Pattern Recognition Letters 114 (2018) 41–52

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Unsupervised similarity learning through Cartesian product of ranking references

Lucas Pascotti Valem[a], Daniel Carlos Guimarães Pedronette[a,*], Jurandy Almeida[b]

[a] Department of Statistics, Applied Mathematics and Computing, State University of São Paulo (UNESP), Av. 24-A, 1515, Rio Claro, SP 13506-900, Brazil
[b] Institute of Science and Technology, Federal University of São Paulo (UNIFESP), Av. Cesare M. G. Lattes, 1201, São José dos Campos, SP 12247-014, Brazil

A R T I C L E I N F O

Article history:
Available online 16 October 2017

Keywords:
Content-based image retrieval
Unsupervised learning
Cartesian product
Effectiveness
Efficiency

A B S T R A C T

Despite the consistent advances in visual features and other Multimedia Information Retrieval (MIR) techniques, measuring the similarity among multimedia objects is still a challenging task for an effective retrieval. In this scenario, similarity learning approaches capable of improving the effectiveness of retrieval in an unsupervised way are indispensable. A novel method, called Cartesian Product of Ranking References (CPRR), is proposed with this objective in this paper. The proposed method uses Cartesian product operations based on rank information for exploiting the underlying structure of datasets. Only subsets of ranked lists are required, demanding low computational efforts. An extensive experimental evaluation was conducted considering various aspects, seven public multimedia datasets (images and videos) and several different features. Besides effectiveness, experiments were also conducted to assess the efficiency of the method, considering parallel and heterogeneous computing on CPU and GPU devices. The proposed method achieved significant effectiveness gains, including competitive state-of-the-art results on popular benchmarks.

1. Introduction

Over the last years, the availabilty of multimedia data has exponentially increased due to the rapid evolution of technologies for acquisition and sharing multimedia content. As a consequence, significant amounts of multimedia material from a wide variety of sources are being accumulated daily [1], enabling the creation of huge multimedia collections. In this scenario, multimedia information retrieval (MIR) systems have emerged as a promising solution for searching and indexing multimedia data by their content.

MIR systems are currently an essential tool for supporting many applications. In general, these systems rely on the use of several features for encoding multimedia content into feature vectors and a proper distance measure for assessing the similarity among multimedia objects based on their corresponding feature vectors. In spite of all the advances in the development of multimedia retrieval approaches, effectively measuring the similarity among multimedia objects remains a challenging task. This has motivated many research efforts on other stages of the retrieval process not directly related to feature extraction procedures [2].

Many post-processing methods have been proposed to improve the effectiveness of multimedia retrieval tasks in an unsupervised fashion [3–5]. In general, such methods aims to replace pairwise distances by more global affinity measures capable of considering the dataset structure [4]. Although effective, approaches based on diffusion processes [3] and graphs [6] often require high computational efforts, thus they are not suitable for several applications.

As a promising alternative, rank-based approaches have attracted a lot of attention due to their capacity to take into account both effectiveness and efficiency aspects. The analysis of rankings can provide a rich and reliable source of information for context-based measures, as demonstrated in several recent studies [7–9]. Different models can be exploited to analyze the rank information, such as similarity of ranked lists [7] or sets [8], rank-based recommendations [9], and rank consistency verifications [10]. In general, rank-based approaches are computationally efficient, since the required computational efforts can be substantially reduced by considering only the most relevant information, which is located at top positions of the ranked lists. In addition, they model the similarity information using an uniform representation, which does not depend on the distance measures used for comparing multimedia objects.

In this paper, we present a novel unsupervised similarity learning method for improving the effectiveness of multimedia retrieval tasks, named as Cartesian product of ranking references (CPRR).

* Corresponding author.
E-mail addresses: lucasvalem@rc.unesp.br (L.P. Valem), daniel@rc.unesp.br (D.C.G. Pedronette), jurandy.almeida@unifesp.br (J. Almeida).

https://doi.org/10.1016/j.patrec.2017.10.013
0167-8655/© 2017 Elsevier B.V. All rights reserved.

The underlying idea of the proposed method is to maximize the similarity information encoded in rankings through Cartesian product operations. While the CPRR algorithm only considers a subset of ranked lists for reducing computational costs, the Cartesian product is used for expanding the similarity relationships. For that, kNN and reverse kNN queries are used for computing sets of multimedia objects, which are used for Cartesian product operations. To the best of our knowledge, this is the first unsupervised similarity learning approach which models the rank information in terms of Cartesian product of neighborhood and reverse neighborhood sets. In addition, the proposed method can be used in rank aggregation tasks and can be efficiently computed through parallel computing.

An extensive experimental evaluation was conducted, considering various aspects and several different retrieval scenarios. The experiments were conducted on seven multimedia datasets, including both images and videos. Several descriptors were also considered, from traditional global to deep-learning based features. Experimental results confirm the effectiveness of the proposed method, consistently improving the retrieval precision and achieving relative gains up to +32.57%. Besides the effectiveness, the efficiency and scalability of the proposed method were also evaluated. Experiments conducted in parallel and heterogeneous environments (CPUs and GPUs) demonstrated that the algorithm presents very small run times for multimedia collections of different sizes. The CPRR algorithm also compares favorably with recent retrieval methods and state-of-the-art approaches, considering both effectiveness and efficiency aspects. The algorithm achieves a N-S score of 3.94 on the popular UKBench [11] dataset.

This work differs from a previously published conference paper [12] in several aspects, including novel contributions and a substantive extension of the experimental evaluation. A significant contribution consists in the proposal of an approximate version of the CPRR algorithm. In this way, the algorithm can be used in situations where the multimedia objects used as queries are not part of the dataset. Other relevant contribution consists in the estimation of an appropriate neighborhood size. Most of the post-processing methods require a k-neighborhood size definition, which is commonly determined empirically. In this paper, an estimation of k is proposed based on the variation of reciprocal rank references. Additionally, a novel exponential function is proposed for assigning weights to top ranking references on Cartesian product operations.

The remainder of this paper is organized as follows. Section 2 describes the multimedia retrieval model considered. Section 3 presents the proposed algorithm. Section 4 discusses the experimental evaluation. Finally, Section 5 draws our conclusions and presents future work.

2. Problem formulation

The retrieval notation used along the paper is formally defined in this section. Let $\mathcal{C} = \{o_1, o_2, \ldots, o_n\}$ be an multimedia collection, where n denotes the size of the collection. Let ρ: $\mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$ be a similarity function, such that $\rho(o_i, o_j)$ denotes the similarity between two multimedia objects $o_i, o_j \in \mathcal{C}$. For simplicity and readability purposes, the notation $\rho(i, j)$ is used in the remainder of the paper.

The similarity among all multimedia objects $o_i, o_j \in \mathcal{C}$ defined by the function $\rho(i, j)$ can be applied for computing an affinity matrix W. The matrix W, in turn, is commonly used as an adjacency matrix by various graph and diffusion-based methods. However, this approach often leads to storage and time complexity of at least $O(n^2)$, so that scalability and efficiency requirements are not met for large collections.

A rank-based modeling of similarity information represents an effective and efficient solution in this scenario. Different from similarity functions which establish relationships only between pairs of objects, the ranked lists encode similarity information among a query and all other collection objects. In addition, although a ranked list can encode information from the entire collection, the most similar objects are expected to be located at top positions. Therefore, a constant $L \ll n$ can be used such that only a subset composed of top-L positions of the ranked list is considered, reducing the computational efforts required.

Formally, the ranked list $\tau_q = (o_1, o_2, \ldots, o_L)$ can be defined as a permutation of the collection $\mathcal{C}_L \subset \mathcal{C}$, which contains the most similar objects to the query o_q, such that and $|\mathcal{C}_L| = L$. A permutation τ_q is a bijection from the set $\mathcal{C}_L$ onto the set $[L] = \{1, 2, \ldots, L\}$. The notation $\tau_q(i)$ can be interpreted as the position (or rank) of object o_i in the ranked list τ_q. If o_i is ranked before o_j in the ranked list of o_q, that is, $\tau_q(i) < \tau_q(j)$, then $\rho(q, i) \geq \rho(q, j)$.

Every multimedia object in the collection can be taken as a query object o_q and a respective ranked list can be computed. In this way, the set of ranked lists $\{\tau_1, \tau_2, \ldots, \tau_n\}$ provides a compact and effective rank-based modeling of similarity information. In this work, an unsupervised method is proposed aiming at exploiting the information encoded in the set of ranked lists for computing new and more effective retrieval results.

3. Cartesian Product of Ranking References (CPRR)

The rank analysis has been established as a rich and reliable source of information for context-based measures. The main objective of the proposed Cartesian product of ranking references (CPRR) is to maximize the available rank information through the use of Cartesian product operations. The Cartesian product over neighborhood sets establishes new pairwise relationships, which are used to discover underlying similarity information. The proposed approach can be broadly divided in two main steps:

- **Rank normalization:** the reciprocal rank references are analyzed aiming at improving the symmetry of neighborhoods and, consequently, the effectiveness of the ranked lists;
- **Cartesian product of ranking references:** the Cartesian product is computed considering the top-k neighborhood and the reverse neighborhood sets. The obtained results are used to define an iterative similarity measure.

Each step of the proposed approach is discussed in the following subsections.

