RHash: Robust Hashing via $\ell_\infty$-norm Distortion

Amirali Aghazadeh$^1$, Andrew Lan$^2$, Anshumali Shrivastava$^1$, Richard Baraniuk$^1$

$^1$Rice University, Houston, TX, USA  
$^2$Princeton University, Princeton, NJ, USA  
{amirali,anshumali,richb}@rice.edu, andrew.lan@princeton.edu

Abstract

Hashing is an important tool in large-scale machine learning. Unfortunately, current data-dependent hashing algorithms are not robust to small perturbations of the data points, which degrades the performance of nearest neighbor (NN) search. The culprit is the minimization of the $\ell_2$-norm, average distortion among pairs of points to find the hash function. Inspired by recent progress in robust optimization, we develop a novel hashing algorithm, dubbed RHash, that instead minimizes the $\ell_\infty$-norm, worst-case distortion among pairs of points. We develop practical and efficient implementations of RHash that couple the alternating direction method of multipliers (ADMM) framework with column generation to scale well to large datasets. A range of experimental evaluations demonstrate the superiority of RHash over ten state-of-the-art binary hashing schemes. In particular, we show that RHash achieves the same retrieval performance as the state-of-the-art algorithms in terms of average precision while using up to 60% fewer bits.

1 Introduction

Hashing, one of the primitive operations in large-scale machine learning systems, seeks a low-dimensional binary embedding of a high-dimensional dataset. Such a binary embedding can increase the computational efficiency of a variety of tasks, including nearest neighbor (NN) search.

The embedding of high-dimensional data into short binary codes naturally distorts the set of top NNs of each data point. To minimize this distortion, a range of hashing schemes have been developed. The simplest, random projection, projects the data onto a lower-dimensional random subspace and then quantizes to binary values. Random projection belongs to the family of probabilistic, locality sensitive hashing (LSH) algorithms that guarantee the preservation of the set of NN points with small distortion [Datar et al., 2004]. The cost, however, is that LSH requires an impractically high number of bits to index real-world, large-scale datasets [Lv et al., 2007]. In response, data-dependent binary hashing algorithms have been developed that require significantly fewer bits; see [Wang et al., 2014] for a survey.

Data-dependent hashing algorithms learn a hash function by minimizing a measure of distortion between certain pairs of data points [Weiss et al., 2009; 2012]. For the task of NN search, it is sufficient to control the pairwise distortion among the NNs of each point plus a few additional points [Kulis and Darrell, 2009]. Given a pair of data points $(x_i, x_j)$ in the ambient space and their corresponding embedded binary codes $(h_i, h_j)$ in the Hamming space, define the pairwise distortion function $\delta_{ij} = L(x_i, x_j; h_i, h_j)$, where $L(\cdot)$ is a chosen loss function. The lion’s share of data-dependent hashing algorithms use an $\ell_2$-norm loss function, which minimizes the average of the square of pairwise distortions $\|\delta\|^2 = \sum_{(i,j) \in \Omega} \delta_{ij}^2$ over a set of selected pairs of points $\Omega$ [Jiang and Li, 2015; Kong and Li, 2012; Kulis and Darrell, 2009; Liu et al., 2011; Weiss et al., 2009; 2012].

Unfortunately, the loss functions employed by the hashing community to date, including the $\ell_2$-norm loss, are not robust to perturbations of the data points. Indeed, the $\ell_2$-norm average distortion is well-known to be sensitive to small disturbances and noise [Chatterjee and Hadi, 2009]. In this paper, we argue that alternative norms, in particular the $\ell_\infty$-norm (which measures worst-case distortion) [Du and Pardalos, 2013; Hartley and Schaffalitzky, 2004; Xu et al., 2009], are more robust to small perturbations and thus more appropriate for designing data-dependent hash functions.

To elucidate the strong effect of the loss function on the robustness of a hashing function, we compare the hash functions learned using the $\ell_2$- and $\ell_\infty$-norm losses on a subset of the MNIST dataset of handwritten digit images [LeCun and Cortes, 1998]. For visualization purposes, we project the data points onto the first two principal components and compute the optimal 1-dimensional binary embeddings generated by minimizing the $\ell_2$-norm distortion and the $\ell_\infty$-norm distortion $\|\delta\|_\infty = \max_{(i,j) \in \Omega} |\delta_{ij}|$ with $|\Omega| = 50$. Figure 1(a) shows the $\ell_2$- and $\ell_\infty$-optimal embeddings that we learned from a grid search. Figure 1(b) summarizes an experiment where we add a small amount of white Gaussian noise to each data point before computing the optimal embeddings. We see that the $\ell_2$-optimal embedding is increasingly less robust than the $\ell_\infty$-optimal embedding as the amount of noise grows.

