Binary Embedding: Fundamental Limits and Fast Algorithm

Xinyang Yi YIXY@UTEXAS.EDU
Constantine Caramanis CONSTANTINE@UTEXAS.EDU
Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712
Eric Price ECPRICE@CS.UTEXAS.EDU
Department of Computer Science, The University of Texas at Austin, Austin, TX 78712

Abstract

Binary embedding is a nonlinear dimension reduction methodology where high dimensional data are embedded into the Hamming cube while preserving the structure of the original space. Specifically, for an arbitrary N distinct points in $\mathbb{S}^{p-1}$, our goal is to encode each point using m-dimensional binary strings such that we can reconstruct their geodesic distance up to δ uniform distortion. Existing binary embedding algorithms either lack theoretical guarantees or suffer from running time $O\big(mp\big)$. We make three contributions: (1) we establish a lower bound that shows any binary embedding oblivious to the set of points requires $m = \Omega(\frac{1}{\delta^2}\log N)$ bits and a similar lower bound for non-oblivious embeddings into Hamming distance; (2) we propose a novel fast binary embedding algorithm with provably optimal bit complexity $m = O\big(\frac{1}{\delta^2}\log N\big)$ and near linear running time $O(p \log p)$ whenever $\log N \ll \delta\sqrt{p}$, with a slightly worse running time for larger $\log N$; (3) we also provide an analytic result about embedding a general set of points $K \subseteq \mathbb{S}^{p-1}$ with even infinite size. Our theoretical findings are supported through experiments on both synthetic and real data sets.

1. Introduction

Low distortion embeddings that transform high-dimensional points to low-dimensional space have played an important role in dealing with storage, information retrieval and machine learning problems for modern datasets. Perhaps one of the most famous results along these lines is the Johnson-Lindenstrauss (JL) lemma Johnson & Lindenstrauss (1984), which shows that N points can be embedded into a $O\big(\delta^{-2}\log N\big)$-dimensional space while preserving pairwise Euclidean distance up to δ-Lipschitz distortion. This δ^{-2} dependence has been shown to be information-theoretically optimal Alon (2003). Significant work has focused on fast algorithms for computing the embeddings, e.g., (Ailon & Chazelle, 2006; Krahmer & Ward, 2011; Ailon & Liberty, 2013; Cheraghchi et al., 2013; Nelson et al., 2014).

Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

More recently, there has been a growing interest in designing binary codes for high dimensional points with low distortion, i.e., embeddings into the binary cube (Weiss et al., 2009; Raginsky & Lazebnik, 2009; Salakhutdinov & Hinton, 2009; Gong & Lazebnik, 2011; Yu et al., 2014). Compared to JL embedding, embedding into the binary cube (also called binary embedding) has two advantages in practice: (i) As each data point is represented by a binary code, the disk size for storing the entire dataset is reduced considerably. (ii) Distance in binary cube is some function of the Hamming distance, which can be computed quickly using computationally efficient bit-wise operators. As a consequence, binary embedding can be applied to a large number of domains such as biology, finance and computer vision where the data are usually high dimensional.

While most JL embeddings are linear maps, any binary embedding is fundamentally a nonlinear transformation. As we detail below, this nonlinearity poses significant new technical challenges for both upper and lower bounds. In particular, our understanding of the landscape is significantly less complete. To the best of our knowledge, lower bounds are not known; embedding algorithms for infinite sets have distortion-dependence δ significantly exceeding their finite-set counterparts; and perhaps most significantly, there are no fast (near linear-time) embedding algorithms with strong performance guarantees. As we explain below, this paper contributes to each of these three areas. First, we detail some recent work and state of the art results.

Recent Work. A common approach pursued by several existing works, considers the natural extension of JL embedding techniques via one bit quantization of the projections:

$$\boldsymbol{b}(\boldsymbol{x}) = \text{sign}(\mathbf{A}\boldsymbol{x}), \tag{1.1}$$

where $\boldsymbol{x} \in \mathbb{R}^p$ is input data point, $\mathbf{A} \in \mathbb{R}^{m \times p}$ is a projection matrix and $\boldsymbol{b}(\boldsymbol{x})$ is the embedded binary code. In particular, Jacques et al. (2011) shows when each entry of $\mathbf{A}$ is generated independently from $\mathcal{N}(0,1)$, with $m > \frac{1}{\delta^2}\log N$ it with high probability achieves at most δ (additive) distortion for N points. Work in Plan & Vershynin (2014) extend these results to arbitrary sets $K \subseteq \mathbb{S}^{p-1}$ where $|K|$ can be infinite. They prove that the embedding with δ-distortion can be obtained when $m \gtrsim w(K)^2/\delta^6$ where $w(K)$ is the *Gaussian Mean Width* of K. It is unknown whether the unusual δ^{-6} dependence is optimal or not. Despite provable sample complexity guarantees, one bit quantization of random projection as in (1.1), suffers from $O\big(mp\big)$ running time for a single point. This quadratic dependence can result in a prohibitive computational cost for high-dimensional data. Analogously to the developments in "fast" JL embeddings, there are several algorithms proposed to overcome this computational issue. Work in Gong et al. (2013) proposes a bilinear projection method. By setting $m = O(p)$, their method reduces the running time from $O(p^2)$ to $O(p^{1.5})$. More recently, work in Yu et al. (2014) introduces a circulant random projection algorithm that requires running time $O\big(p\log p\big)$. While these algorithms have reduced running time, to the best of our knowledge, the measurement complexities of the two algorithms are still unknown. Another line of work considers learning binary codes from data by solving certain optimization problems (Weiss et al., 2009; Salakhutdinov & Hinton, 2009; Norouzi et al., 2012; Yu et al., 2014). Unfortunately, there is no known provable bits complexity result for these algorithms. It is also worth noting that Raginsky & Lazebnik (2009) provide a binary code design for preserving shift-invariant kernels. Their method suffers from the same quadratic computational issue compared with the fully random Gaussian projection method.

Another related dimension reduction technique is locality sensitive hashing (LSH) where the goal is to compute a discrete data structure such that similar points are mapped into the same bucket with high probability (see, e.g., Andoni & Indyk (2006)). The key difference is that LSH preserves short distances, but binary embedding preserves both short and far distances. For points that are far apart, LSH only cares that the hashings are different while binary embedding cares how different they are.

