On Binary Embedding using Circulant Matrices

Felix X. Yu[1]   FELIXYU@GOOGLE.COM
Aditya Bhaskara[2]   BHASKARAADITYA@GMAIL.COM
Sanjiv Kumar[1]   SANJIVK@GOOGLE.COM
Yunchao Gong[3]   YUNCHAO@CS.UNC.EDU
Shih-Fu Chang[4]   SFCHANG@EE.COLUMBIA.EDU
[1]GOOGLE RESEARCH, NEW YORK, NY 10011
[2]UNIVERSITY OF UTAH, SALT LAKE CITY, UT 84112
[3]SNAP, INC., VENICE, CA 90291
[4]COLUMBIA UNIVERSITY, NEW YORK, NY 10027

Abstract

Binary embeddings provide efficient and powerful ways to perform operations on large scale data. However binary embedding typically requires long codes in order to preserve the discriminative power of the input space. Thus binary coding methods traditionally suffer from high computation and storage costs in such a scenario. To address this problem, we propose Circulant Binary Embedding (CBE) which generates binary codes by projecting the data with a circulant matrix. The circulant structure allows us to use Fast Fourier Transform algorithms to speed up the computation. For obtaining k-bit binary codes from d-dimensional data, our method improves the time complexity from $\mathcal{O}(dk)$ to $\mathcal{O}(d \log d)$, and the space complexity from $\mathcal{O}(dk)$ to $\mathcal{O}(d)$.

We study two settings, which differ in the way we choose the parameters of the circulant matrix. In the first, the parameters are chosen randomly and in the second, the parameters are learned using the data. For randomized CBE, we give a theoretical analysis comparing it with binary embedding using an unstructured random projection matrix. The challenge here is to show that the dependencies in the entries of the circulant matrix do not lead to a loss in performance. In the second setting, we design a novel time-frequency alternating optimization to learn data-dependent circulant projections, which alternatively minimizes the objective in original and Fourier domains. In both the settings, we show by extensive experiments that the CBE approach gives much better performance than the state-of-the-art approaches if we fix a running time, and provides much faster computation with negligible performance degradation if we fix the number of bits in the embedding.

Keywords: structured matrix, circulant matrix, dimensionality reduction, binary embedding, FFT

1. Introduction

Sketching and dimensionality reduction have become powerful and ubiquitous tools in the analysis of large high-dimensional data sets, with applications ranging from computer vision, to biology, to finance. The celebrated Johnson-Lindenstrauss lemma says that pro-

jecting high dimensional points to a random $\mathcal{O}(\log N)$-dimensional space approximately preserves all the pairwise distances between a set of N points, making it a powerful tool for nearest neighbor search, clustering, etc. This started the paradigm of designing low dimensional *sketches* (or embeddings) of high dimensional data that can be used for efficiently solving various information retrieval problems.

More recently, *binary* embeddings (or embeddings into $\{0, 1\}^k$ or $\{-1, 1\}^k$) have been developed for problems in which we care about preserving the *angles* between high dimensional vectors (Li et al., 2011; Gong et al., 2013; Raginsky and Lazebnik, 2009; Gong et al., 2012b; Liu et al., 2011). The main appeal of binary embeddings stems from the fact that storing them is often much more efficient than storing real valued embeddings. Furthermore, operations such as computing the Hamming distance in binary space can be performed very efficiently either using table lookup, or hardware-implemented instructions on modern computer architectures.

In this paper, we study binary embeddings of high-dimensional data. Our goal is to address one of its main challenges: even though binary embeddings are easy to manipulate, it has been observed that obtaining high accuracy results *requires* the embeddings to be rather long when the data is high dimensional (Li et al., 2011; Gong et al., 2013; Sánchez and Perronnin, 2011). Thus in applications like computer vision, biology and finance (where high dimensional data is common), the task of computing the embedding is a bottleneck. The natural algorithms have time and space complexity $\mathcal{O}(dk)$ per input point in order to produce a k-bit embedding from a d-dimensional input. Our main contribution in this work is to improve these complexities to $\mathcal{O}(d \log d)$ for time and $\mathcal{O}(d)$ for space complexity.

Our results can be viewed as binary analogs of the recent work on *fast* Johnson-Lindenstrauss transform. Starting with the work of Ailon and Chazelle (Ailon and Chazelle, 2006), there has been a lot of beautiful work on fast algorithms for dimension reduction with the goal of preserving pairwise distances between points. Various aspects, such as exploiting sparsity, and using structured matrices to reduce the space and time complexity of dimension reduction, have been explored (Ailon and Chazelle, 2006; Matoušek, 2008; Liberty et al., 2008). But the key difference in our setting is that binary embeddings are *non-linear*. This makes the analysis tricky when the projection matrices do not have independent entries. Binary embeddings are also better suited to approximate the angles between vectors (as opposed to distances). Let us see why.

The general way to compute a binary embedding of a data point $\mathbf{x} \in \mathbb{R}^d$ is to first apply a linear transformation $\mathbf{A}\mathbf{x}$ (for a $k \times d$ matrix $\mathbf{A}$), and then apply a *binarization* step. We consider the natural binarization of taking the sign. Thus, for a point $\mathbf{x}$, the binary embedding into $\{-1, 1\}^d$ we consider is

$$h(\mathbf{x}) = \text{sign}(\mathbf{A}\mathbf{x}), \qquad (1)$$

where $\mathbf{A} \in \mathbb{R}^{k \times d}$ as above, and sign($\cdot$) is a binary map which returns element-wise sign[1]. How should one pick the matrix $\mathbf{A}$? One natural choice, in light of the Johnson-Lindenstrauss lemma, is to pick it randomly, i.e., each entry is sampled from an independent Gaussian. This *data oblivious* choice is well studied (Charikar, 2002; Raginsky and Lazebnik, 2009),

---

1. A few methods transform the linear projection via a nonlinear map before taking the sign (Weiss et al., 2008; Raginsky and Lazebnik, 2009).

and has the nice property that for two data vectors $\mathbf{x}, \mathbf{y}$, the ℓ_1 distance between their embeddings is proportional to the angle between $\mathbf{x}$ and $\mathbf{y}$, in expectation (over the random entries in $\mathbf{A}$). This is a consequence of the fact that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, if $\mathbf{r}$ is drawn from $\mathcal{N}(0, 1)^d$,

$$\Pr[\text{sign}\, \langle \mathbf{x}, \mathbf{r} \rangle = \text{sign}\, \langle \mathbf{y}, \mathbf{r} \rangle] = \frac{\angle(\mathbf{x}, \mathbf{y})}{\pi}. \tag{2}$$

Other data oblivious methods have also been studied in the literature, by choosing different distributions for the entries of $\mathbf{A}$. While these methods do reasonably well in practice, the natural question is if adapting the matrix to the data allows us to use shorter codes (i.e., have a smaller k) while achieving a similar error. A number of such data-dependent techniques have been proposed with different optimization criteria such as reconstruction error (Kulis and Darrell, 2009), data dissimilarity (Norouzi and Fleet, 2012; Weiss et al., 2008), ranking loss (Norouzi et al., 2012), quantization error after PCA (Gong et al., 2012a), and pairwise misclassification (Wang et al., 2010). As long as data is relatively low dimensional, these methods have been shown to be quite effective for learning compact codes.

However, the $\mathcal{O}(kd)$ barrier on the space and time complexity barrier prevents them from being applied with very high-dimensional data. For instance, to generate 10K-bit binary codes for data with 1M dimensions, a huge projection matrix will be required needing tens of GB of memory.[2]

In order to overcome these computational challenges, Gong et al. (2013) proposed a *bilinear projection* based coding method. The main idea here is to reshape the input vector $\mathbf{x}$ into a matrix $\mathbf{Z}$, and apply a bilinear projection to get the binary code:

$$h(\mathbf{x}) = \text{sign}(\mathbf{R}_1^T \mathbf{Z} \mathbf{R}_2). \tag{3}$$

When the shapes of $\mathbf{Z}, \mathbf{R}_1, \mathbf{R}_2$ are chosen appropriately[3], the method has time and space complexities $\mathcal{O}(d\sqrt{k})$ and $\mathcal{O}(\sqrt{dk})$ respectively. Bilinear codes make it feasible to work with data sets of very high dimensionality and have shown good results for a variety of tasks.

