Infinity Search: Approximate Vector Search with Projections on q-Metric Spaces

Antonio Pariente[*]
pariente@seas.upenn.edu
Ignacio Hounie[*]
ihounie@seas.upenn.edu
Santiago Segarra[†]
segarra@rice.edu
Alejandro Ribeiro[*]
aribeiro@seas.upenn.edu

Abstract

Despite the ubiquity of vector search applications, prevailing search algorithms overlook the metric structure of vector embeddings, treating it as a constraint rather than exploiting its underlying properties. In this paper, we demonstrate that in q-metric spaces, metric trees can leverage a stronger version of the triangle inequality to reduce comparisons for exact search. Notably, as q approaches infinity, the search complexity becomes logarithmic. Therefore, we propose a novel projection method that embeds vector datasets with arbitrary dissimilarity measures into q-metric spaces while preserving the nearest neighbor. We propose to learn an approximation of this projection to efficiently transform query points to a space where euclidean distances satisfy the desired properties. Our experimental results with text and image vector embeddings show that learning q-metric approximations enables classic metric tree algorithms—which typically underperform with high-dimensional data—to achieve competitive performance against state-of-the-art search methods.

1 Introduction

Given a dataset of vector embeddings, a dissimilarity function and a query, nearest neighbor search refers to the problem of finding the point in the dataset that is most similar to the query, or, in its stead, a vector that is among the most similar [Indyk and Motwani, 1998]. It is well known that exact search requires comparison with all the points in the dataset in the worst case [Hanov, 2011] but it is also well known that this search complexity can be reduced with proper organization of the data [Malkov and Yashunin, 2020].

In particular, if we consider search problems in metric spaces, namely, problems in which the dissimilarity function satisfies the triangle inequality, data can be organized in a metric tree that can be searched with smaller average complexity [Uhlmann, 1991]. In this paper we observe that metric spaces are a particular case of a more generic family of q-metric spaces. These spaces satisfy more restrictive triangle inequalities in which the q-th power of each side of a triangle is smaller than the sum of the qth powers of the other two sides. In the limit of growing q we obtain ∞-metric spaces in which each side of a triangle is smaller than the maximum of the other two. The first contribution of this paper is to show that:

(C1) The number of comparisons needed to find the nearest neighbor of a query in an ∞-metric space is, at most, the ceiling of the base-2 logarithm of the size of the dataset (Section 2).

This fact is not difficult to establish. It holds because in an ∞-metric space each comparison in a metric tree discards half of the dataset (Proof of Theorem 1). It is nevertheless remarkable because

[*]Department of Electrical and Systems Engineering, University of Pennsylvania.
[†]Department of Electrical and Systems Engineering, Rice University.

Preprint.

it is an exact bound (not an order) on worst case search performance (not average). We point out that ∞-metric spaces are often called ultrametrics and the ∞-triangle inequality is often called the strong triangle inequality [Dovgoshey, 2025]. Ultrametrics are equivalent to dendrograms used in hierarchical clustering [Draganov et al., 2025], a fact that may help to understand why search in ultrametric spaces has logarithmic complexity.

Because of (C1) we would like to solve search problems in ∞-metric spaces, thus the title of this paper. However, the data is what the data is and most problems in vector search involve dissimilarity functions that are not even metric [Zezula et al., 2006]. Because of this we seek to develop a projection operator that we can use to map a given dataset into a general q-metric space, including an ∞-metric space. Our second contribution is to interpret a dataset as a graph and to show that:

(C2) The set of shortest path distances as measured by the q-norm of paths satisfies the q-triangle inequality. We call this operation the canonical q-metric projection of a dataset (Section 3).

The q-norm of a path is q-root of the sum of the distances in each hop elevated to the power of q, which in the limit of an ∞-metric space reduces the maximum of the dissimilarities in the path. We emphasize for clarity that the q-norm of a path has no relationship with the q-norm of the vector embeddings. As all norms do, this q-norm of the vector embeddings satisfies the standard triangle inequality.

The canonical q-metric projection described in (C2) generates a q-metric space but there may be –indeed, we can construct – other possible projection operators. To argue in favor of canonical projections we draw two axioms from the existing literature [Carlsson et al., 2017]. The Axiom of Projection states that the projection of a q-metric space should be the same q-metric space. The Axiom of Transformation states that the projection of a q-metric space must respect the partial ordering of original spaces. I.e., if the dissimilarities of a dataset dominate *all* the dissimilarities of another, the q-metric distances of the projected datasets must satisfy the same relationship. Our next two contributions are theoretical results that follow from this axiomatic requirements

(C3) The canonical projection of (C2) is the only projection operator that satisfies the Axioms of Projection and Transformation (Section 3).

(C4) Any projection method that satisfies the Axiom of Transformation, the canonical projection in particular, preserves the nearest neighbor of any given query (Section 3.1).

As per (C3), the canonical projection of (C2) is the only reasonable method that we can use to generate a q-metric space. This is to the extent that the Axioms of Projection and Transformation are reasonable, but it is difficult to argue that they are not. Regardless, (C4) states that the canonical projection preserves the nearest neighbor information which is the focus of our search problem.

The combination of contributions (C1)-(C4) dictates that we can project datasets into ∞-metric spaces with a canonical projection to search with logarithmic complexity. The hitch is that q-canonical projections require computation of q-shortest paths. This is not a problem for the dataset, but it is problem for the query. To compute q-distances between the query and the dataset we need to compare the query with all points in the dataset, defeating the purpose of using ∞-metric distances to reduce search complexity. We work around this problem with our fifth contribution:

(C5) We learn a function to maps vector embeddings into separate embeddings such that their Euclidean distances estimate the distances of the q-metric projections of the original embedding. We train this function on the dataset and generalize it to queries (Section 4)

Combining (C1)-(C5) we propose to use canonical q-metric projections of a dataset to generate spaces where we expect nearest neighbor search to be more efficient. We incur this cost once and reutilize it for all subsequent searches. We then use the learned embedding in (C5) to estimate q-metric distances between queries and points in the dataset. The use of approximate computation of q-metric projections renders the resulting search algorithm approximate. Numerical experiments indicate that this approach is competitive with state-of-the-art approximate search algorithms in terms of search complexity and rank of the approximate nearest neighbor (Section 5).

We refer the reader to Appendix A for a discussion of related work.

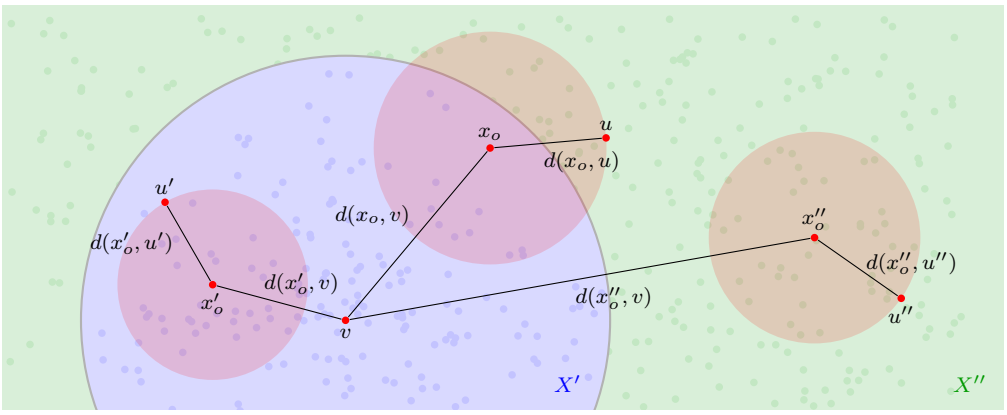


Figure 1: Search in metric spaces requires fewer comparisons than search in arbitrary spaces because for *some* queries – such as x'_o and x''_o – the triangle inequality allows us to restrict comparisons to subsets of the dataset X. Queries, such as x_o, for which triangle inequality bounds are inconclusive, also exist (5). This latter eventuality is impossible when the strong triangle inequality (3) holds, resulting in worst-case logarithmic complexity for search in ultrametric spaces (Theorem 1).

2 Nearest Neighbor Search and Metric Structure

We are given a set X containing m vectors $x \in \mathbb{R}^n$ along with a nonnegative dissimilarity function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$ so that smaller values of $d(x, y)$ represent more similarity between points x and y. The set X along with the set D containing all dissimilarities $d(x, y)$ for all points $x, y \in X$ defines a fully connected weighted graph $G = (X, D)$. We assume here that $d(x, y) = d(y, x)$ for all $x, y \in \mathbb{R}^n$, which, in particular, implies that the graph G is symmetric. Examples of dissimilarity functions used in vector search are the Euclidean distance, Manhattan (1-norm) distance, cosine dissimilarity, and dissimilarities based on the Jaccard index [Zezula et al., 2006]; see E.4.

We are further given a vector x_o which is not necessarily an element of X but for which it is possible to evaluate the dissimilarity function $d(x_o, x)$ for all $x \in X$. The nearest neighbor of x_o in the set X is defined as the element that is most similar to x_o,

$$\hat{x}_o \equiv \operatorname*{argmin}_{x \in X} d(x, x_o). \tag{1}$$

The problem of finding *a* vector $\hat{x}_o$ is termed nearest neighbor search, the vector x_o is called a query and the vector $\hat{x}_o$ is the query's answer [Wang et al., 2014]. It may be that there are several nearest neighbors. In such case we overload $\hat{x}_o$ to denote an arbitrary nearest neighbor and also the set of all nearest neighbors. We make the distinction clear when needed.

We evaluate the performance of a search algorithm by the number $c(x_o)$ of comparisons against elements of X that are needed to find $\hat{x}_o$. Without further assumptions on the dissimilarity function d, we have $c(x_o) = m$ because we need to compare the query x_o to all points in X. This number of comparisons can be reduced if we assume that d has some metric structure [Aksoy and Oikhberg, 2010]. For instance, it may be that d is a proper metric or pseudometric that satisfies the triangle inequality so that for any three points $x, y, z \in \mathbb{R}^n$ we have that

$$d(x, y) \leq d(x, z) + d(y, z). \tag{2}$$

Alternatively, we may consider ultrametric dissimilarity functions d [Simovici et al., 2004]. In this case, triplets of points $x, y, z \in \mathbb{R}^n$ satisfy the strong triangle inequality,

$$d(x, y) \leq \max\Big[\, d(x, z), d(y, z) \Big]. \tag{3}$$

In this paper, we also have an interest in q-metric spaces [Greenhoe, 2016]. These spaces arise when the dissimilarity function is such that any three points $x, y, z \in \mathbb{R}^n$ satisfy the q-triangle inequality,

$$d^q(x, y) \leq d^q(x, z) + d^q(y, z), \tag{4}$$

for some given $q \geq 1$. For $q = 1$, (4) reduces to (2) and as $q \to \infty$, (4) approaches (3). Thus, we can think of q-metric spaces as interpolations between regular metric spaces that sastify the standard

triangle inequality and ultrametric spaces that satisfy the strong triangle inequality. Henceforth, we may refer to (2) as the 1-triangle inequality and to (3) as the ∞-triangle inequality.

Figure 1 illustrates the use of metric structures in search. In this Figure, we have separated the dataset X into sets X' and X'' that contain points whose distance to some *vantage point* v is smaller or larger than a threshold μ – see Appendix C for details. We further consider queries x_o, x'_o, and x''_o for which we have evaluated dissimilarities $d(x_o, u)$, $d(x'_o, u')$, and $d(x''_o, u'')$ to the corresponding points u, u', and u''. If it turns out that the distance $d(x'_o, v)$ between the query x'_o and the vantage point v is such that $d(x'_o, v) + d(x'_o, u') \leq \mu$, the diagram makes it apparent that the nearest neighbor $\hat{x}'_o$ of the query x'_o *cannot* be an element of X''. Thus, we can restrict the nearest neighbor search to points $x' \in X'$ and save the computational cost of evaluating distances to points in X''. Likewise, if it turns out that for point x''_o we have $d(x''_o, v) - d(x''_o, u'') \geq \mu$ we can restrict the nearest neighbor search to points $x'' \in X''$ and save the cost of evaluating distances to points in X' – we give a formal proof of these statements in Proposition 3.

While this argument seems to indicate logarithmic complexity of nearest neighbor search in metric spaces, this is not quite so. The reason is that we may have queries such as x_o for which neither condition is true. I.e., there may be points x_o such that

$$\mu - d(x_o, u) < d(x_o, v) < \mu + d(x_o, u). \tag{5}$$

If $d(x_o, v)$ is such that (5) holds, the nearest neighbor $\hat{x}_o$ may be in X' or X'' and we therefore do not have a reduction in the number of comparisons needed to find $\hat{x}_o$.

A critical observation we make here is that (5) is impossible in an ultrametric space. Indeed, if we have a dissimilarity function d that satisfies the strong triangle inequality in (3), we show in Appendix B that for any three points x_o, u, and v and threshold μ, we either have $d(x_o, v) + d(x_o, u) \leq \mu$ or $d(x_o, v) - d(x_o, u) \geq \mu$. From this lemma we derive the following theorem.

Theorem 1. *Consider a dataset X with m elements and a dissimilarity function d satisfying the strong triangle inequality* (3)*. There is a search algorithm such that the number of comparisons $c(x_o)$ required to find the nearest neighbor $\hat{x}_o \in X$ of any query x_o is bounded as*

$$c(x_o) \leq \left\lceil \log_2 m \right\rceil. \tag{6}$$

The proof of Theorem 1 is constructive. Search with a vantage point (VP) tree attains (6).

It follows from Theorem 1 that, given a choice, we would like to solve the nearest neighbor search in ultrametric spaces. Alas, dissimilarity metrics of interest do not satisfy the strong triangle inequality. Some are not even metric.[3]Due to this mismatch, we propose here an approach to *approximate* nearest neighbor search based on the development of projection and embedding operators to compute and approximate q-metric distances:

Projection Operator. The projection operator P_q maps dissimilarities $d(x, x') \in D$ that do not necessarily satisfy the q-triangle inequality into distances $d_q(x, x') \in D_q$ that do satisfy the q-triangle inequality (Section 3).