Contributions of this paper. In this paper, we address several unanswered problems about binary embedding:

1. We provide two lower bounds for binary embeddings. The first shows that any method for embedding and for recovering a distance estimate from the embedded points that is independent of the data being embedded must use $\Omega(\frac{1}{\delta^2}\log N)$ bits. This is based on a bound on the communication complexity of Hamming distance used by (Jayram & Woodruff, 2013) for a lower bound on the "distributional" JL embedding. Separately, we give a lower bound for arbitrarily data-dependent methods that embed into (any function of) the Hamming distance, showing such algorithms require $m = \Omega(\frac{1}{\delta^2 \log(1/\delta)}\log N)$. This bound is similar to Alon (2003) which gets the same result for JL, but the binary embedding requires a different construction.

2. We provide the first provable fast algorithm with optimal measurement complexity $O\big(\frac{1}{\delta^2}\log N\big)$. The proposed algorithm has running time $O\big(\frac{1}{\delta^2}\log\frac{1}{\delta}\log^2 N\log p\log^3\log N + p\log p\big)$ thus has almost linear time complexity when $\log N \lesssim \delta\sqrt{p}$. Our algorithm is based on two key novel ideas. First, our similarity is based on the median Hamming distance of sub-blocks of the binary code; second, our new embedding takes advantage of a *pair-wise independence argument* of Gaussian Toeplitz projection that could be of independent interest.

3. For arbitrary set $K \subseteq \mathbb{S}^{p-1}$ and the fully random Gaussian projection algorithm, we prove that $m = O(w(K^+)^2/\delta^4)$ is sufficient to achieve δ uniform distortion. Here K^+ is an *expanded* set of K. Although in general $K \subseteq K^+$ and hence $w(K) \le w(K^+)$, for interesting K such as sparse or low rank sets, one can show $w(K^+) = \Theta(w(K)) \ll p$. Therefore applying our theory to these sets results in an improved dependence on δ compared to a recent result in Plan & Vershynin (2014). See Section 3.3 for a detailed discussion.

Discussion. For the fast binary embedding, one simple solution, to the best of our knowledge not previously stated, is to combine a Gaussian projection and the well known results about fast JL. In detail, consider the strategy $\boldsymbol{b}(\boldsymbol{x}) = \text{sign}(\mathbf{AF}\boldsymbol{x})$, where $\mathbf{A}$ is a Gaussian matrix and $\mathbf{F}$ is any fast JL construction such as subsampled Walsh-Hadamard matrix Rudelson & Vershynin (2008) or partial circulant matrix Krahmer et al. (2014) with column flips. A simple analysis shows that this approach achieves measurement complexity $O(\frac{1}{\delta^2}\log N)$ and running time $O(\frac{1}{\delta^4}\log^2 N\log p\log^3\log N + p\log p)$ by following the best known fast JL results. Our fast binary embedding algorithm builds on this simple but effective thought. Instead of using a Gaussian matrix after the fast JL transform, we use a series of Gaussian Toeplitz matrices that have fast matrix vector multiplication. This novel construction improves the running time by δ^2 while keeping measurement complexity the same. In order for this to work, we need to change the estimator from straight Hamming distance to one based on the median of several Hamming distances.

An interesting point of comparison is Ailon & Rauhut (2014), which considers "RIP-optimal" distributions that give JL embeddings with optimal measurement complexity $O(\frac{1}{\delta^2}\log N)$ and running time $O(p\log p)$. They show the existence of such embeddings whenever $\log N < \delta^2 p^{1/2-\gamma}$ for any constant $\gamma > 0$, which is essentially no better than the bound given by the folklore method of composing a Gaussian projection with a subsampled Fourier matrix. In our binary setting, we show how to improve the region of optimality by a factor of δ. It would be interesting to try and translate this result back to the JL setting.

Notations. We use $[n]$ to denote set $\{1, 2, \ldots, n\}$. We use $\boldsymbol{x}_{\mathcal{I}}$ to denote the sub-vector of $\boldsymbol{x}$ with index set $\mathcal{I} \subseteq [n]$. We denote entry-wise vector multiplication as $\boldsymbol{x} \odot \boldsymbol{y} = (x_1y_1, x_2y_2, \ldots, x_ny_n)^\top$. For two random variables X, Y, we denote the statement that X and Y are independent as $X \perp Y$. For two binary strings $\boldsymbol{a}, \boldsymbol{b} \in \{0,1\}^m$, we use $d_{\mathcal{H}}(\boldsymbol{a}, \boldsymbol{b})$ to denote the normalized Hamming distance, i.e., $d_{\mathcal{H}}(\boldsymbol{a}, \boldsymbol{b}) := \frac{1}{m}\sum_{i=1}^m \mathbb{1}(a_i \neq b_i)$.

2. Problem Setup and Preliminaries

In this section, we state our problem formally, give some key definitions and present a simple (known) algorithm that sets the stage for the main results of this paper.

2.1. Problem Setup

Given a set of p-dimensional points, our goal is to find a transformation $f : \mathbb{R}^p \mapsto \{0,1\}^m$ such that the Hamming distance (or other related, easily computable metric) between two binary codes is close to their similarity in the original space. We consider points on the unit sphere $\mathbb{S}^{p-1}$ and use the normalized geodesic distance (occasionally, and somewhat misleadingly, called cosine similarity) as the input space similarity metric. For two points $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^p$, we use $d(\boldsymbol{x}, \boldsymbol{y})$ to denote the geodesic distance, defined as

$$d(\boldsymbol{x}, \boldsymbol{y}) := \frac{\angle(\boldsymbol{x}/\|\boldsymbol{x}\|_2, \boldsymbol{y}/\|\boldsymbol{y}\|_2)}{\pi},$$

where $\angle(\cdot,\cdot)$ denotes the angle between two vectors. For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{S}^{p-1}$, the metric $d(\boldsymbol{x}, \boldsymbol{y})$ is proportional to the length of the shortest path connecting $\boldsymbol{x}, \boldsymbol{y}$ on the sphere.