The robustness of the $\ell_\infty$-optimal embedding translates directly to improved NN preservation. Continuing the MNIST example from above but with no added noise, Figure 1(c) il-
 Ambient NNs: ℓ∞ is significantly more robust to perturbations in the data points. (c) Comparison of the top-5 NNs of the query point q embedding divided by the mean square error of the orientation of the digits “3”, “5”, and “9” and the digit “8” and thus optimal embedding does not preserve some of the distances between pairs. In contrast, the ℓ∞-norm (worst-case distortion) optimal embedding preserves these harder distances and thus preserves the set of original NNs of q with minimal distortion.

Inspired by the robustness of the ℓ∞-norm distortion [Du and Pardalos, 2013; Hartley and Schaffalitzky, 2004; Xu et al., 2009], we propose Robust Hashing (RHash), the first hashing algorithm that learns robust binary hash codes by minimizing the ℓ∞-norm of the pairwise distances vector \( \| \delta \|_\infty = \max_{i,j} d_H(x_i, x_j) \), where \( d_H \) denote the Euclidean and Hamming distances, respectively. The robustness of ℓ∞-norm distortion stems from its dependence on only the pairwise distortions that are hardest to preserve. As long as the hardest pairwise distortions are not affected, small perturbations do not alter the outcome of the embedding. In other words, in the NN retrieval task, where we only care about the distortions of NN pairs, the hardest pair to preserve is the deal-breaker. We emphasize that the worst-case distorted NN pair (under an embedding) is the hardest NN pair to be preserved and not the farthest pair nor a pair of outliers.

### 1.1 Contributions

We make four distinct contributions in this paper. First, conceptually, we advocate minimizing the more robust worst-case ℓ∞-norm distortion measure rather than the more prosaic ℓ2-norm distortion.

Second, algorithmically, since ℓ∞-norm minimization problems are computationally challenging, especially for large datasets, we develop two accelerated and scalable algorithms to find the optimal robust embedding. The first, Robust Hashing (RHash), is based on the alternating direction method of multipliers (ADMM) framework [Boyd et al., 2011]. The second, RHash-CG, is based on an accelerated greedy extension of the RHash algorithm using the concept of column generation [Dantzig and Wolfe, 1960]. RHash-CG can rapidly learn hashing functions from large-scale data sets that require the preservation of billions of pairwise distances.

Third, theoretically, since current data-dependent hashing algorithms do not offer any probabilistic guarantees in terms of preserving NNs, we develop new theory to prove that, under natural assumptions regarding the data distribution and with a notion of hardness of NN search (dubbed the k-order gap), RHash preserves the top NNs with high probability.

Fourth, experimentally, we demonstrate the superior performance of RHash over ten state-of-the-art binary hashing algorithms using an exhaustive set of experimental evaluations involving six diverse datasets and three different performance metrics (distance preservation, Hamming distance ranking, and Kendall’s \( \tau \) ranking performance). In particular, we show that RHash achieves the same retrieval performance as state-of-the-art algorithms in terms of average precision while using up to 60% fewer bits. Our experiments clearly show the advantages of the ℓ∞-norm formulation compared to the more classical ℓ2-norm formulation that underlies many hashing algorithms [Kong and Li, 2012; Kulis and Darrell, 2009]. Our formulation also outperforms recently developed techniques that assume more structure in their hash functions [Heo et al., 2012; Yu et al., 2014].

We note that the ℓ∞-norm has been exploited in the hashing literature before; however, these works are in sharp contrast to the RHash objective and algorithm. For example, antisparsifying coding [Jégou et al., 2012] uses the ℓ∞-norm to

![Figure 1: Comparison of the robustness and nearest neighbor (NN) preservation performance of embeddings based on minimizing the ℓ2-norm (average distortion) vs. the ℓ∞-norm (worst-case distortion) on a subset of the MNIST handwritten digit dataset projected onto its first two principal components.](image)

(a) Optimal embeddings for both distortion measures computed using a grid search over the orientation of the line representing the embedding. (b) Robustness of the embedding orientations to the addition of a small amount of white Gaussian noise to each data point. This plot of the mean square error of the orientation of the ℓ∞-optimal embedding divided by the mean square error of the orientation of the ℓ2-optimal embedding indicates that the latter embedding is significantly more robust to perturbations in the data points. (c) Comparison of the top-5 NNs of the query point q obtained in the ambient space using the ℓ∞- and ℓ2-optimal embeddings (no added noise). (d) Projections of the data points onto the ℓ2- and ℓ∞-optimal embeddings (no added noise).
directly learn binary codes and bypass quantization. More recently, the \( l_\infty \)-norm has been used to train a deep supervised hashing algorithm [Wang et al., 2016]. In contrast, we impose the \( l_\infty \)-norm on the NNs of the dataset in order to learn a robust unsupervised hashing algorithm, which is completely different from the above approaches.