3.1. Rank similarity score

This section defines a rank similarity score, which is used for both rank normalization and Cartesian product procedures. Since the most relevant information about similarity is encoded at top positions of ranked lists, neighborhood sets can be defined at different depths. When considering a given depth d, only the similarity information of top-d ranked objects is considered, avoiding noisy information contained in the remainder of ranked lists.

Aiming at considering only the top-d most similar multimedia objects, a neighborhood set $\mathcal{N}$ is defined. Let d denote the depth of ranked lists considered, and therefore the size of the neighborhood set. Let $\mathcal{N}(i, d)$ be the neighborhood set, which is formally defined as follows:

$$\mathcal{N}(q, d) = \{\mathcal{R} \subseteq \mathcal{C}, |\mathcal{R}| = d \wedge \forall x \in \mathcal{R}, y \in \mathcal{C} - \mathcal{R} : \rho(q, x) \geq \rho(q, y)\}. \quad (1)$$

Taking into account the neighborhood set, a similarity score is defined based on rank information. The rank similarity score $r_d(q, i)$ represents the similarity between multimedia objects o_q and o_i based on the ranked list of o_q analyzed until a given depth d. In this work, two different depths are considered: L, which defines a broader neighborhood used by the rank normalization step; and

k, which defines a local neighborhood used by Cartesian product operations. Once both k and L values are much smaller than n, a sparse matrix structure [9] can be used for storage of similarity scores.

The rank similarity score $r_d(q, i)$ can assume different forms, according to the weight assigned to the position of o_i in the ranked list τ_q. Two different functions are presented: linear and exponential, discussed in the following.

3.1.1. Linear

The weight assigned is proportional to the position of o_i in the ranked list τ_q considering a depth d. The score ranges linearly from k (assigned to the first position) to 1 (assigned to the dth position). The linear score is defined as:

$$r_d(q,i) = \begin{cases} d - \tau_q(i) + 1, & \text{if } o_i \in \mathcal{N}(q,d) \\ 0, & \text{otherwise}. \end{cases} \tag{2}$$

3.1.2. Exponential

In order to make the method more flexible to different retrieval scenarios (with more or less relevant items at top positions), other functions can be used for adjusting the weight of rank positions in the similarity score. With this purpose, a exponential function is proposed, as follows:

$$r_d(q,i) = \begin{cases} (1+c)^{(d-\tau_q(i)+1)}, & \text{if } o_i \in \mathcal{N}(q,d) \\ 0, & \text{otherwise}. \end{cases} \tag{3}$$

where $c > 0$ is a constant which adjusts the weight of top positions. The higher the value of c, the higher the weight assigned to the top positions. In this work, the value of c is set to 0.1.

3.2. Reciprocal Rank Normalization

Different from most of distance or similarity measures, the k-neighborhood relationships and rank measures are not symmetric. However, the benefits of improving the symmetry of the k-neighborhood relationship are remarkable in multimedia retrieval applications [13]. In this way, various approaches have been proposed for exploiting the reciprocal neighborhood [14].

In this work, a simple rank normalization based on the reciprocal neighborhood is employed as a pre-processing step for the Cartesian product operations. A normalized similarity function $\rho_r(i, j)$ is defined as the sum of reciprocal rank similarity score at a depth L:

$$\rho_r(i,j) = r_L(i,j) + r_L(j,i). \tag{4}$$

The linear weight function is used for the rank normalization. In the following, all the ranked lists are updated according to the similarity function, using a stable sorting algorithm. The update gives rise to a new set of ranked lists, which is used as input for the Cartesian product operations. Notice that a low computational cost algorithm can be derived for computing the rank normalization procedure, once only the top-L positions of ranked lists are considered.

3.3. Cartesian product of neighborhood sets

The ranked lists and the neighborhood sets represent a relevant source of information for context-based similarity functions. While a pairwise measure defines a similarity relation between only two objects, a ranked list establishes a broader relationship among a query and its most similar multimedia objects. Additionally, the neighborhood and rank analysis can be exploited for discovering underlying similarity information among neighbors of a same query object.

In this way, Cartesian product operations of neighborhood sets and rank information are employed for computing a new and more effective similarity measure. The objective is to consider the pairwise relations computed by the Cartesian product weighted by rank information.

The Cartesian product can be defined as the set of all possible pairs of elements whose components are members of two sets. Let $\mathcal{N}(i,k)$ and $\mathcal{N}(j,k)$ be k-neighborhood sets of multimedia objects o_i and o_j, respectively. The Cartesian product of $\mathcal{N}(i,k)$ and $\mathcal{N}(j,k)$ is defined as:

$$\mathcal{N}(i,k) \times \mathcal{N}(j,k) = \{(n_i, n_j) \mid n_i \in \mathcal{N}(i,k),\ n_j \in \mathcal{N}(j,k)\}. \tag{5}$$

Given a query object o_q and its respective neighborhood set $\mathcal{N}(q,k)$, we denote $\mathcal{N}(q,k) \times \mathcal{N}(q,k) = \mathcal{N}(q,k)^2$. The pairs of objects contained in $\mathcal{N}(q,k)^2$ define all possible similarity relationships among the neighbors of o_q. This information is exploited for computing a new similarity score $wc(i,j)$, between o_i and o_j, with $o_i, o_j \in \mathcal{N}(q,k)$.

The similarity score $wc(i,j)$ is computed considering every query object $o_q \in \mathcal{C}$ that has o_i and o_j as neighbors. In addition, the score is weighted according to their rank score, by the term $r_k(q, i) \times r_k(q, j)$. Formally, the score $wc(i, j)$ can be defined as:

$$wc(i,j) = \sum_{q \in \mathcal{C} \wedge (i,j) \in \mathcal{N}(q,k)^2} r_k(q,i) \times r_k(q,j). \tag{6}$$

Algorithmically, the similarity score can be computed with complexity of only $O(n)$, once k is a constant. Algorithm 1 outlines our proposed approach. The main idea consists in performing only top-k rank analysis for computing the Cartesian product for each neighborhood set.

Algorithm 1 Cartesian product of neighborhood sets.

```
Require: Set of ranked lists $\mathcal{R}$
Ensure: Similarity score $wc(\cdot,\cdot)$
1: $wc(\cdot,\cdot) \leftarrow 0$
2: for all $o_q \in \mathcal{C}$ do
3:     for all $o_i \in \mathcal{N}(q,k)$ do
4:         for all $o_j \in \mathcal{N}(q,k)$ do
5:             $wc(i,j) \leftarrow wc(i,j) + r_k(q,i) \times r_k(q,j)$
6:             $wc(j,i) \leftarrow wc(j,i) + r_k(q,i) \times r_k(q,j)$
7:         end for
8:     end for
9: end for
```

3.4. Cartesian product of reverse neighborhood sets

The Cartesian product of ranked lists defines similarity relationships among neighbors of the same query object. On the other hand, information from different queries with a common neighbor is ignored. In other words, the set of multimedia objects that have an item among its neighbors also encodes a relevant similarity information, which can be exploited for improving the effectiveness of retrieval.

With this objective, the Cartesian product of reverse neighborhood sets is considered. Let $\mathcal{N}_r(x)$ be a reverse neighborhood set computed for an object o_x, which is compose by all objects whose neighborhood set contains o_x. Formally, the set $\mathcal{N}_r(x)$ can be defined as follows:

$$\mathcal{N}_r(x) = \{\mathcal{R} \subseteq \mathcal{C}, \forall q \in \mathcal{R} : o_x \in \mathcal{N}(q,k)\}. \tag{7}$$

The Cartesian product of the reverse neighborhood set $\mathcal{N}_r(i)^2$ is used for analysing underlying similarity information. In this way, a similarity score $wr(i, j)$ is defined for increasing the similarity between any given objects o_i and o_j contained in the reverse neighborhood sets. Formally, the score is defined as follows:

$$wr(i, j) = \sum_{x \in \mathcal{C} \wedge (i,j) \in \mathcal{N}_r(x)^2} r_k(i, x) \times r_k(j, x). \tag{8}$$

An algorithmic solution for computing the similarity score based on reverse neighborhood is presented by the Algorithm 2. The reverse neighborhood sets are computed on lines 1-5, while the Cartesian product is computed on lines 6-13.

Algorithm 2 Cartesian product of reverse neighborhood sets.

```
Require: Set of ranked lists $\mathcal{R}$
Ensure: Similarity score $wr(\cdot, \cdot)$
1:  $wr(\cdot, \cdot) \leftarrow 0$
2:  for all $o_q \in \mathcal{C}$ do
3:      for all $o_x \in \mathcal{N}(q, k)$ do
4:          $\mathcal{N}_r(x) \leftarrow \mathcal{N}_r(x) \cup o_q$
5:      end for
6:  end for
7:  for all $o_x \in \mathcal{C}$ do
8:      for all $o_i \in \mathcal{N}_r(x)$ do
9:          for all $o_j \in \mathcal{N}_r(x)$ do
10:             $wr(i, j) \leftarrow wr(i, j) + r_k(i, x) \times r_k(j, x)$
11:             $wr(j, i) \leftarrow wr(j, i) + r_k(i, x) \times r_k(j, x)$
12:         end for
13:     end for
14: end for
```

3.5. Iterative similarity measure

The similarity scores based on the Cartesian product of neighborhood and reverse neighborhood sets are used for computing a new and iterative similarity measure. Let the superscript $^{(t)}$ denote the current iteration, the similarity measure $\rho^{(t+1)}$ can be defined as:

$$\rho^{(t+1)} = wc(i, j) + wr(i, j). \tag{9}$$

The similarity measure $\rho^{(t+1)}$ is used as input for a sorting step, which gives rise to a new set of ranked lists. Once the input of the algorithm is also a set of ranked lists, it can be iteratively executed until a certain number of T iterations.

