Re-ranking Permutation-Based Candidate Sets with the n-Simplex projection

Giuseppe Amato[1], Edgar Chávez[2], Richard Connor[3],
Fabrizio Falchi[1], Claudio Gennaro[1], and Lucia Vadicamo[1]

[1] Institute of Information Science and Technologies (ISTI), CNR, Pisa, Italy
[2] Department of Computer Science, CICESE, Ensenada, Mexico
[3] Department of Computing Science, University of Stirling, FK9 4LA, Scotland
name.surname@isti.cnr.it, elchavez@cicese.mx, richard.connor@stir.ac.uk

Abstract. In the realm of metric search, the permutation-based approaches have shown very good performance in indexing and supporting approximate search on large databases. These methods embed the metric objects into a permutation space where candidate results to a given query can be efficiently identified. Typically, to achieve high effectiveness, the permutation-based result set is refined by directly comparing each candidate object to the query one. Therefore, one drawback of these approaches is that the original dataset needs to be stored and then accessed during the refining step. We propose a refining approach based on a metric embedding, called n-Simplex projection, that can be used on metric spaces meeting the n-point property. The n-Simplex projection provides upper- and lower-bounds of the actual distance, derived using the distances between the data objects and a finite set of pivots. We propose to reuse the distances computed for building the data permutations to derive these bounds and we show how to use them to improve the permutation-based results. Our approach is particularly advantageous for all the cases in which the traditional refining step is too costly, e.g. very large dataset or very expensive metric function.

Keywords: metric search · permutation-based indexing · n-point property · n-Simplex projection · metric embedding · distance bounds

1 Introduction

The problem of searching data objects that are close to a given query object, under some metric function, has a vast number of applications in many branches of computer science, including pattern recognition, computational biology and multimedia information retrieval, to name but a few. This search paradigm, referred to as *metric search*, is based on the assumption that data objects are represented as elements of a metric space (D, d) where the *metric*[4] function $d : D \times D \to \mathbb{R}^+$ provides a measure of the closeness of the data objects.

[4] Throughout this paper, we use the term "metric" and "distance" interchangeably to indicate a function satisfying the metric postulates [23].

In metric search, the main concern is processing and structuring a finite set of data $X \subset D$ so that *proximity queries* can be answered quickly and with a low computational cost. A proximity query is defined by a query object $q \in D$ and a proximity condition, such as "find all the objects within a threshold distance of q" (*range query*) or "finding the k closest objects to q" (*k-nearest neighbour query*). The response to a query is the set of all the objects $o \in X$ that satisfy the considered proximity condition. Providing an exact response is not feasible if the search space is very large or if it has a high intrinsic dimensionality since a large fraction of the data needs to be inspected to process the query. In such cases, the exact search rarely outperforms a sequential scan [22]. To overcome the *curse of dimensionality* [19] researchers proposed several *approximate search* methods that are less (but still) affected by this phenomenon.

Many approximate methods are based on the idea of mapping the data objects into a more tractable space in which we can efficiently perform the search. Successful examples are the *Permutation-Based Indexing* (PBI) approaches that represent data objects as a sequence of identifiers (*permutation*). Typically, the permutation for an object o is computed as a ranking list of some preselected reference points (*pivots*) according to their distance to o. The main rationale behind this approach is that if two objects are very close one to the other, they will sort the set of pivots in a very similar way, and thus the corresponding permutation representations will be close as well. The search in the permutation space is used to build a candidate result set that is normally refined by comparing each candidate object to the query one (according to the metric governing the data space). This refinement step therefore requires access to the original data, which is likely to be too large to fit into main memory. However, some kind of refinement step is likely to be required as the search in the permutation space typically has relatively low precision.

In this paper, we focus on the k-nearest neighbour (k-NN) query search and we investigate several approaches to perform the refining step without accessing the original data, but instead exploiting the distances between the objects and the pivots (calculated at indexing time and stored within the permutations) and the distances between the query and the pivots (evaluated when computing the query permutation). In particular, for a large class of metric spaces that meet the so-called "*n-point property*" [9,11] we propose the use of the *n-Simplex projection* [12] that allows mapping metric objects into a finite dimensional Euclidean space where upper- and lower- bounds for the original distances can be calculated. We show how these distance bounds can be used to refine the permutation-based results, therefore avoiding access to the original dataset.

2 Related Work

The idea of approximating the distance between any two metric objects by comparing their permutation-based representations was originally proposed in [5,8]. Several techniques for indexing and searching permutations were proposed in literature, including indexes based on inverted files, like the *Metric Inverted File*

(MI-File) [4] and its variants, or using prefix trees, like the *Permutation Prefix Index* (PP-Index) [13] and the *Pivot Permutation Prefix Index* (PPP-Index) [17]. The permutation-based approach are *filter* and *refine* methods: a candidate result set is identified by performing the search in the permutation space, then the result set is refined, commonly, by evaluating the actual distance between the query and the candidate objects.

The permutation representation of an object is computed by ordering the identifiers of a set of pivots according to their distances to the object [3]. However, the computation of these distances is just one, yet effective, approach to associate a permutation to each data object. For example, the *Deep Permutations* [2] have been recently proposed as an efficient and effective alternative for generating permutations of emerging deep features. However, this approach is suitable only for specific data domains while the traditional approach is generally applicable since it requires only the existence of a distance function to compare data objects.