Embedding Operator. The embedding operator Φ_q is a learned map from vectors $x \in \mathbb{R}^n$ into vectors $x_q \in \mathbb{R}^s$ such that 2-norms $\|x_q - x'_q\|_2$ approximate q-distances $d_q(x, x')$. The map is trained on the dataset X and applied to queries x_o (Section 4).

We use the projection operator P_q to process the dataset $G = (X, D)$ to produce a graph

$$G_q = (X, D_q)$$

such that any three points $x, y, z \in X$ satisfy the q-triangle inequality. This is a one-time pre-processing cost that we leverage for all queries. We use the learned embedding operator Φ_q to approximate the q-metric distances between a query x_o and arbitrary points $x \in X$ as $d_q(x_o, x) \approx \|x_{oq} - x_q\|_2$ with $x_{oq} = \Phi_q(x_o)$. This is needed because computing the true q-metric distance $d_q(x_o, x)$ requires comparing x_o to all points in X (Section 3). The approximation of q-metric distances with the embedding operator E_q renders this search methodology approximate. We show in numerical experiments that it is competitive with state-of-the-art approximate search approaches (Section 5).

[3]Notice that the usual q-norm $||x||_q$ does not necessarily satisfy the q-triangle inequality.

3 Projection Operator: Projecting Dissimilarities onto q-Metric Spaces

We seek to design a *q-metric projection* $P_q : (X, D) \to (X, D_q)$ where, for arbitrary input dissimilarities D, the distances in D_q satisfy the q-triangle inequality given in (4). Observe that finding a feasible projection is trivial. For example, consider a projection that assigns all distances in D_q to be 1, regardless of the input D. In this case, the image space (X, D_q) becomes an ultrametric (hence, a valid q-metric for all q). However, performing nearest neighbor search in (X, D_q) would be a poor proxy for searching in (X, D), as all distance information was lost in the projection. Motivated by this example, we aim to impose conditions on P_q that preserve meaningful distance information through the projection operation. We formalize these conditions as the following two axioms:

(A1) Axiom of Projection. The q-metric graph $G_q = (X, D_q)$ is a fixed point of the projection map P_q, i.e.,

$$P_q(G_q) = G_q. \tag{A1}$$

(A2) Axiom of Transformation. Consider any two graphs $G = (X, D)$ and $G' = (X', D')$ and a dissimilarity reducing map $\varphi : X \to X'$ such that $D(x, y) \geq D'(\varphi(x), \varphi(y))$ for all $x, y \in X$. Then, the output q-metric graphs $(X, D_q) = P_q(G)$ and $(X', D'_q) = P_q(G')$ satisfy, for all $x, y \in X$,

$$D_q(x, y) \geq D'_q(\varphi(x), \varphi(y)). \tag{A2}$$

Axiom (A1) is natural in our setting: if the graph under study already satisfies the q-triangle inequality, then the projection should introduce no distortion and simply return the same graph. Axiom (A2) enforces a notion of monotonicity: if the distances in one graph dominate those in another, this dominance should be preserved under projection. Though seemingly lax, Axioms (A1) and (A2) impose significant structure on the projection P_q. In fact, we show that *there exists a unique projection P_q that satisfies Axioms (A1) and (A2)*. Moreover, *this projection preserves nearest neighbors*. To formally state these results, we first introduce the canonical q-metric projection.

Canonical q-metric projection $P_q^\star$. Consider a graph equipped with a dissimilarity $G = (X, D)$ and let C_{xy} denote the set of all paths from x to y, where $x, y \in X$. For a given path $c = [x = x_0, x_1, \ldots, x_l = y] \in C_{xy}$, we define its q-length as $\ell_q(c) = \|[d(x, x_1), \ldots, d(x_{l-1}, y)]\|_q$. That is, the q-length of a path is the q-norm of the vector containing the dissimilarities along the path's edges. We define the canonical q-metric projection $(X, D_q) = P_q^\star(X, D)$ as

$$d_q(x, y) = \min_{c \in C_{xy}} \ell_q(c). \tag{7}$$

In words, the canonical q-metric projection computes the all-pairs shortest paths using the q-norm as the path cost function.

A first observation is that $P_q^\star$ is indeed a valid q-metric projection; i.e., that the output graph is guaranteed to satisfy the q-triangle inequality (see Appendix B). More importantly, $P_q^\star$ is uniquely characterized by the axioms of projection and transformation.

Theorem 2 (Existence and uniqueness)**.** *The canonical projection $P_q^\star$ satisfies Axioms (A1) and (A2). Moreover, if a q-metric projection P_q satisfies Axioms (A1) and (A2), it must be that $P_q = P_q^\star$.*

As we discuss next, Theorem 2 has direct practical applications for nearest-neighbor search.

3.1 Search in Projected q-metric Spaces

Because the Axiom of Transformation (A2) is satisfied, we expect $P_q^\star$ to preserve up to some extent the distance ordering of the original graph. In particular, we can show that the nearest neighbors are preserved by this projection. To formalize this, consider the graph $H = (Y, E)$ whose node set $Y = X \cup \{x_o\}$ includes the dataset X and the query x_o. The dissimilarity set E consists of the original dissimilarities D as well as the dissimilarities between the query and each data point, i.e., $E(x_o, x) = d(x_o, x)$ for all $x \in X$. We now apply the canonical projection to this extended graph, obtaining $(Y, E_q) = P_q^\star(H)$, where E_q includes projected distances of the form $E_q(x, x_o)$ between each data point $x \in X$ and the query x_o. We next state that this projection preserves the identity of the nearest neighbor of x_o.

Proposition 1. *Given a graph $G = (X, D)$ and a query x_o, define $H = (Y, E)$ with $Y = X \cup \{x_o\}$ and let $(Y, E_q) = P_q^\star(H)$ be the projected graph. Then, we have that the set of nearest neighbors satisfies*

$$\hat{x}_o \;\equiv\; \operatorname*{argmin}_{x \in X} E(x, x_o) \;\subseteq\; \operatorname*{argmin}_{x \in X} E_q(x, x_o). \tag{8}$$

Proposition 1 shows that the canonical projection $P_q^\star$ in (7) preserves the nearest neighbor. This is enticing because we have argued that search in q-metric spaces is easier (Figure 1) and proved that search in ∞-metric spaces requires a logarithmic number of comparisons (Theorem 1). However, Equation (8) does not immediately help us solve the original nearest neighbor problem in (1). The proposition is stated for the projected dissimilarity set E_q, which requires access to the full set E, including all dissimilarities $E(x_o, x) = d(x_o, x)$ for every $x \in X$. Thus, computing E_q requires comparing x_o with *all* points in X, which defeats the purpose of reducing the number of comparisons $c(x_o)$ through the use of q-metric projections. We address this challenge in the following section by learning an embedding operator that enables us to *approximate* the values $E_q(x, x_o)$ without computing all pairwise distances explicitly.

4 Embedding Operator: Learning to Approximate q-metrics

We approximate q-metrics $E_q(x, x_o)$ with the 2-norm of a parameterized embedding $\Phi(x; \theta)$. Formally, we want to find a function $\Phi(x; \theta) : \mathbb{R}^n \to \mathbb{R}^s$ such that for any $x_o \in \mathbb{R}^n$ and $x \in X$

$$E_q(x, x_o) = \big\|\Phi(x; \theta) - \Phi(x_o; \theta)\big\|, \tag{9}$$

where $\|\cdot\|$ denotes the 2-norm of a vector. We fit the parameter θ to approximate distances $D_q(x, y)$ given by the canonical projection $D_q = P_q^\star(D)$, by minimizing a quadratic loss,

$$\ell_{\mathrm{D}}(x, y) = \Big[\, D_q(x, y) - \big\|\Phi(x; \theta) - \Phi(y; \theta)\big\| \,\Big]^2. \tag{10}$$

We also consider an additional loss that measures the extent to which the q-triangle inequality is violated by the embedded distances. We choose a saturated linear penalty for this loss, explicitly,

$$\ell_{\mathrm{T}}(x, y, z) = \Big[\, \big\|\Phi(x; \theta) - \Phi(y; \theta)\big\|^q - \big\|\Phi(x; \theta) - \Phi(z; \theta)\big\|^q - \big\|\Phi(y; \theta) - \Phi(z; \theta)\big\|^q \,\Big]_+, \tag{11}$$

where $[\cdot]_+$ denotes the projection to the non-negative orthant. The loss ℓ_{T} is positive only when the q-triangle inequality is violated, in which case it takes on the value of the violation.

We minimize a linear combination of the losses $\ell_{\mathrm{D}}(x, y)$ and $\ell_{\mathrm{T}}(x, y, z)$ averaged over the dataset X

$$\theta^\star = \operatorname*{argmin}_{\theta} \alpha_{\mathrm{D}} \sum_{x, y \in X} \ell_{\mathrm{D}}(x, y) + \alpha_{\mathrm{T}} \sum_{x, y, z \in X} \ell_{\mathrm{T}}(x, y, z). \tag{12}$$

Notice that in (12) the loss ℓ_{T} is redundant. It encourages the Euclidean norm $\big\|\Phi(x; \theta) - \Phi(y; \theta)\big\|$ to satisfy the q-triangle inequality when this is already implicit in ℓ_{D} as the latter encourages proximity to distances $D_q(x, y)$ that we know satisfy the q-triangle inequality. We have observed in numerical experiments that adding ℓ_{T} improves performance (see Appendix E). Further notice that although we train θ over the dataset X we expect it to generalize to estimate the q-metric $E_q(x, x_o)$. Our numerical experiments in the next section indicate that this generalization is indeed successful.

5 Experiments

In this section, we validate the properties of the canonical projection E_q as well as the effects of the learned approximation $\Phi(\cdot, \theta^\star)$ in practical vector search settings. Since projecting the data with $P_q^\star$ is computationally expensive, we randomly sample a smaller subset of 1,000 points to evaluate the theoretical properties. To analyze the learned map $\Phi(\cdot, \theta)$, we use 10,000 samples from each dataset. In all approximation experiments, we parameterize the embedding projector Φ using the same fully connected Multi-Layer Perceptron (MLP) architecture. The full dataset X

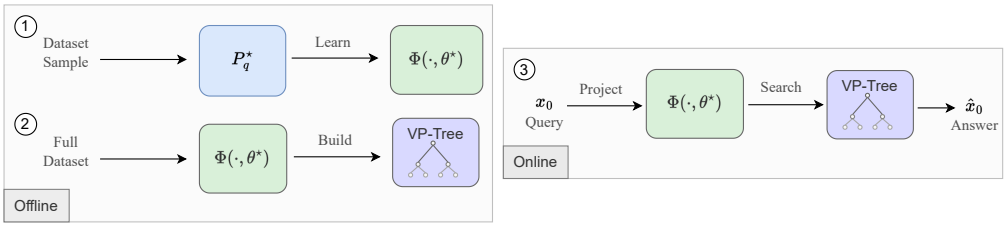
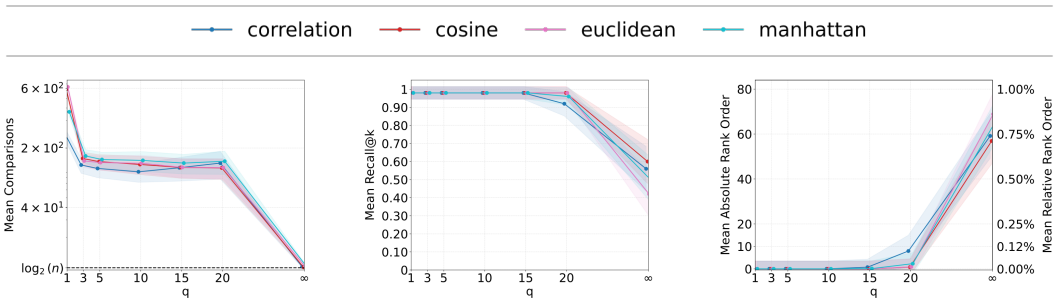
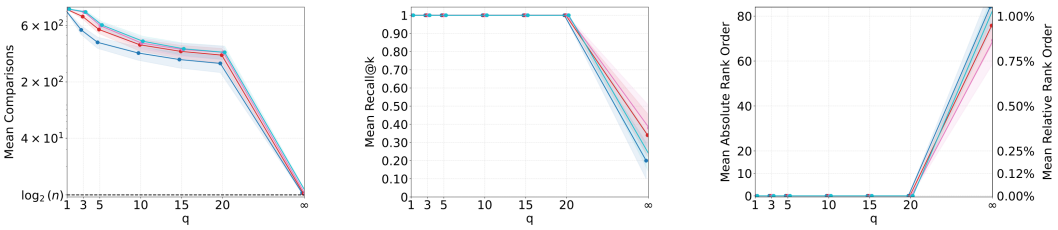


Figure 2: Infinity Search pipeline. Offline (left): Dataset samples are projected into a q-metric space using the canonical projection, and used to fit an embedding projection (1). Then, the dataset is transformed with the learned projection to build the VP-tree (2). Online (right): A query x_o is transformed with the learned projection and then search is conducted using the VP-tree index.

is projected using the trained MLP, and a VP-tree index is built on the resulting embedding. The Infinity Search framework is summarized in Figure 2. Details about the neural network architecture, training hyperparameters, and additional experimental results—including settings such as searching for $k \in 5, 10$ nearest neighbors—are provided in Appendix E. The three real-world datasets used are Fashion-MNIST [Xiao et al., 2017] with $d = 784$, GloVe [Pennington et al., 2014] $d = 200$ and Kosarak [Bodon, 2003] $d = 41{,}000$.