Only the Cartesian product operations are considered for the iterative measure, e.g., the rank normalization is executed only before the first iteration. The method also requires a very small number of iterations for reaching high effectiveness results (as discussed in Section 4).

3.6. Rank aggregation

Different features often provide distinct information about images, videos, and other multimedia objects. In this scenario, different rankings computed for each feature also encode distinct and complementary information. In fact, most of recent retrieval approaches commonly consider various features [15]. Our goal is to use the proposed CPRR algorithm for rank aggregation tasks, aiming at combining rank information computed for different features.

Since the most significant effectiveness gains are obtained at the first iteration, the CPRR algorithm is computed independently for each descriptor considering one iteration. Let $\rho_a^{(1)}$ be the similarity measure computed at the first iteration for a given feature a. Let a be defined in the interval [1, m], where m denotes the number of features considered. The combined similarity measure is computed as follows:

$$\rho^{(1)}(i, j) = \sum_{a=1}^{m} \rho_a^{(1)}(i, j). \tag{10}$$

Based on the similarity score, a new set of ranked lists is generated. Subsequently, the Cartesian product procedures are executed for the combined similarity measures along T iterations. Once all the steps are finished, a final sorting is performed.

3.7. Automatic neighborhood size estimation

Although completely unsupervised, the CPRR algorithm requires the definition of parameters k, which denotes the local neighborhood size, and T, which defines the number of iterations. The algorithm converges very quickly, reaching the best effectiveness results for only two iterations ($T = 2$), as discussed in the experimental evaluation (Section 4). The method is also robust regarding the neighborhood size k, admitting a fixed parameter for most of datasets with high effectiveness results.

However, the neighborhood size can be optimized for each dataset. In fact, several different post-processing approaches [3,7,16] require a similar parameter, which is empirically determined and can assume different values for each dataset [16]. On the other hand, once unsupervised scenarios imply independence of labeled data, human intervention should be avoided. With this purpose, an automatic neighborhood size estimation is proposed.

The essence of the proposed approach consists in analyzing the variation of occurrence of reciprocal neighbors at top positions of ranked lists. The conjecture is that abrupt decreases in the number of reciprocal neighbors indicate that the number of relevant objects is declining fast. Therefore, such variations can represent an effective estimation of the neighborhood size.

Algorithm 3 outlines the proposed method, which searches for an estimation inside an interval [$k_{\text{min}}, k_{\text{max}}$]. First, the algorithm ranges k from 1 to a given k_{max} (lines 3-6), verifying if the object at the kth position is a reciprocal neighbor (line 5). Next, the algorithm computes the variation obtained in the interval (lines 8-10) and selects the estimated k_e through the maximum variation (line 11).

Algorithm 3 Automatic neighborhood size estimation

```
Require: Set of ranked lists $\mathcal{R}$, $k_{min}$, $k_{max}$
Ensure: Estimated $k_e$
1:  $v[k_{min}..k_{max}] = 0$
2:  for all $o_q \in \mathcal{C}$ do
3:      for all $o_i \in \mathcal{N}(q, k_{max})$ do
4:          $k = \tau_q(i)$
5:          $v[k] = v[k] + f_r(q, i, k_{max})$
6:      end for
7:  end for
8:  for $k \in \{k_{min}, ..., k_{max} - 1\}$ do
9:      $d[k] = v[k+1] - v[k]$
10: end for
11: $k_e = argmax(d[k_{min}..k_{max} - 1])$
```

The function f_r (line 5), which verifies if two objects are reciprocal neighbors, also assigns a weight according to the position, as follows:

$$f_r(q, i, k) = \frac{|\mathcal{N}(q, k) \cap \mathcal{N}(i, k) \cap \{o_q, o_i\}|}{2} \times log(k - \tau_i(q) + 1). \tag{11}$$

3.8. Approximate algorithm

The traditional scenario, commonly considered by post-processing methods [6,17], assumes full computed datasets and off-line processing [6]. However, in many situations it is desired to run such methods with on-line response. For example, when the user needs to perform retrieval tasks based on a query object outside of the dataset.

In order to use the CPRR method for an out-of-dataset object, one possible solution is to consider the new multimedia object as part of the dataset. Firstly, a ranked list should be computed for the new object. Next, it is necessary to update the other ranked lists in order to include the new object and, finally, the CPRR algorithm can be performed. In fact, the CPRR algorithm is very efficient and can be fully executed even for on-line response (as showed by conducted experiments detailed in Section 4).

However, despite of the efficiency of the CPRR method, the algorithm still depends on the collection size (asymptotically $O(n)$). Therefore, it is desired to reduce the computational cost for out-of-dataset objects, especially for very large datasets. With this purpose, an approximate version of the algorithm is proposed. While the full algorithm considers the Cartesian product of all ranked lists, the proposed approximation selects only a small subset of ranked lists to be processed. Given a query object out-of-dataset, the most relevant items are at the top positions of τ_o. Therefore, only the ranked lists of objects at top-L positions are processed.

Algorithm 4 outlines the proposed approximate algorithm. Notice the similarity to Algorithm 1, except by line 1, which limits the Cartesian product computations to the top-L positions.

Algorithm 4 Approximate Cartesian product of neighborhood sets.

```
Require: Set of Ranked Lists for top-$L$ of $\tau_o$
Ensure: Similarity score $wc(\cdot,\cdot)$
1: $wc(\cdot,\cdot) \leftarrow 0$
2: for all $o_q \in \mathcal{N}(o, L)$ do
3:     for all $o_i \in \mathcal{N}(q, k)$ do
4:         for all $o_j \in \mathcal{N}(q, k)$ do
5:             $wc(i, j) \leftarrow wc(i, j) + r_k(q, i) \times r_k(q, j)$
6:             $wc(j, i) \leftarrow wc(j, i) + r_k(q, i) \times r_k(q, j)$
7:         end for
8:     end for
9: end for
```

An analogous approach can be applied to the reverse neighborhood sets. Algorithm 5 presents the approximate version of the Cartesian product of reverse neighborhood sets. Lines 1 and 6 reduce the loops from the full dataset for only top-L positions of τ_o.

Algorithm 5 Approximate Cartesian product of reverse neighborhood sets.

```
Require: Set of ranked lists for top-$L$ of $\tau_o$
Ensure: Similarity score $wr(\cdot,\cdot)$
1: $wr(\cdot,\cdot) \leftarrow 0$
2: for all $o_q \in \mathcal{N}(o, L)$ do
3:     for all $o_x \in \mathcal{N}(q, k)$ do
4:         $\mathcal{N}_r(x) \leftarrow \mathcal{N}_r(x) \cup o_q$
5:     end for
6: end for
7: for all $o_x \in \mathcal{N}(o, L)$ do
8:     for all $o_i \in \mathcal{N}_r(x)$ do
9:         for all $o_j \in \mathcal{N}_r(x)$ do
10:            $wr(i, j) \leftarrow wr(i, j) + r_k(i, x) \times r_k(j, x)$
11:            $wr(j, i) \leftarrow wr(j, i) + r_k(i, x) \times r_k(j, x)$
12:        end for
13:    end for
14: end for
```

3.9. Parallel design

The CPRR algorithm can be widely parallelized, specially regarding its Cartesian product operations. This section discusses the parallel design of the CPRR algorithm, using the OpenCL standard. The OpenCL is a low-level API for task-parallel and data-parallel heterogeneous computing. A *kernel* is the name given for pieces of code that can be executed in parallel. Each kernel is executed in parallel by a given number of *work-items*.

The parallel design of the CPRR algorithm is illustrated in Fig. 1. Each main step of the algorithm defines a different kernel, which runs in an OpenCL device (CPU or GPU). Each kernel, in turn, is parallelized in n work-items. Two transfer models were used: *Writer Buffer*, which requires the transfer of the data to the device memory; and *Map Buffer*, which requires only the transfer of data pointers.

Since the Cartesian product operations are processed in parallel and the similarity measure is stored using a global memory, concurrent accesses can cause loss of updates. However, the overhead associated to atomic operations in OpenCL is high. Therefore, direct updates of similarity score are allowed due to the very low impact on the effectiveness of the algorithm (as discussed in the experimental section).

In the end of each iteration, a sorting procedure is executed to update the top positions of ranked lists according to the new similarity measure. The insertion sort algorithm is used, once it tends to be linear when input is almost sorted.

4. Experimental evaluation

Several aspects are considered to assess the effectiveness and efficiency of the proposed method. Section 4.1 describes the experimental setup. Section 4.2 discusses the algorithm settings. Section 4.3 presents the results of the effectiveness evaluation, while Section 4.4 presents the efficiency evaluation. Section 4.5 discusses the evaluation of algorithm variances. Section 4.6 presents a comparison of the proposed method with other state-of-the-art unsupervised learning methods and recent retrieval approaches.

