Noname manuscript No.
(will be inserted by the editor)

Sharing Hash Codes for Multiple Purposes

Wiktor Pronobis[†] · Danny Panknin[†] · Johannes Kirschnick[†] · Vignesh Srinivasan · Wojciech Samek · Volker Markl · Manohar Kaul · Klaus-Robert Müller[*] · Shinichi Nakajima

Received: date / Accepted: date

Abstract Locality sensitive hashing (LSH) is a powerful tool in data science, which enables sublinear-time approximate nearest neighbor search. A variety of hashing schemes have been proposed for different dissimilarity measures. However, hash codes significantly depend on the dissimilarity, which prohibits

[†] contributed equally.

[*] corresponding author.

Wikor Pronobis, Danny Panknin, Klaus-Robert Müller, Shinichi Nakajima
Technische Unversität Berlin, Machine Learning Group, Marchstr. 23, 10587
Tel.: +49-30-314 78620
Fax: +49-30-314 78622
E-mail: wiktor.pronobis@gmail.com, {danny.panknin, klaus-robert.mueller, nakajima}@tu-berlin.de

Johannes Kirschnick
DFKI, Language Technology Lab, Alt-Moabit 91c, Berlin, Germany
E-mail: johannes.kirschnick@dfki.de

Vignesh Srinivasan, Wojciech Samek
Fraunhofer HHI
E-mail: {vignesh.srinivasan@hhi-extern, wojciech.samek@hhi}.fraunhofer.de

Volker Markl
Technische Unversität Berlin, Database Systems and Information Management Group, Einsteinufer 17, 10587 Berlin, Germany
E-mail: volker.markl@tu-berlin.de

Manohar Kaul
IIT Hyderabad
E-mail: mkaul@iith.ac.in

Klaus-Robert Müller
Korea University, and with Max Planck Society

Wojciech Samek, Volker Markl, Klaus-Robert Müller, Shinichi Nakajima
Berlin Big Data Center

Shinichi Nakajima
Center for Advanced Intelligence Project (AIP), RIKEN

users from adjusting the dissimilarity *at query time*. In this paper, we propose *multiple purpose LSH* (mp-LSH) which shares the hash codes for different dissimilarities. mp-LSH supports L2, cosine, and inner product dissimilarities, and their corresponding weighted sums, where the weights can be adjusted at query time. It also allows us to modify the importance of pre-defined groups of features. Thus, mp-LSH enables us, for example, to retrieve similar items to a query with the user preference taken into account, to find a similar material to a query with some properties (stability, utility, etc.) optimized, and to turn on or off a part of multi-modal information (brightness, color, audio, text, etc.) in image/video retrieval. We theoretically and empirically analyze the performance of three variants of mp-LSH, and demonstrate their usefulness on real-world data sets.

Keywords Locality Sensitive Hashing · Approximate Near Neighbor Search · Information Retrieval · Collaborative Filtering

1 Introduction

Statistics and probability theory have been playing the central role in machine learning, artificial intelligance, and related application fields, e.g., text analytics, computer vision, information retrieval, computational biology, and data mining [16, 7]. When the data size and the complexity of the statistical model were moderate, typical machine learning problems such as clustering, regression, and classification were solved by (explicitly or implicitly) estimating the probability distribution.

In recent years when those research fields are generically called *data science*, large amounts of data are used to train statistical models with very high complexity. This arose from the rapid progress of semiconductor devices (CPUs/GPUs, memory, communication devices, etc.), and the breakthrough with deep neural networks, where complex deep architectures have been proven to learn highly non-linear fine structure of data from massive data, further accelerated the demand of large models that can be trained on big data [19, 4, 29, 23, 5, 35].

Rapid increase of data size also necessitated new technologies for basic tools in data analysis. Nearest neighbor search (NNS), which is intensively used in data science, is one of them. In retrieval systems and recommender systems, NNS is used to find items which are closest to (or best match) a given query. NN classifiers have been shown to perform comparably to the state-of-the-art multi-class classifiers [43], which implies that NNS can well approximate (or reflect) the probability distribution when the number of training samples is sufficiently large. NNS has also shown to be useful in extreme classification, where the number of classes is extremely large [42].

Since NNS is required to perform on millions to billions of samples within a few seconds in some real time applications, a naive implementation with linear complexity can be too slow. Thus sublinear methods have become important analysis tools. Locality sensitive hashing (LSH), one of the key technologies

for big data analysis, enables approximate nearest neighbor search (ANNS) in *sublinear* time [20,44]. With LSH functions for a required dissimilarity measure in hand, each data sample is assigned to a *hash bucket* in the pre-processing stage. At runtime, ANNS can be performed by restricting the search to the samples that lie within the hash bucket, to which the query point is assigned, along with the samples lying in the neighbouring buckets. Probability theory provided theoretical guarantees of ANNS performance with LSH [20]. A variety of LSH schemes have been proposed for different dissimilarity measures, including Jaccard distance [8], L_p distance [12], cosine distance [10], chi-squared distance [15], distance to a hyperplane [21], and inner product dissimilarity (maximum inner product search) [36,2,37,32].

A drawback of the existing LSH schemes is that each LSH scheme is specialized for each dissimilarity measure. This can limit the flexibility of the use of LSH. For some data collections, the objective can be clearly expressed from the start, for example, text/image/video/speech analysis. In such cases, the dissimilarity measure can be fixed when LSH codes are given to each sample. However, in other cases such as drug discovery, the material genome project, or climate analysis, the ultimate query structure to such data may still not be fully fixed. In other words, measurements, simulations or observations may be recorded without being able to spell out the full specific purpose (although the general goal, e.g., producing better drugs, finding more potent materials, or detecting anomaly, is clear). Motivated by the latter case, we consider how one can use LSH schemes without defining any specific dissimilarity at the data acquisition and pre-processing phase.

A challenge in developing LSH without defining specific purpose is that the existing LSH schemes, designed for different dissimilarity measures, provide significantly different hash codes. Therefore, a naive realization requires us to prepare the same number of hash tables as the number of possible target dissimilarities, which is not realistic if we need to adjust the importance of multiple criteria. In this paper, we propose three variants of multiple purpose LSH (mp-LSH), which support L2, cosine, and inner product (IP) dissimilarities, and their weighted sums, where the weights can be adjusted at query time.

The first proposed method, called mp-LSH with vector augmentation (mp-LSH-VA), maps the data space into an augmented vector space, so that the squared-L2-distance in the augmented space matches the required dissimilarity measure up to a constant. This scheme can be seen as an extension of recent developments of LSH for maximum IP search (MIPS) [36,2,37,32]. The significant difference from the previous methods is that our method is designed to modify the dissimilarity by changing the augmented query vector. We show that mp-LSH-VA is locality sensitive for L2 and IP dissimilarities and their weighted sums. However, its performance for the L2 dissimilarity is significantly inferior to the standard L2-LSH [12]. In addition, mp-LSH-VA does not support the cosine-distance.

Our second proposed method, called mp-LSH with code concatenation (mp-LSH-CC), concatenates the hash codes for L2, cosine, and IP dissimilarities, and constructs a special structure, called *cover tree* [9], which enables efficient

NNS with the weights for the dissimilarity measures controlled by adjusting the metric in the code space. Although mp-LSH-CC is conceptually simple and its performance is guaranteed by the original LSH scheme for each dissimilarity, it is not memory efficient, which also results in increased query time.

Considering the drawbacks of the aforementioned two variants led us to our final and recommended proposal, called mp-LSH with code augmentation and transformation (mp-LSH-CAT). It supports L2, cosine, and IP dissimilarities by augmenting the hash codes, instead of the original vector. mp-LSH-CAT is memory efficient, since it shares most information over the hash codes for different dissimilarities, so that the augmentation is minimized.

We theoretically and empirically analyze the performance of mp-LSH methods, and demonstrate their usefulness on real-world data sets. Our mp-LSH methods also allow us to modify the importance of pre-defined groups of features. Adjustability of the dissimilarity measure at query time is not only useful in the absence of future analysis plans, but also applicable to multi-criteria searches. The following lists some sample applications of multi-criteria queries in diverse areas:

1. In recommender systems, suggesting items which are similar to a user-provided query and also match the user's preference.
2. In material science, finding materials which are similar to a query material and also possess desired properties such as stability, conductivity, and medical utility.
3. In video retrieval, we can adjust the importance of multimodal information such as brightness, color, audio, and text at query time.

Related Work: After the theoretical relation between the performance of approximate nearest neighbor search and the locality sensitivity of hash functions was established [20], a lot of LSH schems have been proposed for different dissimilarity measures, including Jaccard distance [8], L_p distance [12], cosine distance [10], chi-squared distance [15], distance to a hyperplane [21], and inner product dissimilarity (maximum inner product search) [36, 2, 37, 32]. They are categorized as *data independent* hashing methods where each sample is given a hash code, independently from the other samples [44].

On the other hand, *data dependent* hashing methods have recently been intensively developed, where the code is optimized for the sample distribution. Some of those methods learn the sample distribution by using unsupervised machine learning tools, e.g., PCA [27] and ICA [17], while others additionally use label information by supervised methods, e.g., LDA [41], kernel methods [25], and neural networks [24]. In general, data dependent methods improve the accuracy of the data independent counterpart by learning the sample distribution, while they are less flexible because hashing procedure is fixed only after most of the samples are captured, i.e., they are not suitable for the streaming setting where each sample should be given a hash code right after it is acquired, without waiting the whole data collection process to be completed. In this paper, we propose data independent LSH methods, and therefore, the data dependent methods are out of scope.

Some hashing methods cope with multi-modal data [40,31,45], most of which however are data dependent and do not offer adjustability of the importance weights at query time. To the best of our knowledge, no existing hashing methods can cope with different dissimilarity measures with the weights adjustable at query time.

2 Background

In this section, we briefly overview previous locality sensitive hashing (LSH) techniques.

Assume that we have a sample pool $\mathcal{X} = \{\boldsymbol{x}^{(n)} \in \mathbb{R}^L\}_{n=1}^N$ in L-dimensional space. Given a query $\boldsymbol{q} \in \mathbb{R}^L$, nearest neighbor search (NNS) solves the following problem:

$$\boldsymbol{x}^* = \operatorname*{argmin}_{\boldsymbol{x} \in \mathcal{X}} \mathcal{L}(\boldsymbol{q}, \boldsymbol{x}), \tag{1}$$

where $\mathcal{L}(\cdot, \cdot)$ is a dissimilarity measure. A naive approach computes the dissimilarity from the query to all samples, and then chooses the most similar samples, which takes $O(N)$ time. On the other hand, approximate NNS can be performed in sublinear time. We define the following three terms:

Definition 1 (S_0-near neighbor) For $S_0 > 0$, $\boldsymbol{x}$ is called S_0-near neighbor of $\boldsymbol{q}$, if $\mathcal{L}(\boldsymbol{q}, \boldsymbol{x}) \leq S_0$.

Definition 2 (c-approximate nearest neighbor search) Given $S_0 > 0$, $\delta > 0$, and $c > 1$, c-approximate nearest neighbor search (c-ANNS) reports some cS_0-near neighbor of $\boldsymbol{q}$ with probability $1 - \delta$, if there exists an S_0-near neighbor of $\boldsymbol{q}$ in $\mathcal{X}$.

Definition 3 (Locality sensitive hashing) A family $\mathcal{H} = \{h : \mathbb{R}^L \to \mathcal{K}\}$ of functions is called (S_0, cS_0, p_1, p_2)-sensitive for a dissimilarity measure $\mathcal{L} : \mathbb{R}^L \times \mathbb{R}^L \to \mathbb{R}$, if the following two conditions hold for any $\boldsymbol{q}, \boldsymbol{x} \in \mathbb{R}^L$:

- if $\mathcal{L}(\boldsymbol{q}, \boldsymbol{x}) \leq S_0$ then $\mathbb{P}\left(h(\boldsymbol{q}) = h(\boldsymbol{x})\right) \geq p_1$,
- if $\mathcal{L}(\boldsymbol{q}, \boldsymbol{x}) \geq cS_0$ then $\mathbb{P}\left(h(\boldsymbol{q}) = h(\boldsymbol{x})\right) \leq p_2$,

where $\mathbb{P}(\cdot)$ denotes the probability of the event (with respect to the random draw of hash functions).

Note that $p_1 > p_2$ is required for LSH to be useful. The image $\mathcal{K}$ of hash functions is typically binary or integer. The following proposition guarantees that locality sensitive hashing (LSH) functions enable c-ANNS in sublinear time.

Proposition 1 *[20] Given a family of (S_0, cS_0, p_1, p_2)-sensitive hash functions, there exists an algorithm for c-ANNS with $O(N^\rho \log N)$ query time and $O(N^{1+\rho})$ space, where $\rho = \frac{\log p_1}{\log p_2} < 1$.*

Below, we introduce three LSH families. Let $\mathcal{N}_L(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the L-dimensional Gaussian distribution, $\mathcal{U}_L(\alpha, \beta)$ be the L-dimensional uniform distribution with its support $[\alpha, \beta]$ for all dimensions, and $\boldsymbol{I}_L$ be the L-dimensional identity matrix. The sign function, $\text{sign}(\boldsymbol{z}) : \mathbb{R}^H \mapsto \{-1, 1\}^H$, applies element-wise, giving 1 for $z_h \geq 0$ and -1 for $z_h < 0$. Likewise, the floor operator $\lfloor \cdot \rfloor$ applies element-wise for a vector. We denote by $\sphericalangle(\cdot, \cdot)$ the angle between two vectors, and by a semicolon the row-wise concatenation of vectors, like in matlab.