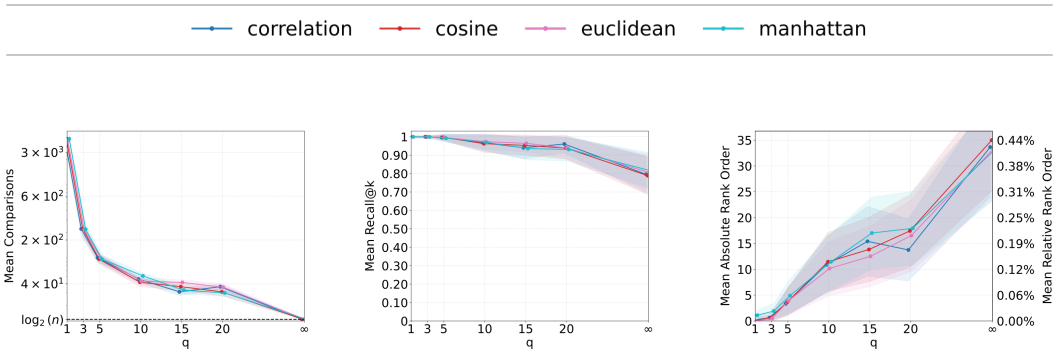
(a) Fashion-MNIST



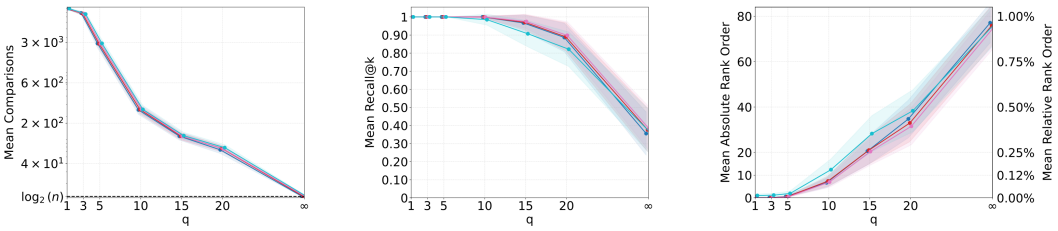
(b) GloVe

Figure 3: Number of comparisons and rank order when searching ($k = 1$) with Canonical Projection E_q for $n = 1{,}000$ points. Solid lines denote the mean and shading the standard deviation computed across queries.

Searching with the canonical projection. To analyze the effect of the q-metric structure on search complexity—decoupling it from that of the learned projection—we conduct experiments searching in the transformed graph G_q. We utilize a subset of the data due to the high computational cost associated with the canonical projection $P_q^\star$. As shown in Figure 3, increasing q leads to a smooth and significant decrease in the number of comparisons. The boundary stated in 1 is reached for $q = \infty$, effectively confirming claim C1. The fact that the projection consistently returns the nearest neighbor, further supports C2, as this behavior relies on the pruning effectiveness caused by the q-triangle inequality. In addition, the relative error shown in the middle column of Figure 3 indicates



(a) Fashion-MNIST



(b) GloVe

Figure 4: Number of comparisons and rank order when searching ($k = 1$) with learned embedding operator $\Phi(\cdot, \theta^\star)$ for $n = 10{,}000$. Solid lines denote the mean and shading the standard deviation computed across queries.

preservation of the nearest neighbor, described in claim C4 and proved in 8. At $q = \infty$, the projection may introduce spurious optima not present in the original nearest neighbor set, thereby affecting accuracy. A similar effect occurs for large q, as distances between points can become artificially close, imitating this behavior.

Learning approximation error. If the learning process attains sufficiently low generalization error, the theoretical properties are expected to hold in practice. Figure 4 shows a reduction in the number of comparisons as q increases, mirroring the trend observed in the Canonical Projection experiments. This reduction is monotonic with q, consistent across dissimilarities, and is accompanied by a moderate increase in rank error and a drop in recall. Recall values above 0.9 still yield speedups of up to three orders of magnitude. In all cases, the rank order remains below 1% of the indexed dataset. The trend observed in absolute rank order reflects partial locality preservation by the learned embedding. A deeper analysis of the learning component–specifically, the projection's ability to fit and generalize–is deferred to Appendix E.

These approximation experiments support the satisfactory fulfillment of claim C5, which concerns the generalization of the learned projection to unseen data. This is supported by the observed preservation of speedup and a generalization error that remains low—reflected in high recall—relative to what the theoretical properties would predict.

5.1 Infinity Search is competitive in ANN-Benchmarks

In this section, our end-to-end approach is compared against modern vector search algorithms, namely, HNSW [Malkov and Yashunin, 2018] as implemented in FAISS [Douze et al., 2024] and ONNG-NG [Arai, 2018]. We use ANN-Benchmarks [Aumüller et al., 2018], a popular evaluation framework for in-memory Approximate Nearest Neighbor algorithms.

In this setting, search algorithms are evaluated based on their ability to trade off search accuracy and speed. Speed is measured by queries-per-second throughput. Note that the learning phase in our method is carried out offline during index building and that computing the embedding projection is a query pre-processing step. As customary, these are not included in search time comparisons. Search

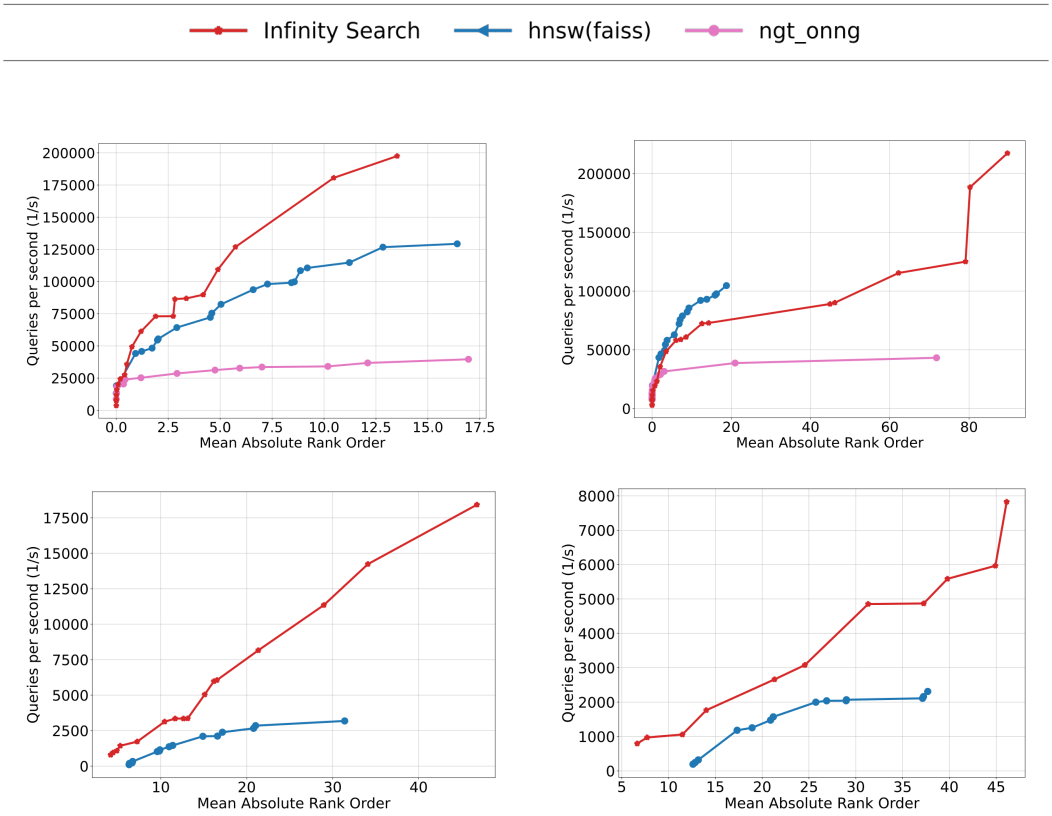


Figure 5: Search speed vs. accuracy on ANN-Benchmarks for Fashion-MNIST (top row) and Kosarak (bottom row). The left plots show nearest neighbor ($k = 1$) results whereas right plots show results for $k = 5$.

accuracy is measured using Rank Order for $k = 1$ – i.e. the position of the vector retrieved. For $k \in 5, 10$, additional nearest neighbor search results are provided in Appendix 16.

Across all datasets, our method either closely matches or outperforms baselines in nearest neighbor search ($k = 1$). Each point in the speed-accuracy curve in Figure 5 depicts a different hyperparameter setting. Notably, our method pareto dominates baselines by a large margin for settings with lower accuracy. That is, considerable speed gains can be obtained, with minimal accuracy trade-offs, if higher order metric structure is imposed. In the case of searching for more neighbours (k=5), we still observe that our method performs competitively. In particular, we observe that our method stands out in Kosarak, which is higher dimensional (40k), sparse and uses Jaccard similarity which is not supported by many methods and renders others ineffective. This highlights the ability of our method to accommodate arbitrary dissimilarities and scale to higher dimensional data, a property further examined in Appendix 8.

6 Conclusion

Driven by the insight that search in ultrametric spaces is logarithmic, in this paper we developed a projection that endows a set of vectors with arbitrary dissimilarities with q-metric structure, while preserving the nearest neighbor. This projection of a dataset is computed as the shortest path in a graph using the q-norm as path length. To address the challenge of computing q-metric distances for queries efficiently, we developed a learning approach that embeds vectors into a space where Euclidean distances approximate q-metric distances. Our experiments real-world datasets, which included high-dimensional sparse data with less common dissimilarity functions, demonstrated that our method is competitive with state-of-the-art approximate search algorithms.

References

[Aksoy and Oikhberg, 2010] Aksoy, A. G. and Oikhberg, T. (2010). Some results on metric trees. *arXiv preprint arXiv:1007.2207*.

[Arai, 2018] Arai, Y. (2018). Onng: Optimized navigable neighborhood graph (ngt variant). `https://github.com/yahoojapan/NGT`. Part of the NGT library, Yahoo Japan Research.

[Aumüller et al., 2018] Aumüller, M., Bernhardsson, E., and Faithfull, A. J. (2018). Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *CoRR*, abs/1807.05614.

[Babenko and Lempitsky, 2014] Babenko, A. and Lempitsky, V. (2014). The inverted multi-index. *IEEE transactions on pattern analysis and machine intelligence*, 37(6):1247–1260.

[Beygelzimer et al., 2006] Beygelzimer, A., Kakade, S., and Langford, J. (2006). Cover trees for nearest neighbor. In *ICML*.

[Bodon, 2003] Bodon, F. (2003). Kosarak click-stream dataset. `http://fimi.uantwerpen.be/data/kosarak.dat.gz`. Last accessed: 2022-08-05.

[Cao et al., 2021] Cao, W., Antonoglou, I., and Vinyals, O. (2021). Neural k-nn graph construction for approximate nearest neighbor search. In *ICLR*.

[Carlsson et al., 2017] Carlsson, G., Mémoli, F., Ribeiro, A., and Segarra, S. (2017). Admissible hierarchical clustering methods and algorithms for asymmetric networks. *IEEE Transactions on Signal and Information Processing over Networks*, 3(4):711–727.

[Cerd'a et al., 2024] Cerd
'a, A., Garc
'ıa, D., and Sainz, C. (2024). Lark: Learning adaptive routing for k-nn graph search. In *ICLR*.

[Dong et al., 2020] Dong, W., Park, J. V. A., and Samet, H. (2020). Diskann: Fast accurate billion-scale nearest neighbor search on a single node. In *ICML*.

[Douze et al., 2024] Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.-E., Lomeli, M., Hosseini, L., and Jégou, H. (2024). The faiss library. arXiv:2401.08281.

[Dovgoshey, 2025] Dovgoshey, O. (2025). Totally bounded ultrametric spaces and locally finite trees. *arXiv preprint arXiv:2502.04228*.

[Draganov et al., 2025] Draganov, A., Weber, P., Jørgensen, R. S. M., Beer, A., Plant, C., and Assent, I. (2025). I want 'em all (at once) – ultrametric cluster hierarchies. *arXiv preprint arXiv:2502.14018*.

[Echihabi et al., 2021] Echihabi, K., Zoumpatianos, K., and Palpanas, T. (2021). High-dimensional similarity search for scalable data science. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 2369–2372.

[Ge et al., 2020] Ge, T., Zhang, S., and Wang, J. (2020). Adapq: Adaptive product quantization for billion-scale approximate nearest neighbor search. In *CVPR*.

[Greenhoe, 2016] Greenhoe, D. J. (2016). Properties of distance spaces with power triangle inequalities. *arXiv preprint arXiv:1610.07594*.

[Guo et al., 2020] Guo, N., Sun, X., Shen, Z., Yu, F., Chen, W., and Wu, Y. (2020). Scann: Efficient vector similarity search at scale. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1571–1580. ACM.

[Hanov, 2011] Hanov, S. (2011). Implementing a fast and memory-efficient trie. `https://stevehanov.ca/blog/?id=130`. Accessed: 2024-11-13.

[Indyk and Motwani, 1998] Indyk, P. and Motwani, R. (1998). Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*.

[Jääsaari et al., 2024] Jääsaari, E., Hyvönen, V., and Roos, T. (2024). Lorann: Low-rank matrix factorization for approximate nearest neighbor search. *Advances in Neural Information Processing Systems*, 37:102121–102153.

[Kalantidis et al., 2014] Kalantidis, Y., Mikolajczyk, B. R., and Croxford, S. (2014). Locally optimized product quantization for approximate nearest neighbor search. In *CVPR*.

[Li et al., 2020] Li, C., Zhang, M., Andersen, D. G., and He, Y. (2020). Improving approximate nearest neighbor search through learned adaptive early termination. In *Proc. SIGMOD*, pages 2539–2554.

[Li et al., 2019] Li, W., Zhang, Y., Sun, Y., Wang, W., Li, M., Zhang, W., and Lin, X. (2019). Approximate nearest neighbor search on high dimensional data—experiments, analyses, and improvement. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1475–1488.

[Malkov and Yashunin, 2018] Malkov, Y. A. and Yashunin, D. A. (2018). Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.

[Malkov and Yashunin, 2020] Malkov, Y. A. and Yashunin, D. A. (2020). Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.

[Oliva and Torralba, 2001] Oliva, A. and Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175.

[Pan et al., 2024] Pan, J. J., Wang, J., and Li, G. (2024). Survey of vector database management systems. *The VLDB Journal*, 33(5):1591–1615.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

[Raguso et al., 2024] Raguso, M., Ri, M. D., and Farinelli, A. (2024). Lion: Learned inverted-file organisation for nearest neighbor retrieval. In *ICLR*.

[Roy et al., 2023] Roy, R., Jaeger, T., and Kannala, J. (2023). Hierarchical quantized autoencoders for efficient similarity search. In *CVPR*.