The distances between the data objects and a set of pivots can be used also to embed the data into another metric space where it is possible to deduce upper- and lower- bounds on the actual distance of any pair of objects. In this context, one of the very first embeddings proposed in a metric search scenario was the one representing each data object with a vector of its distances to the pivots. The LAESA [16] is a notable example of indexing technique using this approach. Recently, Connor et al. [12,11,10] observed that for a large class of metric spaces it is possible to use the distances to a set of n pivots to project the data objects into a n-dimensional Euclidean space such that in the projected space 1) the distances object-pivots are preserved, 2) the Euclidean distance between any two points is a lower-bound of the actual distance, 3) also an upper-bound can be easily computed. They called this approach *n-Simplex projection* and they proved that it can be used in all the metric spaces meeting the *n-point property* [7]. As also pointed out in [9], many common metric spaces meet the desired property, like Cartesian spaces of any dimension with the Euclidean, cosine or quadratic form distances, probability spaces with the Jenson-Shannon or the Triangular distance, and more generally any Hilbert-embeddable space [7,20].

3 Background

In the following, we summarize key concepts of some metric space transformations based on the use of distances between data objects and a set of pivots. The rationale behind these approaches is to project the original data into a space that has better indexing properties than the original, or where the comparison between objects is less expensive than the original distance. In particular, we review data embeddings into permutation spaces, where objects can be efficiently indexed using PBI methods, and other pivot-based embeddings that allow computing upper- and lower- bounds of the actual distance. Table 1 summarizes the notation used.

Table 1: Notation used throughout this paper

Symbol	**Definition**
(D, d)	metric space
X	finite search space, $X \subseteq D$
$\{p_1, \ldots, p_n\}$	set of pivots, $p_i \in D$
n	number of pivots
o, s	data objects, $o, s \in X$
q	query, $q \in D$
k, k'	number of results of a nearest neighbour search
amp	amplification factor
$\mathit{\Pi}_o$	pivot permutation
$\mathit{\Pi}_o^{-1}$	inverted permutation
l	location parameter (permutation prefix length)
$\mathit{\Pi}_{o,l}$	truncated permutation (permutation prefix of length l)
$\mathit{\Pi}_{o,l}^{-1}$	inverted truncated permutation
$PivotSet(\mathit{\Pi}_{o,l})$	the pivots whose identifiers appear in $\mathit{\Pi}_{o,l}$
$\mathit{\Gamma}_{o,q}$	pivots in the intersection $PivotSet(\mathit{\Pi}_{q,l}) \cap PivotSet(\mathit{\Pi}_{o,l})$
$S_{\rho,l}$	Spearman's rho with location parameter l
ℓ_2	Euclidean distance
ℓ_∞	Chebyshev distance
$\lvert \cdot \rvert$	size of a set

3.1 Permutation-based Representation

Let $\mathcal{D}$ a data domain and $d : \mathcal{D} \times \mathcal{D} \to \mathbb{R}^+$ a *metric* function on it[5]. A permutation-based representation $\mathit{\Pi}_o$ (briefly *permutation*) of an object $o \in \mathcal{D}$ with respect to a fixed set of *pivots*, $\{p_1, \ldots, p_n\} \subset \mathcal{D}$, is the sequence of pivots identifiers ordered by their distance to o.

Formally, the permutation $\mathit{\Pi}_o = [\mathit{\Pi}_o(1), \mathit{\Pi}_o(2), ..., \mathit{\Pi}_o(n)]$ lists the pivot identifiers $\{1, \ldots, n\}$ in an order such that $\forall\, i, j \in \{1, \ldots, n\}$

$$\mathit{\Pi}_o(i) < \mathit{\Pi}_o(j) \qquad \Leftrightarrow \qquad \begin{array}{c} d(o, p_{\mathit{\Pi}_o(i)}) < d(o, p_{\mathit{\Pi}_o(j)}) \\ or \\ \big(\, d(o, p_{\mathit{\Pi}_o(i)}) = d(o, p_{\mathit{\Pi}_o(j)}) \;\wedge\; (i < j) \,\big) \end{array} \tag{1}$$

An equivalent permutation-based representation is the *inverted permutation*, defined as $\mathit{\Pi}_o^{-1} = [\mathit{\Pi}_o^{-1}(1), \mathit{\Pi}_o^{-1}(2), \ldots, \mathit{\Pi}_o^{-1}(n)]$, where $\mathit{\Pi}_o^{-1}(i)$ denotes the position of a pivot p_i in the permutation $\mathit{\Pi}_o$. The inverted permutation is such that $\mathit{\Pi}_o(\mathit{\Pi}_o^{-1}(i)) = i$. Note that the value at the coordinate i in the permutation $\mathit{\Pi}_o$ is the identifier of the pivot at i-th position in the ranked list of the nearest pivots to o; the value at the coordinate i in the inverted representation $\mathit{\Pi}_o^{-1}$ is the rank of the pivot p_i in the list of the nearest pivots to o.

[5] In this work, we focus on metric search. The requirement that the function d satisfies the metric postulates is sufficient, but not necessary, to produce a permutation-based representation. For example, d may be a dissimilarity function.

The inverted permutation representation is often used in practice since it allows us to represent permutations in a Cartesian coordinate system and easily compute most of the commonly-used distances between permutations as distances between Cartesian points. In this paper, we use the Spearman Rho that is defined as $S_\rho(\Pi_o, \Pi_s) = \ell_2(\Pi_o^{-1}, \Pi_s^{-1})$ for any two permutations Π_o, Π_s.