Proposition 2 *(L2-LSH) [12] For the L2-distance $\mathcal{L}_{\text{L2}}(\boldsymbol{q}, \boldsymbol{x}) = \|\boldsymbol{q} - \boldsymbol{x}\|_2$, the hash function*

$$h^{\text{L2}}_{\boldsymbol{a},b}(\boldsymbol{x}) = \left\lfloor R^{-1}(\boldsymbol{a}^\top \boldsymbol{x} + b) \right\rfloor, \tag{2}$$

where $R > 0$ is a fixed real number, $\boldsymbol{a} \sim \mathcal{N}_L(\boldsymbol{0}, \boldsymbol{I}_L)$, and $b \sim \mathcal{U}_1(0, R)$, satisfies $\mathbb{P}(h^{\text{L2}}_{\boldsymbol{a},b}(\boldsymbol{q}) = h^{\text{L2}}_{\boldsymbol{a},b}(\boldsymbol{x})) = F^{\text{L2}}_R(\mathcal{L}_{\text{L2}}(\boldsymbol{q}, \boldsymbol{x}))$, where

$$F^{\text{L2}}_R(d) = 1 - 2\Phi(-R/d) - \tfrac{2}{\sqrt{2\pi}(R/d)} \left(1 - e^{-(R/d)^2/2}\right).$$

Here, $\Phi(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} dy$ is the standard cumulative Gaussian.

Proposition 3 *(sign-LSH) [14, 10] For the cosine-distance $\mathcal{L}_{\cos}(\boldsymbol{q}, \boldsymbol{x}) = 1 - \cos \sphericalangle(\boldsymbol{q}, \boldsymbol{x}) = 1 - \frac{\boldsymbol{q}^\top \boldsymbol{x}}{\|\boldsymbol{q}\|_2 \|\boldsymbol{x}\|_2}$, the hash function*

$$h^{\text{sign}}_{\boldsymbol{a}}(\boldsymbol{x}) = \text{sign}(\boldsymbol{a}^\top \boldsymbol{x}), \tag{3}$$

where $\boldsymbol{a} \sim \mathcal{N}_L(\boldsymbol{0}, \boldsymbol{I}_L)$, satisfies $\mathbb{P}\left(h^{\text{sign}}_{\boldsymbol{a}}(\boldsymbol{q}) = h^{\text{sign}}_{\boldsymbol{a}}(\boldsymbol{x})\right) = F^{\text{sign}}(\mathcal{L}_{\cos}(\boldsymbol{q}, \boldsymbol{x}))$, where

$$F^{\text{sign}}(d) = 1 - \tfrac{1}{\pi} \cos^{-1}(1 - d). \tag{4}$$

Proposition 4 *[32] (simple-LSH) Assume that the samples and the query are rescaled so that $\max_{\boldsymbol{x} \in \mathcal{X}} \|\boldsymbol{x}\|_2 \leq 1$, $\|\boldsymbol{q}\|_2 \leq 1$. For the inner product dissimilarity $\mathcal{L}_{\text{ip}}(\boldsymbol{q}, \boldsymbol{x}) = 1 - \boldsymbol{q}^\top \boldsymbol{x}$ (with which the NNS problem (1) is called maximum IP search (MIPS)), the asymmetric hash functions*

$$h^{\text{smp-q}}_{\boldsymbol{a}}(\boldsymbol{q}) = h^{\text{sign}}_{\boldsymbol{a}}(\widetilde{\boldsymbol{q}}) = \text{sign}(\boldsymbol{a}^\top \widetilde{\boldsymbol{q}}) \qquad \textit{where} \qquad \widetilde{\boldsymbol{q}} = (\boldsymbol{q}; 0), \tag{5}$$

$$h^{\text{smp-x}}_{\boldsymbol{a}}(\boldsymbol{x}) = h^{\text{sign}}_{\boldsymbol{a}}(\widetilde{\boldsymbol{x}}) = \text{sign}(\boldsymbol{a}^\top \widetilde{\boldsymbol{x}}) \qquad \textit{where} \qquad \widetilde{\boldsymbol{x}} = (\boldsymbol{x}; \sqrt{1 - \|\boldsymbol{x}\|_2^2}), \tag{6}$$

satisfy $\mathbb{P}\left(h^{\text{smp-q}}_{\boldsymbol{a}}(\boldsymbol{q}) = h^{\text{smp-x}}_{\boldsymbol{a}}(\boldsymbol{x})\right) = F^{\text{sign}}(\mathcal{L}_{\text{ip}}(\boldsymbol{q}, \boldsymbol{x}))$.

These three LSH methods above are standard and state-of-the-art (among the data-independent LSH schemes) for each dissimilarity measure. Although all methods involve the same random projection $\boldsymbol{a}^\top \boldsymbol{x}$, the resulting hash codes are significantly different from each other.

3 Proposed Methods and Theory

In this section, we first define the problem setting. Then, we propose three LSH methods for multiple dissimilarity measures, and conduct a theoretical analysis.

3.1 Problem Setting

Similarly to the simple-LSH (Proposition 4), we rescale the samples so that $\max_{\boldsymbol{x}\in\mathcal{X}} \|\boldsymbol{x}\|_2 \leq 1$. We also assume $\|\boldsymbol{q}\|_2 \leq 1$.[1] Let us assume multi-modal data, where we can separate the feature vectors into G groups, i.e., $\boldsymbol{q} = (\boldsymbol{q}_1; \ldots; \boldsymbol{q}_G)$, $\boldsymbol{x} = (\boldsymbol{x}_1; \ldots; \boldsymbol{x}_G)$. For example, each group corresponds to monochrome, color, audio, and text features in video retrieval. We also accept multiple queries $\{\boldsymbol{q}^{(w)}\}_{w=1}^{W}$ for a single retrieval task. Our goal is to perform ANNS for the following dissimilarity measure, which we call multiple purpose (MP) dissimilarity:

$$\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) = \sum_{w=1}^{W} \sum_{g=1}^{G} \left\{ \gamma_g^{(w)} \|\boldsymbol{q}_g^{(w)} - \boldsymbol{x}_g\|_2^2 + 2\eta_g^{(w)} \left(1 - \frac{\boldsymbol{q}_g^{(w)\top} \boldsymbol{x}_g}{\|\boldsymbol{q}_g^{(w)}\|_2 \|\boldsymbol{x}_g\|_2}\right) + 2\lambda_g^{(w)} \left(1 - \boldsymbol{q}_g^{(w)\top} \boldsymbol{x}_g\right) \right\}, \quad (7)$$

where $\boldsymbol{\gamma}^{(w)}, \boldsymbol{\eta}^{(w)}, \boldsymbol{\lambda}^{(w)} \in \mathbb{R}_+^G$ are the feature weights such that $\sum_{w=1}^{W} \sum_{g=1}^{G} (\gamma_g^{(w)} + \eta_g^{(w)} + \lambda_g^{(w)}) = 1$. In the single query case, where $W = 1$, setting $\boldsymbol{\gamma} = (1/2, 0, 1/2, 0, \ldots, 0), \boldsymbol{\eta} = \boldsymbol{\lambda} = (0, \ldots, 0)$ corresponds to L2-NNS based on the first and the third feature groups, while setting $\boldsymbol{\gamma} = \boldsymbol{\eta} = (0, \ldots, 0), \boldsymbol{\lambda} = (1/2, 0, 1/2, 0, \ldots, 0)$ corresponds to MIPS on the same feature groups. When we like to down-weight the importance of signal amplitude (e.g., brightness of image) of the g-th feature group, we should increase the weight $\eta_g^{(w)}$ for the cosine-distance, and decrease the weight $\gamma_g^{(w)}$ for the squared-L2-distance. Multiple queries are useful when we mix NNS and MIPS, for which the queries lie in different spaces with the same dimensionality. For example, by setting $\boldsymbol{\gamma}^{(1)} = \boldsymbol{\lambda}^{(2)} = (1/4, 0, 1/4, 0, \ldots, 0), \boldsymbol{\gamma}^{(2)} = \boldsymbol{\eta}^{(1)} = \boldsymbol{\eta}^{(2)} = \boldsymbol{\lambda}^{(1)} = (0, \ldots, 0)$, we can retrieve items, which are close to the item query $\boldsymbol{q}^{(1)}$ and match the user preference query $\boldsymbol{q}^{(2)}$. An important requirement for our proposal is that the weights $\{\boldsymbol{\gamma}^{(w)}, \boldsymbol{\eta}^{(w)}, \boldsymbol{\lambda}^{(w)}\}$ can be adjusted at query time.

Our target application is an interactive system, like the demonstration in Section 4.3, where the users modify the weights according to the result with the previous weight setting. Optimizing the weights for some meta objective is out of scope of this paper.

[1] This assumption is reasonable for L2-NNS if the size of the sample pool is sufficiently large, and the query follows the same distribution as the samples. For MIPS, the norm of the query can be arbitrarily modified, and we set it to $\|\boldsymbol{q}\|_2 = 1$.

3.2 Multiple purpose LSH with Vector Augmentation (mp-LSH-VA)

Our first method, called multiple purpose LSH with vector augmentation (mp-LSH-VA), is inspired by the research on asymmetric LSHs for MIPS [36,2,37, 32], where the query and the samples are augmented with additional entries, so that the squared-L2-distance in the augmented space coincides with the target dissimilarity up to a constant. A significant difference of our proposal from the previous methods is that we design the augmentation so that we can adjust the dissimilarity measure (i.e., the feature weights $\{\boldsymbol{\gamma}^{(w)}, \boldsymbol{\lambda}^{(w)}\}$ in Eq.(7)) by modifying the augmented query vector. Since mp-LSH-VA, unfortunately, does not support the cosine-distance, we set $\boldsymbol{\eta}^{(w)} = \mathbf{0}$ in this subsection. We define the weighted sum query by

$$\overline{\boldsymbol{q}} = (\overline{\boldsymbol{q}}_1; \cdots; \overline{\boldsymbol{q}}_G) = \textstyle\sum_{w=1}^{W} \left(\phi_1^{(w)} \boldsymbol{q}_1^{(w)}; \cdots; \phi_G^{(w)} \boldsymbol{q}_G^{(w)}\right), \\ \text{where} \quad \phi_g^{(w)} = \gamma_g^{(w)} + \lambda_g^{(w)}.$$

We augment the queries and the samples as follows:

$$\widetilde{\boldsymbol{q}} = (\overline{\boldsymbol{q}}; \boldsymbol{r}), \qquad\qquad \widetilde{\boldsymbol{x}} = (\boldsymbol{x}; \boldsymbol{y}),$$

where $\boldsymbol{r} \in \mathbb{R}^M$ is a (vector-valued) function of $\{\boldsymbol{q}^{(w)}\}$, and $\boldsymbol{y} \in \mathbb{R}^M$ is a function of $\boldsymbol{x}$. We constrain the augmentation $\boldsymbol{y}$ for the sample vector so that it satisfies, for a constant $c_1 \geq 1$,

$$\|\widetilde{\boldsymbol{x}}\|_2 = c_1, \text{ i.e., } \|\boldsymbol{y}\|_2^2 = c_1^2 - \|\boldsymbol{x}\|_2^2. \tag{8}$$

Under this constraint, the norm of any augmented sample is equal to c_1, which allows us to use sign-LSH (Proposition 3) to perform L2-NNS. The squared-L2-distance between the query and a sample in the augmented space can be expressed as

$$\|\widetilde{\boldsymbol{q}} - \widetilde{\boldsymbol{x}}\|_2^2 = -2\left(\overline{\boldsymbol{q}}^\top \boldsymbol{x} + \boldsymbol{r}^\top \boldsymbol{y}\right) + \text{const.} \tag{9}$$

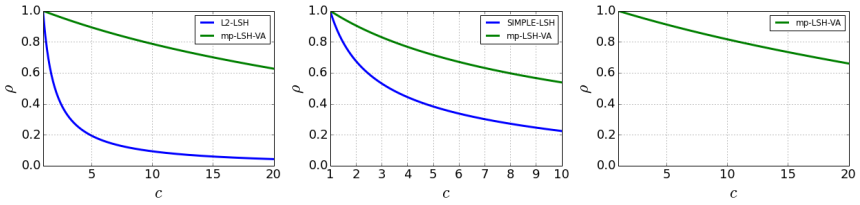
For $M = 1$, only the choice satisfying Eq.(8) is simple-LSH (for $r = 0$), given in Proposition 4. We consider the case for $M \geq 2$, and design $\boldsymbol{r}$ and $\boldsymbol{y}$ so that Eq.(9) matches the MP dissimilarity (7).

The augmentation that matches the MP dissimilarity is not unique. Here, we introduce the following easy construction with $M = G + 3$:

$$\widetilde{\boldsymbol{q}} = \left(\widetilde{\boldsymbol{q}}'; \sqrt{c_2^2 - \|\widetilde{\boldsymbol{q}}'\|_2^2}\right), \quad \widetilde{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}'; 0) \qquad \text{where} \tag{10}$$

$$\widetilde{\boldsymbol{q}}' = \Big(\underbrace{\overline{\boldsymbol{q}}_1; \cdots; \overline{\boldsymbol{q}}_G}_{\overline{\boldsymbol{q}} \in \mathbb{R}^L}; \underbrace{\textstyle\sum_{w=1}^{W} \gamma_1^{(w)}; \cdots; \sum_{w=1}^{W} \gamma_G^{(w)}; 0; \mu}_{\boldsymbol{r}' \in \mathbb{R}^{G+2}}\Big),$$

$$\widetilde{\boldsymbol{x}}' = \Big(\underbrace{\boldsymbol{x}_1; \cdots; \boldsymbol{x}_G}_{\boldsymbol{x} \in \mathbb{R}^L}; \underbrace{-\tfrac{\|\boldsymbol{x}_1\|_2^2}{2}; \cdots; -\tfrac{\|\boldsymbol{x}_K\|_2^2}{2}; \nu; \tfrac{1}{2}}_{\boldsymbol{y}' \in \mathbb{R}^{G+2}}\Big).$$



(a) L2NNS ($\gamma = 1, \lambda = 0$) (b) MIPS ($\gamma = 0, \lambda = 1$) (c) Mixed ($\gamma = 0.5, \lambda = 0.5$)

Fig. 1 Theoretical values $\rho = \frac{\log p_1}{\log p_2}$ (lower is better), which indicates the LSH performance (see Proposition 1). The horizontal axis indicates c for c-ANNS.