Given the success of JL embedding, a natural approach is to consider the one bit quantization of a random projection:

$$\boldsymbol{b} = \text{sign}(\mathbf{A}\boldsymbol{x}), \tag{2.1}$$

where $\mathbf{A}$ is some random projection matrix. Given two points $\boldsymbol{x}, \boldsymbol{y}$ with embedding vectors $\boldsymbol{b}$, and $\boldsymbol{c}$, we have $b_i \neq c_i$ if and only if $\langle \mathbf{A}_i, \boldsymbol{x}\rangle\langle \mathbf{A}_i, \boldsymbol{y}\rangle < 0$. The traditional metric in the embedded space has been the so-called normalized Hamming distance defined as

$$d_{\mathbf{A}}(\boldsymbol{x}, \boldsymbol{y}) := \frac{1}{m}\sum_{i=1}^m \mathbb{1}\Big\{\text{sign}\big(\langle \mathbf{A}_i, \boldsymbol{x}\rangle\big) \neq \text{sign}\big(\langle \mathbf{A}_i, \boldsymbol{y}\rangle\big)\Big\}. \tag{2.2}$$

Definition 2.1. (δ-uniform Embedding) Given a set $K \subseteq \mathbb{S}^{p-1}$ and projection matrix $\mathbf{A} \in \mathbb{R}^{m\times p}$, we say the embedding $\boldsymbol{b} = \text{sign}(\mathbf{A}\boldsymbol{x})$ provides a δ-uniform embedding for points in K if

$$\big|d_{\mathbf{A}}(\boldsymbol{x}, \boldsymbol{y}) - d(\boldsymbol{x}, \boldsymbol{y})\big| \leq \delta,\ \forall\ x, y \in K. \tag{2.3}$$

Note that unlike for JL, we aim to control *additive* error instead of *relative* error. Due to the inherently limited resolution of binary embedding, controlling relative error would force the embedding dimension m to scale inversely with the minimum distance of the original points, and in particular would be impossible for any infinite set.

2.2. Uniform Random Projection

Algorithm 1 Uniform Random Projection

```
input Finite number of points $K = \{\boldsymbol{x}_i\}_{i=1}^{|K|}$ where $K \subseteq \mathbb{S}^{p-1}$, embedding target dimension $m$.
1: Construct matrix $\mathbf{A} \in \mathbb{R}^{m\times p}$ where each entry $\mathbf{A}_{i,j}$ is drawn independently from $\mathcal{N}(0,1)$.
2: for $i = 1, 2, \ldots, |K|$ do
3:     $\boldsymbol{b}_i \leftarrow \text{sign}(\mathbf{A}\boldsymbol{x}_i)$.
4: end for
output $\{\boldsymbol{b}_i\}_{i=1}^{|K|}$
```

Algorithm 1 presents (2.1) formally, when $\mathbf{A}$ is an i.i.d. Gaussian random matrix, i.e., $\mathbf{A}_i \sim \mathcal{N}(0, \mathbf{I}_p),\ \forall\ i \in [m]$. It is easy to observe that for two fixed points $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{S}^{p-1}$ we have

$$\mathbb{E}\Big(\mathbb{1}\big\{\ \text{sign}\left(\langle \mathbf{A}_i, \boldsymbol{x}\rangle\right) \neq \text{sign}\left(\langle \mathbf{A}_i, \boldsymbol{y}\rangle\right)\big\}\Big) = d(\boldsymbol{x}, \boldsymbol{y}). \tag{2.4}$$

The above equality has a geometric explanation: each $\mathbf{A}_i$ actually represents a uniformly distributed random hyperplane in $\mathbb{R}^p$. Then $\text{sign}\left(\langle \mathbf{A}_i, \boldsymbol{x}\rangle\right) \neq \text{sign}\left(\langle \mathbf{A}_i, \boldsymbol{y}\rangle\right)$ holds if and only if hyperplane $\mathbf{A}_i$ intersects the arc between $\boldsymbol{x}$ and $\boldsymbol{y}$. In fact, $d_{\mathbf{A}}(\boldsymbol{x}, \boldsymbol{y})$ is equal to the fraction of such hyperplanes. Under such uniform tessellation, the probability with which the aforementioned event occurs is $d(\boldsymbol{x}, \boldsymbol{y})$. Applying Hoeffding's inequality and probabilistic union bound over N pairs of points, we have the following straightforward guarantee.

Proposition 2.2. Given a set $K \subseteq \mathbb{S}^{p-1}$ with finite size $|K|$, consider Algorithm 1 with $m \geq c(1/\delta^2)\log|K|$ for some absolute constant c. Then with probability at least $1 - 2\exp(-\delta^2 m)$, we have

$$\left| d_{\mathbf{A}}(\boldsymbol{x}, \boldsymbol{y}) - d(\boldsymbol{x}, \boldsymbol{y}) \right| \le \delta, \ \forall\, \boldsymbol{x}, \boldsymbol{y} \in K.$$

Three important questions remain unanswered: (i) Lower Bounds – is the performance guaranteed by Proposition 2.2 optimal? (ii) Fast Embedding – whereas Algorithm 1 is quadratic (depending on the product mp), fast JL algorithms are nearly linear in p; does something similar exist for binary embedding? (iii) Infinite Sets – proposition 2.2 requires $|K|$ is finite; what guarantee can we get for $|K| = \infty$? The rest of the paper focuses on the three problems.

3. Main Results

We now present our main results on lower bounds, on fast binary embedding, and finally, on a general result for infinite sets.

3.1. Lower Bounds

We offer two different lower bounds. The first shows that any embedding technique that is oblivious to the input points must use $\Omega(\frac{1}{\delta^2}\log N)$ bits, regardless of what method is used to estimate geodesic distance from the embeddings. This shows that uniform random projection and our fast binary embedding achieve optimal bit complexity (up to constants). The bound follows from results by (Jayram & Woodruff, 2013) on the communication complexity of Hamming distance.

Theorem 3.1. Consider any distribution on embedding functions $f : \mathbb{S}^{p-1} \to \{0,1\}^m$ and reconstruction algorithms $g : \{0,1\}^m \times \{0,1\}^m \to \mathbb{R}$ such that for any $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in \mathbb{S}^{p-1}$ we have

$$\left| g(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)) - d(\boldsymbol{x}_i, \boldsymbol{x}_j) \right| \le \delta$$

for all $i, j \in [N]$ with probability $1 - \epsilon$. Then $m = \Omega(\frac{1}{\delta^2}\log(N/\epsilon))$.

One could imagine, however, that an embedding could use knowledge of the input point set to embed any specific set of points into a lower-dimensional space than is possible with an oblivious algorithm. In the Johnson-Lindenstrauss setting, Alon (2003) showed that this is not possible beyond (possibly) a $\log(1/\delta)$ factor. We show the analogous result for binary embeddings. Relative to Theorem 3.1, our second lower bound works for data-dependent embedding functions but loses a $\log(1/\delta)$ and requires the reconstruction function to depend only on the Hamming distance between the two strings. This restriction is natural because an unrestricted data-dependent reconstruction function could simply encode the answers and avoid any dependence on δ.

With the scheme given in (2.1), choosing $\mathbf{A}$ as a fully random Gaussian matrix yields $d_{\mathbf{A}}(\boldsymbol{x}, \boldsymbol{y}) \approx d(\boldsymbol{x}, \boldsymbol{y})$. However, an arbitrary binary embedding algorithm may not yield a linear functional relationship between Hamming distance and geodesic distance. Thus for this lower bound, we allow the design of an algorithm with arbitrary link function $\mathcal{L}$.