4.1. Experimental setup

Seven distinct and public datasets are considered in the experiments, including both image and video collections with size ranging from 280 to 87,648 multimedia objects. In order to exploit the different dataset characteristics, several global descriptors are used, considering shape, color, and texture properties. Convolution Neural Network features are extracted using the Caffe framework [33]. CaffeNet was trained to recognize 1000 object categories of ImageNet [52] and is very similar to the AlexNet [53], except that no data-augmentation was used during training and the order of pooling and normalization is switched. From CaffeNet, we used the network definitions provided by Caffe framework [33] for feature extraction,[1] considering the features from the fully connected layers (FC6, FC7, FC8). The input images were resized to 256 × 256 pixels and the feature vectors have 4096 dimensions. Features were considered in the Euclidean space (L2 distance function). Regarding local descriptors, SIFT [38] features are used considering a variant of vocabulary tree based retrieval (VOC) [39]. The datasets and descriptors are briefly described in Table 1.

[1] https://github.com/BVLC/caffe/blob/master/examples/feature_extraction/imagenet_val.proto
(As of August, 2017)

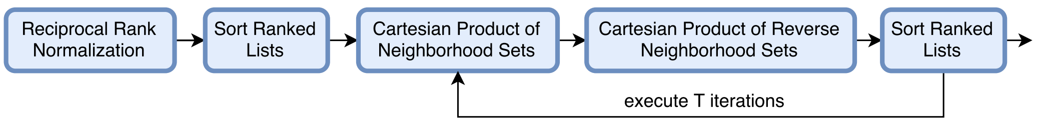
Reciprocal Rank Normalization
Sort Ranked Lists
Cartesian Product of Neighborhood Sets
Cartesian Product of Reverse Neighborhood Sets
Sort Ranked Lists
execute T iterations

Fig. 1. Design of the parallel CPRR algorithm.

Table 1
Datasets and images descriptors used in the experimental evaluation.

Dataset	Size	Type	General Descriptors	Effectiv. Measure	
Soccer [18]	280	Images: color scenes	Dataset composed of images from 7 soccer teams, containing 40 images per class.	Border/Interior Auto Color Correlograms (ACC) [19], Pixel Classification (BIC) [20], and Global Color Histogram (GCH) [21]	MAP
MPEG-7 [22]	1,400	Images: shape	Composed of 1400 shapes divided in 70 classes. Commonly used for evaluation of post-processing methods.	Articulation-Invariant Representation (AIR) [23], Aspect Shape Context (ASC) [24], Beam Angle Statistics (BAS) [25], Contour Features Descriptor (CFD) [26], Shape Context (IDSC) [27], and Segment Saliences (SS) [28]	MAP, Recall@40
Brodatz [29]	1,776	Images: texture	A popular dataset composed of 111 different textures divided into 16 blocks.	Color Co-Occurrence Matrix (CCOM) [30], Local Activity Spectrum (LAS) [31], and Local Binary Patterns (LBP) [32]	MAP
UKBench [11]	10,200	Images: objects/ scenes	Composed of 2,550 objects or scenes. Each object/scene is captured 4 times from different viewpoints, distances, and illumination conditions.	ACC [19], BIC [20], Convolutional Neural Network by Caffe [33] framework (CNN-Caffe) Color and Edge Directivity Descriptor (CEED) [34], Fuzzy Color and Texture Histogram (FCTH) [35], FCTH Spatial Pyramid (FCTH-SPy) [35,36], Joint Composite Descriptor (JCD) [37], Scale-Invariant Feature Transform (SIFT) [38], and Vocabulary Tree (VOC) [39]	MAP, N-S Score
ALOI [40]	72,000	Images: objects	Images from 1,000 classes of objects, with different viewpoint and illumination.	ACC [19], BIC [20], GCH [21], Color Coherence Vectors (CCV) [41], Local Color Histograms (LCH) [42]	MAP
MediaEval [43]	14,838	Videos	A total of 3,288 hours of video collected from blip.tv for the Video Genre Tagging Task at the MedialEval 2012. These videos are distributed among 26 genre categories.	Bag-of-Scenes (BoS) [44], Histogram of Motion Patterns (HMP) [45], Pooling over Pooling (PoP) [46]	MAP
FCVID [47]	87,648	Videos	A total of 4,232 hours of video collected from YouTube and annotated manually according to 233 categories.	Convolutional Neural Network (CNN) [48], Improved Dense Trajectories (IDT) [49], Mel-Frequency Cepstral Coefficients (MFCC) [50], Scale-Invariant Feature Transform (SIFT) [38]. Four descriptors were computed for each trajectory obtained by IDT: Histogram of Oriented Gradients (IDT-HOG), Histogram of Optical Flow (IDT-HOF), Motion Boundary Histogram (IDT-MBH), and Trajectory Shape Descriptor (IDT-TRAJ). The SIFT, IDT and MFCC features were quantized using a Bag-of-Words (BoW) representation.	MAP

The effectiveness evaluation considers all multimedia objects of each dataset as query objects. As effectiveness measure, the Mean Average Precision (MAP) is used for most of the datasets, except for the UKBench [11] dataset which uses the N-S Score. For the MPEG-7 [22] dataset, the Recall@40 is also considered in addition to MAP. Most of the experiments also report the effectiveness gains: let S_b and S_a be the effectiveness scores before and after the algorithm execution, the gain is defined as $(S_a - S_b)/S_b$.

For the efficiency evaluation experiments, the average run time of 10 executions and 95% confidence intervals are considered. The hardware environment is composed of an Intel Xeon E3-1240 CPU and an AMD Radeon HD 7900 GPU. The software environment is given by the operating system Linux 3.11.0-15 - Ubuntu 12.04 and OpenCL 1.2 AMD-APP. The code was compiled using g++ 4.6.3 with the flag -O3. The full algorithm with fixed neighborhood size and the linear function is considered for the experiments, except for the Section 4.5, which discusses the other variances.

4.2. Impact of parameters

Two parameters are considered in the proposed algorithm: *(i)* k: the number of nearest neighbors; and *(ii)* T: the number of iterations. Additionally, the constant L defines a trade-off between effectiveness and efficiency. A set of experiments were conducted for evaluating the impact of different parameter settings on the retrieval scores.

The first experiment aims at analyzing the impact of different combination of parameters k and T. The MAP scores are computed ranging the parameter k in the interval [0, 30] and the parameter T from 1 to 5, considering the MPEG-7 [22] dataset and the CFD [26] shape descriptor. Fig. 2 shows the variation of MAP according to k and T. The joined growth of k and MAP scores can be observed until a stabilization, with values of k near 20. For the parameter T, the most significant effectiveness gains are obtained for the first iteration. Considering these results, the parameter values of $k = 20$ and $T = 2$ were used for all of the remaining experiments, except only for the UKBench [11] dataset, which used $k = 4$

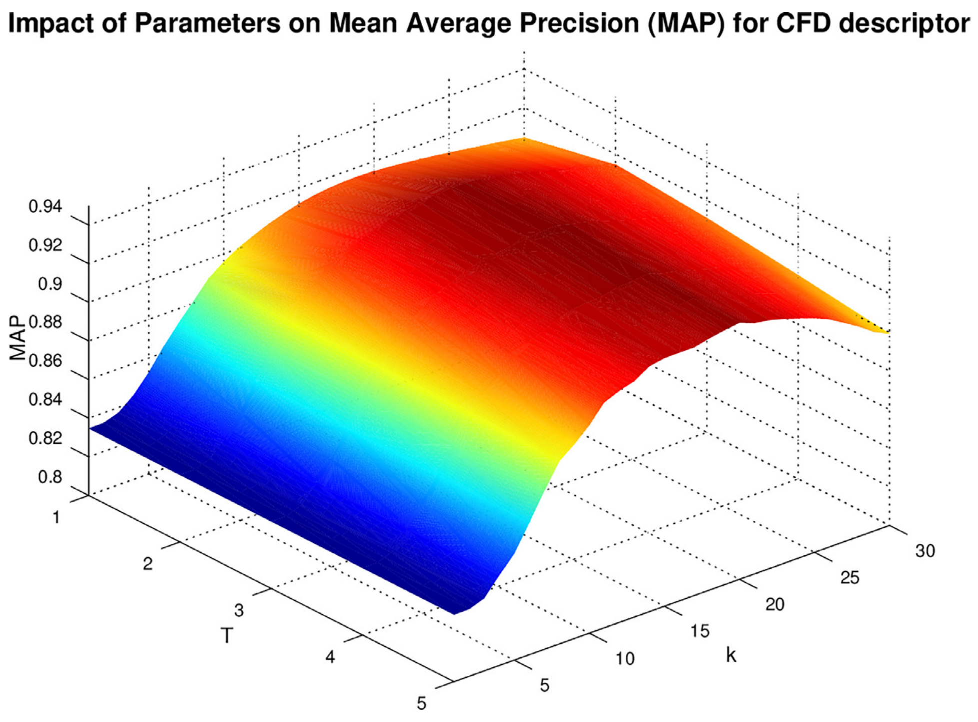

Fig. 2. Impact of k and T on effectiveness.

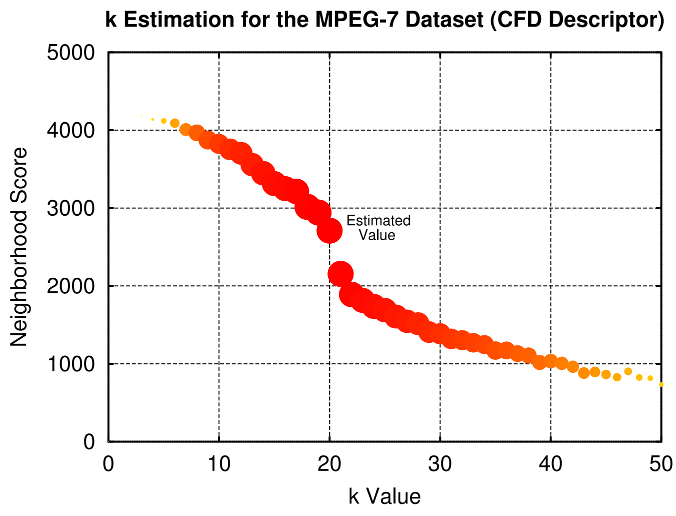

Fig. 3. k estimation for the MPEG-7 dataset.