2 Robust hashing via \( l_\infty \)-norm distortion

In this section, we formulate the RHash algorithm. We pick a simple formulation for data dependent binary hash function that embeds a data point \( x \in \mathbb{R}^N \) into the low-dimensional Hamming space \( \mathcal{H} = \{0, 1\}^M \) by first multiplying it by an embedding matrix \( W \in \mathbb{R}^{M \times N} \) and then quantizing the entries of the product \( Wx \) to binary values: \( h(Wx) = (1 + \text{sgn}(Wx))/2 \). The function \( \text{sgn}(\cdot) \) operates element-wise on the entries of \( Wx \), transforming the real-valued vector \( Wx \) into a set of binary codes depending on the sign of the entries in \( Wx \).

2.1 Problem formulation

Consider the design of a binary embedding \( f \) that maps \( Q \) high-dimensional data vectors \( X = \{x_1, x_2, \ldots, x_Q\} \) in the ambient space \( \mathbb{R}^N \) into low-dimensional binary codes \( \mathcal{H} = \{h_1, h_2, \ldots, h_Q\} \) in the Hamming space with \( h_i \in \{0, 1\}^M \), where \( h_i = f(x_i), i = 1, \ldots, Q \), and \( M \ll N \). Define the distortion of the embedding by

\[
\delta = \inf_{h \in \mathcal{H}} \sup_{(i,j) \in \Omega} |\lambda d_H(h_i, h_j) - d(x_i, x_j)|,
\]

where \( d(x_i, x_j) \) denotes the Euclidean distance between the data points \( x_i, x_j \), \( d_H(h_i, h_j) \) denotes the Hamming distance between the binary codes \( h_i \) and \( h_j \), \( \Omega \) indexes the set containing a selected pair of points (more details in section 3), and \( \lambda \) is a positive scaling variable. The distortion \( \delta \) measures the worst-case deviation from perfect distance preservation [Plan and Vershynin, 2014]. Define the secant set \( S(X) \) as \( S(X) = \{x_i - x_j : (i,j) \in \Omega\} \), i.e., the set of pairwise difference vectors indexed by \( \Omega \). Let \( |S(X)| = |\Omega| \) denote the size of the secant set [Hegde et al., 2015].

RHash minimizes the worst-case distortion parameter \( \delta \) via the following optimization:

\[
\text{minimize } \sup_{(i,j) \in \Omega} |\lambda d_H(h_i, h_j) - d(x_i, x_j)|.
\]

Since the squared \( \ell_2 \)-distance between a pair of binary codes is equivalent to their Hamming distance up to a scaling factor that can be absorbed into \( \lambda \), we can rewrite the above optimization as

\[
(P') \quad \text{minimize } \|\lambda v'(W) - c\|_{\infty},
\]

where \( v' \in \mathbb{R}^{[Q]} \) is a vector containing the pairwise \( \ell_2 \)-distances between the embedded data vectors \( |h(Wx_i) - h(Wx_j)|_2^2 \), and \( c \) is a vector containing the pairwise \( \ell_2 \)-distances between the original data vectors. Intuitively, the \( l_\infty \)-norm objective optimizes the worst-case distortion among NN’s distances.
Optimize over $W$ via $W^{(\ell+1)} \leftarrow \text{arg min}_W \frac{1}{2} \sum_{(i,j) \in S} (u_{ij}^{(\ell)} - \lambda^{(\ell)} \| \mathbf{x}_i^{(\ell)} - \mathbf{x}_j^{(\ell)} \|^2 + \lambda^{(\ell)} \| u_{ij}^{(\ell)} \|^2)^2$, where $\lambda^{(\ell)}$ denotes the value of $\lambda$ in the $\ell$th iteration. We also use $u_{ij}^{(\ell)}$ and $y_{ij}^{(\ell)}$ to denote the entries in $u^{(\ell)}$ and $y^{(\ell)}$ that correspond to the pair $(\mathbf{x}_i, \mathbf{x}_j)$ in the dataset $X$. We show in our experiments below that using the accelerated first-order gradient descent algorithm [Nesterov, 2007] to solve this subproblem results in good convergence performance (see Figure 2).

Optimize over $u$ while holding the other variables fixed; this corresponds to solving the proximal problem of the $l_\infty$-norm $u^{(\ell+1)} \leftarrow \text{arg min}_u \frac{1}{2} \| u^{(\ell)} - \lambda^{(\ell)} y^{(\ell)} + c \|^2_2$. We use the low-cost algorithm described in [Studer et al., 2014] to perform the proximal operator update.

Optimize over $\lambda$ while holding the other variables fixed; this corresponds to a positive least squares problem, where $\lambda$ is updated as $\lambda^{(\ell+1)} \leftarrow \text{arg min}_{\lambda > 0} \frac{1}{2} \| u^{(\ell+1)} - \lambda y^{(\ell+1)} + c \|^2_2$. We perform this update using the non-negative least squares algorithm.