Definition 3.2. (Data-dependent binary embedding problem)
Let $\mathcal{L} : [0,1] \to [0,1]$ be a monotonic and continuous function. Given a set of points $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N \in \mathbb{S}^{p-1}$, we say a binary embedding mapping f solves the binary embedding problem in terms of link function $\mathcal{L}$, if for any $i, j \in [N]$,

$$\left| d_{\mathcal{H}}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)) - \mathcal{L}\big(d(\boldsymbol{x}_i, \boldsymbol{x}_j)\big) \right| \le \delta. \tag{3.1}$$

Although the choice of $\mathcal{L}$ is flexible, note that for the same point, we always have $d_{\mathcal{H}}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_i)) = d(\boldsymbol{x}_i, \boldsymbol{x}_i) = 0$, thus (3.1) implies $\mathcal{L}(0) < \delta$. We can just let $\mathcal{L}(0) = 0$. In particular, we let $\mathcal{L}_{\max} = \mathcal{L}(1)$. We have the following lower bound:

Theorem 3.3. There exist $2N$ points $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_{2N} \in \mathbb{S}^{N-1}$ such that for any binary embedding algorithm f on $\{\boldsymbol{x}_i\}_{i=1}^{2N}$, if it solves the data-dependent binary embedding problem defined in 3.2 in terms of link function $\mathcal{L}$ and any $\delta \in (0, \frac{1}{16\sqrt{e}}\mathcal{L}_{\max})$, it must satisfy

$$m \ge \frac{1}{128e}\left(\frac{\mathcal{L}_{\max}}{\delta}\right)^2 \frac{\log N}{\log \frac{\mathcal{L}_{\max}}{2\delta}}. \tag{3.2}$$

Remark 3.4. We make two remarks for the above result. (1) When $\mathcal{L}_{\max}$ is some constant, our result implies that for general N points, any binary embedding algorithm (even data-dependent) must have $\Omega(\frac{1}{\delta^2 \log\frac{1}{\delta}}\log N)$ number of measurements. This is analogous to Alon's lower bound in the JL setting. It is worth highlighting two differences: (i) The JL setting considers the same metric (Euclidean distance) for both the input and the embedded spaces. In binary embedding, however, we are interested in showing the relationship between Hamming distance and geodesic distance. (ii) Our lower bound is applicable to a broader class of binary embedding algorithms as it involves arbitrary, even data-dependent, link function $\mathcal{L}$. Such an extension is not considered in the lower bound of JL. (2) The stated lower bound only depends on $\mathcal{L}_{\max}$ and does not depend on any curvature information of $\mathcal{L}$. The constraint $\mathcal{L}_{\max} > 16\sqrt{e}\delta$ is critical for our lower bound to hold, but some such restriction is necessary because for $\mathcal{L}_{\max} < \delta$, we are able to embed all points into just one bit. In this case $d_{\mathcal{H}}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)) = 0$ for all pairs and condition (3.1) would hold trivially.

3.2. Fast Binary Embedding

In this section, we present a novel fast binary embedding algorithm. We then establish its theoretical guarantees. There are two key ideas that we leverage: (i) instead of normalized Hamming distance, we use a related metric, the median of the normalized Hamming distance applied to sub-blocks; and (ii) we show a key pair-wise independence lemma for partial Gaussian Toeplitz projection, that allows us to use a concentration bound that then implies nearness in the median-metric we use.

3.2.1. Method

Our algorithm builds on sub-sampled Walsh-Hadamard matrix and partial Gaussian Toeplitz matrices with random column flips. In particular, an m-by-p partial Walsh-Hadamard matrix has the form

$$\mathbf{\Phi} := \mathbf{P} \cdot \mathbf{H} \cdot \mathbf{D}. \tag{3.3}$$

The above construction has three components. We characterize each term as follows: Term $\mathbf{D}$ is a p-by-p diagonal matrix with diagonal terms $\{\zeta_i\}_{i=1}^p$ that are drawn from i.i.d. Rademacher sequence, i.e, for any $i \in [p]$, $\Pr(\zeta_i = 1) = \Pr(\zeta_i = -1) = 1/2$. Term $\mathbf{H}$ is a p-by-p scaled Walsh-Hadamard matrix such that $\mathbf{H}^\top \mathbf{H} = \mathbf{I}_p$. Term $\mathbf{P}$ is an m-by-p sparse matrix where one entry of each row is set to be 1 while the rest are 0. The nonzero coordinate of each row is drawn independently from uniform distribution. In fact, the role of $\mathbf{P}$ is to randomly select p rows of $\mathbf{H} \cdot \mathbf{D}$.

An m-by-n partial Gaussian Toeplitz matrix has the form

$$\mathbf{\Psi} := \mathbf{P} \cdot \mathbf{T} \cdot \mathbf{D}. \tag{3.4}$$

We introduce each term as follows: Term $\mathbf{D}$ a is n-by-n diagonal matrix with diagonal terms $\{\zeta_i\}_{i=1}^n$ that are drawn from i.i.d. Rademacher sequence. Term $\mathbf{T}$ is a n-by-n Toeplitz matrix constructed from $(2n-1)$-dimensional vector $\boldsymbol{g}$ such that $\mathbf{T}_{i,j} = g_{i-j+n}$ for any $i, j \in [n]$. In particular, $\boldsymbol{g}$ is drawn from $\mathcal{N}(0, \mathbf{I}_{2n-1})$. Term $\mathbf{P}$ is an m-by-n sparse matrix where $\mathbf{P}_i = \boldsymbol{e}_i^\top$ for any $i \in [m]$. Equivalently, we use $\mathbf{P}$ to select the first m rows of $\mathbf{TD}$. It's worth to note we actually only need to select any distinct m rows.

With the above constructions in hand, we present our fast algorithm in Algorithm 2. At a high level, Algorithm 2 consists of two parts: First, we apply column flipped partial Hadamard transform to convert p-dimensional point into n-dimensional intermediate point. Second, we use B independent (m/B)-by-n partial Gaussian Toeplitz matrices and sign operator to map an intermediate point into B blocks of binary codes. In terms of similarity computation for the embedded codes, we use the median of each block's normalized Hamming distance. In detail, for $\boldsymbol{b}, \boldsymbol{c} \in \{0,1\}^m$, B-wise normalized Hamming distance is defined as

$$d_{\mathcal{H}}(\boldsymbol{b}, \boldsymbol{c}; B) := \text{median}\left(\left\{ d_{\mathcal{H}}\left(\boldsymbol{b}_{T_i}, \boldsymbol{c}_{T_i}\right) \right\}_{i=0}^{B-1}\right) \tag{3.5}$$

where $T_i = [i+1, \ldots, i+m/B]$.

