This is a post-peer-review, pre-copyedit version of a paper published in Amato G., Gennaro C., Oria V., Radovanović M. (eds) *Similarity Search and Applications. SISAP 2019*. Lecture Notes in Computer Science, vol 11807. Springer, Cham. The final authenticated version is available online at: https://doi.org/10.1007/978-3-030-32047-8_4

SPLX-Perm: A Novel Permutation-Based Representation for Approximate Metric Search

Lucia Vadicamo[1], Richard Connor[2], Fabrizio Falchi[1], Claudio Gennaro[1], and Fausto Rabitti[1]

[1] Institute of Information Science and Technologies (ISTI), CNR, Pisa, Italy
[2] Division of Mathematics and Computing Science, University of Stirling, Scotland
{name.surname}@isti.cnr.it, richard.connor@stir.ac.uk

Abstract. Many approaches for approximate metric search rely on a permutation-based representation of the original data objects. The main advantage of transforming metric objects into permutations is that the latter can be efficiently indexed and searched using data structures such as inverted-files and prefix trees. Typically, the permutation is obtained by ordering the identifiers of a set of pivots according to their distances to the object to be represented. In this paper, we present a novel approach to transform metric objects into permutations. It uses the object-pivot distances in combination with a metric transformation, called n-Simplex projection. The resulting permutation-based representation, named *SPLX-Perm*, is suitable only for the large class of metric space satisfying the n-point property. We tested the proposed approach on two benchmarks for similarity search. Our preliminary results are encouraging and open new perspectives for further investigations on the use of the n-Simplex projection for supporting permutation-based indexing.

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

1 Introduction

Searching a data set for the most similar objects to a given query is a fundamental task in computer science. Over the years several methods for *exact similarity search* were proposed in the literature. These approaches guarantee to find the true result set. However, they scale poorly with the dimensionality of the data (a phenomenon known as "*curse of dimensionality*") and mostly they are not convenient to deal with very large data sets. To overcome these issues, the research community has developed a wide spectrum of techniques for *approximate similarity search*, which have higher efficiency though at the price of some imprecision in the results (e.g. some relevant results might be missing or some ranking errors might occur). Among them, we can distinguish between 1) approaches specialised for a particular kind of data (e.g. Euclidean vectors), and 2) techniques applicable to generic metric data objects. assessing the dissimilarity of any two objects. The advantage of the former class of approaches,

like the Product Quantization [13] and the Inverted Multi-Index [5], is that they have very high efficiency and effectiveness. However, the engineering effort to design a method specialised for any particular data or application is typically too high. The metric approaches, instead, overcome this issue since they are applicable to generic metric objects without assuming a prior knowledge of the nature of the data. Successful examples of metric approximate indexing and searching techniques are the *Permutation-based Indexing* (PBI) ones, such as [4,7,14].

PBI techniques leverage the idea of transforming each metric object into a permutation of a finite set of integers in such a way that similar objects have similar permutations. The main advantage is that the permutations can be efficiently indexed and searched, e.g., using inverted files. The similarity queries are then performed in the permutation space by selecting objects whose permutations are the most similar to the query permutation. The common approach to generate a permutation-based representation of a data object is based on selecting a finite set of *pivots* (reference objects) and measuring the distances of each pivot to the object to be represented: the permutation is obtained as the list of the pivot identifiers ordered according to their distance to the object.

The main contribution of this paper is describing a novel approach to generate permutations associated with metric data objects. The proposed technique is applicable only to the large class of metric spaces satisfying the so-called *n-point property* [8,6]. This class encompasses many commonly used metric spaces, such as Cartesian spaces of any dimensionality regarded with the Euclidean, Cosine, Jensen-Shannon or Quadratic Form distances, and more generally any Hilbert-embeddable space [6]. Our technique exploits the *n-Simplex projection* [10], which is a metric transformation that allows projecting the data objects into a finite-dimensional Euclidean space. Starting from the idea that this space transformation maps similar objects into similar Euclidean vectors, we propose to process each projected vector to further generate a permutation-based representation. We show that, in most of the tested cases, our permutations are more effective than traditional permutations. Therefore, we believe that our technique may be relevant for many permutation-based indexing and searching techniques, even though we are aware that it may require more work to mature.