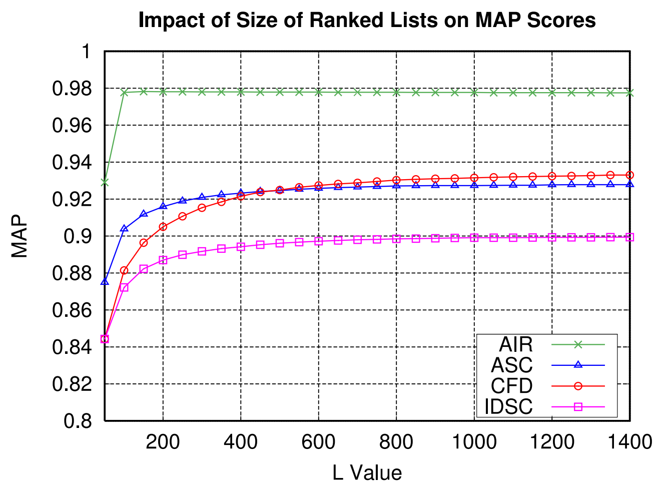

Fig. 4. Impact of L on effectiveness.

due to very small number of images per class. Fig. 3 presents an analysis of the proposed approach for the automatic k estimation. The estimation is based on abrupt changes in reciprocal references (y axis) according to the variations of k (x axis). The size and color of markers are proportional to the effectiveness scores (MAP) obtained to the respective k. Notice that the estimated value ($k = 20$) achieves very high effectiveness scores.

Since the CPRR algorithm does not require the use of the entire ranked list, another experiment analyzed the impact of the size of ranked lists on the effectiveness results. The higher the L value, the greater the effectiveness but also the greater the run time required. The experiment conducted on the MPEG-7 [22] dataset analyzed MAP scores according to different values of L ranging in the interval [50, 1400]. The results for four different descriptors are shown in Fig. 4. As it can be observed, the most significant gains are obtained for small values of L. In addition, for higher values of L, the MAP scores reach an asymptote. In this way, the value of $L = 400$ was used for most of the experiments. The value of L used for the Soccer [18] dataset is limited by the dataset size ($L = 280$). For the UKBench [11] dataset, we used $L = 200$ due to very small number of relevant images for each query. For larger datasets (ALOI, MediaEval, FCVID), we used $L = 1000$.

4.3. Effectiveness evaluation

Various experiments were conducted aiming at evaluating the effectiveness of the proposed CPRR algorithm. Diverse public datasets, several descriptors and various baselines were considered. Firstly, the proposed algorithm is evaluated in generic image retrieval tasks, considering three datasets and different global features (shape, color, and texture). The MAP results are shown in Table 2, considering both serial and parallel implementations of CPRR algorithm. As it can be observed, the effectiveness results for

Table 2
Effectiveness evaluation of the CPRR algorithm considering various datasets, descriptors, and baselines (MAP as effectiveness measure).

Dataset	Descriptor	Original MAP	Pairwise Rec.[51]	RL-Rec.[9]	CPRR Serial	CPRR Parallel	Gain
MPEG-7	SS	37.67%	39.90%	48.68%	49.94%	49.947 % ±0.009	+32.57%
	BAS	71.52%	77.65%	79.58%	80.60%	80.611 % ±0.006	+12.70%
	IDSC	81.70%	86.83%	88.80%	89.42%	89.432 % ±0.004	+9.45%
	CFD	80.71%	91.38%	91.39%	92.15%	92.157 % ±0.005	+14.17%
	ASC	85.28%	91.80%	91.34%	92.32%	92.323 % ±0.005	+8.26%
	AIR	89.39%	95.50%	96.12%	97.80%	97.796 % ±0.007	+9.41%
Soccer	GCH	32.24%	32.35%	34.38%	35.47%	35.307 % ±0.054	+10.02%
	ACC	37.23%	40.31%	41.23%	47.14%	46.965 % ±0.072	+26.62%
	BIC	39.26%	42.64%	45.15%	47.29%	47.172 % ±0.095	+20.45%
Brodatz	LBP	48.40%	51.92%	51.26%	49.07%	49.073 % ±0.006	+1.38%
	CCOM	57.57%	66.46%	64.34%	64.81%	64.816 % ±0.007	+12.58%
	LAS	75.15%	80.73%	79.71%	79.34%	79.346 % ±0.004	+5.58%

Table 3
Effectiveness evaluation of the CPRR algorithm on the UKBench [11] dataset, considering the N-S score.

Descriptor	Type	Original Score	RL-Rec. [9]	CPRR	Gain
SIFT	Local	2.54	2.88	2.99	+17.72%
CEED	Color/text.	2.61	2.72	2.83	+8.43%
FCTH	Color/text.	2.73	2.80	2.90	+6.23%
JCD	Color/text.	2.79	2.88	3.00	+7.53%
FCTH-Spy	Color/text.	2.91	3.05	3.21	+10.31%
BIC	Color	3.04	3.15	3.28	+7.89%
Caffe-FC6	CNN	3.05	3.30	3.40	+11.48%
Caffe-FC8	CNN	3.18	3.30	3.47	+9.12%
Caffe-FC7	CNN	3.31	3.46	3.61	+9.06%
ACC	Color	3.36	3.53	3.62	+7.74%
VOC	BoW	3.54	3.65	3.72	+5.08%

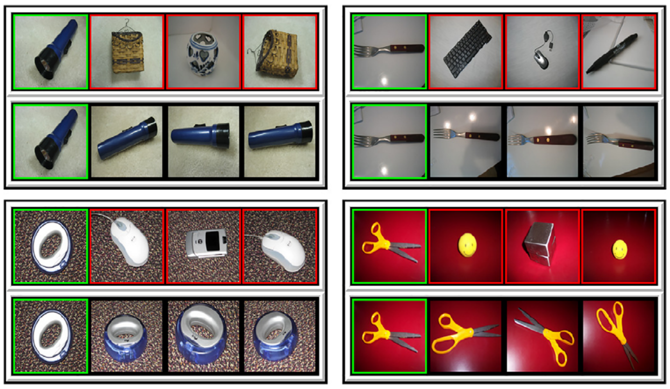
Fig. 5. Impact of the CPRR on the UKBench [11] dataset.

serial and parallel implementations are very similar. The relative effectiveness gains, computed based on serial execution, achieve very high values up to +32.57%. The results of recent unsupervised learning approaches [9,51] are reported as baselines. We can observe that the proposed CPRR algorithm yields the best scores for most of descriptors.

An experiment considering natural image retrieval tasks and a very distinct set of descriptors was conducted on the UKBench [11] dataset. Table 3 presents the effectiveness results given by the N-S score. The N-S score corresponds to the number of relevant images among the first four images returned, defined in the interval [1,4] (the highest achievable score is 4). The small number of images per class (only 4) makes this dataset a very challenging one for unsupervised learning algorithms. Despite this fact, the CPRR achieved high gains ranging from +5.08% to +17.72% and superior to a recent baseline [9].

Fig. 5 illustrates four visual examples of the impact of the CPRR algorithm on retrieval results obtained for the UKBench [11] dataset and the ACC [19] descriptor. The query images are presented in green borders and wrong results in red borders. The first line represents the original retrieval results and the second line, the results after the algorithm execution.

A large-scale experiment was conducted on the ALOI [40] dataset, considering color descriptors. The MAP scores for the ALOI [40] dataset are presented in Table 4. The gains obtained by the CPRR are also very significant for this dataset, ranging from +5.48% to +13.73%.

Table 5 displays the results for the video datasets. Despite the low original MAP scores, our approach presents considerable gains ranging from +4.81% to +25.21%.

The proposed algorithm was also evaluated in rank aggregation tasks, considering the best descriptors for each dataset. Table 6 presents the effectiveness scores for various datasets. We can observe that aggregated results are superior to isolated descriptors in the majority of the cases, reaching very high scores for all datasets.

Table 4
Effectiveness evaluation on the ALOI [40] dataset.

Descriptor	Type	Original MAP	RL-Sim [54]	CPRR	Gain
ACC	Color	43.77%	46.12%	47.45%	+8.41%
CCV	Color	47.49%	50.96%	50.57%	+6.48%
GCH	Color	50.56%	53.14%	53.33%	+5.48%
LCH	Color	58.55%	66.03%	66.59%	+13.73%
BIC	Color	71.75%	78.84%	76.90%	+7.19%

Table 5
Effectiveness evaluation on video datasets.

Descriptor	Dataset	Original MAP	CPRR	Gain
$BoS_{hard,max100}$	MediaEval	1.76%	2.04%	+15.63%
$BoS_{soft,max100}$	MediaEval	2.23%	2.47%	+10.70%
PoP	MediaEval	2.53%	2.95%	+16.47%
HMP	MediaEval	3.85%	4.03%	+4.81%
MFCC	FCVID	1.77%	1.87%	+5.69%
SIFT	FCVID	2.24%	2.60%	+16.28%
IDT-TRAJ	FCVID	2.73%	3.03%	+10.70%
IDT-HOF	FCVID	3.65%	4.02%	+10.24%
IDT-HOG	FCVID	3.80%	4.54%	+19.45%
IDT-MBH	FCVID	4.61%	5.31%	+15.10%
CNN	FCVID	8.42%	10.54%	+25.21%

Table 6
Effectiveness evaluation of rank aggregation tasks.