[Simovici et al., 2004] Simovici, D., Vetro, R., and Hua, K. (2004). Ultrametricity of dissimilarity spaces and its significance for data mining. In *Proceedings of the 2004 SIAM International Conference on Data Mining (SDM)*, pages 130–141. SIAM.

[Sun et al., 2023] Sun, P., Simcha, D., Dopson, D., Guo, R., and Kumar, S. (2023). Soar: improved indexing for approximate nearest neighbor search. *Advances in Neural Information Processing Systems*, 36:3189–3204.

[Uhlmann, 1991] Uhlmann, J. K. (1991). Satisfying general proximity / similarity queries with metric trees. *Information Processing Letters*, 40(4):175–179.

[Ukey et al., 2023] Ukey, N., Yang, Z., Li, B., Zhang, G., Hu, Y., and Zhang, W. (2023). Survey on exact knn queries over high-dimensional data space. *Sensors*, 23(2):629.

[Wang et al., 2014] Wang, J., Shen, H. T., Song, J., and Ji, J. (2014). Hashing for similarity search: A survey. *ACM Computing Surveys (CSUR)*, 46(1):1–29.

[Xiao et al., 2017] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

[Yang et al., 2021] Yang, K., Wang, H., Xu, B., Wang, W., Xiao, Y., Du, M., and Zhou, J. (2021). Tao: A learning framework for adaptive nearest neighbor search using static features only. *arXiv preprint arXiv:2110.00696*.

[Yianilos, 1993] Yianilos, P. N. (1993). Data structures and algorithms for nearest neighbor search in general metric spaces. In *SODA*.

[Yu et al., 2015] Yu, J., Ma, H., and Li, S. (2015). Spann: Highly-efficient billion-scale approximate nearest neighbor search. In *VLDB*.

[Zezula et al., 2006] Zezula, P., Amato, G., Dohnal, V., and Batko, M. (2006). *Similarity Search: The Metric Space Approach*. Springer.

[Zhao et al., 2021] Zhao, L., Zheng, Y., and Shao, J. (2021). Autoindex: A learned index for high-dimensional similarity search. In *NeurIPS*.

A Related Work

The literature on vector-based nearest–neighbour search is vast and spans many decades. We briefly review the families that are most germane to our contributions and direct the reader to existing surveys [Echihabi et al., 2021, Ukey et al., 2023, Li et al., 2019] for a broader analysis.

For data sets endowed with a proper metric, pruning rules derived from the standard triangle inequality have long been exploited through pivot- or vantage-point trees, k-d trees, ball trees, and their variants [Yianilos, 1993, Uhlmann, 1991, Beygelzimer et al., 2006]. Although these methods can yield exact results, they typically struggle with dimensionality and rely on existing metric structure of the date have been largely ouperformed by inverted file indexes [Babenko and Lempitsky, 2014], hashing [Indyk and Motwani, 1998, Wang et al., 2014], quantisation [Kalantidis et al., 2014, Ge et al., 2020] and graph based methods [Malkov and Yashunin, 2018] such as HNSW [Malkov and Yashunin, 2018]. Approaches—such as DiskANN [Dong et al., 2020], SPANN [Yu et al., 2015], and ONNG [Arai, 2018]— have optimized memory layout, link degree, or storage hierarchy to scale to billion-point datasets.

While many of these methods can offer strong performance, recent works increasingly seek to embed learning into the indexing process for data-aware improvements. ScaNN [Guo et al., 2020] jointly optimizes clustering and quantization, SOAR adds orthogonality-amplified residuals to further boost ScaNN's efficiency [Sun et al., 2023], and AutoIndex [Zhao et al., 2021] automates the selection of IVF–PQ parameters. Learned policies can also steer the search itself: L2-KNN [Cao et al., 2021] and LARK [Cerd'a et al., 2024] guide edge traversal at search time in graph based methods to reduce distance computations, AdaptNN predicts early-termination thresholds online [Li et al., 2020], and Tao learns to set them *before* query execution using only static local-intrinsic-dimension features [Yang et al., 2021]. Differentiable hierarchies, such as Hierarchical Quantized Autoencoders [Roy et al., 2023] and LION [Raguso et al., 2024], train the entire coarse-to-fine quantization pipeline end-to-end. LoRANN replaces PQ scoring with a supervised low-rank regressor that outperforms traditional IVF on billion-scale data [Jääsaari et al., 2024].

Industrial adoption of large language models has driven the emergence of "vector DBMSs", see [Pan et al., 2024] and references therein. Despite many developments in compression, partitioning, graph navigation, and hybrid attribute-vector query planning, most if not all systems index the *original* dissimilarities - or an approximation – and thus inherit their metric limitations. In contrast, we propose to pre-process data to obtain a q-metric space whose structure is provably more favourable for search.

B Projection

In this Appendix, theoretical properties exposed in Section 3, will be proved. The proofs rely on the Axioms A1 and A2, that are imposed on the Canonical Projection. These two requirements are enough to imbue the projection with beneficial properties on Nearest Neighbor search.

B.1 $P_q^\star$ exists, is unique, and satisfies the q-triangle inequality

For a given q, the Canonical Projection $P_q^\star$ as defined in 7, is guaranteed to satisfy the q-triangle inequality. Moreover, it meets axioms of Projection A1 and Transformation A2.

Lemma 1 (Satisfaction of q-triangle inequality)**.** *The canonical Projection as defined in 7, satisfies the q-triangle inequality.*

Proof. To verify the q-triangle inequality, let $c_{xx'}$ and $c_{x'x''}$ be paths that achieve the minimum in 7 for $d_q(x, x')$ and $d_q(x', x'')$, respectively. Then, from the definition in 7, it follows that:

$$d_q(x, x'')^q = \min_{c_{xx''}} \|c\|_q^q \leq \|c_{xx'} \oplus c_{x'x''}\|_q^q = \|c_{xx'}\|_q^q + \|c_{x'x''}\|_q^q = d_q(x, x')^q + d_q(x', x'')^q,$$

where the inequality holds because the concatenated path $c_{xx'} \oplus c_{x'x''}$ is a valid (though not necessarily optimal) path between x and x'', while $d_q(x, x'')$ minimizes the q-norm across all such paths. □

Theorem 3 (Existence and uniqueness)**.** *The canonical projection $P_q^\star$ satisfies Axioms A1 and A2. Moreover, if a q-metric projection P_q satisfies Axioms A1 and A2, it must be that $P_q = P_q^\star$.*

Proof. We first prove that d_q is indeed a q-metric on the node space X. That $d_q(x, x') = d_q(x', x)$ follows from the fact that the original graph G is symmetric, and that the norms $\|\cdot\|_q$ are symmetric in their arguments for all q. Moreover, $d_q(x, x') = 0$ if and only if $x = x'$, due to the positive definiteness of the q-norms.

To see that the Axiom of Projection A1 is satisfied, let $M = (X, d) \in \mathcal{M}_q$ be an arbitrary q-metric space, and denote $(X, d_q) = P_q^\star(M)$, the output of applying the canonical q-metric projection. For any pair $x, x' \in X$, we have:

$$d_q(x, x') = \min_{c \in C_{xx'}} \|c\|_q \leq \|[x, x']\|_q = d(x, x'),$$

for all q, where the inequality comes from choosing the trivial path $[x, x']$ consisting of a single edge from x to x'.

Let $c^\star_{xx'} = [x = x_0, x_1, \ldots, x_\ell = x']$ be the path that achieves the minimum in the above expression. Using Lemma 1 we know d_q satisfies the q-triangle inequality:

$$d(x, x') \leq \left(\sum_{i=0}^{\ell-1} d(x_i, x_{i+1})^q \right)^{1/q} = \|c^\star_{xx'}\|_q = d_q(x, x').$$

Substituting this into the previous inequality, we find that:

$$d(x, x') = d_q(x, x').$$

Since x and x' were arbitrary, we conclude $d \equiv d_q$, hence $P_q^\star(M) = M$, as required.

To show that the Axiom of Transformation A2 holds, consider two graphs $G = (X, D)$ and $G' = (X', D')$, and a dissimilarity-reducing map $\varphi : X \to X'$. Let $(X, d_q) = P_q^\star(G)$ and $(X', d'_q) = P_q^\star(G')$ be the projected spaces.

For a pair $x, x' \in X$, let $c^\star_{xx'} = [x = x_0, x_1, \ldots, x_\ell = x']$ be a path achieving the minimum for $d_q(x, x') = \|c^\star_{xx'}\|_q$. Consider the image path in X':

$$P^\star_{\varphi(x)\varphi(x')} = [\varphi(x_0), \varphi(x_1), \ldots, \varphi(x_\ell)].$$

Since φ is dissimilarity-reducing, we have:

$$D'(\varphi(x_i), \varphi(x_{i+1})) \leq D(x_i, x_{i+1}) \quad \text{for all } i.$$

Hence,

$$\|P^\star_{\varphi(x)\varphi(x')}\|_q \leq \|c^\star_{xx'}\|_q = d_q(x, x').$$

Now, since $d'_q(\varphi(x), \varphi(x'))$ is defined as the minimum over all such paths in X', we get:

$$d'_q(\varphi(x), \varphi(x')) \leq \|P^\star_{\varphi(x)\varphi(x')}\|_q \leq d_q(x, x'),$$

which completes the proof of Axiom A2. □

B.2 Canonical Projection preserves the Nearest Neighbor

As a consequence of the Axioms A1 and A2, the Canonical Projection $P_q^\star$ will preserve the Nearest Neighbor. This, when combined with Proposition 1, makes it suitable for Nearest Neighbor Search.

Proposition 2. *Given a graph $G = (X, D)$ and a query x_o, define $H = (Y, E)$ with $Y = X \cup \{x_o\}$ and let $(Y, E_q) = P_q^\star(H)$ be the projected graph. Then, we have that the set of nearest neighbors satisfies*

$$\hat{x}_o \;\equiv\; \operatorname*{argmin}_{x \in X} E(x, x_o) \;\subseteq\; \operatorname*{argmin}_{x \in X} E_q(x, x_o). \tag{13}$$

Proof. Let us construct a new graph $G' = (X', E',)$ as follows:

- $X' = \{x'_1, x'_2\}$
- $D' = \begin{bmatrix} 0 & E(\hat{x}_o, x_o) \\ E(\hat{x}_o, x_o) & 0 \end{bmatrix}$

Notice that, since (X', E') is a metric space, any admissible projection will leave the space unchanged. This is a direct consequence of Axiom 1 in (A1).

Let us now map the original graph G to the new one:

$$\varphi(y) = \begin{cases} x'_1 & \text{if } x = x_o, \\ x'_2 & \text{otherwise.} \end{cases} \tag{14}$$

Since the only existing edge has the minimum value possible $E(\hat{x}_o, x_o)$, φ is can be considered dissimilarity-reducing map. As a consequence of Axiom 2 in (A2), since

$$E(x, x_o) \geq E(\hat{x}_o, x_o) = E'(x'_1, x'_2)$$

therefore, any admissible projection will satisfy:

$$E'(x, x_o) \geq E'(x'_1, x'_2) = E(\hat{x}_o, x_o)$$

This enforces any admissible map deliver distances greater than the one to the original Nearest Neighbor. Moreover, if we reason analogously but interchanging the roles of x_o and its Nearest Neighbor, we can state that the distance is preserved:

$$E'(x, x_o) = E'(x'_1, x'_2)$$

By taking into account that the canonical projection is the only admissible map, since E' must be satisfied by $E_q = P_q^\star(H)$. This, proofs the preservation of the optima:

$$\hat{x}_o \;\equiv\; \operatorname*{argmin}_{x \in X} \; E(x, x_o) \;\subseteq\; \operatorname*{argmin}_{x \in X} \; E_q(x, x_o).$$

□

The following example illustrates a case where the set of original set of nearest neighbors can be extended if $q = \infty$:

Example 1. Let x_1, x_2, x_3 be three points equipped with the original distances

$$D(x_1, x_2) = 3, \quad D(x_2, x_3) = 2, \quad D(x_1, x_3) = 5.$$

Clearly $D(x_1, x_2) < D(x_1, x_2)$, so the unique nearest neighbor of x_1 in $G = (\{x_1, x_2, x_3\}, D)$ is x_2. We now enforce the strong triangle inequality by projecting:

$$D'(x_2, x_3) \;=\; \max\big\{D(x_1, x_2),\, D(x_2, x_3)\big\} \;=\; 3,$$

while leaving $d'(x_1, x_2) = 3$ and $D'(x_2, x_3) = 2$. Under the new distance D', one finds

$$D'(x_1, x_2) = 3 \quad \text{and} \quad D'(x_1, x_3) = 3,$$

so x_1 is equidistant from x_2 and x_3. In particular, the original nearest neighbor x_2 is no longer uniquely closest—any nearest-neighbor search on $G' = (\{x_1, x_2, x_3\}, D')$ may return x_3 instead of x_2, demonstrating that ultra-projecting a metric can extend original nearest neighbor set.

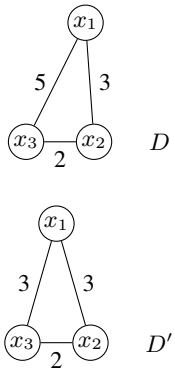


Figure 6: Ultrametric projection added nearest neighbors to the original set.

The proposition guarantees that the Nearest Neighbor is preserved under the mapping. Therefore, solving the solution found when solving Nearest Neighbor Search in the transformed space is a relaxation of the original problem. Although this relaxation provides speedup, it can also make the problem more difficult if many solutions are added to the nearest neighbor set. This problem is addressed in Section E, by further exploiting metric tree structure.

C Vantage Point Trees

In this section, we present our modification of the classical VP-Tree [Hanov, 2011] that leverages the q-triangle inequality. We analyze its properties including optimality preservation for q-metric spaces, and a probabilistic bound regarding the number of comparisons.

C.1 q-VPTree

The VP-Tree is a binary tree constructed by recursively partitioning the points based on pairwise distances, as detailed in Algorithm 1. After the tree is constructed, given a query point x it can be traversed as described in Algorithm 2.