1.1 Our results

In this work, we propose a novel technique, called Circulant Binary Embedding (CBE), which is even faster than the bilinear coding. The main idea is to impose a *circulant* (described in detail in Section 3) structure on the projection matrix $\mathbf{A}$ in (1). This special structure allows us to compute the product $\mathbf{A}\mathbf{x}$ in time $\mathcal{O}(d \log d)$ using the Fast Fourier Transform (FFT), a tool of great significance in signal processing. The space complexity is also just $\mathcal{O}(d)$, making it efficient even for very high dimensional data. Table 1 compares the time and space complexity for the various methods outlined above.

Given the efficiency of computing the CBE, two natural questions arise: how good is the obtained embedding for various information retrieval tasks? and how should we pick the parameters of the circulant $\mathbf{A}$?

In Section 4, we study the first question for *random* CBE, i.e., when the parameters of the circulant are picked randomly (independent Gaussian, followed by its shifts). Specifically,

---

2. In the oblivious case, one can generate the random entries of the matrix on-the-fly (with fixed seeds) without needing to store the matrix, but this increases the computational time even further.

3. Specifically, $\mathbf{Z} \in \mathbb{R}^{\sqrt{d} \times \sqrt{d}}$, $\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{R}^{\sqrt{k} \times \sqrt{d}}$.

Method	Time	Space	Time (optimization)
Unstructured	$\mathcal{O}(dk)$	$\mathcal{O}(dk)$	$\mathcal{O}(Nd^2k)$
Bilinear	$\mathcal{O}(d\sqrt{k})$	$\mathcal{O}(\sqrt{dk})$	$\mathcal{O}(Nd\sqrt{k})$
Circulant $(k \leq d)$	$\mathcal{O}(d\log d)$	$\mathcal{O}(d)$	$\mathcal{O}(Nd\log d)$
Circulant $(k > d)$	$\mathcal{O}(k\log d)$	$\mathcal{O}(k)$	$\mathcal{O}(Nk\log d)$

Table 1: Comparison of the time and space complexities. d is the input dimensionality, and k is the output dimensionality (number of bits). N is the number of instances used for learning data-dependent projection matrices. See Section 3.3 for discussions on $k < d$ and $k > d$.

we analyze the *angle estimating* property of binary embeddings (Eq.(1)), which is the basis for its use in applications. Under mild assumptions, we show that using a random circulant **A** has the same qualitative guarantees as using fully random **A**. These results provide some of the few theoretical guarantees we are aware of, for non-linear circulant-based embeddings. We defer the formal statements of our results to Section 4, Theorems 3 and 4. We note that in independent and very recent work, Choromanska et al. (Choromanska et al., 2016) obtain a qualitatively similar analysis of CBE, however the bounds are incomparable to ours.

In Section 5, we study the second question, i.e., learning data-dependent circulant matrices. We propose a novel and efficient algorithm, which alternatively optimizes the objective in the original and frequency domains.

Finally in Section 7, we study the empirical performance of circulant embeddings via extensive experimentation. Compared to the state-of-the-art, our methods improve the performance dramatically for a fixed computation time. If we instead fix the number of bits in the embedding, we observe that the performance degradation is negligible, while speeding up the computation many-fold (see Section 7).

2. Background and related work

The lemma of Johnson and Lindenstrauss (Johnson and Lindenstrauss, 1984) is a fundamental tool in the area of sketching and dimension reduction. The lemma states that if we have N points in d-dimensional space, projecting them to an $\mathcal{O}(\log N)$ dimensional space (independent of d!) preserves all pairwise distances. One way of doing the projection is by a random Gaussian matrix. Formally,

Lemma 1 (Johnson Lindenstrass lemma) *Let S be a set of N points in $\mathbb{R}^d$. Let $\mathbf{A} \in \mathbb{R}^{k \times d}$ be a matrix whose entries are drawn i.i.d from $\mathcal{N}(0,1)$. Then with probability at least $1 - 2N^2 e^{-(\epsilon^2-\epsilon^3)k/4}$*

$$(1-\epsilon)\|\mathbf{x}-\mathbf{y}\|_2 \leq \frac{1}{\sqrt{k}}\|\mathbf{A}(\mathbf{x}-\mathbf{y})\|_2 \leq (1+\epsilon)\|\mathbf{x}-\mathbf{y}\|_2 \tag{4}$$

for any $\mathbf{x}, \mathbf{y} \in S$.

When $k = \mathcal{O}(\log N/\epsilon^2)$, the probability above can be made arbitrarily close to 1. The result has also been shown to be optimal even if the projection can be chosen data-dependently (Larsen and Nelson, 2016). Due to the simplicity and theoretical support, random projection based dimensionality reduction has been applied in broad applications including approximate nearest neighbor research (Indyk and Motwani, 1998), dimensionality reduction in databases (Achlioptas, 2003), and bi-Lipschitz embeddings of graphs into normed spaces (Frankl and Maehara, 1988).

However a serious concern in a few applications is the dependence of k on the accuracy ($\mathcal{O}(1/\epsilon^2)$). The space and time complexity of dimension reduction are $\mathcal{O}(kd)$, if the computation is done in the natural way. Are there faster methods when k is reasonably large? As mentioned earlier, the line of work starting with (Ailon and Chazelle, 2006) aims to improve the time and space complexity of dimension reduction. This led to work showing Johnson-Lindenstruss-type guarantees with *structured* matrices (with some randomness), including Hadamard matrices along with a sparse random Gaussian matrix (Ailon and Chazelle, 2006), sparse matrices (Matoušek, 2008), and Lean Walsh Transformations (Liberty et al., 2008). The advantage of using structured matrices is that the space and computation cost can be dramatically reduced, yet the distance preserving property remains to be competitive.

In this context, randomized circulant matrices (which are also the main tool in our work) have been studied, starting with the works (Hinrichs and Vybíral, 2011; Vybíral, 2011). The dimension reduction comprises of random sign flips followed by multiplication by a randomized circulant matrix. For d-dimensional input, reducing the dimension to k for $k < d$ has time complexity $\mathcal{O}(d \log d)$ and space complexity $\mathcal{O}(d)$, independent of k. Proving bounds similar to Lemma 1 turns out to be much more challenging because the entries of the projection matrix are now highly dependent, and thus concentration bounds are hard to prove. The first analysis (Hinrichs and Vybíral, 2011) showed that reducing to $\mathcal{O}(\log^3 N/\epsilon^2)$ dimensions (compared to $\mathcal{O}(\log N/\epsilon^2)$ in Lemma 1) preserves all pairwise distances with high probability. This was improved to $\mathcal{O}(\log^2 N/\epsilon^2)$ in (Vybíral, 2011), and furthermore to $\mathcal{O}(\log^{(1+\delta)} N/\epsilon^2)$ in (Zhang and Cheng, 2013), using matrix-valued Bernstein inequalities. These works provide the motivation for our theoretical results, however the key difference for us is the *binarization* step, which is highly non-linear. Thus we need to develop new machinery for our analysis.

Binary embeddings. Recently, structured matrices used in the context of the fast JL transform (a combination of Hadamard and sparse random Gaussian matrices) have also been studied for binary embedding (Dasgupta et al., 2011), and more recently (Yi et al., 2015). In particular, Yi et al. (2015) showed that the method can achieve ϵ distance preserving error with $\mathcal{O}(\log N/\epsilon^2)$ bits and $\mathcal{O}(d \log d)$ computational complexity, for N points ($N \ll \epsilon\sqrt{d}$). In this work, we study the application of using the circulant matrix for binary embedding. The work extends and provides theoretical justification for our previous conference paper on this topic (Yu et al., 2014).