It is worth noting that our first step is one construction of fast JL transform. In fact any fast JL transform would work for our construction, but we choose a standard one with real value: based on Rudelson & Vershynin (2008); Cheraghchi et al. (2013); Krahmer & Ward (2011), it is known that with $m = O\left(\epsilon^{-2} \log N \log p \log^3(\log N)\right)$ measurements, a subsampled Hadamard matrix with column flips becomes an ϵ-JL matrix for N points.

The second part of our algorithm follows framework (2.1). By choosing a Gaussian random vector in each row of Ψ, from our previous discussion in Section 2.2, the probability that such a hyperplane intersects the arc between two points is equal to their geodesic distance. Compared to a fully random Gaussian matrix, as used in Algorithm 1, the key difference is that the hyperplanes represented by rows of Ψ are not independent to each other; this imposes the main analytical challenge.

Algorithm 2 Fast Binary Embedding

```
input Finite number of points $\{\boldsymbol{x}_i\}_{i=1}^N$ where each point $\boldsymbol{x}_i \in \mathbb{S}^{p-1}$, embedded dimension $m$, intermediate dimension $n$, number of blocks $B$.
 1: Draw a $n$-by-$p$ sub-sampled Walsh-Hadamard matrix $\mathbf{\Phi}$ according to (3.3). Draw $B$ independent partial Gaussian Toeplitz matrices $\{\mathbf{\Psi}^{(j)}\}_{j=1}^B$ with size $(m/B)$-by-$n$ according to (3.4).
 2: {Part I: Fast JL}
 3: for $i = 1, 2, \ldots, N$ do
 4:     $\boldsymbol{y}_i \leftarrow \mathbf{\Phi} \cdot \boldsymbol{x}_i$.
 5: end for
 6: {Part II: Partial Gaussian Toeplitz Projection}
 7: for $i = 1, 2, \ldots, N$ do
 8:     for $j = 1, 2, \ldots, B$ do
 9:         $\boldsymbol{c}_j \leftarrow \text{sign}\left(\mathbf{\Psi}^{(j)} \cdot \boldsymbol{y}_i\right)$.
10:     end for
11:     $\boldsymbol{b}_i \leftarrow [\boldsymbol{c}_1; \boldsymbol{c}_2; \ldots; \boldsymbol{c}_B]$
12: end for
output $\{\boldsymbol{b}_i\}_{i=1}^N$
```

3.2.2. Analysis

We give the analysis for Algorithm 2. We first review a well known result about fast JL transform. It can be proved by combining Theorem 14 in Cheraghchi et al. (2013) and Theorem 3.1 in Krahmer & Ward (2011).

Lemma 3.5. Consider the column flipped partial Hadamard matrix defined in (3.3) with size m-by-p. For N points $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N \in \mathbb{S}^{p-1}$, let $\boldsymbol{y}_i = \sqrt{\frac{p}{m}}\Phi(\boldsymbol{\zeta}) \cdot \boldsymbol{x}_i, \ \forall \ i \in [N]$. For some absolute constant c, suppose $m \geq c\delta^{-2}\log N \log p \log^3(\log N)$, then with probability at least 0.99, we have that

$$\big|\|\boldsymbol{y}_i\|_2 - 1\big| \leq \delta, \text{ for any } i \in [N]; \tag{3.6}$$

$$\big|\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2 - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2\big| \leq \delta\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2 \tag{3.7}$$

for any $i, j \in [N]$.

The above result suggests that the first part of our algorithm reduces the dimension while preserving well the Euclidean distance of each pair. Under this condition, all the pairwise geodesic distances are also well preserved as confirmed by the following result.

Lemma 3.6. Consider the set of embedded points $\{\boldsymbol{y}_i\}_{i=1}^N$ defined in Lemma 3.5. Suppose conditions (3.7)-(3.6) hold with $\delta > 0$. Then for any $i, j \in [N]$,

$$\big|d(\boldsymbol{y}_i, \boldsymbol{y}_j) - d(\boldsymbol{x}_i, \boldsymbol{x}_j)\big| \leq C\delta \tag{3.8}$$

holds with some absolute constant C.

The next result is our independence lemma, and is one of the key technical ideas that make our result possible. The result shows that for any fixed $\boldsymbol{x}$, Gaussian Toeplitz projection (with column flips) plus sign$(\cdot)$ generate pair-wise independent binary codes.

Lemma 3.7. Let $\boldsymbol{g} \sim \mathcal{N}(0, \mathbf{I}_{2n-1})$, $\boldsymbol{\zeta} = \{\zeta_i\}_{i=1}^n$ be an i.i.d. Rademacher sequence. Let $\mathbf{T}$ be a random Toeplitz matrix constructed from $\boldsymbol{g}$ such that $\mathbf{T}_{i,j} = g_{i-j+n}$. Consider any two distinct rows of $\mathbf{T}$ say $\boldsymbol{\xi}$, $\boldsymbol{\xi}'$. For any two fixed vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, we define the following random variables

$$X = \text{sign}\left\langle \boldsymbol{\xi} \odot \boldsymbol{\zeta},\ \boldsymbol{x}\right\rangle,\ \ X' = \text{sign}\left\langle \boldsymbol{\xi}' \odot\ \boldsymbol{\zeta}, \boldsymbol{x}\right\rangle;$$
$$Y = \text{sign}\left\langle \boldsymbol{\xi} \odot \boldsymbol{\zeta},\ \boldsymbol{y}\right\rangle,\ \ \ Y' = \text{sign}\left\langle \boldsymbol{\xi}' \odot\ \boldsymbol{\zeta}, \boldsymbol{y}\right\rangle.$$

We have that $X \perp X', X \perp Y', Y \perp X', Y \perp Y'$.

With these ingredients in hand, we are ready to prove the following result.