Most of the PBI methods, e.g. [4,13,17], use only a fixed-length prefix of the permutations in order to represent or compare objects. This choice is based on the intuition that the most relevant information in the permutation is present in its very first elements, i.e. the identifiers of the closest pivots. Moreover, using the positions of the nearest l out of n pivots often leads to obtaining better or similar effectiveness to using the full-length permutation [4,3], resulting also in a more compact data encoding. The permutation prefixes are compared using *top-l distances* [14], like the Spearman Rho with location parameter l defined as $S_{\rho,l},(\Pi_o, \Pi_s) = \ell_2(\Pi_{o,l}^{-1}, \Pi_{s,l}^{-1})$, where $\Pi_{o,l}^{-1}$ is the *inverted truncated permutation*:

$$\Pi_{o,l}^{-1}(i) = \begin{cases} \Pi_o^{-1}(i) & \text{if} \quad \Pi_o^{-1}(i) \leq l \\ l+1 & \text{otherwise} \end{cases} \tag{2}$$

3.2 Pivoted embedding

The distances between metric objects and a set of pivots $\{p_1, \ldots, p_n\} \subset \mathcal{D}$ can be also used to embed a metric space into $(\mathbb{R}^n, \ell_\infty)$:

$$\begin{aligned} f_n : (D, d) &\to (\mathbb{R}^n, \ell_\infty) \\ o &\to [d(o, p_1), \ldots, d(o, p_n)] \end{aligned}$$

Using the triangle inequality of the metric governing the space is possible to prove that

$$\max_{i=1,\ldots,n} |d(o, p_i) - d(s, p_i)| \leq d(o, s) \leq \min_{i=1,\ldots,n} |d(o, p_i) + d(s, p_i)| \tag{3}$$

which means that $\ell_\infty(f_n(o), f_n(s))$ is a lower-bound of $d(o, s)$ and that also an upper-bound can be defined using the projected objects $f_n(o)$, $f_n(s)$ (see [23, pp.28]). In the following we referred to these bounds to as *Pivoted embedding* bounds. Please note that if we use just a subset of size l of the pivots $\{p_1, \ldots, p_n\}$, the corresponding mapping f_l provides upper- and lower- bounds that are less tight than that obtained using f_n.

This family of embeddings are typically used in indexing tables like LAESA [16] or for space pruning [23]. However, as further described in Section 4, in this work we used them not for indexing purpose, but rather as techniques to approximate the distances between a query and data objects already indexed using a permutation-based approach.

3.3 n-Simplex projection

In [9,12] it was observed that there exists a large class of metric spaces that satisfy the so-called *n-point property*, which provides geometric guarantees stronger than triangle inequality.

A metric space meets the n-point property if, and only if, any set of n points of the space can be embedded into a $(n-1)$-dimensional Euclidean space while preserving all the $\binom{n}{2}$ inter-points distances. This property was exploited in [12] to define an embedding of the considered metric space into a finite-dimensional Euclidean space. Specifically, they defined a family of functions $\phi_n : (D, d) \to (\mathbb{R}^n, \ell_2)$, where $\phi_n(o)$ is obtained using the distances between o and a set of pivots $\{p_1, \ldots, p_n\} \subset \mathcal{D}$. They provided also an inductive algorithm for determining the Cartesian coordinates of $\phi_n(o)$ that, given the distances $d(o, p_i)$, requires the computations of $O(n)$ Euclidean distances between vectors having less than n dimensions. The core idea of their approach is computing the vector $\phi_n(o)$ as the apex of a n-dimensional simplex[6] where the length of the i-th edge connecting the apex and a simplex base corresponds to the actual distance $d(o, p_i)$. The simplex base is computed using the distances $d(p_i, p_j)$ for all $i, j \in \{1, \ldots, n\}$.

One of the main outcomes of this embedding is that it allows deriving upper- and lower-bounds of the actual distance by computing the Euclidean distance between two Cartesian points. In facts, given the apexes

$$\begin{aligned}\phi_n(o) &= [x_1, x_2, \ldots, x_{n-1}, x_n] \\ \phi_n(s) &= [y_1, y_2, \ldots, y_{n-1}, y_n]\end{aligned}$$

it holds

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \leq \quad d(o, s) \quad \leq \quad \sqrt{\sum_{i=1}^{n-1}(x_i - y_i)^2 + (x_n + y_n)^2} \tag{4}$$

So, if defining $\phi_n^-(s) = [y_1, y_2, \ldots, y_{n-1}, -y_n]$, we have that $\ell_2(\phi_n(o), \phi_n(s))$ and $\ell_2(\phi_n(o), \phi_n^-(s))$ are respectively a lower- and and upper-bound for $d(o, s)$. Connor et al. [12][7] proved that, if ϕ_n is the n-Simplex projection based on the pivots $\{p_1, \ldots, p_n\}$, and ϕ_m is the m-Simplex projection based on the pivots $\{p_1, \ldots, p_n, p_{n+1}, \ldots, p_m\}$ then

$$\ell_2(\phi_n(o), \phi_n(s)) \leq \ell_2(\phi_m(o), \phi_m(s)) \leq d(o, s) \leq \ell_2(\phi_m(o), \phi_m^-(s)) \leq \ell_2(\phi_n(o), \phi_n^-(s)).$$

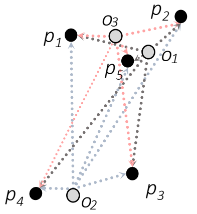
Moreover, they experimentally showed that the so-defined distance bounds converge to the actual distance when increasing the number of pivots.