2 Background

We are interested in searching a (large) finite subset of a metric space (D, d), where D is a domain of objects and $d : D \times D \rightarrow \mathbb{R}^+$ is a metric function [15]. Many methods for approximate metric search rely on transforming the original data objects into a more tractable space, e.g. by exploiting the distances to a set of pivots. In the following, we summarise key concepts of two pivot-based approaches that transform metric objects into permutations and Euclidean vectors, respectively.

Permutation-based representation. For a given metric space (D, d) and a set of pivots $\{p_1, \ldots, p_n\} \subset D$, the traditional permutation-based representation Π_o (briefly *permutation*) of an object $o \in \mathcal{D}$ is the sequence of the pivots

identifiers $\{1,\ldots,n\}$ ordered by their distance to o. Formally, the permutation $\Pi_o = [\Pi_o(1), \Pi_o(2), ..., \Pi_o(n)]$ lists the pivot identifiers in an order such that $\forall\, i \in \{1,\ldots,n-1\}$, $d(o, p_{\Pi_o(i)}) \leq d(o, p_{\Pi_o(i+1)})$. An equivalent representation is the *inverted permutation* Π_o^{-1} whose i-th element denotes the position of the pivot p_i in the permutation Π_o.

Most of the PBI methods, e.g. [4,11,14], use only a fixed-length prefix of the permutations to represent and compare objects. It means that only the positions of the nearest l out of n pivots are used for the data encoding. In this work, we do the same since often the prefix-permutations have better or similar effectiveness than the full-length permutations [4], resulting also in a more compact data encoding. The prefix permutations are compared using *top-l distances* [12]. We use the Spearman Rho with location parameter l, defined as $S_{\rho,l}(\Pi_{o_1}, \Pi_{o_2}) = \ell_2(\Pi^{-1}_{o_1,l}, \Pi^{-1}_{o_2,l})$, where $\Pi^{-1}_{o,l}$ is the *inverted prefix permutation*:

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

n-Simplex Projection. Recently, Connor et al. [8,9,10] investigated how to enhance the metric search on a class of spaces meeting the so-called n-point property, which is a geometrical property stronger than the triangle inequality. A metric space has the n-point property if for any finite set of n objects there exists an *isometric* embedding of those objects into a $(n-1)$-dimensional Euclidean space. They exploited this property to define a space transformation, called *n-Simplex projection*, that allows a metric space to be trasformed into a finite-dimensional Euclidean space. It uses the distances to a set of pivots $\mathcal{P}_n = \{p_1,\ldots,p_n\}$ for mapping metric objects to Euclidean vectors. Formally, the n-Simplex projection associated with the pivot set $\mathcal{P}_n$ is the transformation

$$\begin{aligned} \phi_{\mathcal{P}_n} : (D, d) &\rightarrow (\mathbb{R}^n, \ell_2) \\ o &\mapsto v_o \end{aligned}$$

where v_o is the only vector with a positive last component that preserves the distances of the data object to the pivots, i.e. $\ell_2(v_o, v_{p_i}) = d(o,p)$, $\forall\, i \in \{1,\ldots,n\}$. The algorithm to compute the n-Simplex projected vectors is described in [10].

We recall that one interesting outcome of this space transformation is that the Euclidean distance between any two projected vectors is a lower-bound of the actual distance, and that the lower-bound converges to the actual distance for increasing number of pivots n. Thus, the larger the n the better the preservation of the similarities between the data objects.

3 SPLX-Perm Representation

As recalled above, the traditional approach to associate a permutation to a data object is sorting a set of pivot identifiers in ascending order with respect to the distances of those pivots to the object to be represented. This approach is

Algorithm 1: SPLX-Perm computation

Input : $\mathcal{P}_n = \{p_1, \ldots, p_n\} \subset D$, $o \in D$
Output: The SPLX-Perm Π_o associated to the the object o

1 $v_o \leftarrow \phi_{\mathcal{P}_n}(o)$; `// n-Simplex projection into` $\mathbb{R}^n$
2 $v_o \leftarrow R\, v_o$; `// Rotate the vector using a random rotation matrix` R
3 $[v_{sorted}, v_{index}] = \text{sort}(v_o, \text{ascending})$; `// sorts the vector elements of` v_o `in ascending order;` v_{sorted} `is the sorted array,` v_{index} `is the sort index vector describing the rearrangement of each element of` v_o
4 $\Pi_o \leftarrow v_{index}$

justified by the observation that objects very close to each other should have similar relative distances to the pivots, and thus, similar permutations.