Theorem 3.8. Consider Algorithm 2 with random matrices $\mathbf{\Phi}$, $\mathbf{\Psi}$ defined in (3.3) and (3.4) respectively. For finite number of points $\{\boldsymbol{x}_i\}_{i=1}^N$, let $\boldsymbol{b}_i$ be the binary codes of $\boldsymbol{x}_i$ generated by Algorithm 2. Suppose we set

$$B \geq c\log N,\ n \geq m/B \geq c''(1/\delta^2),$$
$$\text{and}\ \ n \geq c'(1/\delta^2)\log N \log p \log^3(\log N)$$

Algorithm 3 Alternative Fast Binary Embedding

```
input Finite number of points $\{\boldsymbol{x}_i\}_{i=1}^N$ where each point
      $\boldsymbol{x}_i \in \mathbb{S}^{p-1}$, embedded dimension $m$, intermediate dimension $n$.
1: Draw a $n$-by-$p$ sub-sampled Walsh-Hadamard matrix $\mathbf{\Phi}$ according to (3.3). Construct $m$-by-$n$ matrix $\mathbf{A}$ where each entry is drawn independently from $\mathcal{N}(0,1)$.
2: for $i = 1, 2, \ldots, N$ do
3:     $\boldsymbol{b}_i \leftarrow \text{sign}(\mathbf{A}\mathbf{\Phi}\boldsymbol{x}_i)$
4: end for
output $\{\boldsymbol{b}_i\}_{i=1}^N$
```

with some absolute constants c, c', c'', then with probability at least 0.98, we have that for any $i, j \in [N]$

$$\big|d_{\mathcal{H}}(\boldsymbol{b}_i, \boldsymbol{b}_j; B) - d(\boldsymbol{x}_i, \boldsymbol{x}_j)\big| \leq \delta.$$

Similarity metric $d_{\mathcal{H}}(\cdot, \cdot; B)$ is the median of normalized Hamming distance defined in (3.5).

The above result suggests that the measurement complexity of our fast algorithm is $O\big(\frac{1}{\delta^2}\log N\big)$ which matches the performance of Algorithm 1 based on fully random matrix. Note that this measurement complexity can not be improved significantly by any data-oblivious binary embedding with any similarity metric, as suggested by Theorem 3.1.

Running time: The first part of our algorithm takes time $O\big(p\log p\big)$. Generating a single block of binary codes from partial Toeplitz matrix takes time $O\big(n\log(\frac{1}{\delta})\big)$[1]. Thus the total running time is $O\big(Bn\log\frac{1}{\delta} + p\log p\big) = O\big(\frac{1}{\delta^2}\log\frac{1}{\delta}\log^2 N \log p \log^3(\log N) + p\log p\big)$. By ignoring the polynomial $\log\log$ factor, the second term $O\big(p\log p\big)$ dominates when $\log N \lesssim \delta\sqrt{p/\log\frac{1}{\delta}}$.

Comparison to an alternative algorithm: Instead of utilizing the partial Gaussian Toeplitz projection, an alternative method, to the best of our knowledge not previously stated, is to use fully random Gaussian projection in the second part of our algorithm. We present the details in Algorithm 3. By combining Proposition 2.2 and Lemma 3.5, it is straightforward to show this algorithm still achieves the same measurement complexity $O\big(\frac{1}{\delta^2}\log N\big)$. The corresponding running time is $O\big(\frac{1}{\delta^4}\log^2 N \log p \log^3(\log N) + p\log p\big)$, so it is fast when $\log N \lesssim \delta^2\sqrt{p}$. Therefore our algorithm has an improved dependence on δ. This improvement comes from fast multiplication of partial Toeplitz matrix and a pair-wise independence argument shown in Lemma 3.7.

[1]Matrix-vector multiplication for m-by-n partial Toeplitz matrix can be implemented in running time $O\big(n\log m\big)$.

3.3. δ-uniform Embedding for General K

In this section, we turn back to the fully random projection binary embedding (Algorithm 1). Recall that in Proposition 2.2, we show for finite size K, $m = O(\frac{1}{\delta^2}\log|K|)$ measurements are sufficient to achieve δ-uniform embedding. For general K, the challenge is that there might be an infinite number of distinct points in K, so Proposition 2.2 cannot be applied. In proving the JL lemma for an infinite set K, the standard technique is either constructing an ϵ-net of K or reducing the distortion to the deviation bound of a Gaussian process. However, due to the non-linearity essential for binary embedding, these techniques cannot be directly extended to our setting. Therefore strengthening Proposition 2.2 to infinite size K imposes significant technical challenges. Before stating our result, we first give some definitions.

Definition 3.9. (Gaussian mean width) Let $\boldsymbol{g} \sim \mathcal{N}(0, \mathbf{I}_p)$. For any set $K \subseteq \mathbb{S}^{p-1}$, the Gaussian mean width of K is defined as

$$w(K) := \mathbb{E}_{\boldsymbol{g}} \sup_{\boldsymbol{x}\in K} \left|\langle \boldsymbol{g}, \boldsymbol{x}\rangle\right|.$$

Here, $w(K)^2$ measures the effective dimension of set K. In the trivial case $K = \mathbb{S}^{p-1}$, we have $w(K)^2 \lesssim p$. However, when K has some special structure, we may have $w(K)^2 \ll p$. For instance, when $K = \{\boldsymbol{x} \in \mathbb{S}^{p-1} : |\operatorname{supp}(\boldsymbol{x})| \le s\}$, it has been shown that $w(K) = \Theta(\sqrt{s\log(p/s)})$ (see Lemma 2.3 in Plan & Vershynin (2013)).

For a given δ, we define K_δ^+, the *expanded version* of $K \subseteq \mathbb{S}^{p-1}$ as:

$$K_\delta^+ := K \bigcup \big\{ \boldsymbol{z} \in \mathbb{S}^{p-1} : \boldsymbol{z} = \frac{\boldsymbol{x}-\boldsymbol{y}}{\|\boldsymbol{x}-\boldsymbol{y}\|_2}, \quad \forall\, \boldsymbol{x},\boldsymbol{y} \in K \text{ if } \delta^2 \le \|\boldsymbol{x}-\boldsymbol{y}\|_2 \le \delta \big\}. \tag{3.9}$$

-0.1in In other words, K_δ^+ is constructed from K by adding the normalized differences between pairs of points in K that are within δ but not closer than δ^2. Now we state the main result as follows.

Theorem 3.10. Consider any $K \subseteq \mathbb{S}^{p-1}$. Let $\mathbf{A} \in \mathbb{R}^{m\times p}$ be an i.i.d. Gaussian matrix where each row $\mathbf{A}_i \sim \mathcal{N}(0, \mathbf{I}_p)$. For any two points $\boldsymbol{x},\boldsymbol{y} \in K$, $\boldsymbol{d}_{\mathbf{A}}(\boldsymbol{x},\boldsymbol{y})$ is defined in (2.2). Expanded set K_δ^+ is defined in (3.9). When

$$m \ge c\frac{w(K_\delta^+)^2}{\delta^4},$$

with some absolute constant c, then we have that

$$\sup_{\boldsymbol{x},\boldsymbol{y}\in K} \left| d_{\mathbf{A}}(\boldsymbol{x},\boldsymbol{y}) - d(\boldsymbol{x},\boldsymbol{y})\right| \le \delta$$

holds with probability at least $1 - c_1\exp(-c_2\delta^2 m)$ where c_1, c_2 are absolute constants.