Update $y$ via $y^{(\ell+1)} \leftarrow y^{(\ell)} + \eta (u^{(\ell+1)} - \lambda^{(\ell+1)} y^{(\ell+1)} + c)$, where the parameter $\eta$ controls the dual update step size.

2.3 Accelerated RHash for large-scale datasets

The ADMM-based RHash algorithm is efficient for small-scale datasets (e.g., for secant sets of size $|S| < 5000$ or so). However, the memory requirement of RHash is quadratic in $|X|$, which could be problematic for applications involving large numbers of data points and secants. In response, we develop an algorithm that approximately solves (P) while scaling very well to large-scale problems. The key idea comes from classical results in optimization theory related to column generation (CG) [Dantzig and Wolfe, 1960; Hegde et al., 2015].

The optimization (S) is an $l_\infty$-norm minimization problem with an equality constraint on each secant. The Karush-Kuhn-Tucker (KKT) condition for this problem states that, if strong duality holds, then the optimal solution is entirely specified by a (typically very small) portion of the constraints. Intuitively, the secants corresponding to these constraints are the pairwise distances of $k$NNs that are harder to preserve in the low-dimensional Hamming space. We call this set of secants the active set. In order to solve (P), it suffices to find the active secants and solve the RHash optimization with that much smaller number of active constraints. To leverage the idea of the active set, we iteratively run RHash on a small subset of all of the secants that violate the distance preservation, as detailed below:

- Solve (P) with a small random subset $S_0$ of all of the secants $S(X)$ using RHash to obtain the initial estimates $\hat{W}$, $\hat{\delta}$, and $\hat{\lambda}$ of the parameters. Identify the active set $S_{\text{aug}}$. Fix $\lambda = \hat{\lambda}$ for the rest of the algorithm.
- Randomly select a new subset $S_{\text{aug}} \subset S$ of secants that violate the distance preservation condition using the current estimates of $\hat{W}$, $\hat{\delta}$, and $\hat{\lambda}$. Then, form the augmented secant set $S_{\text{aug}} = S_{\text{aug}} \cup S_{\text{active}}$.
- Solve (P) with the secants in the set $S_{\text{aug}}$ using the RHash algorithm.

We dub this approach RHash-CG. RHash-CG iterates over the above steps until no new violating secants are added to the active set. Since the algorithm searches over all secants for violations in each iteration, RHash-CG ensures that all of the constraints are satisfied when it terminates.

A key benefit of RHash-CG is that only the set of active secants (and not all of the secants) needs to be stored in memory. This leads to significant improvements in memory complexity over competing algorithms, since the size of the secant set grows quadratically with the number of data points and can quickly exceed the system memory capacity in large-scale applications. The $l_\infty$-norm distortion function enables RHash to leverage the simple CG approach in order to scale to larger datasets, another advantage of $l_\infty$-norm for large-scale hashing.

2.4 RHash and NN preservation guarantee

Now that we have fully described the RHash algorithm, we develop a theory that bounds the probability of preserving all $k$NNs as a function of the $l_\infty$-norm distortion measure $\delta$ minimized by RHash.

Inspired by the definition of relative contrast [He et al., 2012], we define a more generalized measure of data separability to preserve $k$NN that we call the $k$-order gap $\Delta_k := d(\mathbf{x}_0, \mathbf{x}_{k+1}) - d(\mathbf{x}_0, \mathbf{x}_k)$, where $\mathbf{x}_0$ is a query point and $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ are its $k^\text{th}$ and $k + 1^\text{st}$ $k$NNs, respectively. We formally show that, if the data is sufficiently separable ($\Delta_k$ is large), then RHash preserves all $k$NNs with high probability under assumptions that are typical in the NN search literature [He et al., 2012] (see the Appendix for a proof and discussion).

Theorem 1. Assume that all of the data points are independently generated from a mixture of Gaussians distribution, i.e., $\mathbf{x}_i \sim \sum_{p=1}^{P} \pi_p \mathcal{N}(\mu_p, \Sigma_p)$. Let $\mathbf{x}_0 \in \mathbb{R}^D$ denote a query data point in the ambient space, and let the other data points $\mathbf{x}_i$ be ordered such that $d(\mathbf{x}_0, \mathbf{x}_1) < d(\mathbf{x}_0, \mathbf{x}_2) < \ldots <
\[ d(x_0, x_0) \]. Let \( \delta \) denote the \( \ell_{\infty} \)-norm distortion parameter in RHash, and let \( c \) denote a positive constant. Then, if 
\[
\mathbb{E}_x[\Delta_k] \geq 2\delta + \sqrt{\frac{1}{2} \log \frac{Q_k}{\epsilon}}, 
\]
then RHash preserves all \( k \) NNs of the query point \( x_0 \) with probability at least \( 1 - \epsilon \).