The main goal of such kind of metric transformation is that the similarity between the permutations reflects as much as possible the similarity of the original data objects. Starting from this concept we observe that, on one hand, the traditional permutation representation takes in consideration only the relative distances to the pivots, i.e. which is the closest pivot, the second closest pivot, etc. On the other hand, the recently proposed n-Simplex projection maps the data objects to Euclidean vectors by taking into consideration both object-pivot and pivot-pivot distances. Moreover, the Euclidean distance between those projected vectors well approximates the actual distance, especially when using a large number of pivots. Therefore, our idea is to start from these good approximations of the data objects and further transform them into permutations. Since we are now working in a Euclidean space, and the Euclidean distance does not mix the contribution of values in different dimensions of the vectors, it is reasonable to think that two vectors are very close to each other if they have similar components in each dimension. By exploiting this idea, we propose to generate the permutations by ordering the dimensional indexes of the n-Simplex projected vectors in ascending order with respect to their corresponding values. For example, the Euclidean vector $[0.4, 1.6, 0.3, 0.5]$ is transformed into the permutation $[3, 1, 4, 2]$, since the third element of the vector is the smallest one, the first element is the second smallest one, and so on.

The idea of generating a permutation from a Euclidean vector by ordering its dimensional indexes was investigated also in [2], where only the case of features extracted from images using a deep Convolutional Neural Network was analysed. Moreover, in [2] the intuition was that individual dimensions of the deep feature vectors represent some sort of visual concepts and that the value of each dimension specifies the importance of that visual concept in the image. Here we observe that a similar approach can be applied to Euclidean data in general, and thanks to the use of the n-Simplex projection it can be extended to a large class of metric objects as well. The only problem on applying this approach on general Euclidean vectors is that the variance of the values in a given dimensional position might be very different when varying the considered position. This happens, for example, in the case of vectors obtained using the

Principal Component Analysis where elements in the first dimensional positions have higher variance than elements in the other dimensions. Other examples are the vectors obtained with the n-Simplex projection that, by construction, have higher values in top position and values that decrease to zero in the last components. To overcome this issue we propose to randomly rotate the vectors before transforming them into permutations. In facts, the random rotation distributes the information equally along all the dimensions of the vectors while preserving the Euclidean distance.

In summary, given a set of pivots $\mathcal{P}_n$, the proposed approach to associate a permutation to an object $o \in D$ is 1) compute the n-Simplex projected vector $\phi_{\mathcal{P}_n}(o)$; 2) randomly rotate the obtained vector (the same rotation matrix is used for all the data objects); 3) generate the permutation by ordering the values of the rotated vectors. We use the term *SPLX-Perms* for referring to the so obtained permutations (a pseudo-code is reported in Algorithm 1).

4 Experiments

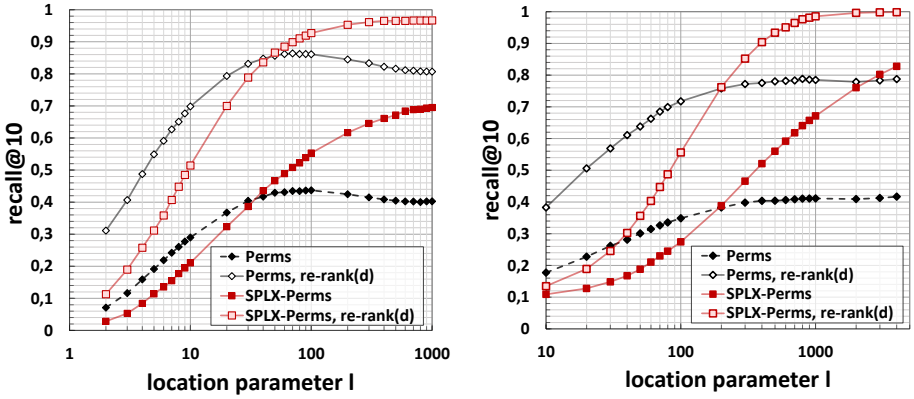
We compared our permutation representations (`SPLX-Perms`) with the traditional permutation-based representations (`Perms`) in an approximate similarity search scenario. The experiments were conducted on two publicly available data sets:

SISAP colors is a benchmark for metric indexing. It contains about 113K color histograms of medical images, each represented as 112-dimensional vector.

YFCC100M is a collection of almost 100M images from Flickr. 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 [16] after ReLu and ℓ_2 normalization. The resulting features are 4,096-dimensional vectors.