4 Re-ranking Permutation-Based Candidate Set

The permutation-based methods are filter-and-refine approaches that map original data (X, d) into a permutation space. The permutation representations are used to identify a set of candidate results for a given query $q \in D$. The candidate results are then refined, typically by comparing the candidate objects with the

[6] A simplex is a generalisation of a triangle or a tetrahedron in arbitrary dimensions. We refer to [12] for further details.

[7] See also the on-line Appendix at `http://arxiv.org/abs/1707.08370`.



Permutation representations (*location parameter* $l = 3$)

$\Pi_{o_1} = [5,2,1,3,4]$	$\Pi_{o_2} = [4,3,5,1,2]$	$\Pi_{o_3} = [5,1,2,3,4]$
$\Pi^{-1}_{o_1} = \overset{p_1,p_2,p_3,p_4,p_5}{[3,2,4,5,1]}$	$\Pi^{-1}_{o_2} = \overset{p_1,p_2,p_3,p_4,p_5}{[4,5,2,1,3]}$	$\Pi^{-1}_{o_3} = \overset{p_1,p_2,p_3,p_4,p_5}{[2,3,4,5,1]}$
$\Pi^{-1}_{o_1,l} = \overset{p_1,p_2,p_3,p_4,p_5}{[3,2,4,4,1]}$	$\Pi^{-1}_{o_2,l} = \overset{p_1,p_2,p_3,p_4,p_5}{[4,4,2,1,3]}$	$\Pi^{-1}_{o_3,l} = \overset{p_1,p_2,p_3,p_4,p_5}{[2,3,4,4,1]}$

Posting lists

p_1	$(o_3, 2)$	$(o_1, 3)$	
p_2	$(o_1, 2)$	$(o_3, 3)$	
p_3	$(o_2, 2)$	$(o_3, 3)$	
p_4	$(o_2, 1)$		
p_5	$(o_1, 1)$	$(o_3, 1)$	$(o_2, 3)$

Posting lists with distances

p_1	$(o_3, 2, d(p_1, o_3))$	$(o_1, 3, d(p_1, o_1))$	
p_2	$(o_1, 2, d(p_2, o_1))$	$(o_3, 3, d(p_2, o_3))$	
p_3	$(o_2, 2, d(p_3, o_2))$	$(o_3, 3, d(p_3, o_3))$	
p_4	$(o_2, 1, d(p_4, o_2))$		
p_5	$(o_1, 1, d(p_5, o_1))$	$(o_3, 1, d(p_5, o_3))$	$(o_2, 3, d(p_5, o_2))$

Fig. 1: Example of posting lists and posting list with distances generated to index three objects using five pivots and a location parameter $l = 3$

query one according to the actual distance d. In the following, we investigate the use of other refining approaches to answer a k-NN query. The aim is improving the permutation-based results while getting rid of the original dataset.

Let $CandSet(q)$ the set of candidate results selected using the permutation-based encoding, where $|CandSet(q)| = k' \geq k$. The candidate result set can be built, for example, by performing a k'-NN search in the permutation space (e.g. using the MI-File [4]) or by finding objects with a common permutation prefix (e.g. using the PP-codes [13]). In any case here we assume to have access only to the permutation prefixes and not to the full-length permutations, as done in many PBI approach [4,13,17].

Let $PivotSet(\Pi_{o,l})$ the set of the l closest pivots to the object o, i.e. the pivots whose identifiers appear in the prefix permutation $\Pi_{o,l}$. We assume that the distances between each object and its l closest pivots are stored and indexed within the object prefix permutation. This can be done with a slight modification of the used permutation-based index. In the following, we assume that the objects are indexed using inverted files. Figure 1 shows a naive example for integrating the object-pivot distances into the posting lists, such as the ones used in the MI-file [4]. However, the approach presented in this paper can be extended to cope with different permutation-based indexes.

We propose to refine the candidate result set by selecting the top-k candidate objects ranked according to some pivot-based dissimilarity function. To this end, we tested the *Pivoted embedding* and the *n-Simplex projection* distance bounds, computed using the metric mapping described in Section 3.2 and 3.3. Specifically, at query time, for each object $o \in CandSet(q)$ we approximate the actual distance $d(o, q)$ on the basis of the distances $d(q, p_j)$, $d(o, p_j)$ for $p_j \in \Gamma_{o,q} = PivotSet(\Pi_{q,l}) \cap PivotSet(\Pi_{o,l})$ as follows:

Pivoted embedding - As a consequence of Equation 3 we have

$$\max_{p_j \in \Gamma_{o,q}} |d(o,p_j) - d(q,p_j)| \leq d(o,q) \leq \min_{p_j \in \Gamma_{o,q}} |d(o,p_i) + d(q,p_i)| \qquad (5)$$

so we consider three possible re-rankings of the candidate objects, based on the following dissimilarity measures

$$P_{lwb}(o,q) = \max_{p_j \in \Gamma_{o,q}} |d(o,p_j) - d(q,p_j)| \qquad \text{lower-bound}$$

$$P_{upb}(o,q) = \min_{p_j \in \Gamma_{o,q}} (d(o,p_j) + d(q,p_j)) \qquad \text{upper-bound}$$