The idea of using structured matrices to speed up linear projection has also be exploited under the settings of deep neural networks (Cheng et al., 2015; Yang et al., 2015), and kernel approximation (Yu et al., 2015; Le et al., 2013) .

3. Circulant Binary Embedding

Let us start by describing our framework and setting up the notation that we use in the rest of the paper.

3.1 The Framework

We will now describe our algorithm for generating k-bit binary codes from d-dimensional real vectors. We start by discussing the case $k = d$ and move to the general case in Section 3.3. The key player is the circulant matrix, which is defined by a real vector $\mathbf{r} = (r_0, r_1, \cdots, r_{d-1})^T$ (Gray, 2006).

$$
\mathcal{C}_{\mathbf{r}} := \begin{bmatrix}
r_0 & r_{d-1} & \cdots & r_2 & r_1 \\
r_1 & r_0 & r_{d-1} & & r_2 \\
\vdots & r_1 & r_0 & \ddots & \vdots \\
r_{d-2} & & \ddots & \ddots & r_{d-1} \\
r_{d-1} & r_{d-2} & \cdots & r_1 & r_0
\end{bmatrix}. \tag{5}
$$

Let $\mathbf{D}$ be a diagonal matrix with each diagonal entry σ_i, $i = 0, \cdots, d-1$, being a Rademacher variable (± 1 with probability $1/2$):

$$
\mathbf{D} = \begin{bmatrix}
\sigma_0 & & & & \\
& \sigma_1 & & & \\
& & \sigma_2 & & \\
& & & \ddots & \\
& & & & \sigma_{d-1}
\end{bmatrix}. \tag{6}
$$

For $\mathbf{x} \in \mathbb{R}^d$, its d-bit Circulant Binary Embedding (CBE) with $\mathbf{r} \in \mathbb{R}^d$ is defined as:

$$
h(\mathbf{x}) = \text{sign}(\mathcal{C}_{\mathbf{r}} \mathbf{D} \mathbf{x}), \tag{7}
$$

where $\mathcal{C}_{\mathbf{r}}$ is defined as above. Note that applying $\mathbf{D}$ to $\mathbf{x}$ is equivalent to applying a random sign flip to each coordinate of $\mathbf{x}$. The necessity of such an operation is discussed in the introduction of Section 4. Since sign flipping can be carried out as a preprocessing step for each input $\mathbf{x}$, here onwards for simplicity we will drop explicit mention of $\mathbf{D}$. Hence the binary code is given as $h(\mathbf{x}) = \text{sign}(\mathcal{C}_{\mathbf{r}} \mathbf{x})$.

3.2 Computational Complexity

The main advantage of a circulant based embedding is that it can be computed quickly using the Fast Fourier Transform (FFT). The following is a folklore result, whose proof we include for completeness.

Proposition 2 *For a d-dimensional vector $\mathbf{x}$ and any $\mathbf{r} \in \Re^d$, the d-bit CBE $\text{sign}(\mathcal{C}_r(\mathbf{D}\mathbf{x}))$ can be computed using $\mathcal{O}(d)$ space and $\mathcal{O}(d \log d)$ time.*

Proof The space complexity comes only from the storage of the vector $\mathbf{r}$ and the signs $\mathbf{D}$ (which amount to $\mathcal{O}(d)$). We never need to store the full matrix $\mathcal{C}_{\mathbf{r}}$ explicitly.

The main property of a circulant matrix is that for any vector $\mathbf{y} \in \mathbf{R}^d$, we can compute $\mathcal{C}_{\mathbf{r}}\mathbf{y}$ in time $\mathcal{O}(d \log d)$. This is because

$$\mathcal{C}_{\mathbf{r}} = \mathcal{F}_d^{-1} \operatorname{diag}(\mathcal{F}_d \mathbf{r}) \, \mathcal{F}_d, \tag{8}$$

where $\mathcal{F}_d$ is the matrix corresponding to the Discrete Fourier Transform (DFT) of periodicity N, i.e., whose (i, j)th entry is given by

$$\mathcal{F}_d(i, j) = \omega^{ij}, \tag{9}$$

where ω is the Nth root of unity $e^{-2\pi\iota/N}$. The celebrated Fast Fourier Transform algorithm (Oppenheim et al., 1999) says that for any $\mathbf{z} \in \mathbf{R}^d$, we can compute $\mathcal{F}_d\mathbf{z}$ and $\mathcal{F}_d^{-1}\mathbf{z}$ in time $\mathcal{O}(d \log d)$, using $\mathcal{O}(d)$ space. This immediately implies that we can compute $\mathcal{C}_{\mathbf{r}}\mathbf{y}$ within the same space and time complexity bounds. ∎

3.3 Generalizing to $k \neq d$

The computation above assumed that number of bits we produce (k) is equal to the input dimension. Let us now consider the general case.

When $k < d$, we still use the circulant matrix $\mathbf{R} \in \mathbb{R}^{d \times d}$ with d parameters, but the output is set to be the first k elements in (7). This is equivalent to the operation

$$\Phi(\mathbf{x}) := \operatorname{sign}(\mathcal{C}_{\mathbf{r},k}\mathbf{D}\mathbf{x}), \tag{10}$$

where $\mathcal{C}_{\mathbf{r},k}$ the so-called *partial circulant matrix*, which is $\mathcal{C}_{\mathbf{r}}$ truncated to k rows. We note that CBE with $k < d$ is not computationally more efficient than that with $k = d$.

When $k > d$, using a single $\mathbf{r}$ causes repetition of bits, so we propose using $\mathcal{C}_{\mathbf{r}}$ for multiple $\mathbf{r}$, and concatenating their output. This gives the computational complexity $\mathcal{O}(k \log d)$, and space complexity $\mathcal{O}(k)$. Note that as the focus of this paper is on binary embedding on high-dimensional data, from here onwards, we assume $k \leq d$. The $k > d$ case is useful in other applications such as neural network (Cheng et al., 2015) and kernel approximation (Yu et al., 2015).

3.4 Choosing the Parameters r

We have presented the general framework as well as its space and computation efficiency in this section. One critical question left unanswered is how to decide the parameter $\mathbf{r}$. As mentioned in the introduction, we consider two solutions. In Section 4, we study the randomized version, where each element of $\mathbf{r}$ is independently sampled from a unit Gaussian distribution. This is inspired by the popular Locality Sensitive Hashing (simhash) approach. Section 5 introduces an optimized version, where the parameters are optimized based on training data and an distance preserving objective function.

4. Randomized CBE – A Theoretical Analysis

We now analyze the angle preserving properties of CBE when the circulant matrix used is generated from a random d-dimensional vector. Formally, we consider the partial circulant

matrix $\mathcal{C}_{\mathbf{r},k}$, for $\mathbf{r} \sim \mathcal{N}(0,1)^d$. The embedding we consider for an $\mathbf{x} \in \mathbb{R}^d$ is given by

$$\Phi(\mathbf{x}) := \text{sign}(\mathcal{C}_{\mathbf{r},k}\mathbf{D}\mathbf{x}). \tag{11}$$

As before, $\mathbf{D}$ is a diagonal matrix of signs. Hence the embedding uses $2d$ independent 'units' of randomness.

Now, for any two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have that

$$\mathbb{E}\left[\frac{1}{2k}\|\Phi(\mathbf{x}) - \Phi(\mathbf{y})\|_1\right] = \frac{\angle(\mathbf{x}, \mathbf{y})}{\pi}, \tag{12}$$

implying that the random variable $(1/2k)\|\Phi(\mathbf{x}) - \Phi(\mathbf{y})\|_1$ provides an estimate for θ/π, where $\theta := \angle(\mathbf{x}, \mathbf{y})$.