The metrics used in the experiments are the *Jensen-Shannon distance* for the SISAP colors data, and the *Euclidean distance* for the YFCC100M deep features. For each data set, we considered 1,000 randomly selected queries and we built the ground-truth for the exact k-NN query search. The approximate results set for a given query is selected by performing the k-NN search in the permutation space. The quality of the approximate results was evaluated using the *recall@k*, that is $|\mathcal{R} \cap \mathcal{R}^A|/k$ where $\mathcal{R}$ is the result set of the exact k-NN search, and $\mathcal{R}^A$ is the set of the k approximate results.

To have a better overview of the tested approaches, we also consider the case in which the permutations are used to select a *candidate result set* to be re-ranked using the original distance d. In such cases, a k'-NN search (with $k' > k$) is performed in the permutation space in order to select the candidate result set. Then the candidate results are re-ranked according to the actual distance d, and the top-k objects are selected to form the final approximate result set $\mathcal{R}^A$. In the experiments, we used $k' = 100$ and $k = 10$, if not specified otherwise.



(a) `SISAP colors, Jensen-Shannon dist.`, $n = 1,000$

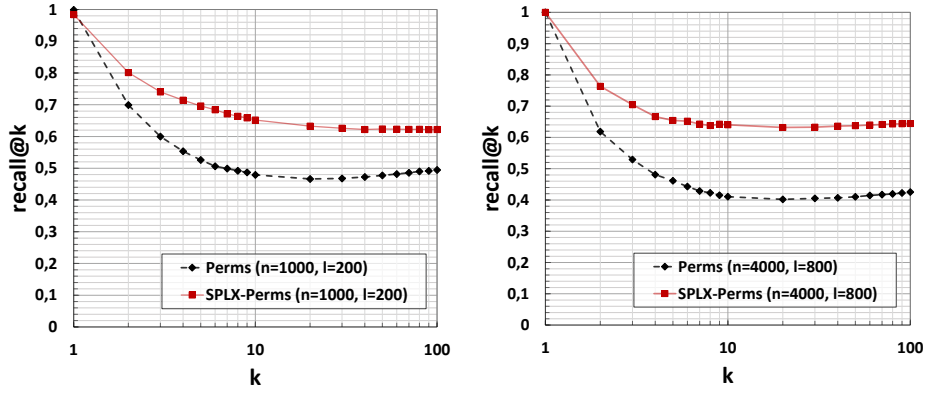
(b) `YFCC100M(1M), Euclidean dist.`, $n = 4,000$

Fig. 1: *Recall*@10 varying the location parameter l (i.e. the prefix length).

To generate the permutation-based representations we used $n = 1,000$ pivots for the SISAP Colors data set, and $n = 4,000$ pivots for the YFCC100M data set. We tested the quality of the results obtained using either the full-length permutations or a fixed-length prefix of the permutations. The metric used in the permutation space is the *Spearman's rho with location parameter* l, where the location parameter l is the length of the prefix permutation.

4.1 Results

Figures 1a and 1b show the *recall*@10 for the SISAP Colors and YFCC100M data sets, respectively. Lines "`Perms`" and "`SPLX-Perms`" refer to the cases in which the permutations are used to select the approximate result set by performing a 10-NN search in the permutation space. Lines "`Perms, re-rank(d)`" and "`SPLX-Perms, re-rank(d)`" refer to the cases in which the permutation-representation are used to select a candidate result set (obtained by performing a 100-NN search) that is then re-ranked using the actual distance d. It is interesting to note that on YFCC100M data, our *full-length* SPLX-Perms representation allowed us to achieve a recall that not only is better than that achieved using the traditional full-length permutation, but it is even better than that obtained by the re-ranked approach. However, we also observe that for very short prefix-lengths the traditional permutations shown better performance than our technique. Another interesting aspect is that when considering the traditional permutation-based representation there is usually an optimal prefix length $l < n$ for which the best recall is achieved or for which the recall curve shows a plateau. This is evident in Fig. 1, where the recall lightly decrease as the location parameter l grows. This is a phenomenon experimentally observed also in other data



(a) `SISAP Colors, Jensen-Shannon dist.` (b) `YFCC100M(1M), Euclidean dist.`

Fig. 2: *Recall@k* varying k (fixed location parameter l)

sets as shown in several works (see e.g., [3,4]). Our SPLX-Perms seems to be not affected by this phenomenon since its recall increases when considering larger l. Moreover, the re-ranking of candidate results selected using our permutations achieved a recall very close to one for large prefix-lengths.