$$P_{mean}(o,q) = (P_{upb}(o,q) + P_{lwb}(o,q))/2 \qquad \text{mean}$$

Simplex projection - For each candidate object o, the pivots in $\Gamma_{o,q}$ are used to build a simplex base. The simplex base and the distances $d(o,p_j)$, $d(q,p_j)$ with $p_j \in \Gamma_{o,q}$ are used to compute the apexes $\phi_h(o), \phi_h(q), \phi_h^-(q) \in \mathbb{R}^h$, where $h = |\Gamma_{o,q}| \leq l$. We consider the re-rankings of the candidate objects based on the following dissimilarity measures:

$$S_{lwb}(o,q) = \ell_2(\phi_h(o), \phi_h(q)) \qquad \text{lower-bound}$$

$$S_{upb}(o,q) = \ell_2(\phi_h(o), \phi_h^-(q)) \qquad \text{upper-bound}$$

$$S_{mean}(o,q) = (S_{upb}(o,q) + S_{lwb}(o,q))/2 \qquad \text{mean}$$

The Simplex bounds are highly affected by the number h of pivots used to build the simplex base (the higher h, the tighter the bounds), moreover note that the number h and the used simplex base change when changing the candidate object o. This means that the quality of the simplex-based approximation of the distance $d(o,q)$ may vary significantly when changing the considered candidate object. To overcome this issue, we also considered the re-ranking according to

$$SN_{mean}(o,q) = S_{mean}(o,q)/g(h) \qquad \text{normalized mean}$$

where $g(h)$ is a normalization factor, further discussed in section 5.2.

The lower-bounds S_{lwb} and P_{lwb} are metrics, while the other considered measures are just dissimilarity functions.

Note that for all those approaches no new object-pivot distances are evaluated at either indexing or query time, since the used distances are already computed for building the permutation-based representations of the objects/query. Moreover, the distances $d(o,p_j)$ with $p_j \in \Gamma_{o,q}$ are retrieved while scanning the posting list to build the candidate result set, therefore the considered re-ranking approaches do not require further disk accesses in addition to the index accesses already made to find the candidate results.

5 Experiments

In this section, we evaluate the quality of the re-ranking approach discussed above. We first describe the experimental setup and then we report results and their analysis.

5.1 Experimental settings

The experiments were conducted using three publicly available datasets, namely YFCC100M [21], Twitter-Glove [18], and SISAP colors [15].

YFCC100M collection contains almost 100M images, all uploaded to Flickr between 2004 and 2014. In the experiments, we used a subset of 1M deep Convolutional Neural Network features extracted by Amato et al. [1] and available at `http://www.deepfeatures.org/`. Specifically, we used the activations of the *fc6* layer of the HybridNet [24] after ReLu and ℓ_2 normalization. The resulting features are 4,096-dimensional vectors. We followed the common choice of using the *Euclidean distance* to compare them.

Twitter-GloVe is a collection of 1.2M GloVe [18] features (word embeddings) trained on tweets. We used the 100-dimensional pre-trained word vector available at `https://nlp.stanford.edu/projects/glove/`. These word vectors are often used as vocabulary terms to embed a document into a vector representation, for example by averaging the vectors of the terms contained in the text. In such cases, the space of the vocabulary word embeddings is representative of the space of the document embeddings. We used the *Cosine distance*, i.e. $d_{\mathrm{Cos}}(x,y) = \sqrt{1 - \frac{x \cdot y}{\|x\|_2 \|y\|_2}}$, to compare the GloVe vectors.

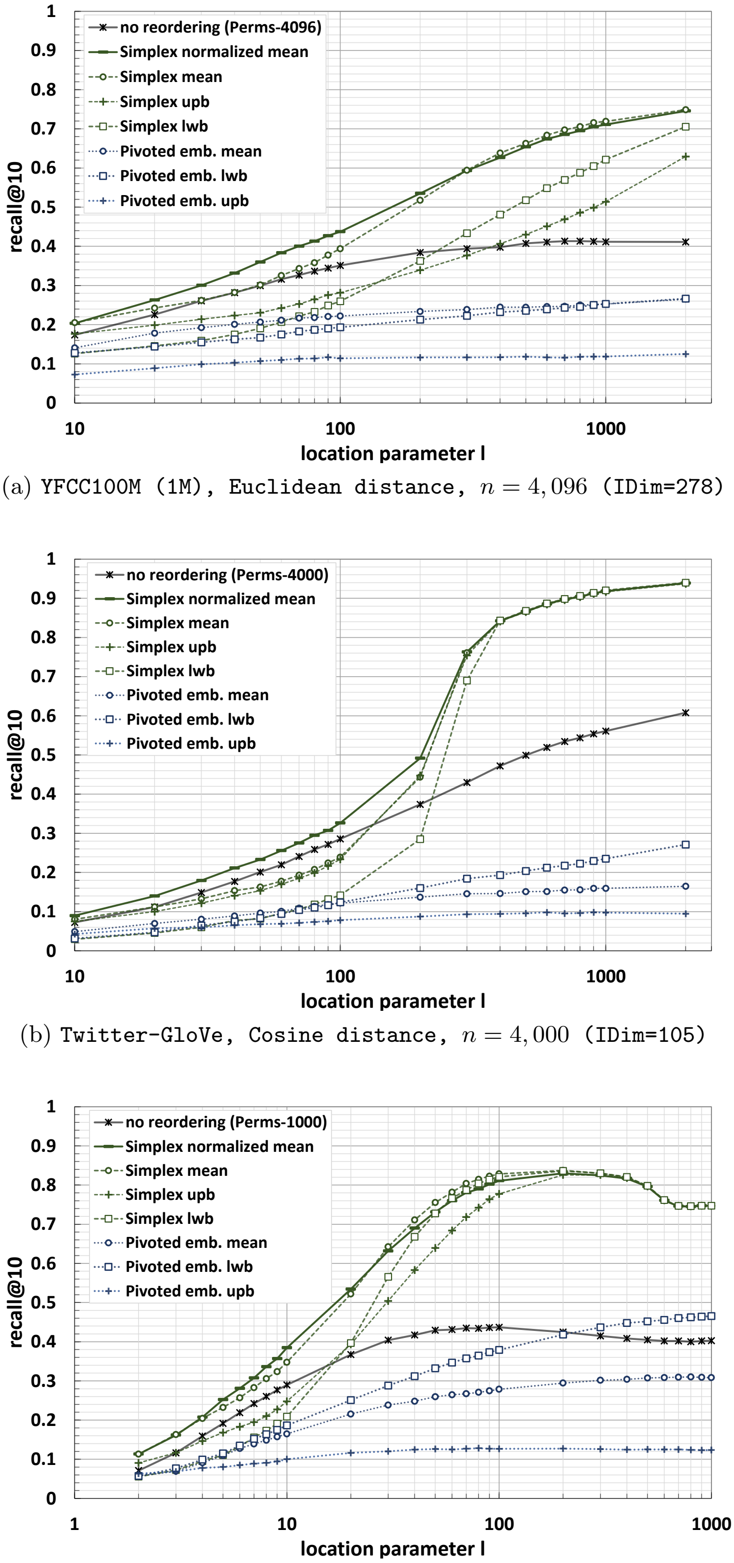
SISAP colors is a commonly used benchmark for metric indexing approaches. It contains about 113K feature vectors of dimensions 112, representing color histograms of medical images. We used the *Jenson-Shannon distance*, defined as in [9], for the feature comparison.