## 3 Experiments

In this section, we validate the RHash and RHash-CG algorithms via experiments using a range of synthetic and real-world datasets, including three small-scale, three medium-scale, and one large-scale datasets. We use three metrics to compare RHash against ten state-of-the-art binary hashing algorithms, including binary reconstructive embedding (BRE) [Kulis and Darrell, 2009], spectral hashing (SH) [Weiss et al., 2009], anchor graph hashing (AGH) [Liu et al., 2011], multi-dimensional spectral hashing (MDSH) [Weiss et al., 2012], scalable graph hashing (SGH) [Jiang and Li, 2015], PCA hashing (PCAH), isotropic hashing (IsoHash) [Kong and Li, 2012], spherical Hamming distance hashing (SHD) [Heo et al., 2012], circulant binary embedding (CBE) [Yu et al., 2014], and locality-sensitive hashing (LSH) [Indyk and Motwani, 1998].

### 3.1 Performance metrics and datasets

We compare the algorithms using the following three metrics:

- **Maximum distortion** \( \delta = \inf_{\lambda > 0} \| x \hat{v} - c \|_{\infty} \), where the vector \( \hat{v} \) contains the pairwise Hamming distances between the learned binary codes. This metric quantifies the distance preservation after projecting the training data in the ambient space into binary codes. We also define the maximum distortion for unseen test data \( \delta_{\text{test}} \), which measures the distance preservation on a hold-out test dataset using the hash function learned from the training dataset.

- **Mean average precision (MAP)** for NN preservation in the Hamming space. MAP is computed by first finding the set of \( k \) NNs for each query point in a hold-out test data set in the ambient space \( L^k \) and the corresponding set \( L_H^k \) in the Hamming space and then calculating the average precision \( \text{AP} = |L^k \cap L_H^k|/k \). We then report MAP by calculating the mean value of AP across all data points.

- **Kendall’s \( \tau \) ranking correlation coefficient**. We first rank the set of \( k \) NNs for each data point by increasing distance in the ambient space as \( T(L^k) \) and in the Hamming space as \( T(L_H^k) \). The Kendall \( \tau \) correlation coefficient is a scalar \( \tau \in [-1, 1] \) that measures the similarity between the two ranked sets \( T(L^k) \) and \( T(L_H^k) \). The value of \( \tau \) increases as the similarity between the two rankings increases and reaches the maximum value of \( \tau = 1 \) when they are identical. We report the average value of \( \tau \) achieved across all data points in the training dataset.

To compare the algorithms, we use the following standard datasets from computer vision: Random consists of independently drawn random vectors in \( \mathbb{R}^{100} \) from a multivariate Gaussian distribution with zero mean and identity covariance matrix. Translating squares is a synthetic dataset consisting of \( 10 \times 10 \) images that are translations of a \( 3 \times 3 \) white square.
on a black background [Hegde et al., 2015]. MNIST is a collection of 60,000 28 × 28 greyscale images of handwritten digits [LeCun and Cortes, 1998]. Photo-Tourism is a corpus of approximately 300,000 image patches represented using scale-invariant feature transform (SIFT) features [Lowe, 2004] in $\mathbb{R}^{128}$ [Snively et al., 2006]. LabelMe is a collection of over 20,000 images represented using GIST descriptors in $\mathbb{R}^{512}$ [Torralba et al., 2008]. Peekaboom is a collection of 60,000 images represented using GIST descriptors in $\mathbb{R}^{512}$ [Torralba et al., 2008]. Following the experimental approaches of the hashing literature [Kulis and Darrell, 2009; Norouzi and Fleet, 2011], we pre-process the data by subtracting the mean and then normalizing all points to lie on the unit sphere.

### 3.2 Small- and medium-scale experiments

We start by evaluating the performance of RHash and RHash-CG using a subset of the first three datasets. Small-scale datasets enable comparisons between the asymptotic behaviors of the algorithms in preserving distances in a regime where number of bits is high enough (compared to the total number of NNs) to preserve a large portion of NN distances.

**Experimental setup.** We randomly select $Q = 100$ data points from the Random, Translating squares, and MNIST datasets. We then apply the RHash, RHash-CG, and all of the baseline algorithms on each dataset for different target binary code word lengths $M$ from 1 to 70 bits. We set the RHash and RHash-CG algorithm parameters to the common choice of $\rho = 1$ and $\eta = 1.6$. To generate a hash function of length $M$ for LSH, we draw $M$ random vectors from a Gaussian distribution with zero mean and an identity covariance matrix. We use the same random vectors to initialize RHash and the other baseline algorithms. In the NN preservation experiments, to demonstrate the direct advantage of minimizing $\ell_\infty$-norm over $\ell_2$-norm, we followed the exact procedure described in BRE [Kulis and Darrell, 2009] to select the training secants, i.e., we apply the RHash algorithm on the lowest 5% of the pairwise distances (which are set to zero as in BRE) combined with the highest 2% of the pairwise distances.