Algorithm 1 Construction Phase of q-VPTree

function BUILDVPTREE (P)
  if $P = \emptyset$ **then**
    return `None`
  end if
  Select a vantage point $v \in P$
  Let $D \leftarrow \{d(x, v) \mid x \in P \setminus \{v\}\}$
  Let $\mu_v \leftarrow$ median of D
  Partition $P \setminus \{v\}$ into:
  $P_{\text{in}} \leftarrow \{x \in P \setminus \{v\} : d(x, v) \leq \mu_v\}$
  $P_{\text{out}} \leftarrow \{x \in P \setminus \{v\} : d(x, v) > \mu_v\}$
  Create node v storing (v, μ_v)
  v.left $\leftarrow$ BUILDVPTREE (P_{in})
  v.right $\leftarrow$ BUILDVPTREE (P_{out})
  return v
end function

Algorithm 2 SEARCHVPTREE: Searching Phase of the q-VPTree

function SEARCHVPTREE(v, x, τ, q)
  if $v =$ `None` **then**
    return
  end if
  Compute $d \leftarrow d(x, v)$ ▷ v is the vantage point at this node
  Evaluate candidate v: update nearest neighbor if $d < \tau$
  if $d(x, v)^q \leq \mu_v^q - \tau^q$ **then**
    SEARCHVPTREE(v.left, x, τ, q)
  else if $\mu_v^q - \tau^q < d(x, v)^q \leq \mu_v^q + \tau^q$ **then**
    SEARCHVPTREE(v.left, x, τ, q)
    SEARCHVPTREE(v.right, x, τ, q)
  else
    SEARCHVPTREE(v.right, x, τ, q)
  end if
end function

At each node in the VP-tree, the points covered by the left child lie inside the closed ball $B_q(v, \mu_v)$, while those in the right child lie outside that ball. The VP index thus partitions the search space into a hierarchy of nested balls. Intuitively, as q increases the balls used to prune comparisons can become tighter delivering a faster search. We formalize this in Section 2, showing that the probability of achieving the optimal search complexity $O(log(n))$ increases. If the q-triangle inequality is satisfied, the search Algorithm 2 yields *exactly* the nearest neighbor.

Despite its advantages, the q-VP Tree relies on the assumption of an underlying q-metric space, thus limiting its applicability. The Projection exposed in Section 3, endows this algorithm with unrestricted use.

C.2 q-VPTree preserves the Nearest Neighbor

Proposition 3. *When using a VP-tree for nearest neighbor search in a q-metric space, the following pruning rules ensure that no potentially optimal node is discarded:*

$$\begin{cases} \textit{i)} \quad D_q^q(z,v) \le \mu_v^q - \tau^q & \Rightarrow \textit{visit only the left child}, \\ \textit{ii)} \quad \mu_v^q - \tau^q < D_q^q(z,v) \le \mu_v^q + \tau^q & \Rightarrow \textit{visit both children}, \\ \textit{iii)} \quad \mu_v^q + \tau^q < D_q^q(z,v) & \Rightarrow \textit{visit only the right child}, \end{cases} \tag{15}$$

where z is the query point, v is the vantage point at the current node of the VP-tree, μ_v is the median distance at that node, τ is the current best-so-far distance, and D_q are the pairwise distances between indexed points living in a q-metric space.

Proof. We show that applying the rules in (15) ensures that no candidate point closer than τ to the query point z is discarded.

Case (i): Assume $D_q^q(z,v) \le \mu_v^q - \tau^q$. Then, by the q-triangle inequality, for any point $t \in X$,

$$D_q^q(v,t) \le D_q^q(v,z) + D_q^q(z,t) \le \mu_v^q - \tau^q + D_q^q(z,t). \tag{16}$$

For every t in the right child, we know by construction that $D_q(v,t) > \mu_v$, so $D_q^q(v,t) > \mu_v^q$. Combining this with (16), we get:

$$\mu_v^q < \mu_v^q - \tau^q + D_q^q(z,t) \quad \Rightarrow \quad D_q^q(z,t) > \tau^q,$$

implying $D_q(z,t) > \tau$. Thus, all points in the right child are farther from z than the current best distance and can be safely pruned.

Case (iii): Assume $\mu_v^q + \tau^q < D_q^q(z,v)$. Using the q-triangle inequality again, for any $t \in X$,

$$D_q^q(z,v) \le D_q^q(z,t) + D_q^q(t,v). \tag{17}$$

Since all points t in the left child satisfy $D_q(t,v) \le \mu_v$, we have $D_q^q(t,v) \le \mu_v^q$. Combining with (17):

$$\mu_v^q + \tau^q < D_q^q(z,v) \le D_q^q(z,t) + \mu_v^q \quad \Rightarrow \quad D_q^q(z,t) > \tau^q,$$

and hence $D_q(z,t) > \tau$, so the left child can be pruned.

Case (ii): This is the ambiguous region where neither child can be safely discarded based solely on the bounds, so both must be visited to guarantee correctness.

This concludes the proof. □

D Learning $\phi(\cdot, \theta)$

D.1 Efficient Approximation of P_q^*

Calculating the canonical projection $P_q^\star$ over training points can incur in prohibitive computing times. Therefore, instead of computing the shortest q-norm path in the complete graph $D \in \mathbb{R}^{n \times n}$, we restrict the search to length l paths and only consider the k-nearest neighbor graph. By doing this, the complexity of computing all pairwise distances is reduced from $O(n^3 log(n))$ to a factor $O(nk^2 l)$.

Given a dissimilarity distance matrix $D \in \mathbb{R}^{n \times n}$ (e.g., cosine or Euclidean), we raise it element-wise to the power q to obtain edge weights consistent with the q-path metric. Then, for each node x_i, we consider only its k nearest neighbors in the original space and perform a fixed number of smoothing updates, which approximate the optimal q-path cost to all other nodes.

At each iteration, for every node x_i and each of its neighbors x_j, we update the path cost $D(x_i, x')$ to a third node x' using the relaxed bound:

$$D(x_i, x') \leftarrow \min \left\{ D(x_i, x'),\ D(x_i, x_j)^q + D(x_j, x')^q \right\}.$$

The final result is root-transformed to return to the original distance scale:

$$d_q^*(x_i, x_j) \approx \left(D(x_i, x_j) \right)^{1/q}.$$

This can be implemented efficiently in GPU using sparse matrix multiplications.

E Experiments & Results

E.1 Experimental settings

Vector Datasets

The experiments were conducted on two commonly used datasets and one with distinct characteristics. We split the data randomly, using 80% for indexing and the remaining 20% is used as queries. We provide a summary of the datasets below, additional details can be found in the references provided.

- **[Pennington et al., 2014]GloVe Embeddings**: Text embeddings extracted from the GloVe (Global Vectors for Word Representation) model, each of dimension 200.
- **[Xiao et al., 2017]Fashion-MNIST**: Image samples from the Fashion-MNIST collection, each flattened into a 784-dimensional vector.
- **[Bodon, 2003]Kosarak**: A real-world sparse binary transaction dataset derived from click-stream data of a Hungarian news portal. Each transaction is represented as a 41,000-dimensional vector.
- **[Oliva and Torralba, 2001]Gist 960 Euclidean**: A set of real-valued GIST descriptors of natural scene images, each represented as a 960-dimensional vector. A GIST descriptor is a global, low-dimensional representation of an image's spatial envelope.

Dissimilarities

Apart from searching using the euclidean distance, as customary for Glove and FashionMNIST, to showcase how our approach can accommodate arbitrary dissimilarity functions, we also evaluate our method and results searching with other dissimilarities on the same datasets. The dissimilarities used are specified in Table 1.

Table 1: Distance and Dissimilarity Metrics used.

Metric	**Formula**
Euclidean Distance	$d(x,y) = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$
Manhattan Distance	$d(x,y) = \sum_{i=1}^{d} \lvert x_i - y_i \rvert$
Cosine Dissimilarity	$d(x,y) = 1 - \frac{x \cdot y}{\lVert x \rVert \, \lVert y \rVert}$
Correlation	$d(x,y) = 1 - \frac{(x-\bar{x})\cdot(y-\bar{y})}{\lVert x-\bar{x} \rVert \, \lVert y-\bar{y} \rVert}$

Metrics

Approximate search algorithms are evaluated along two key dimensions: retrieval quality and search efficiency.

Search efficiency is assessed primarily by the number of comparisons needed to retrieve a result for a given query. Unlike throughput, this metric is agnostic to implementation and hardware, offering a fair basis for comparing algorithmic efficiency.

- **Number of Comparisons**: Every time the q-Metric VP-Tree visits a node. Reflects the computational cost related to search speed.
- **Queries Per Second (qps)**: Measures the throughput of the algorithm, indicating how many queries can be processed per second under the current configuration.

Retrieval quality is evaluated using metrics that capture not just whether relevant results are retrieved, but how well their ordering is preserved. In particular, we rely on Recall@k and Rank Order, which – unlike recall – penalizes deviations in the relative ranking of retrieved results. This is crucial in downstream tasks such as recommendation, where the order of results matters. The metrics used in our evaluation are summarized below:

- **RankOrder@k**: Measures how well the approximate method preserves the original ordering of the true nearest neighbors. Let $\mathcal{N}_k^{\text{true}}(y)$ be the true k-nearest neighbors of a query y,

and $\mathcal{N}_k^{\text{approx}}(y) = \{x_1, \ldots, x_k\}$ the corresponding approximate result. Let $\pi(x_i, \mathcal{N}_k^{\text{true}}(y))$ denote the position of x_i in the true result $\mathcal{N}_k^{\text{true}}(y)$ (or $k+1$ if not found). The metric is defined as:

$$\textbf{Absolute RankOrder@k}(y) = \sum_{i=1}^{k} \left| i - \pi(x_i, \mathcal{N}_k^{\text{true}}(y)) \right| \cdot \frac{1}{k}$$

Lower values indicate better rank preservation, with 0 being optimal.
In addition, a variation of the rank order that takes into account the total number of points in the dataset is also used:

$$\textbf{Relative RankOrder@k}(y) = \sum_{i=1}^{k} \left| i - \pi(x_i, \mathcal{N}_k^{\text{true}}(y)) \right| \cdot \frac{100}{nk}$$

where n is the size of the indexed points, i.e $|X|$. This measure expresses rank order but now as a percentage of the points available for retrieval.

- **Recall@k**: Measures the proportion of true k-nearest neighbors that are successfully retrieved by the approximate method. Let $\mathcal{N}_k^{\text{true}}(y)$ be the true k-nearest neighbors of a query y, and $\mathcal{N}_k^{\text{approx}}(y)$ the corresponding approximate result. The recall is defined as:

$$\texttt{Recall@k}(y) = \frac{|\mathcal{N}_k^{\text{true}}(y) \cap \mathcal{N}_k^{\text{approx}}(y)|}{k}$$

This metric ranges from 0 to 1, where 1 indicates that all true neighbors were retrieved. It reflects the *coverage* of the ground-truth neighbors in the approximate result.

E.2 Canonical Projection $P_q^\star$

We validate the theoretical properties of searching in q-metric and ultrametric spaces presented in Section 2. In particular, we confirm our complexity claims (C1), the preservation of nearest neighbors (C2), and the stability of rank order under projection (C4).

In order to do so, we use the Canonical Projection $P_q^\star$ presented in Section 3 to project distances imposing q-metric structure on FashionMNIST and GloVe. For these two datasets we use four different dissimilarities. After projecting dissimilarities, we search using the q-metric VP tree as described in Appendix C.

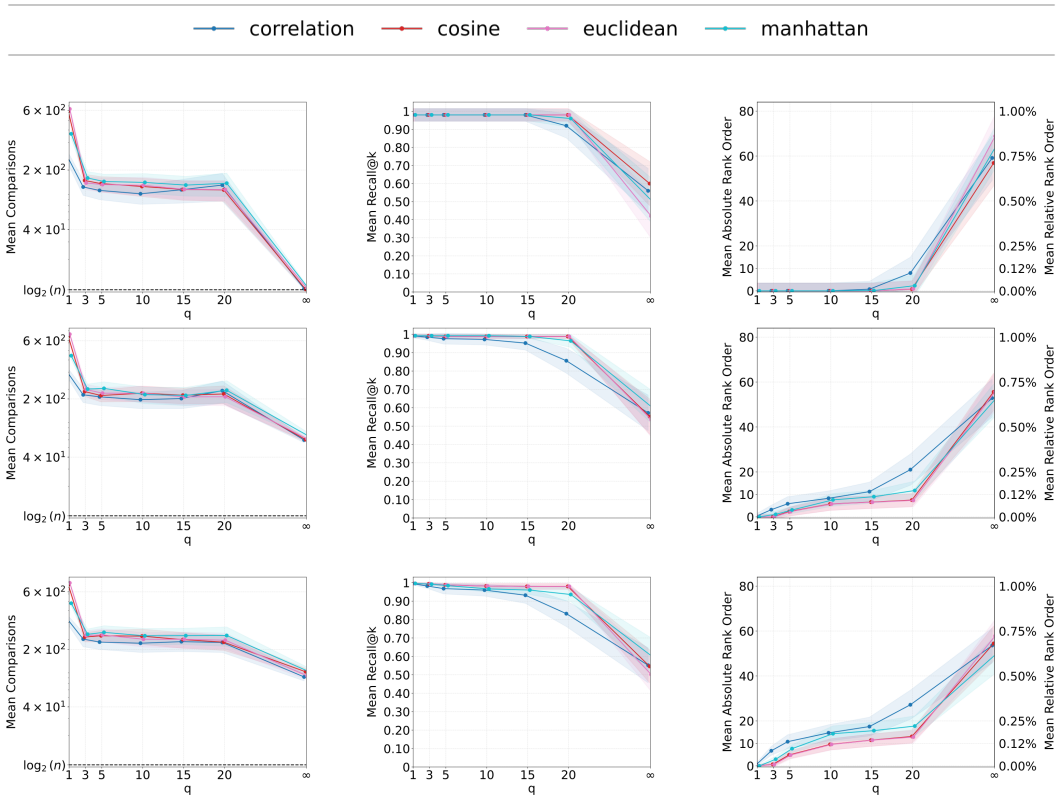
Figures 7a and 7b (first row), show that the number of nodes visited during search decreases monotonically with increasing q, reaching the theoretical minimum of $\log_2(n)$ at $q = \infty$. This directly confirms Theorem 1 and supports claim C1. Additionally, a rank order of zero across all queries–for a wide range of q values (excluding $q = \infty$) and across all dissimilarities– confirms claims C2 and C4, as established in Lemma 1 and Proposition 1. At the same time, this nearest neighbor preservation is also observed at recall, showing perfect matches at $k = 1$ for moderate q values.