Dataset	Descriptors	Score	Metric
MPEG-7	CFD+AIR	99.95%	MAP
MPEG-7	CFD+ASC	98.73%	MAP
Brodatz	CCOM+LAS	83.20%	MAP
Soccer	ACC+BIC	48.25%	MAP
UKBench	VOC+CNN-FC7	3.90	N-S score
UKBench	ACC+CNN-FC7	3.88	N-S score
UKBench	ACC+CNN-FC7+VOC	3.94	N-S score
ALOI	BIC+LCH	76.54%	MAP
MediaEval	HMP+PoP	4.37%	MAP
FCVID	CNN+IDT-HOG	10.54%	MAP

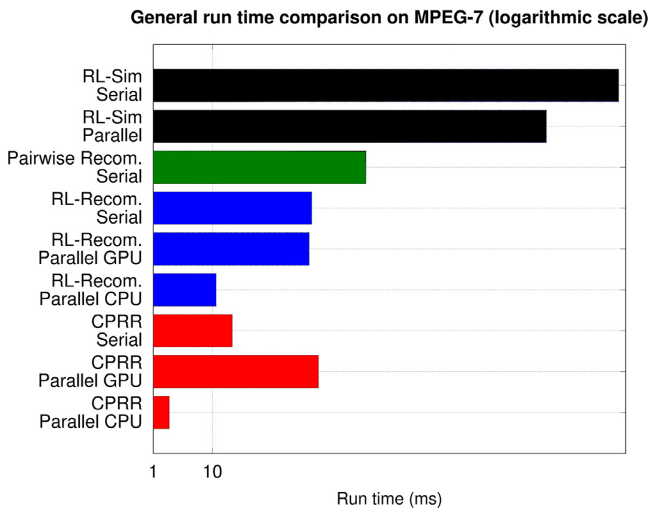

Fig. 6. Run time comparison on MPEG-7 [22] dataset: CPRR and baselines.

4.4. Efficiency evaluation

Experiments were conducted for evaluating the efficiency of the proposed method, considering several aspects, as: different datasets, serial and parallel implementations, different devices (CPU, GPU), and memory transfer models. Table 7 presents the average run time and confidence intervals for the CPRR algorithm considering different criteria. For comparison purposes, the run times of two recent baselines [9,51] are reported. The best performance for each dataset is highlighted in bold. As we can observe, the CPRR algorithm requires very low run times for all datasets, smaller than baselines even for the serial implementation. The OpenCL build and environment time are not considered in the reported results, since the build can be executed once off-line and the environment time is constant independently of dataset sizes.

A general comparison of the CPRR (both serial and parallel) is presented in Fig. 6. The run times of RL-Sim [7,55] (serial and parallel), RL-Recommendation [9] (serial and parallel) and pairwise recommendation [51] (serial), are reported as baselines. The MPEG-7 [22] dataset is considered for the experiment. Notice that, even using a logarithmic scale, the run time of the proposed CPRR (in red) algorithm is significant smaller than other considered approaches. The good performance of the algorithm on CPU can be explained mainly by two factors. First, the operations involved in the CPRR algorithm are very simple, requiring low computational costs. In this scenario, the overhead associated with the data transfer to the GPU memory has a severe impact, contributing favorably to CPU in the comparison. Second, the CPRR algorithm uses iterative sorting steps for ranked lists almost sorted, what constitutes tasks which do not take advantage of all parallel potential of GPUs.

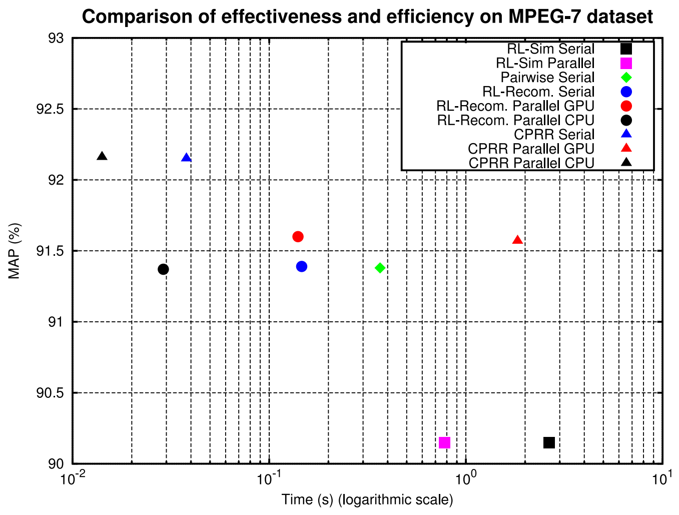

Fig. 7. Effectiveness and efficiency analysis: CPRR and baselines.

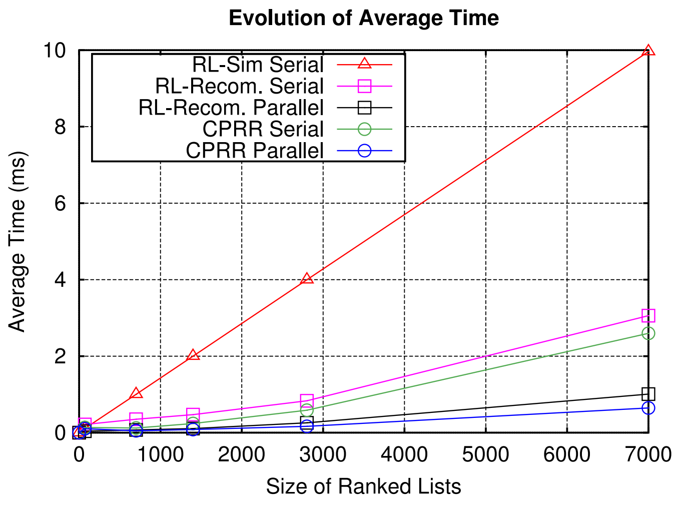

Fig. 8. Scalability analysis on ALOI [40] dataset (time per query).

An experiment analyzing both effectiveness and efficiency aspects was conducted on the MPEG-7 [22] dataset. Fig. 7 presents the results of CPRR and recent baselines (pairwise recommendation [51], RL-Sim [7,55] and RL-Recommendation [9]). The MAP score and the run time determines the position of the algorithms in the graph. Therefore, an ideal algorithm, with high effectiveness and low run time, is positioned at the top-left corner of the graph. Notice that the best positions are occupied by the CPRR algorithm (serial and parallel).

In order to evaluate the scalability of the proposed CPRR algorithm, an experiment varying the size of ranked lists was conducted on the ALOI [40] dataset, considering the LCH [42] descriptor. The size of ranked lists, given by the constant L, establishes an important relationship between effectiveness and efficiency. The L size was varied from 70 to 7000. Aiming at performing a fair comparison of run times, we considered $k = 40$ for this experiment, which is the default value of the baselines. The average time of the unsupervised distance learning per ranked list is reported for each L value. The results are shown in Fig. 8. Results obtained for the RL-Sim [54] and RL-Recommendation [9] algorithms are reported as baselines. Very small average times for growing values of L can be observed for the CPRR algorithm, enabling the use of the algorithm in different scenarios and increasing datasets.

Table 7
Efficiency evaluation: run time (in seconds) of the CPRR for different devices and datasets.

Algorithm	Exec.	Device	Soccer	MPEG-7	Brodatz	UKBench
Pairwise Rec. [51]	Serial	CPU	0.1149 ± 0.00018	0.3663 ± 0.00094	0.6672 ± 0.00140	14.802 ± 0.11059
RL-Rec. [9]	Serial	CPU	0.0607 ± 0.00000	0.1462 ± 0.00021	0.1108 ± 0.00102	0.1868 ± 0.00018
CPRR	Serial	CPU	0.0058 ± 0.00021	0.0381 ± 0.00041	0.0501 ± 0.00077	0.1767 ± 0.00102
CPRR	Parallel	GPU[a]	0.0711 ± 0.00463	0.1640 ± 0.00781	0.1560 ± 0.00570	0.2038 ± 0.00874
CPRR	Parallel	CPU[a]	0.0032 ± 0.00015	0.0164 ± 0.00031	0.0214 ± 0.00054	0.1834 ± 0.00052
CPRR	Parallel	CPU[b]	**0.0027** ± **0.00000**	**0.0131** ± **0.00018**	**0.0143** ± **0.00075**	**0.1082** ± **0.00051**

[a] Memory transfer model: Write buffer
[b] Map buffer.

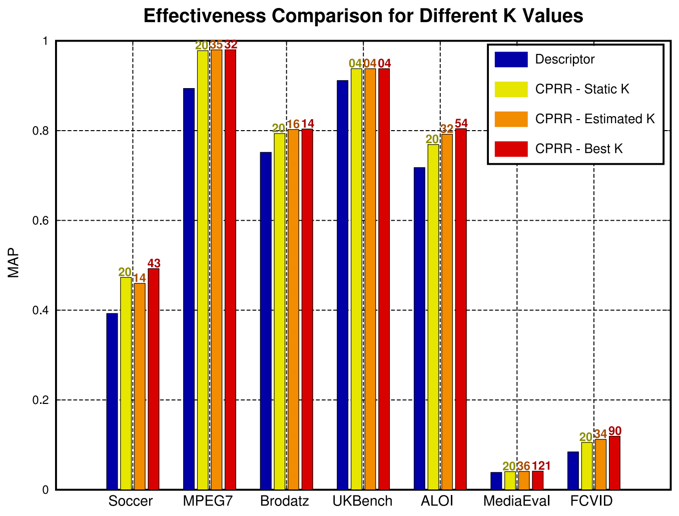

Fig. 9. Effectiveness analysis for neighborhood sizes: fixed, estimated, and best k values.