Here, we defined

$$\begin{aligned}
\mu &= -\textstyle\sum_{w=1}^{W} \sum_{g=1}^{G} \gamma_g^{(w)} \|\boldsymbol{q}_g^{(w)}\|_2^2, \\
\nu &= \sqrt{c_1^2 - \left(\|\boldsymbol{x}\|_2^2 + \tfrac{1}{4} \textstyle\sum_{g=1}^{G} \|\boldsymbol{x}_g\|_2^4 + \tfrac{1}{4} \right)}, \\
c_1^2 &= \max_{\boldsymbol{x} \in \mathcal{X}} \left(\|\boldsymbol{x}\|_2^2 + \tfrac{1}{4} \textstyle\sum_{g=1}^{G} \|\boldsymbol{x}_g\|_2^4 + \tfrac{1}{4} \right), \\
c_2^2 &= \max_{\boldsymbol{q}} \|\widetilde{\boldsymbol{q}}'\|_2^2.
\end{aligned}$$

With the vector augmentation (10), Eq.(9) matches Eq.(7) up to a constant (see Appendix A):

$$\|\widetilde{\boldsymbol{q}} - \widetilde{\boldsymbol{x}}\|_2^2 = c_1^2 + c_2^2 - 2\widetilde{\boldsymbol{q}}^\top \widetilde{\boldsymbol{x}} = \mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) + \mathrm{const.}$$

The collision probability, i.e., the probability that the query and the sample are given the same code, can be analytically computed:

Theorem 1 *Assume that the samples are rescaled so that* $\max_{\boldsymbol{x} \in \mathcal{X}} \|\boldsymbol{x}\|_2 \leq 1$ *and* $\|\boldsymbol{q}^{(w)}\|_2 \leq 1$ *for* $w = 1, \ldots, W$*. For the MP dissimilarity* $\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x})$*, given by Eq.*(7)*, with* $\boldsymbol{\eta}^{(w)} = \boldsymbol{0}$ *for* $w = 1, \ldots, W$*, the asymmetric hash functions*

$$\begin{aligned}
h_{\boldsymbol{a}}^{\mathrm{VA-q}}(\{\boldsymbol{q}^{(w)}\}) &= h_{\boldsymbol{a}}^{\mathrm{sign}}(\widetilde{\boldsymbol{q}}) = \mathrm{sign}(\boldsymbol{a}^\top \widetilde{\boldsymbol{q}}), \\
h_{\boldsymbol{a}}^{\mathrm{VA-x}}(\boldsymbol{x}) &= h_{\boldsymbol{a}}^{\mathrm{sign}}(\widetilde{\boldsymbol{x}}) = \mathrm{sign}(\boldsymbol{a}^\top \widetilde{\boldsymbol{x}}),
\end{aligned}$$

where $\widetilde{\boldsymbol{q}}$ *and* $\widetilde{\boldsymbol{x}}$ *are given by Eq.*(10)*, satisfy*

$$\mathbb{P}\left(h_{\boldsymbol{a}}^{\mathrm{VA-q}}(\{\boldsymbol{q}^{(w)}\}) = h_{\boldsymbol{a}}^{\mathrm{VA-x}}(\boldsymbol{x}) \right) = F^{\mathrm{sign}}\left(1 + \frac{\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) - 2\|\boldsymbol{\lambda}\|_1}{2c_1 c_2} \right).$$

(Proof) Via construction, it holds that $\|\widetilde{\boldsymbol{x}}\|_2 = c_1$ and $\|\widetilde{\boldsymbol{q}}\|_2 = c_2$, and simple calculations (see Appendix A) give $\widetilde{\boldsymbol{q}}^\top \widetilde{\boldsymbol{x}} = \|\boldsymbol{\lambda}\|_1 - \frac{\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x})}{2}$. Then, applying Propostion 3 immediately proves the theorem. □

Figure 1 depicts the theoretical value of $\rho = \frac{\log p_1}{\log p_2}$ of mp-LSH-VA, computed by using Thoerem 1, for different weight settings for $G = 1$. Note that ρ determines the quality of LSH (smaller is better) for c-ANNS performance (see Proposition 1). In the case for L2-NNS and MIPS, the ρ values of the standard

LSH methods, i.e., L2-LSH (Proposition 2) and simple-LSH (Proposition 4), are also shown for comparison.

Although mp-LSH-VA offers attractive flexibility with adjustable dissimilarity, Figure 1 implies its inferior performance to the standard methods, especially in the L2-NNS case. The reason might be a too strong asymmetry between the query and the samples: a query and a sample are far apart in the augmented space, even if they are close to each other in the original space. We can see this from the first G entries in $\boldsymbol{r}$ and $\boldsymbol{y}$ in Eq.(10), respectively. Those entries for the query are non-negative, i.e., $r_m \geq 0$ for $m = 1, \ldots, G$, while the corresponding entries for the sample are non-positive, i.e., $y_m \leq 0$ for $m = 1, \ldots, G$. We believe that there is room to improve the performance of mp-LSH-VA, e.g., by adding constants and changing the scales of some augmented entries, which we leave as our future work.

In the next subsections, we propose alternative approaches, where codes are as symmetric as possible, and down-weighting is done by changing the metric in the code space. This effectively keeps close points in the original space close in the code space.

3.3 Multiple purpose LSH with Code Concatenation (mp-LSH-CC)

Let $\overline{\gamma}_g = \sum_{w=1}^{W} \gamma_g^{(w)}$, $\overline{\eta}_g = \sum_{w=1}^{W} \eta_g^{(w)}$, and $\overline{\lambda}_g = \sum_{w=1}^{W} \lambda_g^{(w)}$, and define the *metric-wise* weighted average queries by $\overline{\boldsymbol{q}}_g^{\mathrm{L2}} = \frac{\sum_{w=1}^{W} \gamma_g^{(w)} \boldsymbol{q}_g^{(w)}}{\overline{\gamma}_g}$, $\overline{\boldsymbol{q}}_g^{\cos} = \sum_{w=1}^{W} \eta_g^{(w)} \frac{\boldsymbol{q}_g^{(w)}}{\|\boldsymbol{q}_g^{(w)}\|_2}$, and $\overline{\boldsymbol{q}}_g^{\mathrm{ip}} = \sum_{w=1}^{W} \lambda_g^{(w)} \boldsymbol{q}_g^{(w)}$.

Our second proposal, called multiple purpose LSH with code concatenation (mp-LSH-CC), simply concatenates multiple LSH codes, and performs NNS under the following distance metric at query time:

$$\begin{aligned}\mathcal{D}_{\mathrm{CC}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) = \sum_{g=1}^{G} \sum_{t=1}^{T} \Big(& \overline{\gamma}_g R \sqrt{\tfrac{\pi}{2}} \left| h_t^{\mathrm{L2}}(\overline{\boldsymbol{q}}_g^{\mathrm{L2}}) - h_t^{\mathrm{L2}}(\boldsymbol{x}_g) \right| \\ & + \|\overline{\boldsymbol{q}}_g^{\cos}\|_2 \left| h_t^{\mathrm{sign}}(\overline{\boldsymbol{q}}_g^{\cos}) - h_t^{\mathrm{sign}}(\boldsymbol{x}_g) \right| \\ & + \|\overline{\boldsymbol{q}}_g^{\mathrm{ip}}\|_2 \left| h_t^{\mathrm{smp-q}}(\overline{\boldsymbol{q}}_g^{\mathrm{ip}}) - h_t^{\mathrm{smp-x}}(\boldsymbol{x}_g) \right| \Big), \quad (11)\end{aligned}$$

where $h_t^{\overline{\ }}$ denotes the t-th independent draw of the corresponding LSH code for $t = 1, \ldots, T$. The distance (11) is a *multi-metric*, a linear combination of metrics [9], in the code space. For a multi-metric, we can use the *cover tree* [6] for efficient (exact) NNS. Assuming that all adjustable linear weights are upper-bounded by 1, the cover tree expresses neighboring relation between samples, taking all possible weight settings into account. NNS is conducted by bounding the code metric for a given weight setting. Thus, mp-LSH-CC allows selective exploration of hash buckets, so that we only need to accurately measure the distance to the samples assigned to the hash buckets within a small code distance. The query time complexity of the cover tree is $O(\kappa^{12} \log N)$, where κ is a data-dependent *expansion constant* [18]. Another good aspect of

the cover tree is that it allows dynamic insertion and deletion of new samples, and therefore, it lends itself naturally to the streaming setting. Appendix F describes further details.

In the pure case for L2, cosine, or IP dissimilarity, the hash code of mp-LSH-CC is equivalent to the base LSH code, and therefore, the performance is guaranteed by Propositions 2–4, respectively. However, mp-LSH-CC is not optimal in terms of memory consumption and NNS efficiency. This inefficiency comes from the fact that it *redundantly* stores the same angular (or cosine-distance) information into each of the L2-, sign-, and simple-LSH codes. Note that the information of a vector is dominated by its angular components unless the dimensionality L is very small.

3.4 Multiple purpose LSH with Code Augmentation and Transformation (mp-LSH-CAT)

Our third proposal, called multiple purpose LSH with code augmentation and transformation (mp-LSH-CAT), offers significantly less memory requirement and faster NNS than mp-LSH-CC by sharing the angular information for all considered dissimilarity measures. Let

$$\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}} = \textstyle\sum_{w=1}^{W} (\gamma_g^{(w)} + \lambda_g^{(w)}) \boldsymbol{q}_g^{(w)}.$$

We essentially use sign-hash functions that we augment with norm information of the data, giving us the following augmented codes:

$$\boldsymbol{H}^{\mathrm{CAT-q}}(\{\boldsymbol{q}^{(w)}\}) = \left(\boldsymbol{H}(\overline{\boldsymbol{q}}^{\mathrm{L2+ip}}); \boldsymbol{H}(\overline{\boldsymbol{q}}^{\cos}); \mathbf{0}_G^\top\right), \tag{12}$$

$$\boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x}) = \left(\widetilde{\boldsymbol{H}}(\boldsymbol{x}); \boldsymbol{H}(\boldsymbol{x}); \boldsymbol{j}^\top(\boldsymbol{x})\right), \tag{13}$$

where

$$\boldsymbol{H}(\boldsymbol{v}) = \Big(\mathrm{sign}(\boldsymbol{A}_1\boldsymbol{v}_1), \ldots, \mathrm{sign}(\boldsymbol{A}_G\boldsymbol{v}_G)\Big), \tag{14}$$

$$\widetilde{\boldsymbol{H}}(\boldsymbol{v}) = \Big(\|\boldsymbol{v}_1\|_2 \mathrm{sign}(\boldsymbol{A}_1\boldsymbol{v}_1), \ldots, \|\boldsymbol{v}_G\|_2 \mathrm{sign}(\boldsymbol{A}_G\boldsymbol{v}_G)\Big),$$

$$\boldsymbol{j}(\boldsymbol{v}) = \Big(\|\boldsymbol{v}_1\|_2^2; \ldots; \|\boldsymbol{v}_G\|_2^2\Big),$$

for a partitioned vector $\boldsymbol{v} = (\boldsymbol{v}_1; \ldots; \boldsymbol{v}_G) \in \mathbb{R}^L$ and $\mathbf{0}_G = (0; \cdots; 0) \in \mathbb{R}^G$. Here, each entry of $\boldsymbol{A} = (\boldsymbol{A}_1, \ldots, \boldsymbol{A}_G) \in \mathbb{R}^{T \times L}$ follows $A_{t,l} \sim \mathcal{N}(0, 1^2)$.

For two matrices $\boldsymbol{H}', \boldsymbol{H}'' \in \mathbb{R}^{(2T+1) \times G}$ in the transformed hash code space, we measure the distance with the following multi-metric:

$$\mathcal{D}_{\mathrm{CAT}}(\boldsymbol{H}', \boldsymbol{H}'') = \textstyle\sum_{g=1}^{G} \Big(\alpha_g \sum_{t=1}^{T} \left|H'_{t,g} - H''_{t,g}\right| + \beta_g \sum_{t=T+1}^{2T} \left|H'_{t,g} - H''_{t,g}\right| + \overline{\gamma}_g \tfrac{T}{2} \left|H'_{2T+1,g} - H''_{2T+1,g}\right| \Big), \tag{15}$$

where $\alpha_g = \|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2$ and $\beta_g = \|\overline{\boldsymbol{q}}_g^{\cos}\|_2$.