At $q = \infty$, the projection may introduce spurious optima not included in the original nearest neighbor set. Although the original nearest neighbours are still a solution in the transformed space, in practice we observe that the spurious optima at $q = \infty$ indeed affect accuracy. A similar effect occurs for large q, as distances between points can become artificially close, imitating this behavior.

Although our theoretical guarantees focus on the $k = 1$ nearest neighbor case, an empirical preservation of locality observed in Figures 7a and 7b. We evaluate this by searching for $k = 5$ and $k = 10$ neighbors using the projected distances, as shown in the second and third rows of Figures 7a and 7b. While the number of comparisons does not decrease as rapidly as in the $k = 1$ case, the method consistently yields improvements across all values of q.

E.3 Approximating the canonical projection with $\Phi(x; \theta^\star)$

We analyze how well the learned distances $\hat{E}_q(x, x') = \|\Phi(x; \theta^\star) - \Phi(x'; \theta^\star)\|$, described in Section 4, reproduce the properties of the true q-metric distances $E_q(x, x')$.



(a) $n = 1{,}000$ points of Fashion-MNIST

Figure 7: Number of comparisons and rank order across different dissimilarities when searching after applying Canonical Projection when a query point is added (E_q). The search was performed with a q-VPTree. Solid lines denote the mean and shading the standard deviation computed across queries. Each row shows results for k-nearest neighbors, with $k = 1, 5, 10$ from top to bottom.

As described in Section 4, the learning process minimizes two loss terms: the stress ℓ_D, which measures the squared error between the learned distances and the true projected distances, and the triangle inequality violation ℓ_T, which penalizes failure to satisfy the q-triangle inequality.

Figure 9 shows that ℓ_D increases monotonically with q, consistent with the trend observed in Section E.4 where retrieval accuracy declined at high q values. This indicates that approximating the Canonical Projection becomes more difficult as q increases.

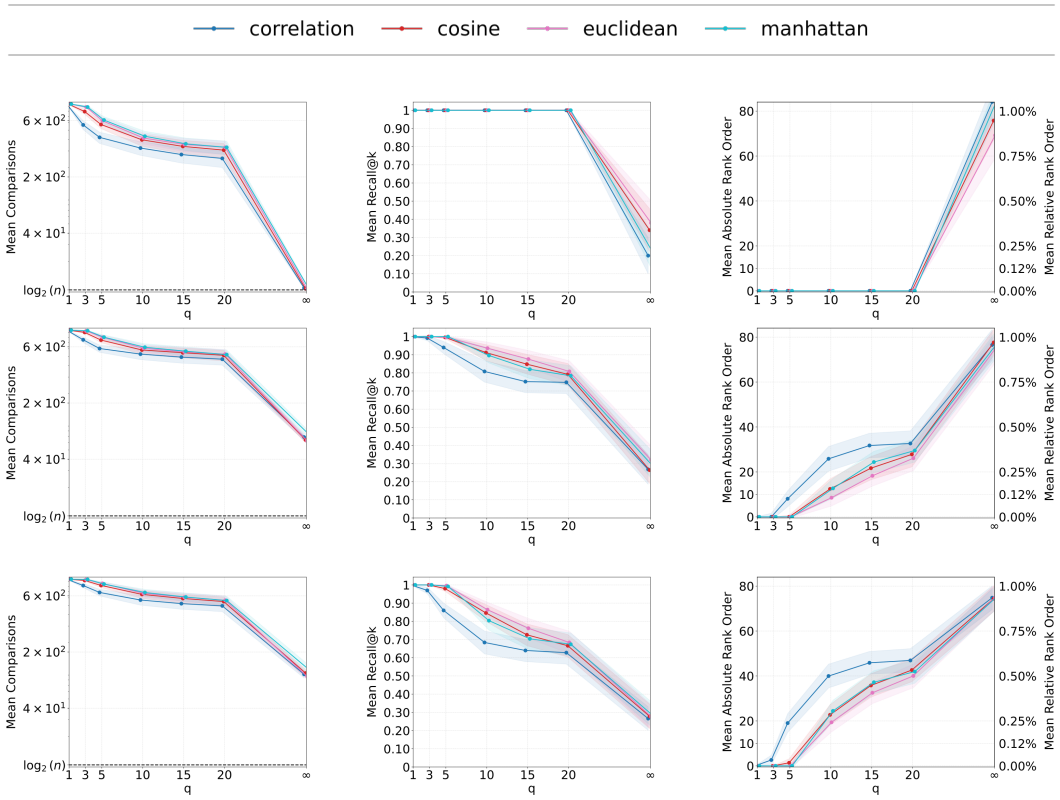
We observe the violation of the q-triangle inequality ℓ_T decreases with increasing q. However, the corresponding increase in accuracy metrics like rank order shows an opposite behavior is observed. That is, we observe ℓ_D to be more correlated with downstream performance than the satisfaction of the q-triangle inequality as measured by ℓ_T.

Predicted and ground truth distances in Figure 10 show similar distributions for training and testing distances albeit for a generalization gap. Again as q increases, so does the approximation error.

E.4 Searching with $\phi_q(\cdot, \theta^\star)$

In this section we demonstrate that the speedup observed in the exact projection experiments of Section E.2 is also attained when replacing the Canonical projection with the learned map $\phi_q(\cdot, \theta^\star)$, at the cost of small errors in search results.

Figures 11a and 11b show a consistent reduction in the number of comparisons as q increases, mirroring the trend observed in the exact projection experiments. This reduction is accompanied by



(b) $n = 1{,}000$ points of GloVe

Figure 7: Number of comparisons and rank order across different dissimilarities when searching after applying Canonical Projection when a query point is added (E_q). The search was performed with a q-VPTree. Solid lines denote the mean and shading the standard deviation computed across queries. The k-nearest neighbors are listed from top to bottom for $k = 1, 5, 10$

a moderate increase in rank error and a decrease in recall, suggesting that the retrieved neighbors remain close but are not always exact. Nevertheless, cases with recall above 0.9 still yield substantial speedups. Since recall captures only exact matches and the method guarantees order preservation only for the 1-nearest neighbor, rank order can provide a complementary view of performance.

Despite the degradation of the approximation quality for high q values, the retrieved neighbors remain within the top 80 nearest, which corresponds to less than 1% of the indexed dataset. When higher precision is required, setting $q = 10$ can yield a two-orders-of-magnitude speedup (Figure 11a) while returning neighbors ranked around 10th, corresponding to a relative error close to 0.12%. These results demonstrate that the learned embeddings preserve local structure to a satisfactory extent.

The results also extend to k-nearest neighbor search with $k > 1$. While the reduction in comparisons is less pronounced than for $k = 1$, the method maintains a consistent speedup across values of q. Rank Order remains low, and in some cases improves as k increases, suggesting that the learned map preserves small-scale neighborhood structure reasonably well even on these datasets.

Examples of retrieval results were also generated. For each dataset, queries were selected from varied categories. In the Fashion-MNIST case (Figure 13), each panel shows the original query image, its true nearest neighbor, and the result returned by Infinity Search. While the exact nearest neighbor was not retrieved in some cases—such as those involving sandals or sneakers—the returned items consistently belonged to the same category as the query. For GloVe-200 text embeddings, Figure 13a includes several exact matches, as well as examples where the retrieved word preserved the semantic meaning of the query.

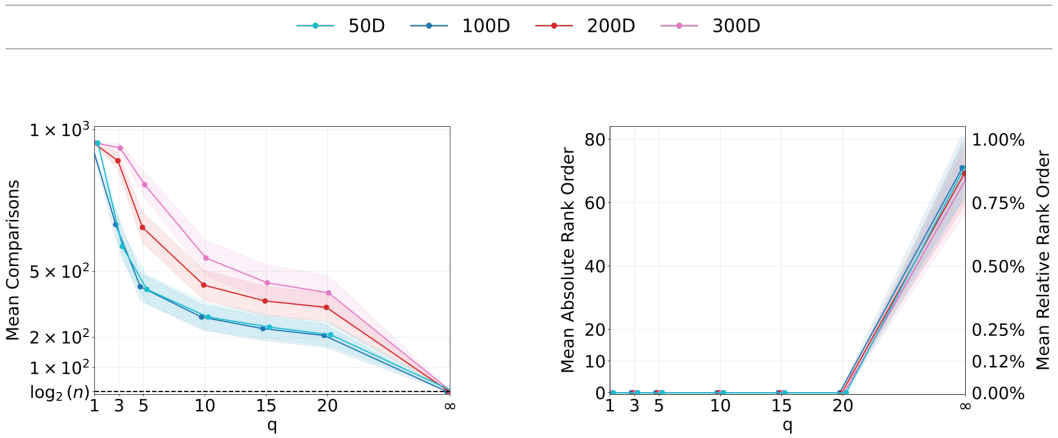
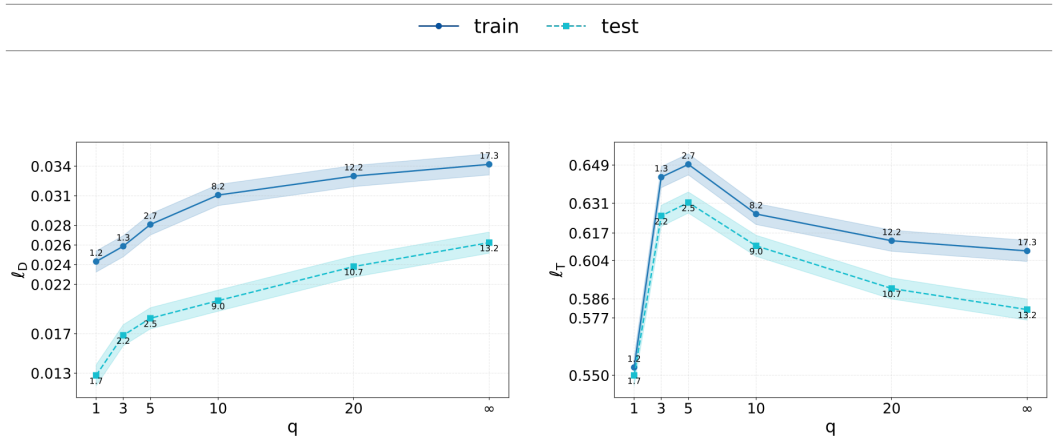


Figure 8: Number of comparisons and rank order across different dimensions when searching after applying Canonical Projection when a query point is added (E_q). Solid lines denote the mean and shading the standard deviation computed across queries. The dataset used was GloVe with $n = 1{,}000$ points and the dissimilarity the standard euclidean distance.



(a) Stress ℓ_D

(b) q-Triangle Inequality Violation ℓ_T

Figure 9: Values of the Stress ℓ_D and the q-triangle inequality regularizer after the training process. Labels of each point represent average Rank Order. This case corresponds to the learning process performed over the Fashion-MNIST dataset for euclidean distance.

E.5 Two-Stage Infinity Search

Although Proposition 8 ensures preservation of the nearest neighbor structure, it also notes that the projected nearest neighbor may not remain unique. This ambiguity can lead to mismatches during retrieval. The effect appears in theoretical settings, as seen in Figures 7a and 7b, where rank order the rank order exhibits a sharp increase at $q = \infty$. A similar trend can be observed in the approximate setting, shown in Figures 11a and 11b. In Figure 14, the distribution of projected q-metrics shows a clustering effect as q increases. Distances become more concentrated around mean, while the frequency of close values also increases. This suggests that distances between points become closer, making true nearest neighbors more difficult to discern.

To address this, one can extend the nearest neighbor set using the Canonical Projection and then prune it to avoid loss in accuracy. This motivates a two-stage modification of the Infinity Search algorithm:

- **Broad Search**: The Infinity Search algorithm is used to retrieve an initial candidate set of K nearest neighbors. This will retrieve close neighbors in the q-metric space.
- **Specific Search**: Once the list of K candidate neighbors is available, the original distance D is used to retrieve the k real nearest neighbors.

This two-stage retrieval strategy is used in some ANN methods, including HNSW [Malkov and Yashunin, 2020]. As shown in the theoretical and approximate experiments of Sections E.2 and E.4, the Canonical Projection preserves locality but not the exact order of nearest neighbors. This makes the two-stage approach suitable for improving accuracy.

Figures 15a and 15b confirm the improvement, showing higher recall and more accurate rank alignment compared to earlier approximations. The gains are particularly notable in rank order, with a 3 to 4 times reduction in error. As expected, this comes with a decrease in speedup, since the method processes a larger candidate set and computes original distances during Specific Search. Although the logarithmic comparison bound ($\log_2(n)$) no longer holds, the resulting speed remains competitive. The original Infinity Search can be recovered by setting $K = k$, while choosing $K > k$ offers additional flexibility to trade off speed and accuracy depending on the requirements of the searching problem.