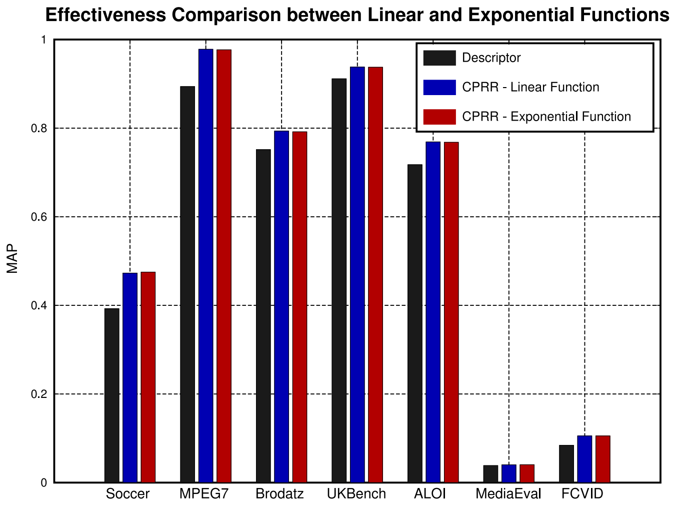

Fig. 10. Effectiveness comparison between linear and exponential functions.

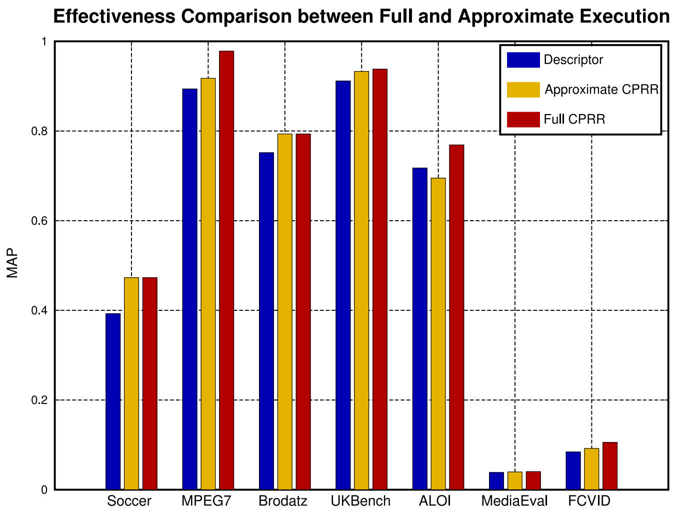

Fig. 11. Effectiveness evaluation of the approximate algorithm.

4.5. Discussion about algorithm variances

The variances of the algorithm were also experimentally evaluated on different datasets. For each dataset, the most effective descriptor is selected. Fig. 9 presents the evaluation of the neighborhood size estimation. The evaluation compared the effectiveness results obtained by: the original descriptor, the fixed size ($k = 20$), the automatic k estimation, and the best k value empirically determined. The estimation was computed considering the interval [4,50] for all datasets, except for FCVID, which used [12,50]. The labels above the bars show the values of k. As it can be observed, the automatic k estimation outperformed the fixed k for most of datasets.

The linear and exponential functions for assigning weights for rank positions were also evaluated. Fig. 10 presents a comparison between both functions and the original descriptor. The results are very similar for all datasets, indicating an equivalence between the functions. Fig. 11 shows the experimental analysis of the approximate algorithm. As it can be observed, the approximate algorithm achieved effectiveness results very similar to the full algorithm on various datasets (Soccer, Brodatz, UKBench, MediaEval).

4.6. Comparison with other approaches

Finally, the CPRR algorithm was also evaluated in comparison with other state-of-the-art unsupervised learning methods and recently proposed retrieval approaches. Experiments were conducted on two image datasets: UKBench [11] and MPEG-7 [22], which are popular datasets commonly used as benchmark for image retrieval and post-processing methods.

Table 8 presents the effectiveness results of CPRR algorithm in comparison with recent retrieval approaches on the UKBench [11] dataset. We can observe that the results achieved by the CPRR algorithm are among the the best results, reaching a N-S score of **3.94** for the aggregation of VOC+ACC+CNN-FC7 features.

A comparison of the proposed method with other state-of-the-art unsupervised methods on the MPEG-7 [22] dataset is shown in Table 9. The effectiveness results obtained by the CPRR algorithm are also comparable or superior to various other approaches.

Table 8
Effectiveness comparison among recent retrieval methods on the UKBench [11] dataset.

N-S scores for recent retrieval methods					
Zheng et al. [56]	Zhang et al. [15]	Zheng et al. [57]	Xie et al. [58]	Bai et al. [59]	Bai et al. [60]
3.57	3.83	3.84	3.89	3.94	3.98
N-S scores for the CPRR method					
VOC+CNN-FC7	ACC+CNN-FC7	ACC+CNN-FC7+VOC			
3.90	3.88	**3.94**			

Table 9
Comparison of post-processing methods on the MPEG-7 [22] dataset - Bull's Eye Score (Recall@40).

Shape descriptors		
CFD	–	84.43%
IDSC	–	85.40%
SC	–	86.80%
ASC	–	88.39%
AIR	–	93.67%
Post-processing methods		
Algorithm	**Descriptor(s)**	**Score**
Graph transduction [17]	IDSC	91.00%
Shortest path propagation [6]	IDSC	93.35%
Smooth neighborhood [60]	IDSC	93.52%
RDP [59]	IDSC	93.78%
RL-Sim [7]	CFD	94.13%
CPRR	**CFD**	**94.77%**
Locally C. diffusion process [3]	ASC	95.96%
RL-recommendation [9]	ASC	94.40%
CPRR	**ASC**	**95.07%**
Tensor product graph [4]	ASC	96.47%
Self-smoothing operator [5]	SC+IDSC	97.64%
Self-smoothing operator [5]	SC+IDSC+DDGM	99.20%
CPRR	**CFD+ASC**	**99.40%**
Pairwise recommendation [51]	CFD+IDSC	99.52%
RL-recommendation [9]	AIR	99.78%
CPRR	**AIR**	**99.93%**
Tensor product graph [4]	AIR	99.99%
Neighbor set similarity [8]	AIR	100%
Smooth neighborhood [60]	AIR	100%
CPRR	**CFD+AIR**	**100%**

5. Conclusions

In this paper, we presented a novel unsupervised similarity learning algorithm for multimedia retrieval tasks. The proposed approach employs Cartesian product operations for analyzing rank information and exploiting the underlying structure of the datasets. The method is capable of estimating an effective size for the neighborhood set and compute the results when the query image is not part of the dataset. The source code is public available,[2] as part of an open-source framework for unsupervised distance learning.

Extensive experiments were conducted considering public datasets (images and videos) and several descriptors. The experimental results and comparisons with other recent state-of-the-art approaches demonstrate the effectiveness and efficiency of the proposed method. As future work, we intend to investigate the use of the proposed method in semi-supervised learning tasks, considering interactive image retrieval scenarios.

Acknowledgments

We thank the São Paulo Research Foundation - FAPESP (grants #2013/08645-0, #2014/04220-8, #2016/06441-7, and #2017/02091-4) and the Brazilian National Council for Scientific and Technological Development - CNPq (grant #423228/2016-1) for financial support. The authors are also grateful to Otávio Augusto Bizetto Penatti for his assistance in extracting the BoS and PoP features from the dataset provided for the Video Genre Tagging Task at the MediaEval 2012.

[2] https://github.com/UDLF/UDLF (As of August, 2017).