For each dataset we build a ground-truth for exact similarity search related to 1,000 randomly-selected queries. The ground-truths are used to evaluate the quality of the approximate results obtained by re-ranking a permutation-based result set of size $k' \geq k$. Specifically, we select as a candidate result set for a k-NN query the set obtained by performing a k'-NN search in the permutation space. Then we re-rank the candidate results and we select the top-k objects. The quality of the so obtained approximate results was evaluated using the *recall@k*, defined as $|\mathcal{R} \cap \mathcal{R}^A|/k$ where $\mathcal{R}$ is the result set of the exact k-NN search in the original metric space and $\mathcal{R}^A$ is the approximate result set. We set $k = 10$ and $k' = 100$, thus, in order to build the candidate result set, we performed a 100-NN search in the permutation space using the Spearman's rho with location parameter l. In all the tests the pivots were randomly selected. We used about $4,000$ pivots for YFCC100M and Twitter-GloVe, and $n = 1,000$ pivots for the smaller SISAP colors dataset.

5.2 Results

Figure 2 reports the $recall$@10 with respect to the length l of the permutation prefixes used to represent the data objects. Please note that, in each test, we fixed the number n of pivots and we varied the prefix length l.

In order to evaluate the permutation-based results without any re-ranking, we selected the first $k = 10$ objects of the candidate set, ordered according to their permutation-based distance to the query that, in our case, was $S_{\rho,l}$. This baseline

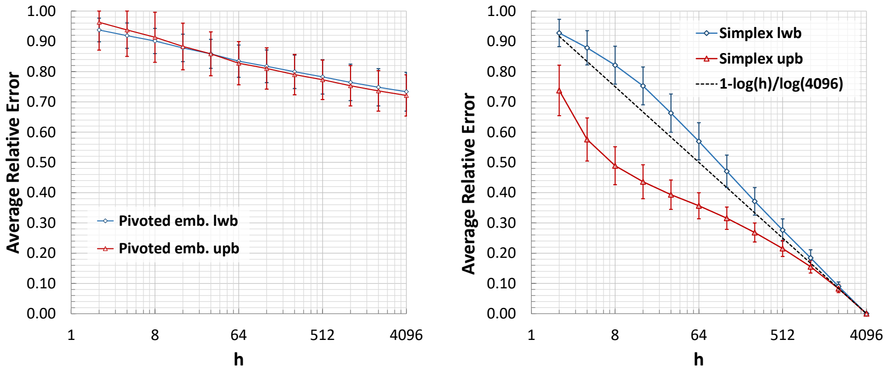


(a) `YFCC100M (1M), Euclidean distance,` $n = 4,096$ `(IDim=278)`

(b) `Twitter-GloVe, Cosine distance,` $n = 4,000$ `(IDim=105)`

(c) `SISAP colors, Jenson-Shannon dist.,` $n = 1,000$ `(IDim=7)`

Fig. 2: *Recall*@10 varying the location parameter l (the number n of pivots is fixed). The candidate set to be reordered is selected with a 100-NN search in the permutation space using the Spearman's rho with location parameter l. In the captions we also report the the Intrinsic Dimensionality (IDim), computed as in [8], for each considered metric space.