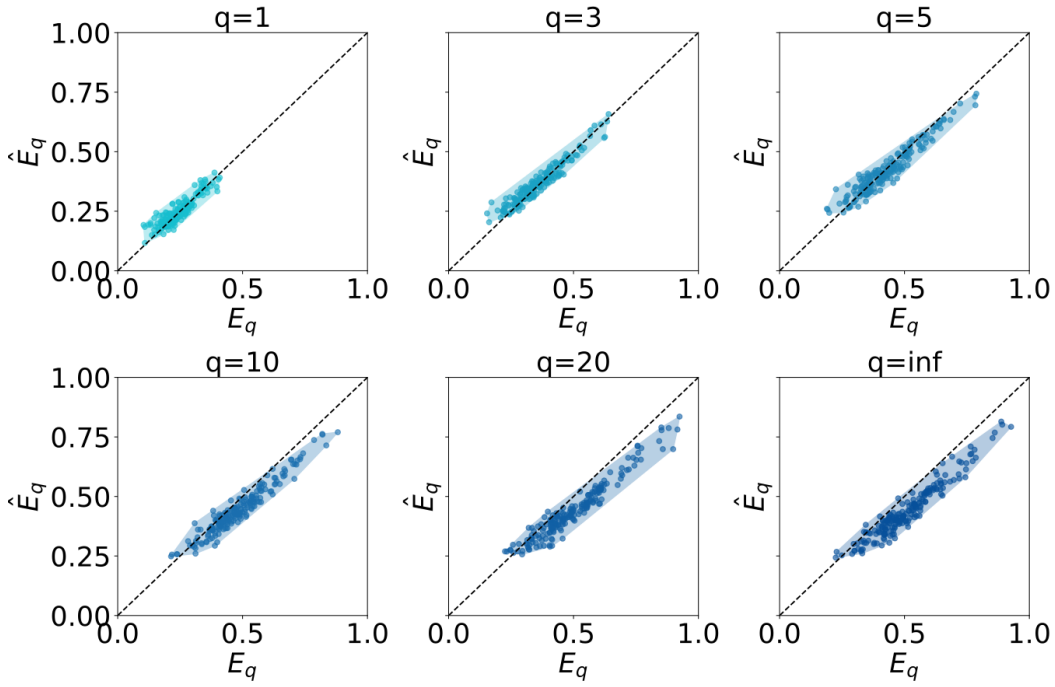
E.6 ANN-Benchmarks

Infinity Search offers a configurable trade-off between query throughput and recall. To assess its competitiveness against state-of-the-art ANN methods, we evaluated it within the ANN-Benchmarks framework [Aumüller et al., 2018]. We ran experiments on three datasets provided by the library and compared against a wide set of algorithms chosen for their balance of speed, accuracy, and open-source availability.

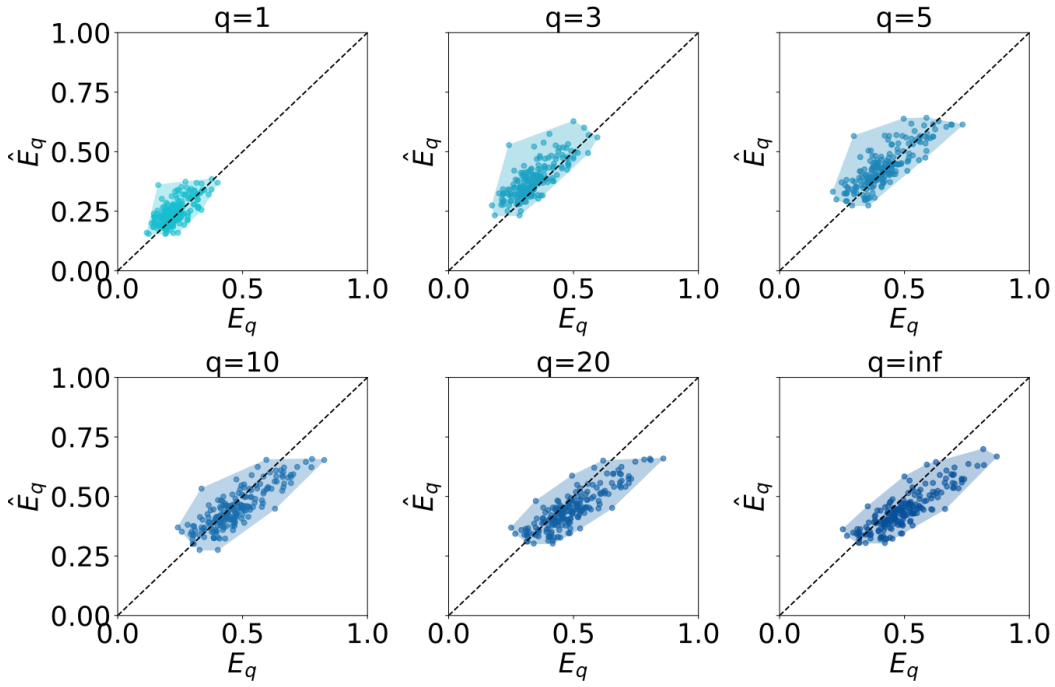
In both theoretical analysis and empirical benchmarks, Infinity Search consistently accelerates nearest-neighbor queries across all tested dissimilarities. On moderate-dimensional datasets such as Fashion-MNIST (Figure 16a) and GIST (Figure 16b), it delivers a clear speedup by, in some cases, sacrificing perfect accuracy. Remarkably, on the high-dimensional Kosarak dataset—with Jaccard dissimilarity—Infinity Search outperforms competing methods by an even wider margin (Figure 16c). This supports the flexibility of the method when less popular dissimilarities are required.

Across all datasets, it offers a favorable speed–accuracy trade-off for the $k = 1$ nearest-neighbor task. For larger neighborhood sizes ($k \in \{5, 10\}$), the increased comparison overhead prevents it from always leading in Rank Order; nevertheless, its performance remains competitive. Note that, since $n = 10{,}000$ in these experiments, modest rank-order errors at extreme speeds still correspond to few misplaced neighbors.

Overall, Infinity Search is a viable alternative when rapid retrieval is required or non-Euclidean or less structured similarity are used.

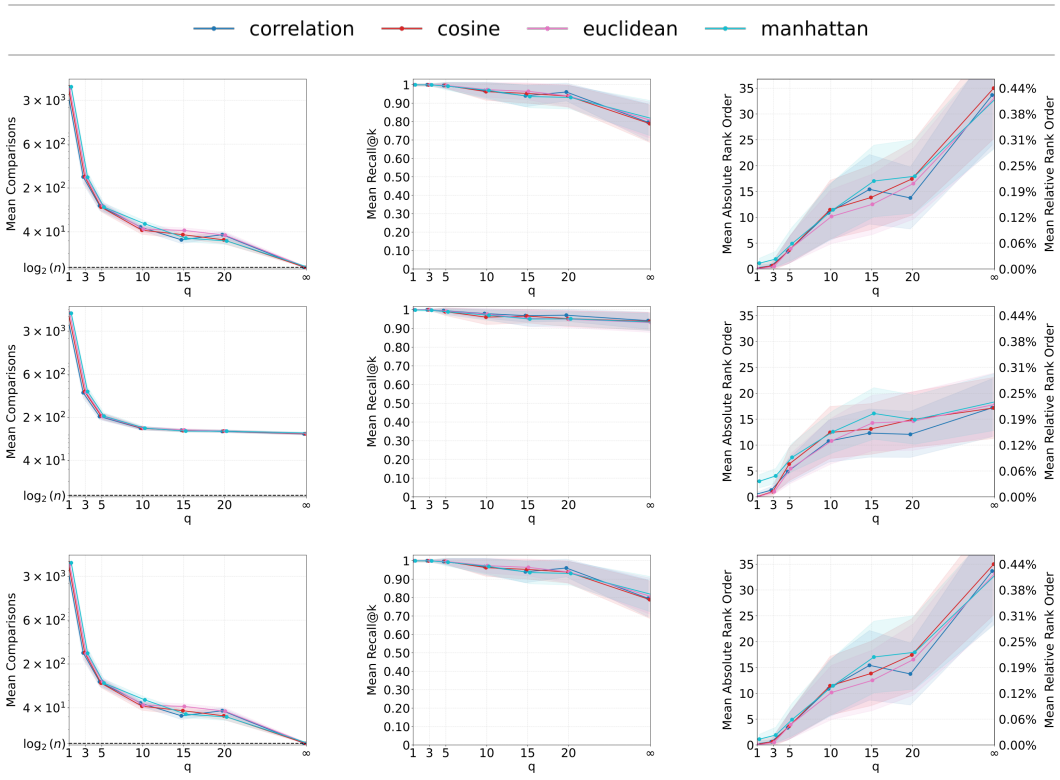


(a) Train



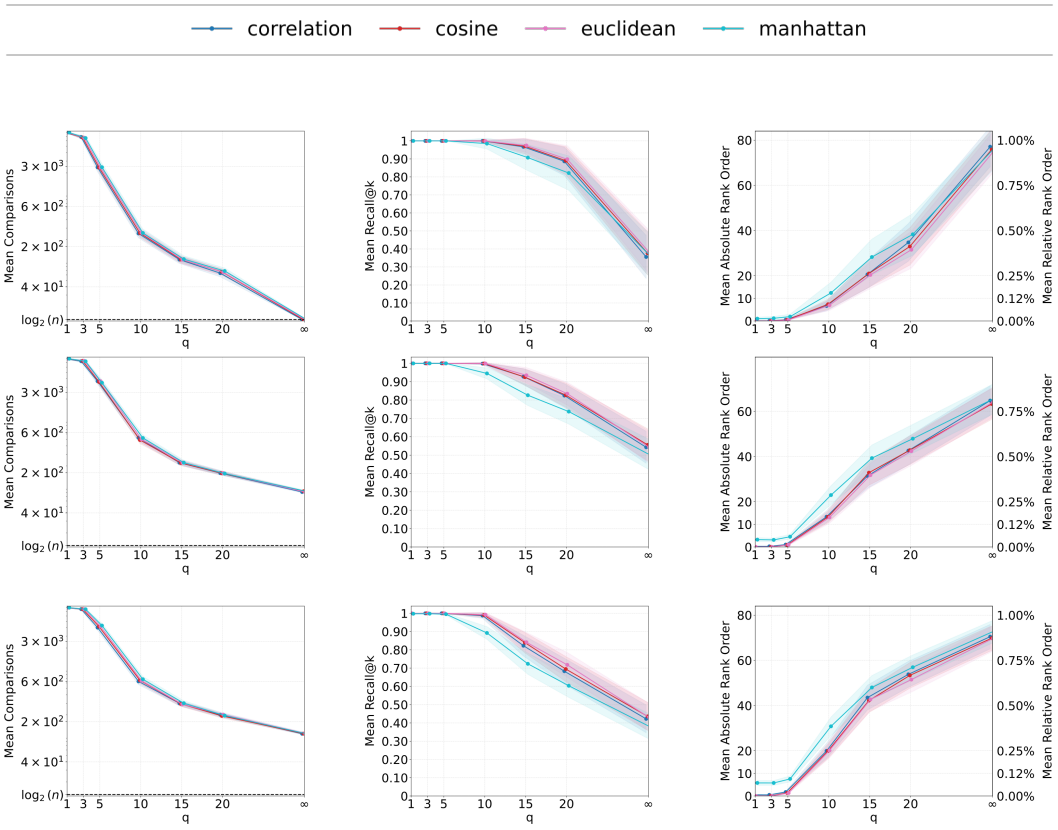
(b) Test

Figure 10: Distribution of the distance to Nearest Neighbor. In the X-axis E_q depicts distance values obtained after projecting when a query point is added. The Y-axis shows the learned approximation of the Canonical projection $\hat{E}_q$. The dashed line represents the perfect match between projected and approximated $x = y$.



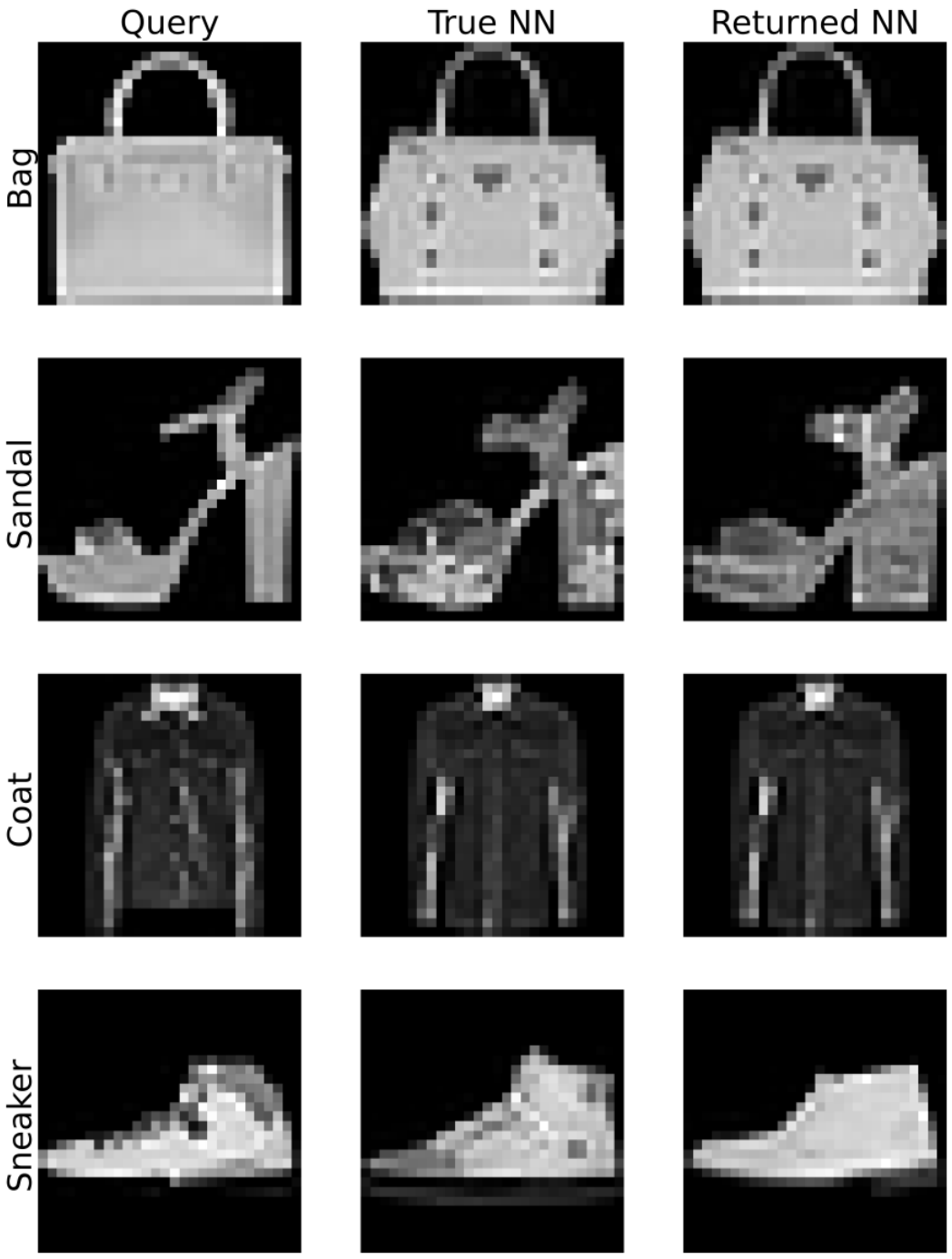
(a) $n = 10{,}000$ points of Fashion-MNIST