We present two main results. In the first, we bound the variance of the above angle estimate for given $\mathbf{x}, \mathbf{y}$. We compare with the variance in the *fully independent* case, i.e., when we consider the embedding $\text{sign}(\mathbf{A}\mathbf{x})$, where $\mathbf{A}$ is a $k \times d$ matrix with all entries being independent (and unit normal). In this case, the variance of the estimator in Eq. (12) is equal to $\frac{1}{k}\frac{\theta}{\pi}\left(1 - \frac{\theta}{\pi}\right)$.[4]

We show that using a circulant matrix instead of A above has a similar dependence on k, as long as the vectors are *well spread*. Formally,

Theorem 3 *Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, such that $\max\{\|\mathbf{x}\|_\infty/\|\mathbf{x}\|_2, \|\mathbf{y}\|_\infty/\|\mathbf{y}\|_2\} \leq \rho$, for some parameter $\rho < 1$, and set $\theta = \angle(\mathbf{x}, \mathbf{y})$. The variance of the averaged hamming distance of k-bit code generated by randomized CBE is*

$$var\left[\frac{1}{2k}\|\Phi_x - \Phi_y\|_1\right] \leq \frac{1}{k}\frac{\theta}{\pi}\left(1 - \frac{\theta}{\pi}\right) + 32\rho. \tag{13}$$

The variance above is over the choice of $\mathbf{r}$ and the random signs $\mathbf{D}$.

Remark. For typical vectors in $\mathbb{R}^d$, we have $\|\mathbf{x}\|_\infty/\|\mathbf{x}\|_2$ to be $O(\log d/\sqrt{d})$. Further, by using the idea from Ailon and Chazelle (Ailon and Chazelle, 2006), we can pre-process the data by multiplying it with a randomly signed Hadamard matrix, and guarantee such an ℓ_∞ bound with high probability.[5] Therefore the second term becomes negligible for large d. The above result suggests that the angle preservation performance of CBE (in term of the variance) is as good as LSH for high-dimensional data.

Our second theorem gives a large-deviation bound for the angle estimate, also assuming that the vectors are well-spread. This will then enable us to obtain a dimension reduction theorem which preserves all angles up to an additive error.

Theorem 4 *Let $\mathbf{x}, \mathbf{y} \in \mathbf{R}^d$ with $\angle(\mathbf{x}, \mathbf{y}) = \theta$, and suppose $\max\{\|\mathbf{x}\|_\infty/\|\mathbf{x}\|_2, \|\mathbf{y}\|_\infty/\|\mathbf{x}\|_2\} \leq \rho$, for some parameter ρ. Now consider the k-dimensional CBE Φ_x, Φ_y of $\mathbf{x}, \mathbf{y}$ respectively, for some $k < d$. Suppose $\rho \leq \frac{\theta^2}{16k\log(k/\delta)}$. For any $\delta > 0$, we have:*

$$\Pr\left[\left|\frac{1}{2k}\|\Phi_x - \Phi_y\|_1 - \frac{\theta}{\pi}\right| > \frac{4\log(k/\delta)}{\sqrt{k}}\right] < \delta. \tag{14}$$

---

4. We are computing the variance of an average of i.i.d. Bernoulli random variables which take value 1 with probability $p = \theta/\pi$.

5. However, applying this pre-processing leads to *dense* vectors, which may be memory intensive for some applications. In this case, dividing the co-ordinates into blocks of size $\sim k^2$ and performing the pre-processing on the blocks separately is better for small k.

Qualitatively, the condition on ρ is similar to the one we implicitly have in Theorem 3. Unless $\rho = o\left(\frac{1}{k}\frac{\theta}{\pi}(1 - \frac{\theta}{\pi})\right)$, the additive term dominates, so for the bound to be interesting, we need this condition on ρ.

We observe that Theorem 4 implies a Johnson-Lindenstrauss type theorem.

Corollary 5 *Suppose we have N vectors $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}$ in $\mathbf{R}^d$, and define*

$$\rho_{ij} = \max\{\|\mathbf{u}_i\|_\infty/\|\mathbf{u}_i\|_2, \|\mathbf{u}_j\|_\infty/\|\mathbf{u}_j\|_2\}, \quad \theta_{ij} = \angle(\mathbf{u}_i, \mathbf{u}_j). \tag{15}$$

Let $\epsilon > 0$ be a given accuracy parameter and let $k = C\log^2 n/\epsilon^2$. Then for all i, j such that $\rho_{ij} < \frac{\theta_{ij}^2}{16k\log(2kN^2)}$, we have

$$\left| \frac{1}{2k}\|\Phi_i - \Phi_j\|_1 - \frac{\theta_{ij}}{\pi} \right| < \epsilon, \tag{16}$$

with probability at least $3/4$.

Proof We can set $\delta = 1/2N^2$ in Theorem 4 and then take a union bound over all $\binom{N}{2}$ choices of pairs i, j to obtain a failure probability $\leq 1/4$. Further, for our choice of k, setting $C = 144$ and assuming N is large enough that $k < N$, we have

$$\frac{4\log(k/\epsilon)}{\sqrt{k}} < \frac{12\delta\log N}{\sqrt{C}\cdot\log N} < \epsilon. \tag{17}$$

∎

In the remainder of the section, we will prove the above theorems. We start with Theorem 3, whose proof will give a basic framework for that of Theorem 4.

4.1 Variance of the angle estimator

For a vector $\mathbf{x}$ and an index i, we denote by $s_{\to i}(\mathbf{x})$ the vector *shifted* by i positions: the jth entry of $s_{\to i}$ is the $((j-i)\bmod d)$'th entry of $\mathbf{x}$. Further, let us define

$$F_i = \frac{1 - \text{sign}(s_{\to i}(\mathbf{r})^T\mathbf{Dx})\,\text{sign}(s_{\to i}(\mathbf{r})^T\mathbf{Dy})}{2} - \frac{\theta}{\pi}. \tag{18}$$

where $s_{\to i}(\cdot)$ is defined as the operator circularly shifting a vector by i elements[6]. By definition, we have

$$var\left[\frac{1}{2k}\|\Phi_x - \Phi_y\|_1\right] = var\left[\frac{1}{k}\sum_{i=1}^{k}F_i\right]. \tag{19}$$

---

6. The above comes with a slight abuse of notation, where the first column (instead of row) of the projection matrix $\mathbf{R}$ is defined as $\mathbf{r}$.

Without loss of generality, we assume $\|\mathbf{x}\|_2, \|\mathbf{y}\|_2 = 1$ (since we only care about the angle). The mean of each F_i is zero, and thus $\mathbb{E}[\frac{1}{k}\sum_{i=1}^{k} F_i] = 0$. Thus the variance is equal to

$$var\left[\frac{1}{k}\sum_{i=0}^{k-1} F_i\right] = \mathbb{E}\left[\frac{1}{k^2}\left(\sum_{i=0}^{k-1} F_i\right)^2\right] \tag{20}$$

$$= \mathbb{E}\left[\frac{\sum_{i=0}^{k-1} F_i^2 + \sum_{i\neq j} F_i F_j}{k^2}\right]$$

$$= \frac{1}{k^2}\left(k \cdot \mathbb{E}F_1^2 + \sum_{i\neq j}\mathbb{E}(F_i F_j)\right)$$

$$= \frac{1}{k}\frac{\theta}{\pi}\left(1 - \frac{\theta}{\pi}\right) + \frac{1}{k^2}\sum_{i\neq j}\mathbb{E}(F_i F_j). \tag{21}$$

The last step is based on variance of the fully independent case: the variance of a Bernoulli random variable which takes value 1 with probability $p = \theta/\pi$ is $\frac{\theta}{\pi}\left(1 - \frac{\theta}{\pi}\right)$.