(a) `Pivoted embedding` (b) `Simplex projection`

Fig. 3: YFCC100M, Euclidean Distance - Average relative error of the `Pivoted embedding` and `Simplex embedding` distance bounds with respect to the actual distance varying the number of h of pivots used to compute the bounds. Similar trends are obtained on Twitter-GloVe and SISAP colors datasets.

approach is indicated as `no reordering` in the graphs. We compared it with the re-rankings based on the `Pivoted embedding` and the `Simplex projection` distance approximations (lower-bound, upper-bound and mean). In all cases, it is important to keep in mind that, for a fixed value l and for a candidate object o, the number h of pivots used to compute the distance approximations is less than l; moreover, it varies when changing the candidate object because it equals the cardinality of $\Gamma_{o,q}$. So, typically h is greater for objects in top positions in the permutation-based candidate set and decrease for far objects. Moreover, the greater the l, the greater the h and so the better the approximation bounds.

Surprisingly, we observed that in almost all the tested cases the `Pivoted embedding` approach greatly degrades the quality of the permutation-based results. Moreover, on YFCC100M and Twitter-GloVe sets it never reaches a *recall* greater than 0.3. So, in our tests, the `Pivoted` distance approximations resulted to be not adequate for the re-ranking purpose. In fact, the considered lower-bounds approximate well the actual distance $d(o, q)$ only if o and q are very close to each other in the original metric space, or if $\Gamma_{o,q}$ contains at least one pivot that is close to q and far to o (or vice versa). However, for randomly selected pivots in high dimensional space this is unlikely to happen: for a random pivots p and for an object o not so close to q, we often have that the distances $d(o, p)$ and $d(q, p)$ are both close to the mean value in the distribution of the data distances, and so the lower-bound results to be close to zero. This means that, when using the `Pivoted` lower-bound for the re-ranking, many objects may be incorrectly swapped and far objects can be assigned in top-positions. More generally, we observed that the `Pivoted` distance bounds have high relative errors with respect to the actual distance and that these errors slightly decrease when increasing the number h of pivots used to compute the bounds (Figure 3a).

The `Simplex` distance bounds showed similar drawbacks when using relatively small prefix lengths. In particular, they are mostly influenced by the fact that the `Simplex` bounds asymptotically approach the true distances when increasing the number h of pivots used to build the simplex base and that the tightness of the bounds highly depends on h. In fact, in all the cases we observed that there exists a value of $\tilde{h}$ for which full convergence is achieved: 4096 for YFCC100M, around 100 for Twitter-GloVe and SISAP colors. Therefore, for two objects $o, s \in CandSet(q)$ with $|\Gamma_{o,q}| < |\Gamma_{s,q}| << \tilde{h}$ we may have $S_{lwb}(o, q) < S_{lwb}(s, q)$ even if $d(o, q) > d(s, q)$. Moreover, when using few pivots the upper-bound particularly fails in approximate small distances (it is not a metric and in particular $S_{upb}(o, o)$ may be much greater than 0 for small h). The effect of the convergence of the `Simplex` bounds is evident in the Twitter-GloVe data (Figure 2b): for $l \geq 200$ we observed that the number of pivots in the intersection $\Gamma_{o,q}$ starts to exceed $\tilde{h} = 100$ for most of the candidate objects o, thus in that case all `Simplex` bounds provide an exact or almost exact approximation of the actual distances. A similar phenomenon was observed on the SISAP colors.

In general, we experimentally observed that very good re-ranking scores can be obtained using the same simplex base (e.g. the one formed by the pivots in the query permutation prefix) to project all the candidate objects. However, this is not feasible because at query time, for each candidate object o, we had access only to the distances $d(o, p)$ with p appearing in both the object and query permutation prefixes. Thus, the simplex base used (and its dimension h) changes when considering different candidate objects. This means that the "quality" (tightness) of the `Simplex`-based approximations of the distances query-objects is not uniform within the set of the candidate objects. To overcome this issue, we tested normalized versions of the `Simplex` distance bounds that take into account the number of pivots used for projecting the data. In Figure 2, we report the normalized mean that was the one obtaining the best results. As normalization factor we used $g(h) = \log(h)$ since we experimentally observed that the relative errors decrease logarithmically with h (e.g. Figure 3b).

In all the tested cases, the re-ranking using the `Simplex normalized mean` improved the permutation-based results. For example, using about $4,000$ pivots and $l = 300$ the $recall$@10 is improved from 0.39 to 0.59 on YFCC100M dataset, and from 0.43 to 0.76 on Twitter-GloVe. On SISAP colors the recall increase from 0.44 to 0.80 for $l = 80$ and $n = 1,000$. We provided examples with $l < n$ since when using inverted files the number of index blocks accessed is proportional to l^2/n; moreover, it does not depend on the number of retrieved objects.

The disk space needed by the inverted file can be estimated in general assuming to encode each entry of the posting lists with $\lceil log_2|X|\rceil + 32$ bits, where $|X|$ is the size of the dataset. This space is largely sufficient to encode both the ID of the object and its distance from the pivot corresponding to the list to which the entry belongs to. As observed in [4], the positions of the objects can be neglected by ordering the entries of the posting list according to the position of the objects. So, for a fixed l, the size of our index is $l|X|(\lceil log_2|X|\rceil + 32)$ bits, e.g. 1.8 GB for indexing one million objects using $l = 300$. For reference, we observe that for the same set-up the size of the traditional permutation-based inverted index is 0.70 GB. However, more efficient approaches to compressing our posting lists might be employed, for example by quantizing the distances to store each of them in less than 32 bits. In facts, preliminary tests on a 1M subset of YFCC100M deep features, confirmed us that the retrieval performance is preserved when using just 8 bits for storing each distance. In such case, the size of our index is 0.98 GB for one million objects and $l = 300$. For lack of space, we reserve further investigation of this aspect for future work.