Figure 11: Number of comparisons and rank order when searching after approximating the Canonical Projection $\hat{E}_q$, with the learned map $\Phi(x;\theta)$. Solid lines denote the mean and shading the standard deviation computed across queries. The k-nearest neighbors are listed from top to bottom for $k = 1, 5, 10$



(b) $n = 10{,}000$ points of Glove

Figure 11: Number of comparisons and rank order when searching after approximating the Canonical Projection $\hat{E}_q$, with the learned map $\Phi(x;\theta)$. Solid lines denote the mean and shading the standard deviation computed across queries. The k-nearest neighbors are listed from top to bottom for $k = 1, 5, 10$

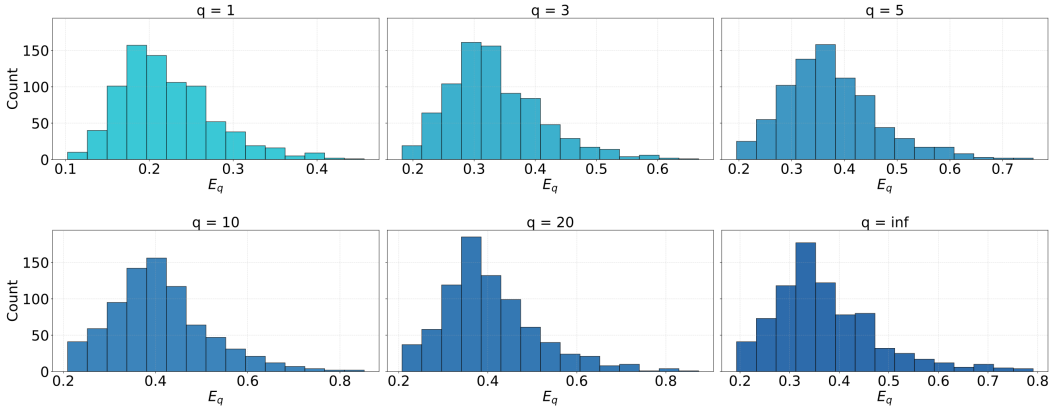


(a) Fashion-MNIST

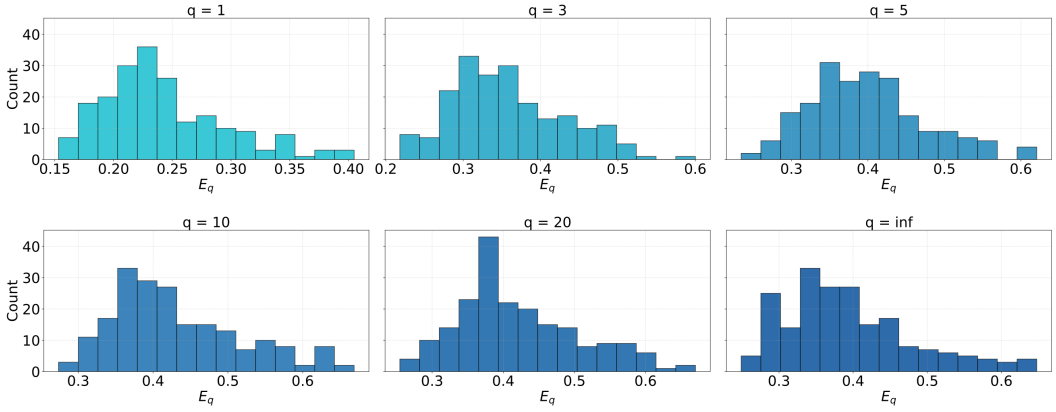
Category	Query	True NN	Returned NN
Animals	dog	dogs	dogs
	lion	wolf	wolf
Colors	blue	pink	purple
	red	pink	purple
Clothing	pants	jeans	jeans
	shirt	shirts	worn
Tools	drill	drilling	drilling
	hammer	throw	flame

(a) GloVe

Figure 13: Retrieval of dataset items with Infinity Search ($q = 5$).

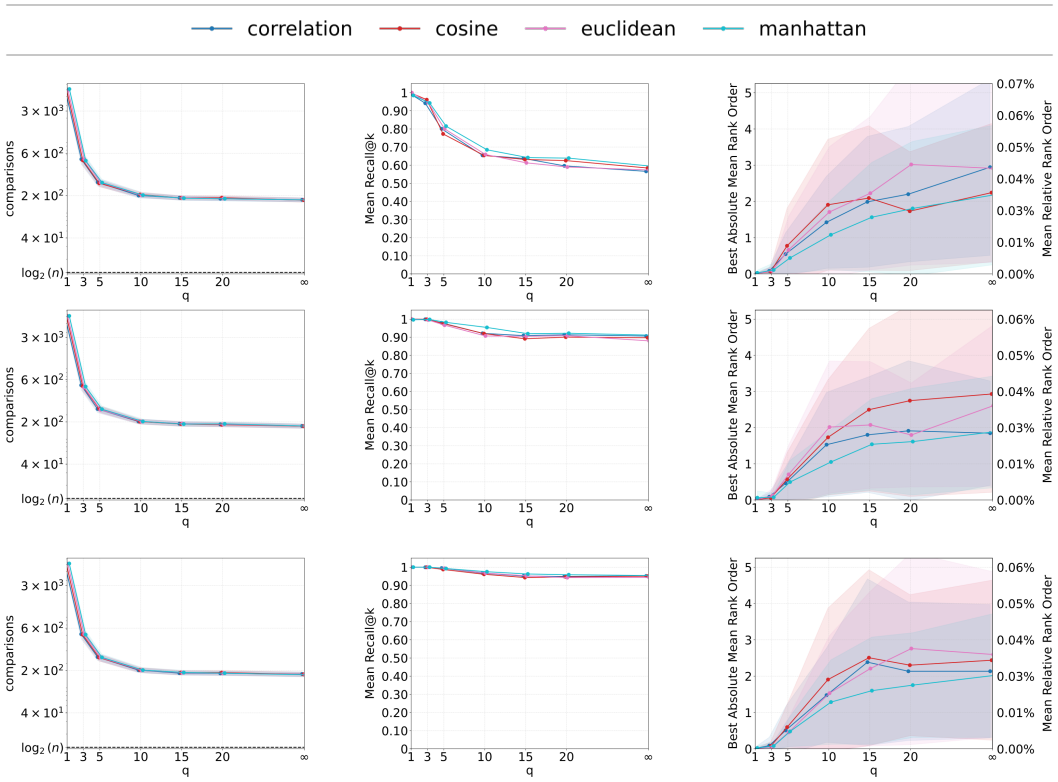


(a) Train



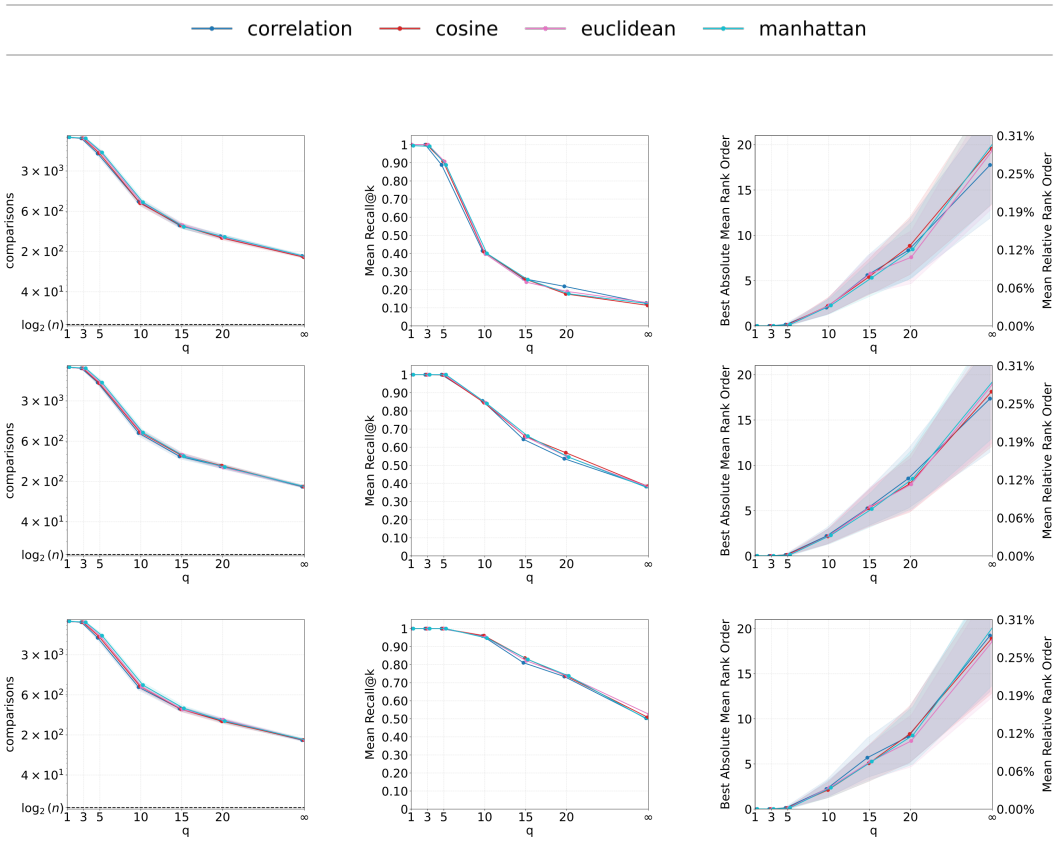
(b) Test

Figure 14: Histogram of the distance to Nearest Neighbor. In the X-axis E_q depicts distance values obtained after projecting when a query point is added. The Y-axis shows number of counts for that distance bin.



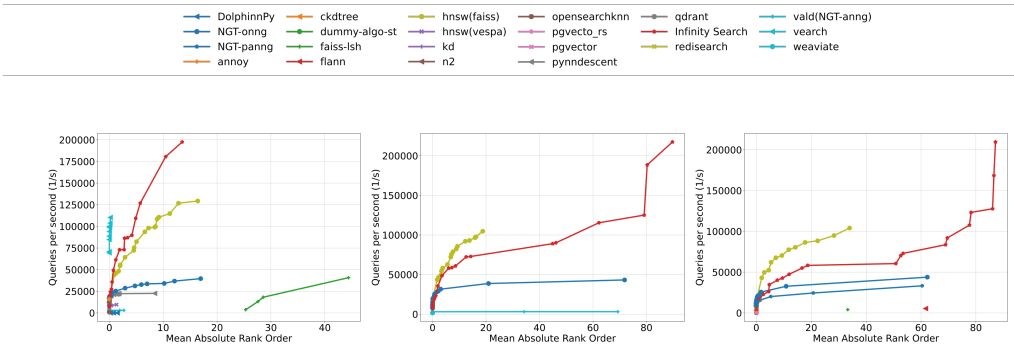
(a) $n = 10{,}000$ points of Fashion-MNIST

Figure 15: Number of comparisons, Recall@k and Rank Order when searching with a two-stage retrieval Infinity Search. Solid lines denote the mean and shading the standard deviation computed across queries. The k-nearest neighbors are listed from top to bottom for $k = 1, 5, 10$

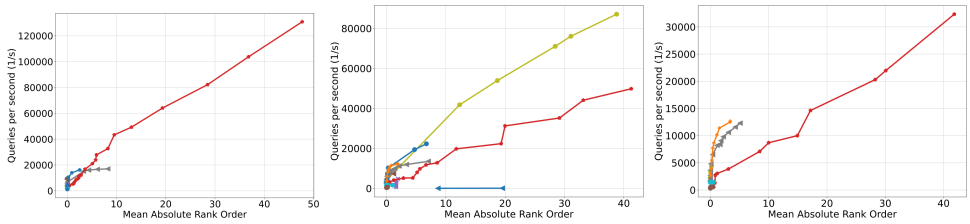


(b) $n = 10{,}000$ points of Glove200

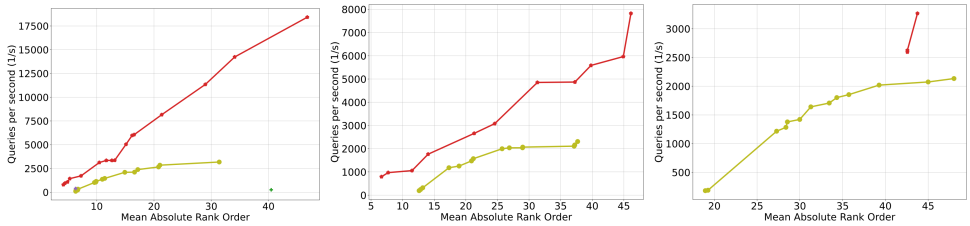
Figure 15: Number of comparisons, Recall@k and Rank Order when searching with a two-stage retrieval Infinity Search. Solid lines denote the mean and shading the standard deviation computed across queries. The k-nearest neighbors are listed from top to bottom for $k = 1, 5, 10$



(a) $n = 10{,}000$ points of Fashion-MNIST with Euclidean dissimilarity



(b) $n = 10{,}000$ points of GIST-960 with Euclidean dissimilarity



(c) $n = 10{,}000$ points of Kosarak-41000 with Jaccard dissimilarity

Figure 16: Search speed vs. accuracy on ANN-Benchmarks for Fashion-MNIST (top row), GIST (middle row) and Kosarak (bottom row). Plots show from left to right, k Nearest Neighbor Search for $k \in \{1, 5, 10\}$.