To prove the theorem, it suffices to show that $\mathbb{E}(F_i F_j) \leq 32\rho$ for all $i \neq j$. Without loss of generality, we can assume that $i = 0$, and consider $\mathbb{E}(F_0 F_j)$. By definition, it is equal to

$$\mathbb{E}\left[\left(\frac{1 - \text{sign}(\mathbf{r}^T\mathbf{Dx})\,\text{sign}(\mathbf{r}^T\mathbf{Dy})}{2} - \frac{\theta}{\pi}\right)\left(\frac{1 - \text{sign}(s_{\to j}(\mathbf{r})^T\mathbf{Dx})\,\text{sign}(s_{\to j}(\mathbf{r})^T\mathbf{Dy})}{2} - \frac{\theta}{\pi}\right)\right].$$

The trick now is to observe that

$$s_{\to j}(\mathbf{r})^T\mathbf{x} = \mathbf{r}^T s_{\to(d-j)}(\mathbf{x}). \tag{22}$$

Thus setting $t = d - j$, we can write the above as

$$\mathbb{E}\left[\left(\frac{1 - \text{sign}(\mathbf{r}^T\mathbf{Dx})\,\text{sign}(\mathbf{r}^T\mathbf{Dy})}{2} - \frac{\theta}{\pi}\right)\left(\frac{1 - \text{sign}(\mathbf{r}^T s_{\to t}(\mathbf{Dx}))\,\text{sign}(\mathbf{r}^T s_{\to t}(\mathbf{Dy}))}{2} - \frac{\theta}{\pi}\right)\right].$$

The key idea is that we expect the vector $s_{\to t}(\mathbf{Dx})$ to be nearly orthogonal to the space containing $\mathbf{Dx}, \mathbf{Dy}$. This is because $\mathbf{D}$ is a diagonal matrix of random signs, and $\mathbf{x}$ and $\mathbf{y}$ are vectors with small ℓ_∞ norm. We show this formally in Lemma 7.

Why does this help? Suppose for a moment that $\mathbf{u} := s_{\to t}(\mathbf{Dx})$ and $\mathbf{v} := s_{\to t}(\mathbf{Dy})$ are both orthogonal to $\text{span}\{\mathbf{Dx}, \mathbf{Dy}\}$. Then for a random Gaussian $\mathbf{r}$, the random variables $\text{sign}(\mathbf{r}^T\mathbf{u})\,\text{sign}(\mathbf{r}^T\mathbf{v})$ and $\text{sign}(\mathbf{r}^T\mathbf{Dx})\,\text{sign}(\mathbf{r}^T\mathbf{Dy})$ are independent, because the former depends only on the projection of $\mathbf{r}$ onto $\text{span}\{\mathbf{u}, \mathbf{v}\}$, while the latter depends only on the projection of $\mathbf{r}$ onto $\text{span}\{\mathbf{Dx}, \mathbf{Dy}\}$. Now if these two spaces are orthogonal, the projections of a Gaussian vector onto these spaces are independent (this is a fundamental property of multidimensional Gaussians). This implies that the expectation of the product above is equal to the product of the expectations, which is zero (each expectation is zero).

The key lemma (see below) now says that even if $\mathbf{u}$ and $\mathbf{v}$ as defined above are *nearly* orthogonal to $\text{span}\{\mathbf{Dx}, \mathbf{Dy}\}$, we still get a good bound on the expectation above.

10

Lemma 6 *Let* $\mathbf{a}, \mathbf{b}, \mathbf{u}, \mathbf{v}$ *be unit vectors in* $\mathbf{R}^d$ *such that* $\angle(\mathbf{a}, \mathbf{b}) = \angle(\mathbf{u}, \mathbf{v}) = \theta$, *and let* Π *be the projector onto* $span\{\mathbf{a}, \mathbf{b}\}$. *Suppose* $\max\{\|\Pi\mathbf{u}\|, \|\Pi\mathbf{v}\|\} = \delta < 1$. *Then we have*

$$\mathbb{E}\left[\left(\frac{1 - \text{sign}(\mathbf{r}^T\mathbf{a})\,\text{sign}(\mathbf{r}^T\mathbf{b})}{2} - \frac{\theta}{\pi}\right)\left(\frac{1 - \text{sign}(\mathbf{r}^T\mathbf{u})\,\text{sign}(\mathbf{r}^T\mathbf{v})}{2} - \frac{\theta}{\pi}\right)\right] \leq 2\delta. \qquad (23)$$

Here, the expectation is over the choice of $\mathbf{r}$.

The proof of the above lemma is moved to Appendix A.1.

We use the lemma with $\mathbf{a} = \mathbf{D}\mathbf{x}$ and $\mathbf{b} = \mathbf{D}\mathbf{y}$. To show Theorem 3, we have to prove that

$$\mathbb{E}\left[\max\{\|\Pi\mathbf{u}\|, \|\Pi\mathbf{v}\|\}\right] \leq 16\rho, \qquad (24)$$

where $\Pi, \mathbf{u}, \mathbf{v}$ are defined as in the statement of Lemma 6. The expectation now is over the choice of $\mathbf{D}$. This leads us to our next lemma.

Lemma 7 *Let* $\mathbf{p}, \mathbf{q} \in \mathbb{R}^d$ *be vectors that satisfy* $\|\mathbf{p}\|_2 = 1$ *and* $\|\mathbf{q}\|_\infty < \rho$ *for some parameter* ρ, *and suppose* $\mathbf{D} := diag(\sigma_0, \sigma_1, \ldots, \sigma_{d-1})$, *where* σ_i *are random* ± 1 *signs. Then for any* $0 < t < d$, *we have*

$$\Pr[\langle \mathbf{D}\mathbf{p}, s_{\to t}(\mathbf{D}\mathbf{q})\rangle > \gamma] \leq e^{-\gamma^2/8\rho^2}. \qquad (25)$$

Note that the probability is over the choice of $\mathbf{D}$.

The proof of the above lemma is moved to Appendix A.2. We remark that the lemma only assumes that $\mathbf{p}$ is a unit vector, it need not have a small ℓ_∞ norm.

We can now complete the proof of our theorem. As noted above, we need to show (24). To recall, Π is the projector onto $span\{\mathbf{D}\mathbf{x}, \mathbf{D}\mathbf{y}\}$, and we need to bound:

$$\mathbb{E}\left[\max\{\|\Pi\mathbf{u}\|, \|\Pi\mathbf{v}\|\}\right] \leq \mathbb{E}[\|\Pi\mathbf{u}\|] + \mathbb{E}[\|\Pi\mathbf{v}\|]. \qquad (26)$$

Let $\mathbf{x}, \mathbf{z}$ be an orthonormal basis for $span\{\mathbf{x}, \mathbf{y}\}$; then it is easy to see that for any diagonal $\mathbf{D}$ with ± 1 entries on the diagonal, $\mathbf{D}\mathbf{x}, \mathbf{D}\mathbf{z}$ is an orthonormal basis for $span\{\mathbf{D}\mathbf{x}, \mathbf{D}\mathbf{y}\}$. Thus

$$\mathbb{E}[\|\Pi\mathbf{u}\|] \leq \mathbb{E}[|\langle \mathbf{u}, \mathbf{D}\mathbf{x}\rangle| + |\langle \mathbf{u}, \mathbf{D}\mathbf{z}\rangle|]. \qquad (27)$$

Now by Lemma 7,

$$\Pr[|\langle \mathbf{u}, \mathbf{D}\mathbf{x}\rangle| > t\rho] \leq e^{-t^2/4}. \qquad (28)$$

Integrating over t, we get $\mathbb{E}[|\langle \mathbf{u}, \mathbf{D}\mathbf{x}\rangle|] \leq 4\rho$. Thus we can bound the LHS of (26) by 16ρ, completing the proof of the theorem.

4.2 The Johnson-Lindenstrauss Type Result

Next, we turn to the proof of Theorem 4, where we wish to obtain a strong tail bound. At a high level, the argument consists of two steps:

- First, show that with probability $1 - \epsilon$ over the choice of $\mathbf{D}$, the k translates of $\mathbf{x}, \mathbf{y}$ satisfy certain orthogonality properties (this is in the same spirit as Lemma 7).
- Second, conditioned on orthogonality as above, with high probability over the choice of $\mathbf{r}$, we have the desired guarantee.