In Figure 2, we also report the *recall@k* with k ranging from 1 to 100 for the baselines approaches (i.e. without considering the re-ranking phase) using a fixed prefix-length l. We can see that the improvement of the proposed approach over the traditional permutation-based representation holds for all ks. Our SPLX-Perm representation seems also to be more stable and provides recall values that are up to 1.6 times higher than that obtained using traditional permutations.

5 Conclusions

In this paper, we presented a novel permutation-based representation for metric objects, called SPLX-Perm. It exploits the n-Simplex projection to map the data object to Euclidean vectors, which are in turn transformed into permutations. The approach used to transform the Euclidean vectors into permutations has some analogies with the Deep Permutation approach that was proposed in [2] for associating permutations to visual deep features. To some extent, our work can be viewed as a generalisation of this technique to the large class of metric space meeting the n-point property. Our preliminary results show that our SPLX-Perms are more effective than the traditional permutations, even if there are some drawbacks with respect to the traditional permutations: 1) worse performance for very small prefix permutation; 2) higher cost for generating the SPLX-Perm since for each object we need to compute both the object-pivot distances and the n-Simplex projection. Nevertheless, we believe that our technique

as a lot of potentialities and deserves further investigations. In this perspective, we plan to extend our experimental evaluation on more data sets and metrics, using a different prefix length for the query object (to reduce the search cost) and using a pivot selection specifically designed for the n-simplex projection.

Acknowledgements

This work was partially supported by VISECH ARCO-CNR, CUP B56J17001330004, the AI4EU project, funded by the EC (H2020 - Contract n. 825619), and the Short-Term-Mobility (STM) program of the CNR.

References

1. Amato, G., Falchi, F., Gennaro, C., Rabitti, F.: YFCC100M-HNfc6: a large-scale deep features benchmark for similarity search. In: Proceedings of SISAP 2016. pp. 196–209. Springer International Publishing (2016)
2. Amato, G., Falchi, F., Gennaro, C., Vadicamo, L.: Deep Permutations: Deep convolutional neural networks and permutation-based indexing. In: Proceedings of SISAP 2016. pp. 93–106. Springer International Publishing (2016)
3. Amato, G., Falchi, F., Rabitti, F., Vadicamo, L.: Some theoretical and experimental observations on permutation spaces and similarity search. In: Proceedings of SISAP 2014. pp. 37–49. Springer (2014)
4. Amato, G., Gennaro, C., Savino, P.: MI-File: Using inverted files for scalable approximate similarity search. Multimed. Tools and Appl. **71**(3), 1333–1362 (2014)
5. Babenko, A., Lempitsky, V.: The inverted multi-index. IEEE Trans. Pattern Anal. Mach. Intell. **37**(6), 1247–1260 (June 2015)
6. Blumenthal, L.M.: Theory and applications of distance geometry. Clarendon Press (1953)
7. Chavez, E., Figueroa, K., Navarro, G.: Effective proximity retrieval by ordering permutations. IEEE Trans. Pattern. Anal. Mach. Intell. **30**(9), 1647–1658 (2008)
8. Connor, R., Cardillo, F.A., Vadicamo, L., Rabitti, F.: Hilbert Exclusion: Improved metric search through finite isometric embeddings. ACM Trans. Inf. Syst. **35**(3), 17:1–17:27 (Dec 2016)
9. Connor, R., Vadicamo, L., Cardillo, F.A., Rabitti, F.: Supermetric search. Information Systems (2018)
10. Connor, R., Vadicamo, L., Rabitti, F.: High-dimensional simplexes for supermetric search. In: Proceedings of SISAP 2017. pp. 96–109. Springer (2017)
11. Esuli, A.: Use of permutation prefixes for efficient and scalable approximate similarity search. Information Processing & Management **48**(5), 889–902 (2012)
12. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. In: Proceedings of SODA 2003. pp. 28–36. Society for Industrial and Applied Mathematics (2003)
13. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. IEEE Trans. Pattern Anal. Mach. Intell. **33**(1), 117–128 (Jan 2011)
14. Novak, D., Zezula, P.: PPP-codes for large-scale similarity searching. In: TLDKS XXIV. pp. 61–87. Springer (2016)
15. Zezula, P., Amato, G., Dohnal, V., Batko, M.: Similarity search: the metric space approach, vol. 32. Springer Science & Business Media (2006)
16. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. In: Proceedings of NIPS 2014, pp. 487–495. Curran Associates, Inc. (2014)