We follow the continuation approach [Wen et al., 2010] to set the value of $\alpha$. We start with a small value of $\alpha$ (e.g., $\alpha = 1$), in order to avoid becoming stuck in bad local minima, and then gradually increase $\alpha$ as the algorithm proceeds. As the algorithm moves closer to convergence and has obtained a reasonably good estimate of the parameters $W$ and $\lambda$, we set $\alpha = 10$, which enforces a good approximation of the sign function (see Lem. 2 in the Appendix for an analysis of the accuracy of this approximation).

**Results.** The plots in the top row of Figure 3 illustrate the value of the distortion parameter $\delta$ as a function of the number of projections (bits) $M$. The performance of RHash and RHash-CG closely follow each other, indicating that RHash-CG is a good approximation to RHash. Both RHash and RHash-CG outperform the other baseline algorithms in terms of the distortion parameter $\delta$. Among these baselines, LSH has the lowest distance preservation performance, which should be expected, since random projections are oblivious to the intrinsic geometry of the training dataset.

To achieve $\delta = 1$, RHash(-CG) requires 60% fewer bits $M$ than CBE and BRE. RHash(-CG) also achieves better distance preservation performance asymptotically, i.e., up to $\delta \approx 0.5$, given a sufficient number of bits ($M \geq 70$), while for most of the other algorithms the performance plateaus after $\delta = 1$. RHash’s superior distance preservation performance extends well to unseen data. RHash achieves the lowest distortion on a test dataset $\delta_{test}$ compared to the other hashing algorithms (Figure 4).

The plots in the middle and bottom row of Figure 3 show the average precision for retrieving training data and the Kendall’s $\tau$ correlation coefficient respectively, as a function of the number of bits $M$. We see that RHash preserves a higher percentage of NNs compared to other baseline algorithms as $M$ increases, with better average ranking among $k = 10$ NNs. We see that RHash preserves a higher per-
Table 1: Comparison of RHash-CG against several baseline binary hashing algorithms on the large-scale MNIST-rotation dataset with over 5 billion secants $|S(X)|$. We tabulate the Hamming ranking performance in terms of mean average precision MAP for different sizes of the dataset. All training times are in seconds.

<table>
<thead>
<tr>
<th>Training size $Q$</th>
<th>1K</th>
<th>10K</th>
<th>100K</th>
<th>240K</th>
<th>240K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secant size $</td>
<td>S(X)</td>
<td>$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100K</td>
<td>54.79 (±0.15)</td>
<td>54.69 (±0.18)</td>
<td>54.93 (±0.23)</td>
<td>55.52 (±0.11)</td>
<td>541.43</td>
</tr>
<tr>
<td>1M</td>
<td>54.79 (±0.15)</td>
<td>54.69 (±0.18)</td>
<td>54.93 (±0.23)</td>
<td>55.52 (±0.11)</td>
<td>541.43</td>
</tr>
<tr>
<td>1B</td>
<td>54.79 (±0.15)</td>
<td>54.69 (±0.18)</td>
<td>54.93 (±0.23)</td>
<td>55.52 (±0.11)</td>
<td>541.43</td>
</tr>
<tr>
<td>5B</td>
<td>54.79 (±0.15)</td>
<td>54.69 (±0.18)</td>
<td>54.93 (±0.23)</td>
<td>55.52 (±0.11)</td>
<td>541.43</td>
</tr>
<tr>
<td>5B</td>
<td>54.79 (±0.15)</td>
<td>54.69 (±0.18)</td>
<td>54.93 (±0.23)</td>
<td>55.52 (±0.11)</td>
<td>541.43</td>
</tr>
</tbody>
</table>

percentage of NNs and achieves a better average ranking performance compared to other baseline algorithms.

Next, we showcase the performance of RHash-CG on the three medium-scale, real-world datasets used in [Kulis and Darrell, 2009; Norouzi and Fleet, 2011], including Phototourism, LabelMe, and Peekaboom for the task of data retrieval. From each dataset we randomly select $Q = 1000$ training points, following the setup in BRE [Kulis and Darrell, 2009], and use them to train RHash-CG and the other baseline algorithms. We then randomly select a separate set of $Q = 1000$ data points and use it to test the performance of RHash-CG and other baseline algorithms in terms MAP with $k = 50$. Figure 5 illustrates the performance of RHash-CG on these datasets. RHash-CG outperforms all of the baseline algorithms by large margins in Hamming ranking performance in terms of MAP with top-50 NNs.