Next will will show the two steps respectively. Throughout this section, we denote by $X_0, X_1, \ldots, X_{k-1}$ the k shifts of $\mathbf{Dx}$, i.e., $X_i = s_{\rightarrow(i)}(\mathbf{Dx})$; define $Y_0, \ldots, Y_{k-1}$ analogously as shifts of $\mathbf{Dy}$. We will also assume that $\rho < \frac{\theta^2}{16k \log(k/\delta)}$.

The structure we require is formally the following.

Definition 8 ((γ, k)-orthogonality) *Two sequences of k unit vectors $X_0, X_1, \ldots, X_{k-1}$ and $Y_0, Y_1, \ldots, Y_{k-1}$ are said to be (γ, k)-orthogonal if there exists a decomposition (for every i)*

$$X_i = \mathbf{u}_i + \mathbf{e}_i \; ; \quad Y_i = \mathbf{v}_i + \mathbf{f}_i \tag{29}$$

satisfying the following properties:
1. *$\mathbf{u}_i$ and $\mathbf{v}_i$ are both orthogonal to $span\{\mathbf{u}_j, \mathbf{v}_j : j \neq i\}$.*
2. *$\max_i\{\|\mathbf{e}_i\|, \|\mathbf{f}_i\|\} < \gamma$.*

The lemma of the first step, as described earlier, is the following:

Lemma 9 *Let $\mathbf{x}, \mathbf{y}$ be unit vectors with $\|\mathbf{x}\|_\infty, \|\mathbf{y}\|_\infty \leq \rho$, and $\theta = \angle(\mathbf{x}, \mathbf{y})$, and let X_i, Y_i be rotations of $\mathbf{Dx}, \mathbf{Dy}$ respectively (as defined earlier). Then w.p. $1 - \delta$ over the choice of $\mathbf{D}$, the vectors $(X_i, Y_i)_{i=1}^k$ are (γ, k) orthogonal, for $\gamma = 4\sqrt{\rho}$.*

The proof of the lemma is quite technical, and is moved to Appendix A.3.

Now suppose we have that the shifts X_i, Y_i satisfy (γ, k)-orthogonality for some $\gamma > 0$. Suppose $\mathbf{u}_i, \mathbf{v}_i, \mathbf{e}_i, \mathbf{f}_i$ are as defined earlier. (γ, k)-orthogonality gives us that $\|\mathbf{e}_i\|, \|\mathbf{f}_i\| < \gamma$, which is $\ll 1$. Roughly speaking, we use this to say that *most of the time*, $\mathrm{sign}(\langle \mathbf{r}, X_i \rangle) = \langle \mathbf{r}, \mathbf{u}_i \rangle)$. Thus determining if $\mathrm{sign}(\langle \mathbf{r}, X_i \rangle) = \mathrm{sign}(\langle \mathbf{r}, Y_i \rangle)$ is essentially equivalent to determining if $\mathrm{sign}(\langle \mathbf{r}, \mathbf{u}_i \rangle) = \mathrm{sign}(\langle \mathbf{r}, \mathbf{v}_i \rangle)$. But the latter quantities, by orthogonality, are indepedent! (because the signs depend only on the projection of $\mathbf{r}$ onto the span of $\mathbf{u}_i, \mathbf{v}_i$, which is independent for different i).[7] The main lemma of the second step is the following:

Lemma 10 *Let $(X_i, Y_i)_{i=1}^k$ be a set of vectors satisfying (γ, k)-orthogonality and $\angle(X_i, Y_i) = \theta$ for all i. Then for any $\delta > 0$ and $k > \max\{1/\gamma, \log(4/\delta)\}$, we have*

$$\Pr\left[\left|\frac{1}{k}\sum_i (\mathrm{sign}\langle \mathbf{r}, X_i \rangle \neq \mathrm{sign}\langle \mathbf{r}, Y_i \rangle) - \frac{\theta}{\pi}\right| > \gamma \cdot (12 \log(2k/\delta))\right] < 1 - \delta. \tag{30}$$

The probability here is over the choice of $\mathbf{r}$.

The proof is deferred to Appendix A.4.

We can now complete the proof of Theorem 4. It essentially follows using Lemma 9 and Lemma 10. Note that we can apply Lemma 10 because the angle between X_i and Y_i is also θ for each i (since they are shifts of $\mathbf{x}, \mathbf{y}$).

Formally, using the value of γ defined in Lemma 9, we have that the vectors X_i, Y_i are (γ, k) orthogonal with probability $1 - \delta$. Conditioned on this, the probability that the conclusion of Lemma 10 holds with probability $1 - \delta$. Thus the overall probability of success is at least $1 - 2\delta$. The theorem is thus easily proved by plugging in the value of γ from Lemma 9, together with $\rho < 1$. This completes the proof of the Theorem.

---

7. Again, using the property of multi-variate Gaussians that the projections onto orthogonal directions are orthogonal.

5. Optimized Binary Embedding

In the previous section, we showed the randomized CBE has LSH-like angle preserving properties, especially for high-dimensional data. One problem with the randomized CBE method is that it does not utilize the underlying data distribution while generating the matrix $\mathbf{R}$. In the next section, we propose to learn $\mathbf{R}$ in a data-dependent fashion, to minimize the distortions due to circulant projection and binarization.

The study in this section is closely related to data-dependent binary embeddings (Wang et al., 2010; Gong et al., 2012a, 2013). Different from the above works, the projection matrix has a circulant structure instead of being unstructured. The method is also related to data-dependent dimensionality reduction techniques such as PCA. Although it has been shown that the Johnson Lindenstrauss bound is tight even in the data-dependent setting (Larsen and Nelson, 2016), it is expected that optimization can improve the result of CBE: CBE differs from conventional dimensionality reduction by using the circulant structure and binarization; the data dependent bound is based on a worst case scenario of the data, whereas we study the empirical performance on real large-scale data sets.

We propose data-dependent CBE (CBE-opt), by optimizing the projection matrix with a novel time-frequency alternating optimization. We consider the following objective function in learning the d-bit CBE. The extension of learning $k < d$ bits will be shown in Section 5.2.

$$\underset{\mathbf{B}, \mathbf{r}}{\operatorname{argmin}} \quad ||\mathbf{B} - \mathbf{X}\mathbf{R}^T||_F^2 + \lambda ||\mathbf{R}\mathbf{R}^T - \mathbf{I}||_F^2 \tag{31}$$

$$\text{s.t.} \quad \mathbf{R} = \operatorname{circ}(\mathbf{r}),$$

where $\mathbf{X} \in \mathbb{R}^{N \times d}$, is the data matrix containing n training points: $\mathbf{X} = [\mathbf{x}_0, \cdots, \mathbf{x}_{N-1}]^T$, and $\mathbf{B} \in \{-1, 1\}^{N \times d}$ is the corresponding binary code matrix.[8]

In the above optimization, the first term minimizes distortion due to binarization. The second term tries to make the projections (rows of $\mathbf{R}$, and hence the corresponding bits) as uncorrelated as possible. In other words, this helps to reduce the redundancy in the learned code. If $\mathbf{R}$ were to be an orthogonal matrix, the second term will vanish and the optimization would find the best rotation such that the distortion due to binarization is minimized. However, being a circulant matrix, $\mathbf{R}$, in general, will not be orthogonal[9]. A similar objective has been used in previous works including (Gong et al., 2012a, 2013) and (Wang et al., 2010).

5.1 The Time-Frequency Alternating Optimization

The above is a difficult non-convex combinatorial optimization problem. In this section we propose a novel approach to efficiently find a local solution. The idea is to alternatively optimize the objective by fixing $\mathbf{r}$, and $\mathbf{B}$, respectively. For a fixed $\mathbf{r}$, optimizing $\mathbf{B}$ can be easily performed in the input domain ("time" as opposed to "frequency"). For a fixed $\mathbf{B}$, the circulant structure of $\mathbf{R}$ makes it difficult to optimize the objective in the input domain.

---

8. If the data is ℓ_2 normalized, we can set $\mathbf{B} \in \{-1/\sqrt{d}, 1/\sqrt{d}\}^{N \times d}$ to make $\mathbf{B}$ and $\mathbf{X}\mathbf{R}^T$ more comparable. This does not empirically influence the performance.