Although the hash codes consist of $(2T+1)G$ entries, we do not need to store all the entries, and computation can be simpler and faster by first computing the total number of collisions in the sign-LSH part (14) for $g = 1, \ldots, G$:

$$\mathcal{C}_g(\boldsymbol{v}', \boldsymbol{v}'') = \sum_{t=1}^{T} \Big\{ \big(\boldsymbol{H}(\boldsymbol{v}')\big)_{t,g} = \big(\boldsymbol{H}(\boldsymbol{v}'')\big)_{t,g} \Big\}. \tag{16}$$

Note that this computation, which dominates the computation cost for evaluating code distances, can be performed efficiently with bit operations. With the total number of collisions (16), the metric (15) between a query set $\{\boldsymbol{q}^{(w)}\}$ and a sample $\boldsymbol{x}$ can be expressed as

$$\begin{aligned} &\mathcal{D}_{\mathrm{CAT}}\Big(\boldsymbol{H}^{\mathrm{CAT-q}}(\{\boldsymbol{q}^{(w)}\}), \boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x})\Big) \\ &\quad = \sum_{g=1}^{G} \Bigg(\alpha_g \Big(T + \|\boldsymbol{x}_g\|_2 \big(T - 2\mathcal{C}_g(\overline{\boldsymbol{q}}^{\mathrm{L2+ip}}, \boldsymbol{x})\big)\Big) \\ &\qquad\qquad + 2\beta_g \big(T - \mathcal{C}_g(\overline{\boldsymbol{q}}^{\cos}, \boldsymbol{x})\big) + \overline{\gamma}_g \tfrac{T}{2} \|\boldsymbol{x}_g\|_2^2 \Bigg). \end{aligned} \tag{17}$$

Given a query set, this can be computed from $\boldsymbol{H}(\boldsymbol{x}) \in \mathbb{R}^{T \times G}$ and $\|\boldsymbol{x}_g\|_2$ for $g = 1, \ldots, G$. Therefore, we only need to store the pure TG sign-bits, which is required by sign-LSH alone, and G additional float numbers.

Similarly to mp-LSH-CC, we use the cover tree for efficient NNS based on the code distance (15). In the cover tree construction, we set the metric weights to their upper-bounds, i.e., $\alpha_g = \beta_g = \overline{\gamma}_g = 1$, and measure the distance between samples by

$$\begin{aligned} &\mathcal{D}_{\mathrm{CAT}}\Big(\boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x}'), \boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x}'')\Big) \\ &\quad = \sum_{g=1}^{G} \Bigg(\big| \|\boldsymbol{x}'_g\|_2 - \|\boldsymbol{x}''_g\|_2 \big| \, \mathcal{C}_g(\boldsymbol{x}', \boldsymbol{x}'') \\ &\qquad\qquad + \big(\|\boldsymbol{x}'_g\|_2 + \|\boldsymbol{x}''_g\|_2 + 2\big)\big(T - \mathcal{C}_g(\boldsymbol{x}', \boldsymbol{x}'')\big) \\ &\qquad\qquad + \tfrac{T}{2} \Big| \|\boldsymbol{x}'_g\|_2^2 - \|\boldsymbol{x}''_g\|_2^2 \Big| \Bigg). \end{aligned} \tag{18}$$

Since the collision probability can be zero, we cannot directly apply the standard LSH theory with the ρ value guaranteeing the ANNS performance. Instead, we show that the metric (15) of mp-LSH-CAT approximates the MP dissimilarity (7), and the quality of ANNS is guaranteed.

Theorem 2 *For $\boldsymbol{\eta}^{(w)} = \boldsymbol{0}$ for $w = 1, \ldots, W$, it holds that*

$$\lim_{T \to \infty} \tfrac{\mathcal{D}_{\mathrm{CAT}}}{T} = \tfrac{1}{2} \mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) + \textit{const.} + \textit{error},$$

with $|\textit{error}| \leq 0.2105 \left(\|\boldsymbol{\lambda}\|_1 + \|\boldsymbol{\gamma}\|_1 \right)$.

(proof is given in Appendix C).

Theorem 3 *For* $\boldsymbol{\gamma}^{(w)} = \boldsymbol{\lambda}^{(w)} = \mathbf{0}$ *for* $w = 1, \ldots, W$*, it holds that*

$$\lim_{T\to\infty} \frac{\mathcal{D}_{\mathrm{CAT}}}{T} = \frac{1}{2}\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) + \textit{const.} + \textit{error},$$

with $|\textit{error}| \leq 0.2105\|\boldsymbol{\eta}\|_1$.

(proof is given in Appendix D).

Corollary 1 *It holds that*

$$2\lim_{T\to\infty} \frac{\mathcal{D}_{\mathrm{CAT}}}{T} = \mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) + \textit{const.} + \textit{error},$$

with

$$|\textit{error}| \leq 0.421.$$

Note that Corollary 1 does not state that the distance in the code space converges to the multiple purpose dissimilarity even in the asymptotic limit $T \to \infty$—there can be a constant worst case error. However, the constant error is bounded by 0.421, which ranges one order of magnitude below the MP dissimilarity having itself a range of 4. The following theorem guarantees ANNS to succeed with mp-LSH-CAT for pure MIPS case with specified probability (proof is given in Appendix E):

Theorem 4 *Let* $S_0 \in (0, 2)$*,* $cS_0 \in (S_0 + 0.2105, 2)$ *and set*

$$T \geq \frac{48}{(t_2 - t_1)^2} \log(\frac{n}{\varepsilon}),$$

where $t_2 > t_1$ *depend on* S_0 *and* c *(see Appendix E for details). With probability larger than* $1 - \varepsilon - \left(\frac{\varepsilon}{n}\right)^{\frac{3}{2}}$*, mp-LSH-CAT guarantees c-ANNS with respect to* $\mathcal{L}_{\mathrm{ip}}$ *(MIPS).*

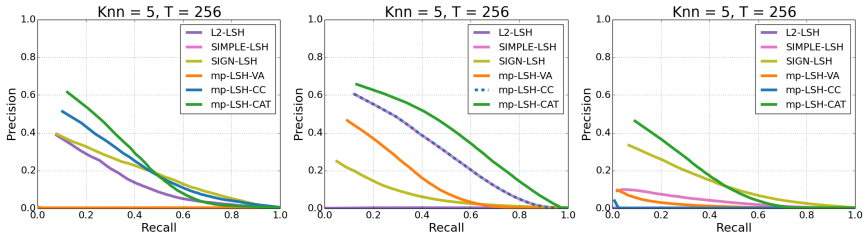
It is straightforward to show Theorem 4 for squared-L2- and cosine-distance.

Because of the constant error, the guarantee by Theorem 4 is applied for c such that $cS_0 \in (S_0 + 0.2105, 2)$. In Section 4, we will empirically show the good performance of mp-LSH-CAT, which supports that the constant error is not very harmful in practice.

3.5 Memory Requirement

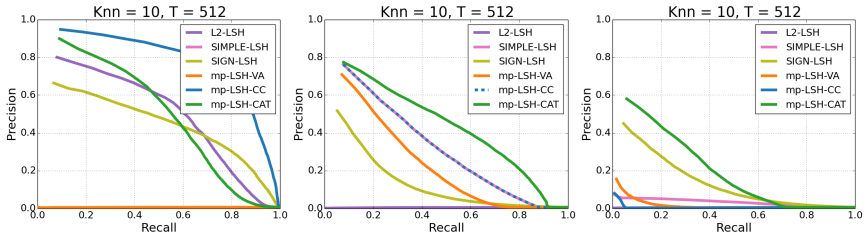
For all LSH schemes, one can trade off the memory consumption and accuracy performance by changing the hash bit length T. However, the memory consumption for specific hashing schemes heavily differs from the other schemes such that a comparison of performance is inadequate for a globally shared T. In this subsection, we derive individual numbers of hashes for each scheme, given a fixed memory budget.

We count the theoretically minimal number of bits required to store the hash code of one data point. The two fundamental components we are confronted with are sign-hashes and discretized reals. Sign-hashes can be represented by exactly one bit. For the reals we choose a resolution such that their discretizations take



(a) L2NNS ($\gamma = 1, \lambda = 0$) (b) MIPS ($\gamma = 0, \lambda = 1$) (c) Mixed ($\gamma = 0.5, \lambda = 0.5$)

Fig. 2 Precision recall curves (higher is better) on MovieLens10M data for $K = 5$ and $T = 256$.



(a) L2NNS ($\gamma = 1, \lambda = 0$) (b) MIPS ($\gamma = 0, \lambda = 1$) (c) Mixed ($\gamma = 0.5, \lambda = 0.5$)

Fig. 3 Precision recall curves on NetFlix data for $K = 10$ and $T = 512$.

values in a set of fixed size. The L2-hash function $h^{\mathrm{L2}}_{\boldsymbol{a},b}(\boldsymbol{x}) = \lfloor R^{-1}(\boldsymbol{a}^\top \boldsymbol{x} + b) \rfloor$ is a random variable with potentially infinite, discrete values. Nevertheless we can come up with a realistic upper-bound of values the L2-hash essentially takes. Note that $R^{-1}(\boldsymbol{a}^\top \boldsymbol{x})$ follows a $\mathcal{N}(\mu = 0, \sigma = (R\|x\|_2)^{-1})$ distribution and $\|x\|_2 \leq 1$. Then $\mathbb{P}(|R^{-1}(\boldsymbol{a}^\top \boldsymbol{x})| > 4\sigma) < 10^{-4}$. Therefore L2-hash essentially takes one of $\frac{8}{R}$ discrete values stored by $3 - \log_2(R)$ bits. Namely, for $R = 2^{-10} \approx 0.001$, L2-hash requires 13 bits. We also store the norm-part of mp-LSH-CAT using 13 bits.

Denote by $\mathrm{stor}_{\mathrm{CAT}}(T)$ the required storage of mp-LSH-CAT. Then $\mathrm{stor}_{\mathrm{CAT}}(T) = T_{\mathrm{CAT}} + 13$, which we set as our fixed memory budget for a given T_{CAT}. The baselines sign- and simple-LSH, so mp-LSH-VA are pure sign-hashes, thus giving them a budget of $T_{\mathrm{sign}} = T_{\mathrm{smp}} = T_{\mathrm{VA}} = \mathrm{stor}_{\mathrm{CAT}}(T)$ hashes. As discussed above, L2-LSH may take $T_{\mathrm{L2}} = \frac{\mathrm{stor}_{\mathrm{CAT}}(T)}{13}$ hashes. For mp-LSH-CC we allocate a third of the budget for each of the three components giving $\boldsymbol{T}_{\mathrm{CC}} = (T^{\mathrm{L2}}_{\mathrm{CC}}, T^{\mathrm{sign}}_{\mathrm{CC}}, T^{\mathrm{smp}}_{\mathrm{CC}}) = \mathrm{stor}_{\mathrm{CAT}}(T) \cdot (\frac{1}{39}, \frac{1}{3}, \frac{1}{3})$. This consideration is used when we compare mp-LSH-CC and mp-LSH-CAT in Section 4.2.

4 Experiment

Here, we conduct an empirical evaluation on several real-world data sets.

Table 1 ANNS Results for mp-LSH-CC with $\boldsymbol{T}_{\mathrm{CC}} = (T_{\mathrm{CC}}^{\mathrm{L2}}, T_{\mathrm{CC}}^{\mathrm{sign}}, T_{\mathrm{CC}}^{\mathrm{smp}}) = (1024, 1024, 1024)$.

	Recall@k			Query time (msec)			Storage
	1	5	10	1	5	10	per sample
L2	0.53	0.76	0.82	2633.83	2824.06	2867.00	4344 bytes
MIPS	0.69	0.77	0.82	3243.51	3323.20	3340.36	4344 bytes
L2+MIPS (.5,.5)	0.29	0.50	0.60	3553.63	3118.93	3151.44	4344 bytes

Table 2 ANNS Results with mp-LSH-CAT with $T_{\mathrm{CAT}} = 1024$.

	Recall@k			Query time (msec)			Storage
	1	5	10	1	5	10	per sample
L2	0.52	0.80	0.89	583.85	617.02	626.02	224 bytes
MIPS	0.64	0.76	0.85	593.11	635.72	645.14	224 bytes
L2+MIPS (.5,.5)	0.29	0.52	0.62	476.62	505.63	515.77	224 bytes

Table 3 ANNS Results for mp-LSH-CC with $\boldsymbol{T}_{\mathrm{CC}} = (T_{\mathrm{CC}}^{\mathrm{L2}}, T_{\mathrm{CC}}^{\mathrm{sign}}, T_{\mathrm{CC}}^{\mathrm{smp}}) = (27, 346, 346)$.

	Recall@k			Query time (msec)			Storage
	1	5	10	1	5	10	per sample
L2	0.35	0.49	0.59	1069.29	1068.97	1074.40	280 bytes
MIPS	0.32	0.56	0.56	363.61	434.49	453.35	280 bytes
L2+MIPS (.5,.5)	0.04	0.07	0.08	811.72	839.91	847.35	280 bytes

4.1 Collaborative Filtering

We first evaluate our methods on collaborative filtering data, the MovieLens10M[2] and the Netflix datasets [13]. Following the experiment in [36,37], we applied PureSVD [11] to get L-dimensional user and item vectors, where $L = 150$ for MovieLens and $L = 300$ for Netflix. We centered the samples so that $\sum_{\boldsymbol{x} \in \mathcal{X}} \boldsymbol{x} = \boldsymbol{0}$, which does not affect the L2-NNS as well as the MIPS solution.

Regarding the L-dimensional vector as a single feature group ($G = 1$), we evaluated the performance in L2-NNS ($W = 1, \gamma = 1, \eta = \lambda = 0$), MIPS ($W = 1, \gamma = \eta = 0, \lambda = 1$), and their weighted sum ($W = 2, \gamma^{(1)} = 0.5, \lambda^{(2)} = 0.5, \gamma^{(2)} = \lambda^{(1)} = \eta^{(1)} = \eta^{(2)} = 0$). The queries for L2-NNS were chosen randomly from the items, while the queries for MIPS were chosen from the users. For each query, we found its $K = 1, 5, 10$ nearest neighbors in terms of the MP dissimilarity (7) by linear search, and used them as the ground truth. We set the hash bit length to $T = 128, 256, 512$, and rank the samples (items) based on the Hamming distance for the baseline methods and mp-LSH-VA. For mp-LSH-CC and mp-LSH-CAT, we rank the samples based on their code distances (11) and (15), respectively. After that, we drew the precision-recall curve, defined as Precision $= \frac{\mathrm{relevantseen}}{k}$ and Recall $= \frac{\mathrm{relevantseen}}{K}$ for different k, where "relevant seen" is the number of the true K nearest neighbors that are ranked within the top k positions by the LSH methods. Figures 2 and 3

[2] `http://www.grouplens.org/`

show the results on MovieLens10M for $K = 5$ and $T = 256$ and NetFlix for $K = 10$ and $T = 512$, respectively, where each curve was averaged over 2000 randomly chosen queries.