Remark 3.11. We compare the above result to Theorem 1.5 from the recent paper (Plan & Vershynin, 2014) where it is proved that for $m \gtrsim w(K)^2/\delta^6$, Algorithm 1 is guaranteed to achieve δ-uniform embedding for general K. Based on definition (3.9), we have

$$w(K) \le w(K_\delta^+) \le \frac{1}{\delta^2} w(K-K) \lesssim \frac{1}{\delta^2} w(K).$$

Thus in the worst case, Theorem 3.10 recovers the previous result up to a factor $\frac{1}{\delta^2}$. More importantly, for many interesting sets one can show $w(K_\delta^+) \lesssim w(K)$; in such cases, our result leads to an improved dependence on δ. We give several such examples as follows:

- **Low rank set.** For some $\mathbf{U} \in \mathbb{R}^{p\times r}$ such that $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_r$, let $K = \{\boldsymbol{x} \in \mathbb{S}^{p-1} : \boldsymbol{x} = \mathbf{U}\boldsymbol{c},\ \forall\, \boldsymbol{c} \in \mathbb{S}^{r-1}\}$. We simply have $K = K_\delta^+$ and $w(K) \lesssim \sqrt{r}$. Our result implies $m = O\big(r/\delta^4\big)$.
- **Sparse set.** $K = \{\boldsymbol{x} \in \mathbb{S}^{p-1} : |\operatorname{supp}(\boldsymbol{x})| \le s\}$. In this case we have $K_\delta^+ \subseteq \{\boldsymbol{x} \in \mathbb{S}^{p-1} : |\operatorname{supp}(\boldsymbol{x})| \le 2s\}$. Therefore $w(K_\delta^+) = \Theta(\sqrt{s\log(p/s)})$. Our result implies $m = O\big(\frac{s\log(p/s)}{\delta^4}\big)$.
- **Set with finite size.** $|K| < \infty$. As $w(K) \lesssim \sqrt{\log|K|}$ and $|K_\delta^+| \le 2|K|$, our result implies $m = O\big(\log|K|/\delta^4\big)$. We thus recover Proposition 2.2 up to factor $1/\delta^2$.

Applying the result from Plan & Vershynin (2014) to the above sets implies similar results but the dependence on δ becomes $1/\delta^6$.

4. Numerical Results

In this section, we present the results of experiments we conduct to validate our theory and compare the performance of the following three algorithms we discussed: uniform random projection (URP) (Algorithm 1), fast binary embedding (FBE) (Algorithm 2) and the alternative fast binary embedding (FBE-2) (Algorithm 3). We first apply these algorithms to synthetic datasets. In detail, given parameters (N, p), a synthetic dataset is constructed by sampling N points from $\mathbb{S}^{p-1}$ uniformly at random. Recall that δ is the maximum embedding distortion among all pairs of points. We use m to denote the number of binary measurements. Algorithm FBE needs parameters n, B, which are intermediate dimension and number of blocks respectively. Based on Theorem 3.8, n is required to be proportional to m (up to some logarithmic factors) and B is required to be proportional to $\log N$. We thus set $n \approx 1.3m$, $B \approx 1.8\log N$. We also set $n \approx 1.3m$ for FBE-2. In addition, we fix $p = 512$. We report our first result showing the functional relationship between (m, N, δ) in Figure 1. In particular, panel 1(a) shows the the change of distortion

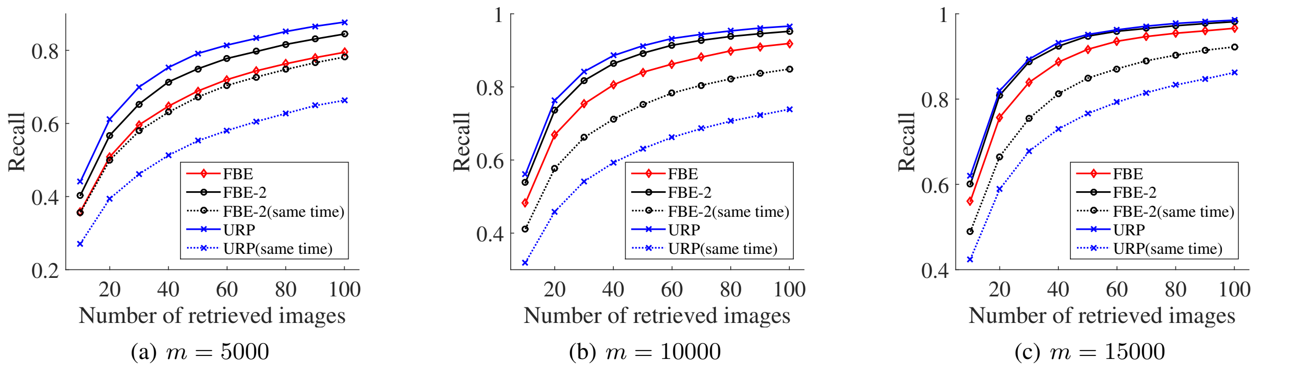
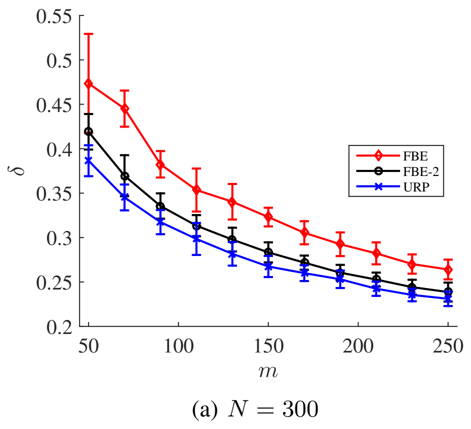


Figure 2. Image retrieval results on Flickr-25600. Each panel presents the recall for specified number of measurements m. Black and blue dot lines are respectively the recall of FBE-2 and URP with less number of measurements but the same running time as FBE.

δ over the number of measurements m for fixed N. We observe that, for all the three algorithms, δ decays with m at the rate predicted by Proposition 2.2 and Theorem 3.8. Panel 1(b) shows the empirical relationship between m and $\log N$ for fixed δ. As predicted by our theory (lower bound and upper bound), m has a linear dependence on $\log N$.