9. We note that the rank of the circulant matrices can range from 1 (an all-1 matrix) to d (an identity matrix).

Hence we propose a novel method, by optimizing $\mathbf{r}$ in the frequency domain based on DFT. This leads to a very efficient procedure.

For a fixed r. The objective is independent on each element of $\mathbf{B}$. Denote B_{ij} as the element of the i-th row and j-th column of $\mathbf{B}$. It is easy to show that $\mathbf{B}$ can be updated as:

$$B_{ij} = \begin{cases} 1 & \text{if } \mathbf{R}_{j\cdot}\mathbf{x}_i \geq 0 \\ -1 & \text{if } \mathbf{R}_{j\cdot}\mathbf{x}_i < 0 \end{cases}, \tag{32}$$

$$i = 0, \cdots, N-1. \quad j = 0, \cdots, d-1.$$

For a fixed B. Define $\tilde{\mathbf{r}}$ as the DFT of the circulant vector $\tilde{\mathbf{r}} := \mathcal{F}(\mathbf{r})$. Instead of solving $\mathbf{r}$ directly, we propose to solve $\tilde{\mathbf{r}}$, from which $\mathbf{r}$ can be recovered by IDFT.

Key to our derivation is the fact that DFT projects the signal to a set of orthogonal basis. Therefore the ℓ_2 norm can be preserved. Formally, according to Parseval's theorem, for any $\mathbf{t} \in \mathbb{C}^d$ (Oppenheim et al., 1999),

$$||\mathbf{t}||_2^2 = (1/d)||\mathcal{F}(\mathbf{t})||_2^2. \tag{33}$$

Denote $\text{diag}(\cdot)$ as the diagonal matrix formed by a vector. Denote $\Re(\cdot)$ and $\Im(\cdot)$ as the real and imaginary parts, respectively. We use $\mathbf{B}_{i\cdot}$ to denote the i-th row of $\mathbf{B}$. With complex arithmetic, the first term in (31) can be expressed in the frequency domain as:

$$||\mathbf{B} - \mathbf{X}\mathbf{R}^T||_F^2 = \frac{1}{d} \sum_{i=0}^{N-1} ||\mathcal{F}(\mathbf{B}_{i\cdot}^T - \mathbf{R}\mathbf{x}_i)||_2^2 \tag{34}$$

$$= \frac{1}{d} \sum_{i=0}^{N-1} ||\mathcal{F}(\mathbf{B}_{i\cdot}^T) - \tilde{\mathbf{r}} \circ \mathcal{F}(\mathbf{x}_i)||_2^2 = \frac{1}{d} \sum_{i=0}^{N-1} ||\mathcal{F}(\mathbf{B}_{i\cdot}^T) - \text{diag}(\mathcal{F}(\mathbf{x}_i))\tilde{\mathbf{r}}||_2^2$$

$$= \frac{1}{d} \sum_{i=0}^{N-1} \left(\mathcal{F}(\mathbf{B}_{i\cdot}^T) - \text{diag}(\mathcal{F}(\mathbf{x}_i))\tilde{\mathbf{r}}\right)^T \left(\mathcal{F}(\mathbf{B}_{i\cdot}^T) - \text{diag}(\mathcal{F}(\mathbf{x}_i))\tilde{\mathbf{r}}\right)$$

$$= \frac{1}{d} \left[\Re(\tilde{\mathbf{r}})^T \mathbf{M} \Re(\tilde{\mathbf{r}}) + \Im(\tilde{\mathbf{r}})^T \mathbf{M} \Im(\tilde{\mathbf{r}}) + \Re(\tilde{\mathbf{r}})^T \mathbf{h} + \Im(\tilde{\mathbf{r}})^T \mathbf{g} \right] + ||\mathbf{B}||_F^2,$$

where,

$$\mathbf{M} = \text{diag}\Big(\sum_{i=0}^{N-1} \Re(\mathcal{F}(\mathbf{x}_i)) \circ \Re(\mathcal{F}(\mathbf{x}_i)) + \Im(\mathcal{F}(\mathbf{x}_i)) \circ \Im(\mathcal{F}(\mathbf{x}_i))\Big), \tag{35}$$

$$\mathbf{h} = -2 \sum_{i=0}^{N-1} \Re(\mathcal{F}(\mathbf{x}_i)) \circ \Re(\mathcal{F}(\mathbf{B}_{i\cdot}^T)) + \Im(\mathcal{F}(\mathbf{x}_i)) \circ \Im(\mathcal{F}(\mathbf{B}_{i\cdot}^T)), \tag{36}$$

$$\mathbf{g} = 2 \sum_{i=0}^{N-1} \Im(\mathcal{F}(\mathbf{x}_i)) \circ \Re(\mathcal{F}(\mathbf{B}_{i\cdot}^T)) - \Re(\mathcal{F}(\mathbf{x}_i)) \circ \Im(\mathcal{F}(\mathbf{B}_{i\cdot}^T)). \tag{37}$$

The above can be derived based on the following fact. For any $\mathbf{Q} \in \mathbb{C}^{d \times d}$, $\mathbf{s}$, $\mathbf{t} \in \mathbb{C}^d$,

$$\begin{aligned}
||\mathbf{s} - \mathbf{Q}\mathbf{t}||_2^2 &= (\mathbf{s} - \mathbf{Q}\mathbf{t})^H(\mathbf{s} - \mathbf{Q}\mathbf{t}) \\
&= \mathbf{s}^H\mathbf{s} - \mathbf{s}^H\mathbf{Q}\mathbf{t} - \mathbf{t}^H\mathbf{Q}^H\mathbf{s} + \mathbf{t}^H\mathbf{Q}^H\mathbf{Q}\mathbf{t} \\
&= \Re(\mathbf{s})^T\Re(\mathbf{s}) + \Im(\mathbf{s})^T\Im(\mathbf{s}) - 2\Re(\mathbf{t})^T(\Re(\mathbf{Q})^T\Re(\mathbf{s}) + \Im(\mathbf{Q})^T\Im(\mathbf{s})) \\
&\quad + 2\Im(\mathbf{t})^T(\Im(\mathbf{Q})^T\Re(\mathbf{s}) - \Re(\mathbf{Q})^T\Im(\mathbf{s})) + \Re(\mathbf{t})^T(\Re(\mathbf{Q})^T\Re(\mathbf{Q}) + \Im(\mathbf{Q})^T\Im(\mathbf{Q}))\Re(\mathbf{t}) \\
&\quad + \Im(\mathbf{t})^T(\Re(\mathbf{Q})^T\Re(\mathbf{Q}) + \Im(\mathbf{Q})^T\Im(\mathbf{Q}))\Im(\mathbf{t}) + 2\Re(\mathbf{t})^T(\Im(\mathbf{Q})^T\Re(\mathbf{Q}) - \Re(\mathbf{Q})^T\Im(\mathbf{Q}))\Im(\mathbf{t}).
\end{aligned} \tag{38}$$

For the second term in (31), we note that the circulant matrix can be diagonalized by DFT matrix $\mathbf{F}_d$ and its conjugate transpose $\mathbf{F}_d^H$. Formally, for $\mathbf{R} = \mathrm{circ}(\mathbf{r})$, $\mathbf{r} \in \mathbb{R}^d$,

$$\mathbf{R} = (1/d)\mathbf{F}_d^H\mathrm{diag}(\mathcal{F}(\mathbf{r}))\mathbf{F}_d. \tag{39}$$