We observe that mp-LSH-VA performs very poorly in L2-NNS (as bad as simple-LSH, which is not designed for L2-distance), although it performs reasonably in MIPS. On the other hand, mp-LSH-CC and mp-LSH-CAT perform well for all cases. Similar tendency was observed for other values of K and T. Since poor performance of mp-LSH-VA was shown in theory (Figure 1) and experiment (Figures 2 and 3), we will focus on mp-LSH-CC and mp-LSH-CAT in the subsequent subsections.

4.2 Computation Time in Query Search

Next, we evaluate query search time and memory consumption of mp-LSH-CC and mp-LSH-CAT on the texmex dataset[3] [22], which was generated from millions of images by applying the standard SIFT descriptor [26] with $L = 128$. Similarly to Section 4.1, we conducted experiment on L2-NNS, MIPS, and their weighted sum with the same setting for the weights $\boldsymbol{\gamma}, \boldsymbol{\eta}, \boldsymbol{\lambda}$. We constructed the cover tree with $N = 10^7$ samples, randomly chosen from the ANN_SIFT1B dataset. The queries were chosen from the defined *query set*, and the query for MIPS is normalized so that $\|\boldsymbol{q}\|_2 = 1$.

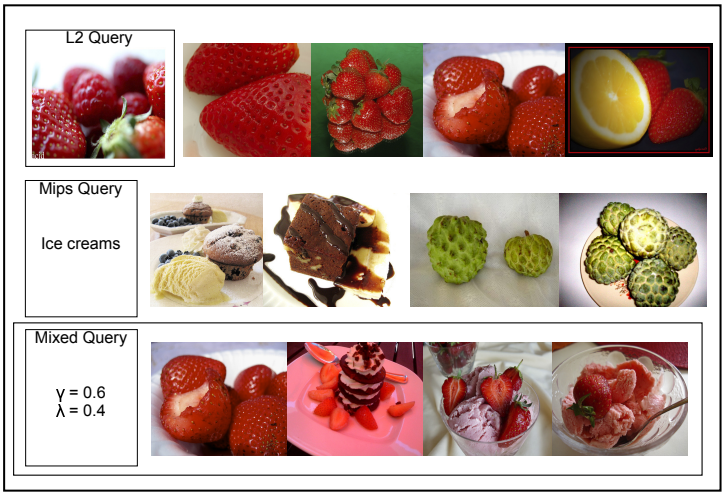
We ran the performance experiment on a machine with 48 cores (4 AMD Opteron$^{\text{TM}}$6238 Processors) and 512 GB main memory on Ubuntu 12.04.5 LTS. Tables 1–3 summarize recall@k, query time, and required memory storage. Here, recall@k is the recall for $K = 1$ and given k. All reported values are averaged over 100 queries.

We see that mp-LSH-CC (Table 1) and mp-LSH-CAT (Table 2) for $T = 1024$ perform comparably well in terms of accuracy (see the columns for recall@k). But mp-LSH-CAT is much faster (see query time) and requires significantly less memory (see storage per sample). Table 3 shows the performance of mp-LSH-CC with equal memory requirement to mp-LSH-CAT for $T = 1024$. More specifically, we use different bit length for each dissimilarity measure, and set them to $\boldsymbol{T}_{\text{CC}} = (T_{\text{CC}}^{\text{L2}}, T_{\text{CC}}^{\text{sign}}, T_{\text{CC}}^{\text{smp}}) = (27, 346, 346)$, with which the memory budget is shared equally for each dissimilarity measure, according to Section 3.5. By comparing Table 2 and Table 3, we see that mp-LSH-CC for $\boldsymbol{T}_{\text{CC}} = (27, 346, 346)$, which uses similar memory storage per sample, gives significantly worse recall@k than mp-LSH-CAT for $T = 1024$.

Thus, we conclude that both mp-LSH-CC and mp-LSH-CAT perform well, but we recommend the latter for the case of limited memory budget, or in applications where the query search time is crucial.

[3] `http://corpus-texmex.irisa.fr/`

(a) Trench coats



(b) Ice creams

Fig. 4 Image retrieval results with mixed queries. In both of (a) and (b), the top row shows L2 query (left end) and the images retrieved (by ANNS with mp-LSH-CAT for $T = 512$) according to the L2 dissimilarity ($\gamma^{(1)} = 1.0$ and $\lambda^{(2)} = 0.0$), the second row shows MIPS query and the images retrieved according to the IP dissimilarity ($\gamma^{(1)} = 0.0$ and $\lambda^{(2)} = 1.0$), and the third row shows the images retrieved according to the mixed dissimilarity for $\gamma^{(1)} = 0.6$ and $\lambda^{(2)} = 0.4$.

4.3 Demonstration of Image Retrieval with Mixed Queries

Finally, we demonstrate the usefulness of our flexible mp-LSH in an image retrieval task on the ILSVRC2012 data set [34]. We computed a feature vector for each image by concatenating the 4096-dimensional fc7 activations of the

trained VGG16 model [39] with 120-dimensional color features[4]. Since user preference vector is not available, we use classifier vectors, which are the weights associated with the respective ImageNet classes, as MIPS queries (the entries corresponding to the color features are set to zero). This simulates users who like a particular class of images.

We performed ANNS based on the MP dissimilarity by using our mp-LSH-CAT with $T = 512$ in the sample pool consisting of all $N \approx 1.2M$ images. In Figure 4(a), each of the three rows consists of the query at the left end, and the corresponding top-ranked images. In the first row, the shown black dog image was used as the L2 query $\boldsymbol{q}^{(1)}$, and similar black dog images were retrieved according to the L2 dissimilarity ($\gamma^{(1)} = 1.0$ and $\lambda^{(2)} = 0.0$). In the second row, the VGG16 classifier vector for *trench coats* was used as the MIPS query $\boldsymbol{q}^{(2)}$, and images containing trench coats were retrieved according to the MIPS dissimilarity ($\gamma^{(1)} = 0.0$ and $\lambda^{(2)} = 1.0$). In the third row, images containing black trench coats were retrieved according to the mixed dissimilarity for $\gamma^{(1)} = 0.6$ and $\lambda^{(2)} = 0.4$. Figure 4(b) shows another example with a strawberry L2 query and the *ice creams* MIPS query. We see that, in both examples, mp-LSH-CAT handles the combined query well: it brings images that are close to the L2 query, and relevant to the MIPS query. Other examples can be found through our online demo.[5]

5 Conclusion

When querying huge amounts of data, it becomes mandatory to increase efficiency, i.e., even linear methods may be too computationally involved. Hashing, in particular locality sensitive hashing (LSH) has become a highly efficient workhorse that can yield answers to queries in sublinear time, such as L2-/cosine-distance nearest neighbor search (NNS) or maximum inner product search (MIPS). While for typical applications the type of query has to be fixed beforehand, it is not uncommon to query with respect to several aspects in data, perhaps, even reweighting this dynamically at query time. Our paper contributes exactly herefore, namely by proposing three multiple purpose locality sensitive hashing (mp-LSH) methods which enable L2-/cosine-distance NNS, MIPS, and their weighted sums. A user can now indeed and efficiently change the importance of the weights at query time without recomputing the hash functions. Our paper has placed its focus on proving the feasibilty and efficiency of the mp-LSH methods, and introducing the very interesting cover tree concept (which is less commonly applied in the machine learning world) for fast querying over the defined multi-metric space. Finally we provide a demonstration on the usefulness of our novel technique.

[4] We computed histograms on the central crop of an image (covering 50% of the area) for each rgb color channel with 8 and 32 bins. We normalized the histograms and concatenate them.

[5] `http://bbdcdemo.bbdc.tu-berlin.de/`

Future studies will extend the possibilities of mp-LSH for further including other types of dissimilarity measure, e.g., the distance from hyperplane [21], and further applications with combined queries, e.g., retrieval with one complex multiple purpose query, say, a pareto-front for subsequent decision making. Another future direction would be to analyze the interpretability of NNS systems, specifically for recommender systems with nonlinear query mechanism, in terms of salient features that have led to the query result. This is in the line of research on "explaining learning machines", i.e., answering to the question which part of the data is responsible for specific decisions made by learning machines [3,38,46,1,33,28,30]. This question is non-trivial when the learning machines are complex and non-linear. Our mp-LSH enables complex nonlinear query mechanism, and therefore, it would be a useful tool if we could, for example, develop a method which can explain why a NNS system with mixed queries recommended a specific set of items, and analyze the dependency on the weight setting.

Acknowledgements This work was supported by the German Research Foundation (GRK 1589/1) by the Federal Ministry of Education and Research (BMBF) under the project Berlin Big Data Center (FKZ 01IS14013A) and the BMBF project ALICE II, Autonomous Learning in Complex Environments (01IB15001B).

On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., , Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS One **10**(7): e0130140 (2015)
2. Bachrach, Y., Finkelstein, Y., Gilad-Bachrach, R., Katzir, L., Koenigstein, N., Nice, N., Paquet, U.: Speeding up the Xbox recommender system using a euclidean transformation for inner-product spaces. In: Proc. of RecSys (2014)
3. Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., Muller, K.R.: How to explain individual classification decisions. Journal of Machine Learning Research **11**(Jun), 1803–1831 (2010)
4. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends in Machine Learning **2**(1) (2009)
5. Bengio, Y., LeCun, Y., Hinton, G.: Deep learning. Nature **521** (2015)
6. Beygelzimer, A., Kakade, S., Langford, J.: Cover trees for nearest neighbor. In: ICML, pp. 97–104 (2006)
7. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York, NY, USA (2006)
8. Broder, A.Z., Glassman, S.C., Manasse, M.S., Zweig, G.: Syntactic clustering of the web. Computer Networks **29**, 1157–1166 (1997)
9. Bustos, B., Kreft, S., Skopal, T.: Adapting metric indexes for searching in multi-metric spaces. Multimedia Tools and Applications **58**(3), 467–496 (2012)
10. Charikar, M.S.: Similarity estimation techniques from rounding algorithms. In: STOC, pp. 380–388 (2002)
11. Cremonesi, P., Koren, Y., Turrin, R.: Performance of recommender algorithms on top-n recommendation tasks. In: Proc. of RecSys, pp. 39–46 (2010)
12. Datar, M., Immorlica, N., Indyk, P., Mirrokn, V.S.: Locality-sensitive hashing scheme based on p-stable distributions. In: SCG, pp. 253–262 (2004)

13. Funk, S.: Try this at home. http://sifter.org/˜simon/journal/20061211.html (2006)
14. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of ACM **42**(6), 1115–1145 (1995)
15. Gorisse, D., Cord, M., Precioso, F.: Locality-sensitiv hashing for chi2 distance. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(2), 402–409 (2012)
16. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer (2001)
17. He, J., Chang, S.F., Radhakrishnan, R., Bauer, C.: Compact hashing with joint optimization of search accuracy and time. In: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 753–760 (2011)
18. Heinonen, J.: Lectures on analysis on metric spaces. Universitext (2001)
19. Hinton, G.: Learning multiple layers of representation. Trends in Cognitive Sciences **11** (2007)
20. Indyk, P., Motwani, R.: Approximate nearest neighbors: Towards removing the curse of dimensionality. In: STOC, pp. 604–613 (1998)
21. Jain, P., Vijayanarasimhan, S., Grauman, K.: Hashing hyperplane queries to near points with applications to large-scale active learning. In: Advances in NIPS (2010)
22. Jégou, H., Tavenard, R., Douze, M., Amsaleg, L.: Searching in one billion vectors: re-rank with source coding. In: ICASSP, pp. 861–864 (2011)
23. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS), vol. 25 (2012)
24. Lin, K., Yang, H.F., Hsiao, J.H., Chen, C.S.: Deep learning of binary hash codes for fast image retrieval. In: Proceedings of Computer Vision and Pattern Recognition Workshops (2015)
25. Liu, G., Xu, H., Yan, S.: Exact subspace segmentation and outlier detection by low-rank representation. In: Proc. of AISTATS (2012)
26. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision **60**(2), 91–110 (2004)
27. Matsushita, Y., Wada, T.: Principal component hashing: An accelerated approximate nearest neighbor search. In: Proceedings of Pacific-Rim Symposium on Image and Video Technology (PSIVT), pp. 374–385 (2009)
28. Montavon, G., Lapuschkin, S., Binder, A., Samek, W., Müller, K.R.: Explaining nonlinear classification decisions with deep taylor decomposition. Pattern Recognition **65**, 211–222 (2017)
29. Montavon, G., Orr, G., Müller, K.R.: Neural Networks: Tricks of the Trade. Springer, New York, NY, USA (2012)
30. Montavon, G., Samek, W., Muller, K.R.: Methods for interpreting and understanding deep neural networks. Digital Signal Processing **73**, 1–15 (2018)
31. Moran, S., Lavrenko, V.: Regularized cross-modal hashing. In: Proc. of SIGIR (2015)
32. Neyshabur, B., Srebro, N.: On symmetric and asymmetric lshs for inner product search. In: ICML, vol. 32 (2015)
33. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should I trust you? In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)
34. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). DOI 10.1007/s11263-015-0816-y
35. Schütt, K., Arbabzadah, F., Chmiela, S., Müller, K.R., Tkatchenko, A.: Quantum-chemical insights from deep tensor neural networks. Nature Communications **8**: 13890 (2017)
36. Shrivastava, A., Li, P.: Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In: NIPS, vol. 27 (2014)
37. Shrivastava, A., Li, P.: Improved asymmetric locality sensitive hashing (ALSH) for maximum inner product search (MIPS). Proc. of UAI (2015)
38. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. In: ICLR Workshop 2014 (2014)

39. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
40. Song, J., Yang, Y., Huang, Z., Schen, H.T., Luo, J.: Effective multiple feature hashing for large-scale near-duplicate video retrieval. IEEE Trans. on Multimedia **15**(8), 1997–2008 (2013)
41. Strecha, C., Bronstein, A.M., Bronstein, M.M., Fua, P.: LDA hash: Improved matching with smaller descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(1), 66–78 (2012)
42. Tagami, Y.: AnnexML: Approximate nearest neighbor search for extreme multi-label classification. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 455–464 (2017)
43. Torralba, A., Fergus, R., Freeman, W.: 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(11), 1958–1970 (2008)
44. Wang, J., Schen, H.T., Song, J., Ji, J.: Hashing for similarity search: A survey. arXiv:1408.2927v1 [cs.DS] (2014)
45. Xu, S., Wang, S., Y.Zhang: Summarizing complex events: a cross-modal solution of storylines extraction and reconstruction. In: Proc. of EMNLP, pp. 1281–1291 (2013)
46. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Proceedings of European Conference on Computer Vision, pp. 818–833 (2014)

A Derivation of Inner Product in Proof of Theorem 1

The inner product between the augmented vectors $\widetilde{\boldsymbol{q}}$ and $\widetilde{\boldsymbol{x}}$, defined in Eq.(10), is given by

$$\begin{aligned}\widetilde{\boldsymbol{q}}^\top \widetilde{\boldsymbol{x}} &= \textstyle\sum_{w=1}^{W}\sum_{g=1}^{G}\left((\gamma_g^{(w)}+\lambda_g^{(w)})\boldsymbol{q}_g^{(w)\top}\boldsymbol{x}_g - \frac{1}{2}\sum_{g=1}^{G}\gamma_g^{(w)}\left(\|\boldsymbol{q}_g^{(w)}\|_2^2+\|\boldsymbol{x}_g\|_2^2\right)\right)\\ &= -\tfrac{1}{2}\textstyle\sum_{w=1}^{W}\sum_{g=1}^{G}\left(-2\lambda_g^{(w)}\boldsymbol{q}_g^{(w)\top}\boldsymbol{x}_g+\gamma_g^{(w)}\underbrace{\left((\|\boldsymbol{q}_g^{(w)}\|_2^2+\|\boldsymbol{x}_g\|_2^2)-2\boldsymbol{q}_g^{(w)\top}\boldsymbol{x}_g\right)}_{\|\boldsymbol{q}_g^{(w)}-\boldsymbol{x}_g\|_2^2}\right)\\ &= \|\boldsymbol{\lambda}\|_1 - \tfrac{\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\},\boldsymbol{x})}{2}.\end{aligned}$$

B Lemma: Inner Product Approximation

For $\boldsymbol{q},\boldsymbol{x}\in\mathbb{R}^L$ let

$$d_T(\boldsymbol{q},\boldsymbol{x}) = \tfrac{1}{T}\textstyle\sum_{t=1}^{T}\left|\boldsymbol{H}(\boldsymbol{q})_{t1}-\widetilde{\boldsymbol{H}}(\boldsymbol{x})_{t1}\right|$$

with expectation

$$d(\boldsymbol{q},\boldsymbol{x}) = \mathbb{E}d_T(\boldsymbol{q},\boldsymbol{x}) = \mathbb{E}\left|\boldsymbol{H}(\boldsymbol{q})_{11}-\widetilde{\boldsymbol{H}}(\boldsymbol{x})_{11}\right|$$

and define

$$L(\boldsymbol{q},\boldsymbol{x}) = 1-\tfrac{\boldsymbol{q}^\top\boldsymbol{x}}{\|\boldsymbol{q}\|_2}.$$

Lemma 1 *The following statements hold:*
(a): It holds that

$$d(\boldsymbol{q},\boldsymbol{x}) = 1-\|\boldsymbol{x}\|_2\left(1-\tfrac{2}{\pi}\sphericalangle(\boldsymbol{q},\boldsymbol{x})\right)$$

(b): For $\mathcal{E}_x = 0.2105\|\boldsymbol{x}\|_2$ *it is*

$$|L(\boldsymbol{q},\boldsymbol{x})-d(\boldsymbol{q},\boldsymbol{x})| \le \mathcal{E}_x \tag{19}$$

(c): Let $b(\boldsymbol{q},\boldsymbol{x}) = 1 - \frac{2}{\pi}\frac{\boldsymbol{q}^\top \boldsymbol{x}}{\|\boldsymbol{q}\|_2}$*, then for* $L(\boldsymbol{q},\boldsymbol{x}) \leq 1$ *it is*

$$L(\boldsymbol{q},\boldsymbol{x}) \leq d(\boldsymbol{q},\boldsymbol{x}) \leq b(\boldsymbol{q},\boldsymbol{x}) \leq 1$$

and for $L(\boldsymbol{q},\boldsymbol{x}) \geq 1$ *it is*

$$L(\boldsymbol{q},\boldsymbol{x}) \geq d(\boldsymbol{q},\boldsymbol{x}) \geq b(\boldsymbol{q},\boldsymbol{x}) \geq 1$$

(d): It holds that

$$|L(\boldsymbol{q},\boldsymbol{x}) - d(\boldsymbol{q},\boldsymbol{x})| \leq \min\{(1-\frac{2}{\pi})|L(\boldsymbol{q},\boldsymbol{x}) - 1|, \mathcal{E}_x\}$$

and for $s_{\boldsymbol{x}} = 0.58\|\boldsymbol{x}\|_2$*, if* $|L(\boldsymbol{q},\boldsymbol{x}) - 1| \leq s_{\boldsymbol{x}}$*, it is*

$$(1-\frac{2}{\pi})|L(\boldsymbol{q},\boldsymbol{x}) - 1| \leq \mathcal{E}_x.$$

Proof (a):

Defining $p_{col} = 1 - \frac{1}{\pi}\sphericalangle(\boldsymbol{q},\boldsymbol{x})$ we have

$$\begin{aligned}\mathbb{E}\left|\boldsymbol{H}(\boldsymbol{q})_{11} - \widetilde{\boldsymbol{H}}(\boldsymbol{x})_{11}\right| &= \big(1-\|\boldsymbol{x}\|_2\big)p_{col} + \big(1+\|\boldsymbol{x}\|_2\big)\big(1-p_{col}\big)\\ &= 1 - \|\boldsymbol{x}\|_2\big(2p_{col}-1\big) = 1 - \|\boldsymbol{x}\|_2(1-\frac{2}{\pi}\sphericalangle(\boldsymbol{q},\boldsymbol{x})).\end{aligned}$$

Proof (b):

$$\begin{aligned}|L(\boldsymbol{q},\boldsymbol{x}) - d(\boldsymbol{q},\boldsymbol{x})| &= \|\boldsymbol{x}\|_2\Big|\frac{\boldsymbol{q}^\top\boldsymbol{x}}{\|\boldsymbol{q}\|_2\|\boldsymbol{x}\|_2} - 1 + \frac{2}{\pi}\sphericalangle(\boldsymbol{q},\boldsymbol{x})\Big|\\ &\leq \|\boldsymbol{x}\|_2 \max_{z\in[-1,1]}|z - 1 + \frac{2}{\pi}\arccos(z)|.\end{aligned}$$

For $z^* = \sqrt{1-\frac{4}{\pi^2}}$ we obtain the maximum

$$\mathcal{E}_x = \|\boldsymbol{x}\|_2|z^* - 1 + \frac{2}{\pi}\arccos(z^*)| \approx 0.2105\|\boldsymbol{x}\|_2.$$

Proof (c):

We treat the case $L(\boldsymbol{q},\boldsymbol{x}) \leq 1$, noting that the others case is analogous due to symmetry. Observe that $\frac{\boldsymbol{q}^\top\boldsymbol{x}}{\|\boldsymbol{q}\|_2} \geq 0$, providing

$$b(\boldsymbol{q},\boldsymbol{x}) = 1 - \frac{2}{\pi}\frac{\boldsymbol{q}^\top\boldsymbol{x}}{\|\boldsymbol{q}\|_2} \leq 1.$$

As arccos is a concave function on $[0,1]$, it is

$$\begin{aligned}\arccos(z) &= \arccos(0(1-z) + 1(z))\\ &\geq (1-z)\arccos(0) + z\arccos(1) = \frac{\pi}{2}(1-z).\end{aligned}$$

Define $z = \frac{\boldsymbol{q}^\top\boldsymbol{x}}{\|\boldsymbol{q}\|_2\|\boldsymbol{x}\|_2}$. Then we have

$$d(\boldsymbol{q},\boldsymbol{x}) - L(\boldsymbol{q},\boldsymbol{x}) = \|\boldsymbol{x}\|_2\left(z - 1 + \frac{2}{\pi}\arccos(z)\right) \geq 0,$$

from which $L(\boldsymbol{q},\boldsymbol{x}) \leq d(\boldsymbol{q},\boldsymbol{x})$ follows. Noting that

$$\max_{z\in[0,1]}\frac{d\arccos}{\delta z}(z) = \max_{z\in[0,1]}\frac{-1}{\sqrt{1-z^2}} = -1,$$

and $\arccos(0) = \frac{\pi}{2}$, it is

$$\arccos(z) - \arccos(0) = \int_0^z \frac{d\arccos}{\delta z}(t)dt \leq -\int_0^z dt = -z,$$

such that

$$\arccos(z) \leq \frac{\pi}{2} - z.$$

Therefore it is

$$b(\boldsymbol{q}, \boldsymbol{x}) - d(\boldsymbol{q}, \boldsymbol{x}) = \|\boldsymbol{x}\|_2 \left(1 - \frac{2}{\pi}z - \frac{2}{\pi}\arccos(z)\right) \geq 0$$

assuring $d(\boldsymbol{q}, \boldsymbol{x}) \leq b(\boldsymbol{q}, \boldsymbol{x})$.
Proof (d):

The inequality follows from (b) and (c). Letting

$$s_{\boldsymbol{x}} = \frac{\mathcal{E}_x}{1 - \frac{2}{\pi}} \approx 0.58\|\boldsymbol{x}\|_2,$$

the first bound is tighter than $\mathcal{E}_x$, if $|L(\boldsymbol{q}, \boldsymbol{x}) - 1| \leq s_{\boldsymbol{x}}$.

□

Note that $d_T(\boldsymbol{q}, \boldsymbol{x}) \to d(\boldsymbol{q}, \boldsymbol{x})$ as $T \to \infty$. Therefore all statements are also valid, replacing $d(\boldsymbol{q}, \boldsymbol{x})$ by $d_T(\boldsymbol{q}, \boldsymbol{x})$ with T large enough.

C Proof of Theorem 2

For $\boldsymbol{\eta}^{(w)} = \mathbf{0}$ for $w = 1, \ldots, W$ we have

$$\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) = \sum_{w=1}^{W}\sum_{g=1}^{G} \gamma_g^{(w)} \|\boldsymbol{q}_g^{(w)} - \boldsymbol{x}_g\|_2^2 + 2\lambda_g^{(w)} \left(1 - \boldsymbol{q}_g^{(w)\top}\boldsymbol{x}_g\right).$$

Recall that $\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}} = \sum_{w=1}^{W}(\gamma_g^{(w)} + \lambda_g^{(w)})\boldsymbol{q}_g^{(w)}$. Therefore

$$\frac{1}{T}\mathcal{D}_{\mathrm{CAT}}\Big(\boldsymbol{H}^{\mathrm{CAT-q}}(\{\boldsymbol{q}^{(w)}\}), \boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x})\Big) = \sum_{g=1}^{G}\left(\frac{\overline{\gamma}_g}{2}\|\boldsymbol{x}_g\|_2^2 + \|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2\Big(1 + \|\boldsymbol{x}_g\|_2\big(1 - \frac{2}{T}\mathcal{C}_g(\overline{\boldsymbol{q}}^{\mathrm{L2+ip}}, \boldsymbol{x})\big)\Big)\right).$$

We use that

$$1 - \frac{2}{T}\mathcal{C}_g(\overline{\boldsymbol{q}}^{\mathrm{L2+ip}}, \boldsymbol{x}) = -1 + \frac{1}{T}\sum_{t=1}^{T}\left|\boldsymbol{H}(\boldsymbol{x})_{tg} - \boldsymbol{H}(\overline{\boldsymbol{q}}^{\mathrm{L2+ip}})_{tg}\right| = -1 + d_T\left(\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}, \frac{\boldsymbol{x}_g}{\|\boldsymbol{x}_g\|_2}\right)$$
$$\overset{T\to\infty}{\to} -1 + d\left(\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}, \frac{\boldsymbol{x}_g}{\|\boldsymbol{x}_g\|_2}\right) \overset{(19)}{=} -1 + L\left(\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}, \frac{\boldsymbol{x}_g}{\|\boldsymbol{x}_g\|_2}\right) + e_g = -\frac{\boldsymbol{x}_g^\top \overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}}{\|\boldsymbol{x}_g\|_2\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2} + e_g,$$

where $|e_g| \leq \mathcal{E}_1$ such that

$$
\begin{aligned}
&\frac{1}{T}\mathcal{D}_{\mathrm{CAT}}\Big(\boldsymbol{H}^{\mathrm{CAT-q}}(\{\boldsymbol{q}^{(w)}\}), \boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x})\Big) \\
&\quad= \sum_{g=1}^{G}\left(\frac{\overline{\gamma}_g}{2}\|\boldsymbol{x}_g\|_2^2 + \|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2\Big(1 - \frac{\boldsymbol{x}_g^\top \overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}}{\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2} + \|\boldsymbol{x}_g\|_2 e_g\Big)\right) \\
&\quad= \sum_{g=1}^{G}\left(\frac{\overline{\gamma}_g}{2}\|\boldsymbol{x}_g\|_2^2 - \boldsymbol{x}_g^\top \overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\right) + \sum_{g=1}^{G}\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2 + \underbrace{\sum_{g=1}^{G}\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2\|\boldsymbol{x}_g\|_2 e_g}_{\text{error}} \\
&\quad= \frac{1}{2}\sum_{g=1}^{G}\sum_{w=1}^{W}\left[\gamma_g^{(w)}\|\boldsymbol{q}_g^{(w)} - \boldsymbol{x}_g\|_2^2 + 2\lambda_g^{(w)}(1 - \boldsymbol{x}_g^\top \boldsymbol{q}_g^{(w)})\right] \\
&\qquad\qquad + \underbrace{\sum_{g=1}^{G}\left(\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2 - \frac{1}{2}\sum_{w=1}^{W}\left(\gamma_g^{(w)}\|\boldsymbol{q}_g^{(w)}\|_2^2 + 2\lambda_g^{(w)}\right)\right)}_{\text{const}} + \text{error} \\
&\quad= \frac{1}{2}\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) + \text{ const } + \text{error}.
\end{aligned}
$$