### 3.3 Large-scale experiments

To demonstrate how RHash-CG scales to large-scale datasets, we use the full MNIST dataset with 60,000 training images and augment it with three rotated versions of each image (rotations of 90°, 180°, and 270°) to create a larger dataset with $Q = 240,000$ data points. Next, we construct 4 training sets with 1,000, 10,000, 100,000, and 240,000 images out of this large set. We train all algorithms with $M = 30$ bits and compare their performance on a test set of 10,000 images. The results for all algorithms are given in Table 1; we tabulate their performance in terms of MAP for the top-500 NNs.\(^1\)

The performance of RHash-CG is significantly better than the baseline algorithms and, moreover, improves as the size of the training set grows.

### 4 Discussion

We have demonstrated that our worst-case, $\ell_\infty$-norm-based, Robust Hashing (RHash) algorithm is superior to a wide range of algorithms based on the more traditional average-case, $\ell_2$-norm. Despite its non-convexity and non-smoothness, RHash admits an efficient optimization algorithm that converges to a high-performing local minimum. Moreover, RHash-CG, the accelerated version of RHash, provides significant memory advantages over existing algorithms; the memory requirement of RHash-CG is linear in the number of active secants rather than the total number of secants, a significant advantage over the existing reconstructive hashing methods which minimize the reconstruction error between the original distances in ambient space and the Hamming distances in embedded space. Our experiments with six datasets, three metrics, and ten algorithms have demonstrated the superiority of RHash over other state-of-the-art data-dependent hashing algorithms. Thus, our results provide a strong motivation for exploring $\ell_\infty$-norm and other worst-case formulations for robust binary hashing.

### 5 Appendix

Here, we prove Thm. 1 on the performance of RHash for $k$ nearest neighbor ($k$NN) preservation.

#### 5.1 Sigmoid Approximation

We start with a result that bounds the error of the sigmoid approximation in RHash.

**Lemma 2.** Let $x$ be a Gaussian random variable distributed as $x \sim \mathcal{N}(\mu, \sigma^2)$. Define the distortion of the sigmoid approximation at $x$ by $|h(x) - \sigma_\alpha(x)|$. Then, the expected distortion is bounded by

$$E_x[|h(x) - \sigma_\alpha(x)|] \leq \frac{1}{\sigma \sqrt{2\pi}} + 2e^{-\left(\sqrt{\alpha} + c/\alpha\sigma^2\right)},$$

where $c$ is a positive constant. As $\alpha \to \infty$, the expected distortion $\to 0$.

**Proof.** It is easy to see that the maximum distortion $|h(x) - \sigma_\alpha(x)|$ occurs at $x = 0$. Therefore, among different values of $\mu$, $\mu = 0$ gives the largest distortion, since the density of $x$ peaks at $x = 0$. Therefore, we bound the distortion at setting $\mu = 0$, which is an upper bound of the distortion when $\mu \neq 0$. By definition (1) above, $h(x)$ can be written as

$$h(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

\(^1\)BRE fails to execute on a standard desktop PC with 12 GB of RAM due to the size of the secant set.
When $x \sim \mathcal{N}(0, \sigma^2)$ and we set $x_0 = \frac{1}{\sqrt{\alpha}}$, we can write

$$\mathbb{E}_x[|h(x) - \sigma_\alpha(x)|] = \int_{-\infty}^{\infty} |h(x) - \sigma_\alpha(x)| \mathcal{N}(x; 0, \sigma^2)dx$$

$$= 2 \int_{0}^{\infty} (h(x) - \sigma_\alpha(x)) \mathcal{N}(x; 0, \sigma^2)dx$$

$$= 2 \int_{0}^{x_0} (h(x) - \sigma_\alpha(x)) \mathcal{N}(x; 0, \sigma^2)dx$$

$$+ 2 \int_{x_0}^{\infty} (h(x) - \sigma_\alpha(x)) \mathcal{N}(x; 0, \sigma^2)dx$$

$$\leq 2 \int_{0}^{x_0} \frac{1}{2} \mathcal{N}(x; 0, \sigma^2)dx + 2 \int_{x_0}^{\infty} \frac{1}{1 + e^{ax_0}} \mathcal{N}(x; 0, \sigma^2)dx$$

$$\leq \frac{x_0}{\sqrt{2\pi\sigma}} + 2 e^{-c\sigma^2/\sigma^2}$$

$$\leq \frac{1}{\sigma \sqrt{2\pi}} + 2 e^{-\sqrt{\pi}c/\sigma^2},$$

where $c$ is a positive constant. In (6), we used the fact that $\sigma_\alpha(x)$ and $h(x)$ are symmetric with respect to the point $(0, \frac{1}{2})$. (7) follows from the properties of the sigmoid function; (8) follows from the Gaussian concentration inequality [Ledoux and Talagrand, 1991]; and (9) follows from the inequality $1/(1 + e^{ax_0}) \leq e^{-\alpha x_0}$. The fact that $\mathbb{E}_x[|h(x) - \sigma_\alpha(x)|] \rightarrow 0$ as $\alpha \rightarrow \infty$ is obvious from the bound above.

