocode in [5]. All of our source codes are publicly published. We set $\alpha = 0.99$, following previous work [6, 33, 34]. For *MCMR*, we set c to be 1000 for each dataset. The effect of constant c in *MCMR* is evaluated and the results can be found in our technical report [11] due to space limit.

Datasets. We used the four images datasets following [5]: (1) COIL, (2) NUS-WIDE, (3) Flickr, and (4) Pub-Fig. Due to space limit, more details about dataset and all experimental results on dataset Pub-Fig can be found in [11]. Among all existing studies mentioned in this paper, only [5] has image datasets of size larger than 100,000 images. Thus, we followed [5] to include all image datasets in [5]. Table II shows the statistics of each dataset. Besides, following [5], we used k from the set of $\{5, 10, 15, 20\}$.

Accuracy metric. For each dataset, we randomly selected 50 query nodes. The ground truth for each dataset was obtained by using *Power* which stops when the absolute error $|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}|$ dropped below 10^{-10} (see Table II for the number of iterations T required). Note that we could not directly compute the matrix inverse to obtain the ground truth since it ran out of memory for the large datasets. We evaluated the accuracy of the top-k search results of each method by using the classic metrics: Precision P@k, which is defined as $\frac{|V_k \cap V'_k|}{k}$, where V_k is the true top-k set and V'_k is the top-k set

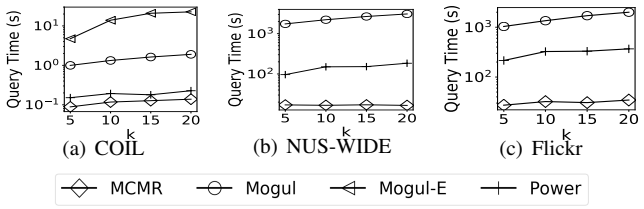

Fig. 2: Query time vs k on different datasets.

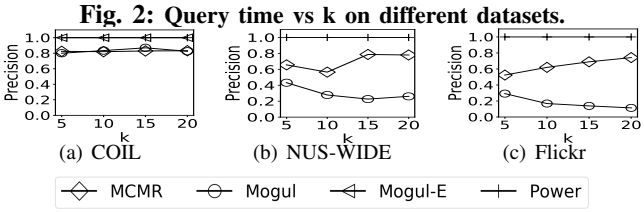

Fig. 3: Precision vs k on different datasets.

returned by each method.

(B) Efficiency of MCMR: In this section, we evaluated the efficiency of our proposed methods in the query phase. For each dataset, the average query time of each method was plotted in Fig. 2. We can see that *MCMR* are faster than *Mogul* and *Mogul-E* by 1 to 4 orders of magnitude, satisfying the efficiency requirement. Note that we did not include *Mogul-E* in Fig. 2(b) and Fig. 2(c) because the query time of *Mogul-E* on both datasets is very large (i.e., more than 4 days).

(C) Precision of MCMR: In this section, we evaluated the quality of the top-k results obtained by each method. Over 50 queries, the average precision of each method is shown in Fig. 3. Although *Mogul-E* can return the exact top-k results, its query time is slower than *MCMR* by up to 4 orders of magnitude. In addition, the precision of *MCMR* outperforms *Mogul* on all datasets. Especially on the largest dataset Flickr, the precision of *Mogul* is below 0.3 varying k. Because *Mogul* directly sets some entries in the factorized matrices to zero for achieving high efficiency, resulting in that the estimated MR scores are approximate with high errors.

VI. Conclusion

To efficiently address the top-k MRIR search, we propose a novel approach based on Random Walk Sampling. We prove the correctness of our new Random Walk Sampling strategy and its time complexity. There are possible future directions. One possible direction is to consider pruning some nodes which do not have high estimated MR scores in the algorithm for efficiency. Another possible direction is to consider how to perform MRIR on dynamic datasets.

Acknowledgement. We are grateful to the anonymous reviewers for their constructive comments on this paper. The research is supported by HKRGC GRF 14205117.

References

[1] S. Bai, X. Bai, and Q. Tian. Scalable person re-identification on supervised smoothed manifold. In *CVPR*, 2017.

[2] R. Datta, D. Joshi, J. Li, and J. Z Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Csur*, 2008.

[3] R. Datta, J. Li, and J. Z Wang. Content-based image retrieval: approaches and trends of the new age. In *SIGMM*, 2005.

[4] W. Dong, C. Moses, and K. Li. Efficient k-nearest neighbor graph construction for generic similarity measures. In *WWW*, 2011.