A popular application of binary embedding is image retrieval, as considered in (Gong & Lazebnik, 2011; Gong et al., 2013; Yu et al., 2014). We thus conduct an experiment on the Flickr-25600 dataset that consists of $10k$ images from Internet. Each image is represented by a 25600-dimensional normalized Fisher vector. We take 500 randomly sampled images as query points and leave the rest as base for retrieval. The *relevant images* of each query are defined as its 10 nearest neighbors based on geodesic distance. Given m, we apply FBE, FBE-2 and URP to convert all images into m-dimensional binary codes. In particular, we set $B = 10$ for FBE and $n \approx 1.3m$ for FBE and FBE-2. Then we leverage the corresponding similarity metrics, (3.5) for FBE and Hamming distance for FBE-2 and URP, to retrieve the nearest images for each query. The performance of each algorithm is characterized by *recall*, i.e., the number of retrieved *relevant* images divided by the total number of relevant images. We report our second result in Figure 2. Each panel shows the average recall of all queries for a specified m. We note that FBE-2, as a fast algorithm, performs as well as URP with the same number of measurements. In order to show the running time advantage of our fast algorithm FBE, we also present the performance of FBE-2 and URP with fewer measurements such that they can be computed with the same time as FBE. As we observe, with large number of measurements, FBE-2 and URP perform marginally better than FBE while FBE has a significant improvement over the two algorithms under identical time constraint.



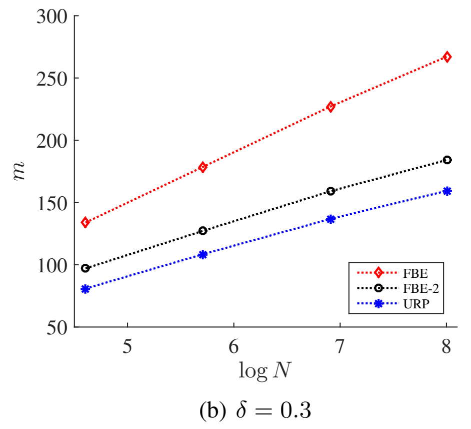


Figure 1. Results on synthetic datasets. (a) Each point, along with the standard deviation represented by the error bar, is an average of 50 trials each of which is based on a fresh synthetic dataset with size $N = 300$ and newly constructed embedding mapping. (b) Each point is computed by slicing at $\delta = 0.3$ in similar plots like (a) but with the corresponding N.

Acknowledgments

C. Caramanis and X. Yi were supported by NSF grants 1056028, 1302435 and 1116955. This research was also partially supported by the U.S. Department of Transportation through the Data-Supported Transportation Operations and Planning (D-STOP) Tier 1 University Transportation Center.

References

Ailon, Nir and Chazelle, Bernard. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pp. 557–563. ACM, 2006.

Ailon, Nir and Liberty, Edo. An almost optimal unrestricted fast johnson-lindenstrauss transform. *ACM Transactions on Algorithms (TALG)*, 9(3):21, 2013.

Ailon, Nir and Rauhut, Holger. Fast and rip-optimal transforms. *Discrete & Computational Geometry*, 52(4):780–798, 2014.

Alon, Noga. Problems and results in extremal combinatoricsâĂŤi. *Discrete Mathematics*, 273(1):31–53, 2003.

Andoni, Alexandr and Indyk, Piotr. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pp. 459–468. IEEE, 2006.

Cheraghchi, Mahdi, Guruswami, Venkatesan, and Velingker, Ameya. Restricted isometry of fourier matrices and list decodability of random linear codes. *SIAM Journal on Computing*, 42(5):1888–1914, 2013.

Gong, Yunchao and Lazebnik, Svetlana. Iterative quantization: A procrustean approach to learning binary codes. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 817–824, 2011.

Gong, Yunchao, Kumar, Sanjiv, Rowley, Henry A, and Lazebnik, Svetlana. Learning binary codes for high-dimensional data using bilinear projections. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 484–491. IEEE, 2013.

Jacques, Laurent, Laska, Jason N, Boufounos, Petros T, and Baraniuk, Richard G. Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *arXiv preprint arXiv:1104.3160*, 2011.

Jayram, TS and Woodruff, David P. Optimal bounds for johnson-lindenstrauss transforms and streaming problems with subconstant error. *ACM Transactions on Algorithms (TALG)*, 9(3):26, 2013.

Johnson, William B and Lindenstrauss, Joram. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

Krahmer, Felix and Ward, Rachel. New and improved johnson-lindenstrauss embeddings via the restricted isometry property. *SIAM Journal on Mathematical Analysis*, 43(3):1269–1281, 2011.

Krahmer, Felix, Mendelson, Shahar, and Rauhut, Holger. Suprema of chaos processes and the restricted isometry property. *Communications on Pure and Applied Mathematics*, 67(11):1877–1904, 2014.

Ledoux, Michel and Talagrand, Michel. *Probability in Banach Spaces: isoperimetry and processes*, volume 23. Springer, 1991.

Nelson, Jelani, Price, Eric, and Wootters, Mary. New constructions of rip matrices with fast multiplication and fewer rows. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1515–1528. SIAM, 2014.

Norouzi, Mohammad, Blei, David M, and Salakhutdinov, Ruslan. Hamming distance metric learning. In *Advances in Neural Information Processing Systems*, pp. 1061–1069, 2012.

Plan, Yaniv and Vershynin, Roman. Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach. *Information Theory, IEEE Transactions on*, 59(1):482–494, 2013.

Plan, Yaniv and Vershynin, Roman. Dimension reduction by random hyperplane tessellations. *Discrete & Computational Geometry*, 51(2):438–461, 2014.

Raginsky, Maxim and Lazebnik, Svetlana. Locality-sensitive binary codes from shift-invariant kernels. In *Advances in neural information processing systems*, pp. 1509–1517, 2009.

Rudelson, Mark and Vershynin, Roman. On sparse reconstruction from fourier and gaussian measurements. *Communications on Pure and Applied Mathematics*, 61(8): 1025–1045, 2008.

Salakhutdinov, Ruslan and Hinton, Geoffrey. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

Weiss, Yair, Torralba, Antonio, and Fergus, Rob. Spectral hashing. In *Advances in neural information processing systems*, pp. 1753–1760, 2009.

Yu, Felix X, Kumar, Sanjiv, Gong, Yunchao, and Chang, Shih-Fu. Circulant binary embedding. *arXiv preprint arXiv:1405.3162*, 2014.