We can bound the error-term by

$$
\begin{aligned}
|\text{error}| &\leq \max_{g\in\{1,\ldots,G\}}|e_g|\sum_{g=1}^{G}\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2\|\boldsymbol{x}_g\|_2 \\
&\leq \mathcal{E}_1\left\|\left(\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2\right)_g\right\|_2\|\boldsymbol{x}\|_2 \leq \mathcal{E}_1\left\|\left(\|\overline{\boldsymbol{q}}_g^{\mathrm{L2+ip}}\|_2\right)_g\right\|_1 \\
&\leq \mathcal{E}_1\sum_{g=1}^{G}\sum_{w=1}^{W}(\gamma_g^{(w)} + \lambda_g^{(w)})\|\boldsymbol{q}_g^{(w)}\|_2 \leq \mathcal{E}_1\left(\|\boldsymbol{\lambda}\|_1 + \|\boldsymbol{\gamma}\|_1\right).
\end{aligned}
$$

□

D Proof of Theorem 3

For $\boldsymbol{\gamma}^{(w)} = \boldsymbol{\lambda}^{(w)} = \mathbf{0}$ for $w = 1, \ldots, W$, we have

$$
\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) = 2\sum_{w=1}^{W}\sum_{g=1}^{G}\eta_g^{(w)}\left(1 - \frac{\boldsymbol{q}_g^{(w)\top}\boldsymbol{x}_g}{\|\boldsymbol{q}_g^{(w)}\|_2\|\boldsymbol{x}_g\|_2}\right).
$$

Recall that $\overline{\boldsymbol{q}}_g^{\cos} = \sum_{w=1}^{W} \eta_g^{(w)} \frac{\boldsymbol{q}_g^{(w)}}{\|\boldsymbol{q}_g^{(w)}\|_2}$. Therefore

$$\begin{aligned}
&\frac{1}{T}\mathcal{D}_{\mathrm{CAT}}\Big(\boldsymbol{H}^{\mathrm{CAT-q}}(\{\boldsymbol{q}^{(w)}\}), \boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x})\Big) \\
&\quad= \sum_{g=1}^{G} 2\|\overline{\boldsymbol{q}}_g^{\cos}\|_2 \big(1 - \frac{1}{T}\mathcal{C}_g(\overline{\boldsymbol{q}}^{\cos}, \boldsymbol{x})\big) \\
&\quad\overset{(19)}{\to} \sum_{g=1}^{G} \|\overline{\boldsymbol{q}}_g^{\cos}\|_2 \big(1 - \frac{\boldsymbol{x}_g^\top \overline{\boldsymbol{q}}_g^{\cos}}{\|\boldsymbol{x}_g\|_2 \|\overline{\boldsymbol{q}}_g^{\cos}\|_2} + e_g\big) \\
&\quad= -\sum_{g=1}^{G}\sum_{w=1}^{W} \eta_g^{(w)} \frac{\boldsymbol{x}_g^\top \boldsymbol{q}_g^{(w)}}{\|\boldsymbol{x}_g\|_2 \|\boldsymbol{q}_g^{(w)}\|_2} + \sum_{g=1}^{G} \|\overline{\boldsymbol{q}}_g^{\cos}\|_2 + \underbrace{\sum_{g=1}^{G} e_g \|\overline{\boldsymbol{q}}_g^{\cos}\|_2}_{\text{error}} \\
&\quad= \sum_{g=1}^{G}\sum_{w=1}^{W} \eta_g^{(w)} \left(1 - \frac{\boldsymbol{x}_g^\top \boldsymbol{q}_g^{(w)}}{\|\boldsymbol{x}_g\|_2 \|\boldsymbol{q}_g^{(w)}\|_2}\right) + \underbrace{\sum_{g=1}^{G} \left(\|\overline{\boldsymbol{q}}_g^{\cos}\|_2 - \sum_{w=1}^{W} \eta_g^{(w)}\right)}_{\text{const}} + \text{error} \\
&\quad= \frac{1}{2}\mathcal{L}_{\mathrm{mp}}(\{\boldsymbol{q}^{(w)}\}, \boldsymbol{x}) + \text{const} + \text{error},
\end{aligned}$$

where

$$\begin{aligned}
|\text{error}| &\le \max_{g\in\{1,\dots,G\}} |e_g| \sum_{g=1}^{G} \|\overline{\boldsymbol{q}}_g^{\cos}\|_2 \\
&\le \mathcal{E}_1 \sum_{g=1}^{G}\sum_{w=1}^{W} \eta_g^{(w)} \Big\| \boldsymbol{q}_g^{(w)} \Big/ \|\boldsymbol{q}_g^{(w)}\|_2 \Big\|_2 = \mathcal{E}_1 \|\boldsymbol{\eta}\|_1.
\end{aligned}$$

□

E Proof of Theorem 4

Without loss of generality we prove the theorem for the plain MIPS case with $G = 1$, $W = 1$ and $\lambda = 1$. Then $\alpha = 1$ and the measure simplifies to

$$\mathcal{D}_{\mathrm{CAT}}\Big(\boldsymbol{H}^{\mathrm{CAT-q}}(\{\boldsymbol{q}^{(w)}\}), \boldsymbol{H}^{\mathrm{CAT-x}}(\boldsymbol{x})\Big) = T d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}).$$

For $\mathcal{C}_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x})$ with $\mu = \mathbb{E}\mathcal{C}_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) = T(1 - \frac{1}{\pi}\sphericalangle(\boldsymbol{x}, \boldsymbol{q}^{\mathrm{ip}}))$ and $0 < \delta_1 < 1$, $\delta_2 > 0$ we use the following *Chernoff*-bounds:

$$\mathbb{P}\big(\mathcal{C}_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \le (1-\delta_1)\mu\big) \le \exp\Big\{-\frac{\mu}{2}\delta_1^2\Big\} \tag{20}$$

$$\mathbb{P}\big(\mathcal{C}_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \ge (1+\delta_2)\mu\big) \le \exp\Big\{-\frac{\mu}{3}\min\{\delta_2, \delta_2^2\}\Big\} \tag{21}$$

The approximate nearest-neighbor problem with $r > 0$ and $c > 1$ is defined as follows: If there exists an $\boldsymbol{x}^*$ with $\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}^*) \le r$ then we return an $\widetilde{\boldsymbol{x}}$ with $\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \widetilde{\boldsymbol{x}}) < cr$. For $cr > r + \mathcal{E}_1$ we can set T logarithmically dependent on the dataset size to solve the approximate nearest-neighbor problem for $\mathcal{L}_{\mathrm{ip}}$, using d_T with constant success probability: For this we require a viable t that fulfills

$$\begin{aligned}
&\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > cr \Rightarrow d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > t \text{ and} \\
&\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \le r \Rightarrow d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) <= t.
\end{aligned}$$

Namely set $t = \frac{t_1+t_2}{2}$, where

$$t_1 = \begin{cases} r + \mathcal{E}_1, & r \leq 1 - s_1 \\ 1 - \frac{2(1-r)}{\pi}, & r \in (1 - s_1, 1) \\ r, & r \geq 1 \end{cases}$$
$$\text{and } t_2 = \begin{cases} cr, & cr \leq 1 \\ 1 + \frac{2(cr-1)}{\pi}, & cr \in (1, 1 + s_1) \\ cr - \mathcal{E}_1, & cr \geq 1 + s_1 \end{cases}.$$

In any case it is $t_2 > t_1$:

First note that t_1 and t_2 are strictly monotone increasing in r and cr, respectively. It therefore suffices to show $\underline{t_2} \geq t_1$ for the lower bound $\underline{t_2}$ based on $\underline{cr} = r + \mathcal{E}_1$.

(Case $r \leq 1 - s_1$): It is $t_1 = r + \mathcal{E}_1$ and $\underline{t_2} = \underline{cr}$, where

$$t_1 = r + \mathcal{E}_1 = \underline{cr} = \underline{t_2}$$

(Case $r \in (1 - s_1, 1 - \mathcal{E}_1]$): It is $t_1 = 1 - \frac{2}{\pi}(1 - r)$ and $\underline{t_2} = \underline{cr}$ such that

$$\begin{aligned} & t_1 = 1 - \frac{2}{\pi}(1 - r) \leq r + \mathcal{E}_1 = \underline{cr} = \underline{t_2} \\ \Leftrightarrow & (1 - \frac{2}{\pi})(1 - r) \leq \mathcal{E}_1 \Leftrightarrow (1 - r) \leq s_1 \Leftrightarrow r \geq 1 - s_1 \end{aligned}$$

(Case $r \in (1 - \mathcal{E}_1, 1]$): It is $t_1 = 1 - \frac{2}{\pi}(1 - r)$ and $\underline{t_2} = 1 + \frac{2}{\pi}(\underline{cr} - 1)$ with $\underline{cr} > 1$ such that

$$t_1 = 1 - \frac{2}{\pi}(1 - r) \leq 1 \leq 1 + \frac{2}{\pi}(\underline{cr} - 1) = \underline{t_2}$$

(Case $r \in (1, 1 + s_1 - \mathcal{E}_1]$): It is $t_1 = r$ and $\underline{t_2} = 1 + \frac{2}{\pi}(\underline{cr} - 1)$ such that

$$\begin{aligned} & t_1 = r \leq 1 + \frac{2}{\pi}(r + \mathcal{E}_1 - 1) = 1 + \frac{2}{\pi}(\underline{cr} - 1) = \underline{t_2} \\ \Leftrightarrow & (1 - \frac{2}{\pi})r \leq (1 - \frac{2}{\pi}) - (1 - \frac{2}{\pi})\mathcal{E}_1 + \mathcal{E}_1 \\ \Leftrightarrow & r \leq 1 + s_1 - \mathcal{E}_1 \end{aligned}$$

(Case $r > 1 + s_1 - \mathcal{E}_1$): It is $t_1 = r$ and $\underline{t_2} = \underline{cr} - \mathcal{E}_1$, where

$$t_1 = r = \underline{cr} - \mathcal{E}_1 = \underline{t_2}$$

□

Now, define

$$\delta = \left| \frac{t - d(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x})}{1 + \|\boldsymbol{x}\|_2 - d(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x})} \right| = \left| T \frac{t - d(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x})}{2\|\boldsymbol{x}\|_2 \mu} \right|.$$

For $\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x}) \leq r$ we can lower bound the probability of $d_T(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x})$ not exceeding the specified threshold:

$$\begin{aligned} \mathbb{P}\big(d_T(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x}) \leq t\big) &= \mathbb{P}\big(\mathcal{C}(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x}) \geq (1 - \delta)\mu\big) \\ &= 1 - \mathbb{P}\big(\mathcal{C}(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x}) \leq (1 - \delta)\mu\big) \\ &\overset{(20)}{\geq} 1 - \exp\Big\{-\frac{\mu}{2}\delta^2\Big\}. \end{aligned}$$

We can show $d(\boldsymbol{q}^{\mathrm{iP}}, \boldsymbol{x}) \leq t_1$, using Lemma 1, (c) and (d):

(Case $r \leq 1 - s_1$):

$$d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) - \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq \mathcal{E}_1 \Rightarrow d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq r + \mathcal{E}_1$$

(Case $r \in (1 - s_1, 1)$):

$$d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) - \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq (1 - \frac{2}{\pi})(1 - \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}))$$
$$\Rightarrow d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq 1 - \frac{2}{\pi}(1 - \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x})) \leq 1 - \frac{2}{\pi}(1 - r) = t_1$$

(Case $r \geq 1$): For $\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq 1$ it is $d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq 1$. Else $d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x})$ such that

$$d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq \max\{1, \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x})\} \leq r = t_1$$

Thus we can bound

$$\delta \overset{d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq t_1 < t}{\geq} \frac{T(t - t_1)}{2\|\boldsymbol{x}\|_2 \mu} \overset{\|\boldsymbol{x}\|_2 \leq 1}{\geq} \frac{T(t - t_2)}{2\mu} = \frac{T(t_2 - t_1)}{4\mu}$$

and

$$\delta^2 \mu \geq \frac{T^2(t_2 - t_1)^2}{16\mu} \overset{\mu \leq T}{\geq} \frac{T(t_2 - t_1)^2}{16},$$

such that

$$\mathbb{P}\big(d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq t\big) \geq 1 - \exp\left\{-\frac{(t_2 - t_1)^2}{32} T\right\}.$$