**Proof of Thm. 1**

In order to prove Thm. 1, we need the following Lemma:

**Lemma 3.** Let $x_0, \ldots, x_N$ and $\Delta_k$ be defined as in Thm. 1. Then, there exists a constant $c$ such that $P(\Delta_k - \mathbb{E}_x[\Delta_k] < t) \leq e^{-ct^2}$ for $t > 0$.

**Proof.** Since the data points $x_0, x_k, x_{k+1}$ are independently generated from a finite mixture of Gaussian distributions, the random variable of their concatenation $y = [x_0, x_k, x_{k+1}]^T \in \mathbb{R}^{3N}$ is sub-Gaussian [Wainwright, 2009]. Then, we have

$$\Delta_k(y) = ||x_0 - x_{k+1}||_2 - ||x_0 - x_k||_2$$

$$= \frac{1}{2} ||\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{pmatrix} y||_2 - ||\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix} y||_2$$

$$\leq \frac{1}{2} ||\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} y||_2$$

$$\leq 2||y||_2,$$

where we have used the triangular inequality in the second to last step, the Rayleigh-Ritz theorem, and the fact that the maximum singular value of the diagonal matrix above is 2. This means that $\Delta_k(y)$ is a Lipschitz function of $y$. Thus, by Talagrand’s inequality [Ledoux and Talagrand, 1991], we have that $P(\Delta_k - \mathbb{E}_x[\Delta_k] < -t) \leq e^{-ct^2}$ for some positive constant $c$ and $t > 0$, since $y$ is sub-Gaussian.

We are now ready to prove Thm. 1.

**Proof.** Let $E$ denote the event that the set of top $k$ NNs is not preserved in the Hamming space. Then, we have $E = \cup_{e_{m,n}}$, where $e_{m,n}$ denotes the event that $d_H(x_0, x_m) > d_H(x_0, x_n)$, with $m \in \{1, \ldots, k\}$ and $n \in \{k + 1, \ldots, Q\}$. Then, using the union bound, we have

$$P(E) \leq \sum_{m,n} P(e_{m,n}) \leq k(Q - k)P(\epsilon_{k,k+1})$$

$$= k(Q - k)P(d_H(x_0, x_k) > d_H(x_0, x_{k+1}))$$

$$= k(Q - k)P(d_H(x_0, x_{k+1}) < d_H(x_0, x_k)),$$

where we have used the fact that the most probable among all $e_{m,n}$ events is the one corresponding to the order mismatch between the $k^th$ and $k + 1^st$ NN. Now, note that the RHsh output $\delta$ satisfies $\max_{i,j}|d_H(x_i, x_j) - d(x_i, x_j)| \leq \delta^2$. Observe that $\Delta_k = d(x_0, x_{k+1}) - d(x_0, x_k) \geq 2\delta$ is a sufficient condition for $d_H(x_0, x_{k+1}) \geq d_H(x_0, x_k)$, since

$$d_H(x_0, x_{k+1}) - d_H(x_0, x_k) \geq d(x_0, x_{k+1}) - d(x_0, x_k) \geq 2\delta - 2\delta = 0$$

by the triangular inequality. This leads to

$$P(d_H(x_0, x_{k+1}) < d_H(x_0, x_k))$$

$$= 1 - P(d_H(x_0, x_{k+1}) \geq d_H(x_0, x_k))$$

$$\leq 1 - P(\Delta_k \geq 2\delta) = P(\Delta_k < 2\delta).$$

Combining all of the above with Lem. 3, the probability that the $k$ NN is not preserved is bounded by

$$P(E) \leq k(Q - k)P(d_H(x_0, x_{k+1}) < d_H(x_0, x_k))$$

$$\leq k(Q - k)P(\Delta_k < 2\delta)$$

$$\leq k(Q - k)P(\Delta_k - \mathbb{E}_x[\Delta_k] < -2\delta)$$

$$\leq k(Q - k)e^{-c(\mathbb{E}_x[\Delta_k] - 2\delta)^2}.$$

Now, by letting $kQe^{-c(\mathbb{E}_x[\Delta_k] - 2\delta)^2} \leq \epsilon$, we have that the requirement for the top $k$ NNs to be exactly preserved with probability at least $1 - \epsilon$ is

$$\mathbb{E}_x[\Delta_k] \geq 2\delta + \sqrt{\frac{1}{c} \log \frac{kQ}{\epsilon}}.$$

**References**


---

3Here we assume $\lambda = 1$ without loss of generality.