[5] Y. Fujiwara, G. Irie, S. Kuroyama, and M. Onizuka. Scaling manifold ranking based image retrieval. *PVLDB*, 2014.

[6] J. He, M. Li, H.J. Zhang, and H. Tong. Manifold-ranking based image retrieval. In *MM*, 2004.

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

[8] R. He, Y. Zhu, and W. Zhan. Fast manifold-ranking for content-based image retrieval. In *ISECS ICCCCM*, 2009.

[9] M. Jian, L. Wu, X. Zhang, and Y. He. Manifold ranking-based kernel propagation for saliency estimation. In *ICCAR*. IEEE, 2018.

[10] A. Krizhevsky, I. Sutskever, and G. E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[11] D. Lin, J. Wei, and R. C.-W. Wong. First index-free manifold ranking-based image retrieval with output bound- (technical report). In *http://www.cse.ust.hk/~raywong/paper/MRIR-technical.pdf*, 2019.

[12] C. C. Loy, C. Liu, and S. Gong. Person re-identification by manifold ranking. In *ICIP*, 2013.

[13] R Motwani and P Raghavan. Chap. randomized algorithms, 2010.

[14] W. Qi, M.M. Cheng, A. Borji, H. Lu, and L.F. Bai. Saliencyrank: Two-stage manifold ranking for salient object detection. *Computational Visual Media*, 2015.

[15] R. Quan, J. Han, D. Zhang, and F. Nie. Object co-segmentation via graph optimized-flexible manifold ranking. In *CVPR*, 2016.

[16] F. Radenović, G. Tolias, and O. Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In *ECCV*. Springer, 2016.

[17] F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *TPAMI*, 2018.

[18] L. Shen, Q. Cao, D.and Xu, X. Huang, N. Xiao, and Y. Liang. A novel local manifold-ranking based k-nn for modeling the regression between bioactivity and molecular descriptors. *Chemometrics and Intelligent Laboratory Systems*, 2016.

[19] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[20] D. Tao, J. Cheng, M. Song, and X. Lin. Manifold ranking-based matrix factorization for saliency detection. *TNNLS*, 2016.

[21] J. Wan, D. Wang, S. Chu Hong Hoi, J. Wu, P.and Zhu, Y. Zhang, and J. Li. Deep learning for content-based image retrieval: A comprehensive study. In *MM*. ACM, 2014.

[22] X. Wan, J. Yang, and J. Xiao. Towards a unified approach to document similarity search using manifold-ranking of blocks. *Information Processing & Management*, 2008.

[23] H. Wang, Y. Cai, Y. Zhang, H. Pan, and and.Han H. Lv, W. Deep learning for image retrieval: what works and what doesn't. In *ICDMW*. IEEE, 2015.

[24] J. Wang, P. Huang, H. Zhao, Z. Zhang, B. Zhao, and D. L. Lee. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *SIGKDD*. ACM, 2018.

[25] Q. Wang, J. Lin, and Y. Yuan. Salient band selection for hyperspectral image classification via manifold ranking. *TNNLS*, 2016.

[26] Z. Wang, J. Chen, and S. CH Hoi. Deep learning for image super-resolution: A survey. *arXiv*, 2019.

[27] Z. Wei, X. He, X. Xiao, S. Wang, S. Shang, and J.R. Wen. Topppr: top-k personalized pagerank queries with precision guarantees on large graphs. In *SIGMOD*. ACM, 2018.

[28] J. Wu, Y. He, X. Guo, Y. Zhang, and N. Zhao. Heterogeneous manifold ranking for image retrieval. *IEEE Access*, 2017.

[29] B. Xu, J. Bu, C. Chen, D. Cai, X. He, W. Liu, and J. Luo. Efficient manifold ranking for image retrieval. In *SIGIR*. ACM, 2011.

[30] S. Lin H. Xie M. Lv Y. Xu J. C.S. Lui Y. Li, Z. Wu. Walking with perception: Efficient random walk sampling via common neighbor awareness. *IEEE ICDE*, 2019.

[31] Y. Zhang, P. Pan, Y. Zheng, K. Zhao, Y. Zhang, X. Ren, and R. Jin. Visual search at alibaba. In *SIGKDD*. ACM, 2018.

[32] L. Zheng, Y. Yang, and Q. Tian. Sift meets cnn: A decade survey of instance retrieval. *PAMI*, 2018.

[33] D. Zhou, O. Bousquet, Thomas N Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, 2004.

[34] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2004.

[35] W. Zhou, H. Li, and Q. Tian. Recent advance in content-based image retrieval: A literature survey. *arXiv preprint*, 2017.