Finally, we observe that for $l = 300$ the time cost at query time for computing all the simplex bases and projecting both the query and the candidate objects is about 300ms. As future work, we plan to reduce the query runtime cost by optimizing the construction of the simplex bases, e.g. by re-using some of them instead of computing a new simplex base from scratch for each candidate object.

5.3 Conclusions

In this article, we presented an approach that proposes to exploit the n-Simplex projection to reorder the candidate list of an approximate k-NN-based search system based on permutations, without accessing the original data. However, our approach can be generalized to other types of approximate search provided that they are based on the use of anchor objects from which we must pre-calculate the distances for other purposes. For example, some data structures use inverted indices, as the inverted multi-index [6], in which objects belonging to a Voronoi cell are inserted in a posting list associated with the centroid of the cell from which we calculated the distance. Other indexes that can benefit from our approach are those based on permutation prefix trees, like PP-Index [13] and PPP-Index [17]. We also intend to investigate developments of our approach using the aggregation of the rankings provided by the permutation representations and the rankings obtained with various n-Simplex bounds, using techniques as that proposed in [17], instead of just re-ranking the former one.

Acknowledgements

The work was partially funded by Smart News, “Social sensing for breaking news”, CUP CIPE D58C15000270008, and by VISECH, ARCO-CNR, CUP B56J17001330004.

References

1. G. Amato, F. Falchi, C. Gennaro, and F. Rabitti. YFCC100M-HNfc6: a large-scale deep features benchmark for similarity search. In *Proceedings of SISAP 2016*, pages 196–209. Springer International Publishing, 2016.
2. G. Amato, F. Falchi, C. Gennaro, and L. Vadicamo. Deep Permutations: Deep convolutional neural networks and permutation-based indexing. In *Proceedings of SISAP 2016*, pages 93–106. Springer International Publishing, 2016.
3. G. Amato, F. Falchi, F. Rabitti, and L. Vadicamo. Some theoretical and experimental observations on permutation spaces and similarity search. In *Proceedings of SISAP 2014*, pages 37–49. Springer International Publishing, 2014.

4. G. Amato, C. Gennaro, and P. Savino. MI-File: Using inverted files for scalable approximate similarity search. *Multimed. Tools Appl.*, 71(3):1333–1362, 2014.
5. G. Amato and P. Savino. Approximate similarity search in metric spaces using inverted files. In *Proceedings of InfoScale 2008*, pages 28:1–28:10. ICST, 2008.
6. A. Babenko and V. Lempitsky. The inverted multi-index. In *Proceedings of CVPR 2012*, pages 3069–3076. IEEE, 2012.
7. L. M. Blumenthal. *Theory and applications of distance geometry*. Clarendon Press, 1953.
8. E. Chávez, K. Figueroa, and G. Navarro. Effective proximity retrieval by ordering permutations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(9):1647–1658, 2008.
9. R. Connor, F. A. Cardillo, L. Vadicamo, and F. Rabitti. Hilbert Exclusion: Improved metric search through finite isometric embeddings. *ACM Trans. Inf. Syst.*, 35(3):17:1–17:27, December 2016.
10. R. Connor, L. Vadicamo, F. A. Cardillo, and F. Rabitti. Supermetric search with the four-point property. In *Proceedings of SISAP 2016*, pages 51–64. Springer International Publishing, 2016.
11. R. Connor, L. Vadicamo, F. A. Cardillo, and F. Rabitti. Supermetric search. *Information Systems*, 2018.
12. R. Connor, L. Vadicamo, and F. Rabitti. High-dimensional simplexes for supermetric search. In *Proceedings of SISAP 2017*, pages 96–109. Springer International Publishing, 2017.
13. A. Esuli. Use of permutation prefixes for efficient and scalable approximate similarity search. *Information Processing & Management*, 48(5):889–902, 2012.
14. R. Fagin, R. Kumar, and D. Sivakumar. Comparing top k lists. In *Proceedings of SODA 2003*, pages 28–36. Society for Industrial and Applied Mathematics, 2003.
15. K. Figueroa, G. Navarro, and E. Chávez. Metric spaces library. `www.sisap.org/library/manual.pdf`, 2007.
16. M. L. Micó, J. Oncina, and E. Vidal. A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing time and memory requirements. *Pattern Recogn. Lett.*, 15(1):9–17, January 1994.
17. D. Novak and P. Zezula. PPP-codes for large-scale similarity searching. In *TLDKS XXIV*, pages 61–87. Springer Berlin Heidelberg, 2016.
18. J. Pennington, R.Socher, and C. D. Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of EMNLP 2014*, pages 1532–1543, 2014.
19. V. Pestov. Indexability, concentration, and vc theory. *J. Discrete Algoritms*, 13:2–18, 2012.
20. I. J. Schoenberg. Metric spaces and completely monotone functions. *Annals of Mathematics*, 39(4):811–841, 1938.
21. B. Thomee, B. Elizalde, D. .A Shamma, K. Ni, G. Friedland, D. Poland, D. Borth, and L-J Li. YFCC100M: The new data in multimedia research. *Commun. ACM*, 59(2):64–73, 2016.
22. R. Weber, H-J Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *VLDB*, volume 98, pages 194–205, 1998.
23. P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity search: the metric space approach*, volume 32. Springer Science & Business Media, 2006.
24. B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems 27*, pages 487–495. Curran Associates, Inc., 2014.