References

[1] B. Thomee, M. Lew, Interactive search in image retrieval: a survey, Int. J. Multimedia Inf. Retrieval 1 (2) (2012) 71–86.
[2] Y. Liu, D. Zhang, G. Lu, W.-Y. Ma, A survey of content-based image retrieval with high-level semantics, Pattern Recognit. 40 (1) (2007) 262–282.
[3] X. Yang, S. Koknar-Tezel, L.J. Latecki, Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 357–364.
[4] X. Yang, L. Prasad, L. Latecki, Affinity learning with diffusion on tensor product graph, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 28–38.
[5] J. Jiang, B. Wang, Z. Tu, Unsupervised metric learning by self-smoothing operator, in: IEEE International Conference on Computer Vision (ICCV), 2011, pp. 794–801.
[6] J. Wang, Y. Li, X. Bai, Y. Zhang, C. Wang, N. Tang, Learning context-sensitive similarity by shortest path propagation, Pattern Recognit. 44 (10-11) (2011) 2367–2374.
[7] D.C.G. Pedronette, R.d. S. Torres, Image re-ranking and rank aggregation based on similarity of ranked lists, Pattern Recognit. 46 (8) (2013) 2350–2360.
[8] X. Bai, S. Bai, X. Wang, Beyond diffusion process: neighbor set similarity for fast re-ranking, Inf. Sci. 325 (2015) 342–354.
[9] L.P. Valem, D.C.G. Pedronette, R.d. S. Torres, E. Borin, J. Almeida., Effective, efficient, and scalable unsupervised distance learning in image retrieval tasks, in: ACM International Conference on Multimedia Retrieval (ICMR), 2015.
[10] Y. Chen, X. Li, A. Dick, R. Hill, Ranking consistency for image matching and object retrieval, Pattern Recognit. 47 (3) (2014) 1349–1360.
[11] D. Nistér, H. Stewénius, Scalable recognition with a vocabulary tree, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 2, 2006, pp. 2161–2168.
[12] L.P. Valem, D.C.G. Pedronette, Unsupervised similarity learning through cartesian product of ranking references for image retrieval tasks, in: Conference on Graphics, Patterns and Images (SIBGRAPI), 2016, pp. 249–256.
[13] H. Jegou, C. Schmid, H. Harzallah, J. Verbeek, Accurate image search using the contextual dissimilarity measure, IEEE Trans. Pattern Anal. Mach. Intell. 32 (1) (2010) 2–11.
[14] D. Qin, S. Gammeter, L. Bossard, T. Quack, L. van Gool, Hello neighbor: accurate object retrieval with k-reciprocal nearest neighbors, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011, pp. 777–784.
[15] S. Zhang, M. Yang, T. Cour, K. Yu, D. Metaxas, Query specific rank fusion for image retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 37 (4) (2015) 803–815.
[16] S. Bai, X. Bai, Sparse contextual activation for efficient visual re-ranking, IEEE Trans. Image Process. 25 (3) (2016) 1056–1069.
[17] X. Yang, X. Bai, L.J. Latecki, Z. Tu, Improving shape retrieval by learning graph transduction, in: Eur. Conf. Comput. Vision (ECCV), 4, 2008, pp. 788–801.
[18] J. van de Weijer, C. Schmid, Coloring local feature extraction, in: European Conference on Computer Vision (ECCV), 2006, pp. 334–348.
[19] J. Huang, S.R. Kumar, M. Mitra, W.-J. Zhu, R. Zabih, Image indexing using color correlograms, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1997, pp. 762–768.
[20] R.O. Stehling, M.A. Nascimento, A.X. Falcão, A compact and efficient image retrieval approach based on border/interior pixel classification, in: ACM International Conference on Information and Knowledge Management (CIKM), 2002, pp. 102–109.
[21] M.J. Swain, D.H. Ballard, Color indexing, Int. J. Comput. Vis. 7 (1) (1991) 11–32.
[22] L.J. Latecki, R. Lakmper, U. Eckhardt, Shape descriptors for non-rigid shapes with a single closed contour, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2000, pp. 424–429.
[23] R. Gopalan, P. Turaga, R. Chellappa, Articulation-invariant representation of non-planar shapes, in: European Conference on Computer Vision (ECCV), 3, 2010, pp. 286–299.

[24] H. Ling, X. Yang, L.J. Latecki, Balancing deformability and discriminability for shape matching, in: European Conference on Computer Vision (ECCV), Vol. 3, 2010, pp. 411–424.
[25] N. Arica, F.T.Y. Vural, BAS: a perceptual shape descriptor based on the beam angle statistics, Pattern Recognit. Lett. 24 (9-10) (2003) 1627–1639.
[26] D.C.G. Pedronette, R.d. S. Torres, Shape retrieval using contour features and distance optimization, in: International Conference on Computer Vision Theory and Applications (VISAPP), Vol. 1, 2010, pp. 197–202.
[27] H. Ling, D.W. Jacobs, Shape classification using the inner-distance, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2) (2007) 286–299.
[28] R.d. S. Torres, A.X. Falcão, Contour salience descriptors for effective image retrieval and analysis, Image Vis. Comput. 25 (1) (2007) 3–13.
[29] P. Brodatz, Textures: A Photographic Album for Artists and Designers, Dover Publications, New York, 1966.
[30] V. Kovalev, S. Volmer, Color co-occurence descriptors for querying-by-example, in: International Conference on Multimedia Modeling (ICMM), 1998, p. 32.
[31] B. Tao, B.W. Dickinson, Texture recognition and image retrieval using gradient indexing, J. Visual Commun. Image Represent. 11 (3) (2000) 327–342.
[32] T. Ojala, M. Pietikäinen, T. Mäenpää, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Trans. Pattern Anal. Mach. Intell. 24 (7) (2002) 971–987.
[33] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional Architecture for Fast Feature Embedding, Proceedings of the 22Nd ACM International Conference on Multimedia, 2014, pp. 675–678.
[34] S.A. Chatzichristofis, Y.S. Boutalis, Cedd: color and edge directivity descriptor: a compact descriptor for image indexing and retrieval, in: International Conference on Computer Vision Systems (ICVS), 2008, pp. 312–322.
[35] S.A. Chatzichristofis, Y.S. Boutalis, Fcth: fuzzy color and texture histogram - a low level feature for accurate image retrieval, in: Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), 2008, pp. 191–196.
[36] M. Lux, Content based image retrieval with LIRe, in: ACM International Conference on Multimedia (ACM-MM), 2011.
[37] K. Zagoris, S. Chatzichristofis, N. Papamarkos, Y. Boutalis, Automatic image annotation and retrieval using the joint composite descriptor, in: Panhellenic Conference on Informatics (PCI), 2010, pp. 143–147.
[38] D. Lowe, Object recognition from local scale-invariant features, in: IEEE International Conference on Computer Vision (ICCV), 1999, pp. 1150–1157.
[39] X. Wang, M. Yang, T. Cour, S. Zhu, K. Yu, T. Han, Contextual weighting for vocabulary tree based image retrieval, in: IEEE International Conference on Computer Vision (ICCV), 2011, pp. 209–216.
[40] J.-M. Geusebroek, G.J. Burghouts, A.W.M. Smeulders, The amsterdam library of object images, Int. J. Comput. Vis. 61 (1) (2005) 103–112.
[41] G. Pass, R. Zabih, J. Miller, Comparing images using color coherence vectors, in: ACM International Conference on Multimedia (ACM-MM), 1996, pp. 65–73.
[42] H. Lu, B. Ooi, K. Tan, Efficient image retrieval by color contents, in: International Conference on Applications of Databases (ADB), 1994, pp. 95–108.
[43] S. Schmiedeke, C. Kofler, I. Ferrané, Overview of mediaeval 2012 genre tagging task, MediaEval, 2012.
[44] O.A.B. Penatti, L.T. Li, J. Almeida, R.S. Torres, A visual approach for video geocoding using bag-of-scenes, in: ACM International Conference on Multimedia Retrieval (ICMR), 2012, pp. 1–8.
[45] J. Almeida, N.J. Leite, R.S. Torres, Comparison of video sequences with histograms of motion patterns, in: IEEE International Conference on Image Processing (ICIP), 2011, pp. 3673–3676.
[46] J. Almeida, D.C.G. Pedronette, O.A.B. Penatti, Unsupervised manifold learning for video genre retrieval, in: Iberoamerican Congress on Pattern Recognition (CIARP), 2014, pp. 604–612.
[47] Y.G. Jiang, Z. Wu, J. Wang, X. Xue, S.F. Chang, Exploiting Feature and Class Relationships in Video Categorization with Regularized Deep Neural Networks, IEEE Transactions on Pattern Analysis and Machine Intelligence (2017) On-line, To appear, doi:10.1109/TPAMI.2017.2670560.
[48] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Annual Conference on Neural Information Processing Systems (NIPS), 2012, pp. 1106–1114.
[49] H. Wang, C. Schmid, Action recognition with improved trajectories, in: IEEE International Conference on Computer Vision (ICCV), 2013, pp. 3551–3558.
[50] S. Davis, P. Mermelstein, Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences, IEEE Trans. Acoust. Speech Sig. Process. 28 (4) (1980) 357–366.
[51] D.C.G. Pedronette, R.d. S. Torres, Exploiting pairwise recommendation and clustering strategies for image re-ranking, Inf. Sci. 207 (2012) 19–34.
[52] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[53] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Neural Inf. Process. Syst. (2012) 1106–1114.
[54] D.C.G.a. Pedronette, J. Almeida, R. Da, S. Torres, A scalable re-ranking method for content-based image retrieval, Inf. Sci. 265 (2014) 91–104.
[55] D.C.G. Pedronette, R.d. S. Torres, E. Borin, M. Breternitz, Image re-ranking acceleration on GPUs, Computer Architecture and High Performance Computing (SBAC-PAD), 2013.
[56] L. Zheng, S. Wang, Q. Tian, Lp-norm idf for scalable image retrieval, IEEE Trans. Image Process. 23 (8) (2014) 3604–3617.
[57] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, Q. Tian, Query-adaptive late fusion for image search and person re-identification, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.
[58] L. Xie, R. Hong, B. Zhang, Q. Tian, Image classification and retrieval are ONE, in: ACM International Conference on Multimedia Retrieval (ICMR), 2015, pp. 3–10.
[59] S. Bai, X. Bai, Q. Tian, L.J. Latecki, Regularized diffusion process for visual retrieval, in: Conference on Artificial Intelligence (AAAI), 2017, pp. 3967–3973.
[60] S. Bai, S. Sun, X. Bai, Z. Zhang, Q. Tian, Smooth neighborhood structure mining on multiple affinity graphs with applications to context-sensitive similarity, in: European Conference on Computer Vision (ECCV), 2016, pp. 592–608.