For $\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > cr$ we can upper bound the probability of $d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x})$ dropping below the specified threshold:

$$\mathbb{P}\big(d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq t\big) = \mathbb{P}\big(\mathcal{C}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq (1 + \delta)\mu\big)$$
$$\overset{(21)}{\leq} \exp\left\{-\frac{\mu}{3} \min\{\delta, \delta^2\}\right\}.$$

We can show $d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq t_2$, using Lemma 1, (c) and (d):

(Case $cr \leq 1$): For $\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq 1$ it is $d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq 1$. Else $d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x})$ such that

$$d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq \min\{1, \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x})\} \geq cr = t_2$$

(Case $cr \in (1, 1 + s_1)$):

$$\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) - d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq (1 - \frac{2}{\pi})(\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) - 1)$$
$$\Rightarrow d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq 1 + \frac{2}{\pi}(\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) - 1) \geq 1 - \frac{2}{\pi}(cr - 1) = t_2$$

(Case $cr \geq 1 + s_1$):

$$\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) - d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq \mathcal{E}_1 \Rightarrow d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq cr - \mathcal{E}_1 = t_2$$

Thus we can bound

$$\delta \overset{d(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \geq t_2 > t}{\geq} \frac{T(t_2 - t)}{2\|\boldsymbol{x}\|_2 \mu} \overset{\|\boldsymbol{x}\|_2 \leq 1}{\geq} \frac{T(t_2 - t)}{2\mu} = \frac{T(t_2 - t_1)}{4\mu},$$

such that

$$\mathbb{P}\big(d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq t\big) \leq \exp\left\{-\min\left\{\tfrac{T(t_2 - t_1)}{12}, \tfrac{T^2(t_2 - t_1)^2}{48\mu}\right\}\right\}$$
$$\overset{\mu \leq T}{\leq} \exp\left\{-\min\left\{\tfrac{T(t_2 - t_1)}{12}, \tfrac{T(t_2 - t_1)^2}{48}\right\}\right\}$$
$$= \exp\left\{-\tfrac{T}{3} \min\left\{\tfrac{t_2 - t_1}{4}, \left(\tfrac{t_2 - t_1}{4}\right)^2\right\}\right\}$$
$$\overset{\frac{t_2 - t_1}{4} < 1}{=} \exp\left\{-\tfrac{T}{3}\left(\tfrac{t_2 - t_1}{4}\right)^2\right\} = \exp\left\{-\tfrac{(t_2 - t_1)^2}{48} T\right\}.$$

Now, define the events

$$E_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}): \text{ either } \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > r \text{ or } d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq t, \tag{22}$$

$$E_2(\boldsymbol{q}^{\mathrm{ip}}): \forall \boldsymbol{x} \in X: \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > cr \Rightarrow d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > t. \tag{23}$$

Assume that there exists $\boldsymbol{x}^*$ with $\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}^*) \leq r$. Then the algorithm is successful if both, $E_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}^*)$ and $E_2(\boldsymbol{q}^{\mathrm{ip}})$ hold simultaneously. Let $T \geq \frac{48}{(t_2-t_1)^2} \log(\frac{n}{\varepsilon})$. It is

$$\begin{aligned}
\mathbb{P}\big(E_2(\boldsymbol{q}^{\mathrm{ip}})\big) &= 1 - \mathbb{P}\big(\exists \boldsymbol{x} \in X: \mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > cr, d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}^*) \leq t\big) \\
&\geq 1 - \sum_{\boldsymbol{x} \in X} \mathbb{P}\big(\mathcal{L}_{\mathrm{ip}}(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) > cr, d_T(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}) \leq t\big) \\
&\geq 1 - n \exp\left\{-\tfrac{(t_2-t_1)^2}{48} T\right\} \geq 1 - \varepsilon.
\end{aligned}$$

Also it holds that

$$\mathbb{P}\big(E_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}^*)\big) \geq 1 - \left(\frac{\varepsilon}{n}\right)^{\frac{3}{2}}.$$

Therefore the probability of the algorithm to perform approximate nearest neighbor search correctly is larger than

$$\mathbb{P}\big(E_2(\boldsymbol{q}^{\mathrm{ip}}), E_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}^*)\big) \geq 1 - \mathbb{P}\big(\neg E_2(\boldsymbol{q}^{\mathrm{ip}})\big) - \mathbb{P}\big(\neg E_1(\boldsymbol{q}^{\mathrm{ip}}, \boldsymbol{x}^*)\big) \geq 1 - \varepsilon - \left(\frac{\varepsilon}{n}\right)^{\frac{3}{2}}.$$

F Details of Cover Tree

Here, we detail how to selectively explore the hash buckets with the code dissimilarity measure in non-increasing order. The difficulty is in that the dissimilarity $\mathcal{D}$ is a *linear combination* of metrics, where the weights are selected at query time. Such a metric is referred to as a *dynamic metric function* or a *multi-metric* [9]. We use a tree data structure, called the *cover tree* [6], to index the metric space.

We begin the description of the cover tree by introducing the *expansion constant* and the *base of the expansion constant*.

Expansion Constant (κ) [18]: is defined as the smallest value $\kappa \geq \psi$ such that every ball in the dataset $\mathcal{X}$ can be covered by κ balls in $\mathcal{X}$ of radius equal $1/\psi$. Here, ψ is the *base of the expansion constant*.

Data Structure: Given a set of data points $\mathcal{X}$, the cover tree $\mathcal{T}$ is a leveled tree where each level is associated with an integer label i, which decreases as the tree is descended. For ease of explanation, let $B_{\psi^i}(\boldsymbol{x})$ denote a *closed ball* centered at point $\boldsymbol{x}$ with radius ψ^i, i.e., $B_{\psi^i}(\boldsymbol{x}) = \{p \in \mathcal{X} : \mathcal{D}(p, \boldsymbol{x}) \leq \psi^i\}$. At every level i of $\mathcal{T}$ (except the root), we create a *union of possibly overlapping closed balls* with radius ψ^i that *cover* (or contain) all the data points $\mathcal{X}$. The centers of this covering set of balls are stored in *nodes* at level i of $\mathcal{T}$. Let $\mathcal{C}_i$ denote the set of nodes at level i. The cover tree $\mathcal{T}$ obeys the following three invariants at all levels:

1. (**Nesting**) $\mathcal{C}_i \subset \mathcal{C}_{i-1}$. Once a point $\boldsymbol{x} \in \mathcal{X}$ is in a node in $\mathcal{C}_i$, then it also appears in all its successor nodes.
2. (**Covering**) For every $\boldsymbol{x}' \in \mathcal{C}_{i-1}$, there exists a $\boldsymbol{x} \in \mathcal{C}_i$ where $\boldsymbol{x}'$ lies inside $B_{\psi^i}(\boldsymbol{x})$, and exactly one such $\boldsymbol{x}$ is a parent of $\boldsymbol{x}'$.
3. (**Separation**) For all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{C}_i$, $\boldsymbol{x}_1$ lies outside $B_{\psi^i}(\boldsymbol{x}_2)$ and $\boldsymbol{x}_2$ lies outside $B_{\psi^i}(\boldsymbol{x}_1)$.

This structure has a space bound of $O(N)$, where N is the number of samples.

Construction: We use the *batch* construction method [6], where the cover tree $\mathcal{T}$ is built in a *top-down* fashion. Initially, we pick a data point $\boldsymbol{x}^{(0)}$ and an integer s, such that the closed ball $B_{\psi^s}(\boldsymbol{x}^{(0)})$ is the tightest fit that covers the entire dataset $\mathcal{X}$.

This point $\boldsymbol{x}^{(0)}$ is placed in a single node, called the *root* of the tree $\mathcal{T}$. We denote the root node as $\mathcal{C}_i$ (where $i = s$). In order to generate the set $\mathcal{C}_{i-1}$ of the child nodes for $\mathcal{C}_i$, we

Algorithm 1 Nearest neighbor search with cover tree

Require: The cover tree $\mathcal{T}$ and the query point $\boldsymbol{q}$.
Ensure: The point $\boldsymbol{x}^*$ which is the closest to $\boldsymbol{q}$.
1: $\mathcal{C}_i \leftarrow \{\boldsymbol{x} \in \mathcal{T}.\text{root}\}$ ▷ set of points in root node
2: **for** $i \leftarrow \mathcal{T}.\text{root}$; $i \neq \mathcal{T}.\text{leaf}$; $i = i - 1$ **do** ▷ descend $\mathcal{T}$ level-wise
3: $\quad \mathcal{C} \leftarrow \{\text{children}(\boldsymbol{x}) : \boldsymbol{x} \in \mathcal{C}_i\}$ ▷ candidate set $\mathcal{C}$: children of $\mathcal{C}_i$
4: $\quad \mathcal{C}_{i-1} \leftarrow \{\boldsymbol{x} \in \mathcal{C} : \mathcal{D}(\boldsymbol{q}, \boldsymbol{x}) \leq \min_{\boldsymbol{x}' \in \mathcal{C}} \mathcal{D}(\boldsymbol{q}, \boldsymbol{x}') + \psi^i\}$ ▷ next cover set
5: $\quad$ **if** $\mathcal{C}_{i-1} = \mathcal{C}_i$ **then** ▷ no change in candidate set
6: $\quad\quad$ Exit the loop.
7: $\quad$ **end if**
8: **end for**
9: **return** $\arg\min_{\boldsymbol{x} \in \mathcal{C}_{i-1}} \mathcal{D}(\boldsymbol{q}, \boldsymbol{x})$

greedily pick a set of points (including point $\boldsymbol{x}^{(0)}$ from $\mathcal{C}_i$ to satisfy the *Nesting* invariant) and generate closed balls of radius ψ^{i-1} centered on them, in such a way that: (a) all center points lie inside $B_{\psi^i}(\boldsymbol{x}^{(0)})$ (*Covering* invariant), (b) no center point intersects with other balls of radius ψ^{i-1} at level $i-1$ (*Separation* invariant), and (c) the union of these closed balls covers the entire dataset $\mathcal{X}$. These chosen center points form the set of nodes $\mathcal{C}_{i-1}$. Child nodes are *recursively* generated from each node in $\mathcal{C}_{i-1}$, until each data point in $\mathcal{X}$ is the center of a closed ball and resides in a leaf node of $\mathcal{T}$.

Note that, while we construct our cover tree, we use our distance function $\mathcal{D}$ with all the weights set to 1.0, which upper bounds all subsequent distance metrics that depend on the queries. The construction time complexity is $O(\kappa^{12} N \ln N)$.

To achieve a more compact cover tree, we store only element identification numbers (IDs) in the cover tree, and not the original vectors. Furthermore, we store the hash bits using *compressed representation bit-sets* that reduce the storage size compared to a naive implementation down to T bits.

Querying: The nearest neighbor query in a cover tree is illustrated in Algorithm 1. The search for the nearest neighbor begins at the root of the cover tree and descends level-wise. On each descent, we build a candidate set $\mathcal{C}$ (Line 3), which holds all the child nodes (center points of our closed balls). We then *prune* away centers (nodes) in $\mathcal{C}$ (Line 4) that cannot possibly lead to a nearest neighbor to the query point $\boldsymbol{q}$, if we descended down them.

The pruning mechanism is predicated on a proven result in [6] which states that for any point $\boldsymbol{x} \in \mathcal{C}_{i-1}$, the distance between $\boldsymbol{x}$ and any descendant $\boldsymbol{x}'$ is upper bounded by ψ^i. Therefore, on Line 4, the $\min_{\boldsymbol{x}' \in \mathcal{C}} \mathcal{D}(\boldsymbol{q}, \boldsymbol{x}')$ term on the right-hand side of the inequality, computes the shortest distance from every center point to the query point $\boldsymbol{q}$. Any center point whose distance from $\boldsymbol{q}$ exceeds $\min_{\boldsymbol{x}' \in \mathcal{C}} \mathcal{D}(\boldsymbol{q}, \boldsymbol{x}') + \psi^i$ cannot possibly have a descendant that can replace the current closest center point to $\boldsymbol{q}$ and hence can safely be pruned. We add an additional check (lines 5–6) to speedup the search by not always descending to the leaf node. The time complexity of querying the cover tree is $O(\kappa^{12} \ln N)$.

Effect of multi-metric distance while querying: It is important to note that minimizing overlap between the closed balls on higher levels (i.e., closer to the root) of the cover tree can allow us to effectively prune a very large portion of the search space and compute the nearest neighbor faster.

Recall that the cover tree is constructed by setting our distance function $\mathcal{D}$ with all the weights set to 1.0. During querying, we allow $\mathcal{D}$ to be a linear combination of metrics, where the weights lie in the range $[0, 1]$, which means that the distance metric $\mathcal{D}$ used during querying always *under-estimates* the distances and reports lower distances. During querying, the cover tree's structure is still intact and all the invariant properties satisfied. The main difference occurs on Line 4 with the $\min_{\boldsymbol{x}' \in \mathcal{C}} \mathcal{D}(\boldsymbol{q}, \boldsymbol{x}')$ term, which is the shortest distance from a center point to the query $\boldsymbol{q}$ (using the new distance metric). Interestingly, this new distance gets even smaller, thus reducing our search radius (i.e., $\min_{\boldsymbol{x}' \in \mathcal{C}} \mathcal{D}(\boldsymbol{q}, \boldsymbol{x}') + \psi^i$) centered at $\boldsymbol{q}$, which in turn implies that at every level we manage to prune more center points, as the overlap between the closed balls also is reduced.

Streaming: The cover tree lends itself naturally to the setting where nearest neighbor computations have to be performed on a stream of data points. This is because the cover tree allows dynamic insertion and deletion of points. The time complexity for both these operations is $O(\kappa^6 \ln N)$, which is faster than querying.

Parameter choice: In our implementation for experiment, we set the *base of expansion constant* to $\psi = 1.2$, which we empirically found to work best on the texmex dataset.