Let $\mathrm{Tr}(\cdot)$ be the trace of a matrix. Therefore,

$$\begin{aligned}
||\mathbf{R}\mathbf{R}^T - \mathbf{I}||_F^2 &= ||\frac{1}{d}\mathbf{F}_d^H(\mathrm{diag}(\tilde{\mathbf{r}})^H\mathrm{diag}(\tilde{\mathbf{r}}) - \mathbf{I})\mathbf{F}_d||_F^2 \\
&= \mathrm{Tr}\left[\frac{1}{d}\mathbf{F}_d^H(\mathrm{diag}(\tilde{\mathbf{r}})^H\mathrm{diag}(\tilde{\mathbf{r}}) - \mathbf{I})^H(\mathrm{diag}(\tilde{\mathbf{r}})^H\mathrm{diag}(\tilde{\mathbf{r}}) - \mathbf{I})\mathbf{F}_d\right] \\
&= \mathrm{Tr}\left[(\mathrm{diag}(\tilde{\mathbf{r}})^H\mathrm{diag}(\tilde{\mathbf{r}}) - \mathbf{I})^H(\mathrm{diag}(\tilde{\mathbf{r}})^H\mathrm{diag}(\tilde{\mathbf{r}}) - \mathbf{I})\right] \\
&= ||\tilde{\mathbf{r}}^H \circ \tilde{\mathbf{r}} - \mathbf{1}||_2^2 = ||\Re(\tilde{\mathbf{r}})^2 + \Im(\tilde{\mathbf{r}})^2 - \mathbf{1}||_2^2.
\end{aligned} \tag{40}$$

Furthermore, as $\mathbf{r}$ is real-valued, additional constraints on $\tilde{\mathbf{r}}$ are needed. For any $u \in \mathbb{C}$, denote $\bar{u}$ as its complex conjugate. We have the following result (Oppenheim et al., 1999): For any real-valued vector $\mathbf{t} \in \mathbb{C}^d$, $\mathcal{F}(\mathbf{t})_0$ is real-valued, and

$$\mathcal{F}(\mathbf{t})_{d-i} = \overline{\mathcal{F}(\mathbf{t})_i}, \quad i = 1, \cdots, \lfloor d/2 \rfloor. \tag{41}$$

From $(34) - (41)$, the problem of optimizing $\tilde{\mathbf{r}}$ becomes

$$\begin{aligned}
\underset{\tilde{\mathbf{r}}}{\mathrm{argmin}} \quad &\Re(\tilde{\mathbf{r}})^T\mathbf{M}\Re(\tilde{\mathbf{r}}) + \Im(\tilde{\mathbf{r}})^T\mathbf{M}\Im(\tilde{\mathbf{r}}) + \Re(\tilde{\mathbf{r}})^T\mathbf{h} \\
&+ \Im(\tilde{\mathbf{r}})^T\mathbf{g} + \lambda d||\Re(\tilde{\mathbf{r}})^2 + \Im(\tilde{\mathbf{r}})^2 - \mathbf{1}||_2^2 \\
\mathrm{s.t.} \quad &\Im(\tilde{r}_0) = 0 \\
&\Re(\tilde{r}_i) = \Re(\tilde{r}_{d-i}), i = 1, \cdots, \lfloor d/2 \rfloor \\
&\Im(\tilde{r}_i) = -\Im(\tilde{r}_{d-i}), i = 1, \cdots, \lfloor d/2 \rfloor.
\end{aligned} \tag{42}$$

The above is non-convex. Fortunately, the objective function can be decomposed, such that we can solve two variables at a time. Denote the diagonal vector of the diagonal matrix $\mathbf{M}$ as $\mathbf{m}$. The above optimization can then be decomposed to the following sets of optimizations.

$$\underset{\tilde{r}_0}{\mathrm{argmin}} \quad m_0\tilde{r}_0^2 + h_0\tilde{r}_0 + \lambda d\left(\tilde{r}_0^2 - 1\right)^2, \; \mathrm{s.t.} \; \tilde{r}_0 = \overline{\tilde{r}_0}. \tag{43}$$

$$\begin{aligned}
\underset{\tilde{r}_i}{\mathrm{argmin}} \quad &(m_i + m_{d-i})(\Re(\tilde{r}_i)^2 + \Im(\tilde{r}_i)^2) + 2\lambda d\left(\Re(\tilde{r}_i)^2 + \Im(\tilde{r}_i)^2 - 1\right)^2 \\
&+ (h_i + h_{d-i})\Re(\tilde{r}_i) + (g_i - g_{d-i})\Im(\tilde{r}_i), \quad i = 1, \cdots, \lfloor d/2 \rfloor.
\end{aligned}$$

In (43), we need to minimize a 4th order polynomial with one variable, with the closed form solution readily available. In (44), we need to minimize a 4th order polynomial with two variables. Though the closed form solution is hard to find (requiring solution of a cubic bivariate system), a local minimum can be found by gradient descent, which in practice has constant running time for such small-scale problems. The overall objective is guaranteed to be non-increasing in each step. In practice, we find that a good solution can be reached within just 5-10 iterations. Therefore in practice, the proposed time-frequency alternating optimization procedure has running time $\mathcal{O}(Nd \log d)$.

5.2 Learning with Dimensionality Reduction

In the case of learning $k < d$ bits, we need to solve the following optimization problem:

$$\underset{\mathbf{B},\mathbf{r}}{\operatorname{argmin}} \quad ||\mathbf{BP}_k - \mathbf{XP}_k^T\mathbf{R}^T||_F^2 + \lambda||\mathbf{RP}_k\mathbf{P}_k^T\mathbf{R}^T - \mathbf{I}||_F^2 \tag{44}$$

$$\text{s.t.} \quad \mathbf{R} = \text{circ}(\mathbf{r}),$$

in which $\mathbf{P}_k = \begin{bmatrix} \mathbf{I}_k & \mathbf{O} \\ \mathbf{O} & \mathbf{O}_{d-k} \end{bmatrix}$, $\mathbf{I}_k$ is a $k \times k$ identity matrix, and $\mathbf{O}_{d-k}$ is a $(d-k) \times (d-k)$ all-zero matrix.

In fact, the right multiplication of $\mathbf{P}_k$ can be understood as a "temporal cut-off", which is equivalent to a frequency domain convolution. This makes the optimization difficult, as the objective in frequency domain can no longer be decomposed. To address this issue, we propose a simple solution in which $B_{ij} = 0$, $i = 0, \cdots, N-1, j = k, \cdots, d-1$ in (31). Thus, the optimization procedure remains the same, and the cost is also $\mathcal{O}(Nd \log d)$. We will show in experiments that this heuristic provides good performance in practice.

6. Discussion

6.1 Limitations of the Theory for Long Codes

As was shown in earlier works (Li et al., 2011; Gong et al., 2013; Sánchez and Perronnin, 2011) and as we see in our experiments (Section 7), long codes are necessary for high-dimensional data for all binary embedding methods, either randomized or optimized.

However, when the code length is too large, our theoretical analysis is not optimal. For instance, consider our variance bound when $k > \sqrt{d}$. Here the ρ term always dominates, because for any vector, we have $\rho \geq 1/\sqrt{d}$ (at least one entry of a unit vector is at least $1/\sqrt{d}$). In numeric simulations, we see that the variance drops as $1/k$ for a larger range of k, roughly up to d. A similar behavior holds in Theorem 4, where the condition $\rho \leq \frac{\theta^2}{16k \log(k/\delta)}$ can hold only when $k < \mathcal{O}(\sqrt{d}/\log d)$. It is an interesting open question to analyze the variance and other concentration properties for larger k.

6.2 Semi-supervised Extension

In some applications, one can have access to a few labeled pairs of similar and dissimilar data points. Here we show how the CBE formulation can be extended to incorporate such

information in learning. This is achieved by adding an additional objective term $J(\mathbf{R})$.

$$\underset{\mathbf{B},\mathbf{r}}{\operatorname{argmin}} \quad ||\mathbf{B} - \mathbf{X}\mathbf{R}^T||_F^2 + \lambda||\mathbf{R}\mathbf{R}^T - \mathbf{I}||_F^2 + \mu J(\mathbf{R}) \tag{45}$$

$$\text{s.t.} \quad \mathbf{R} = \operatorname{circ}(\mathbf{r}),$$

$$J(\mathbf{R}) = \sum_{i,j \in \mathcal{M}} ||\mathbf{R}\mathbf{x}_i - \mathbf{R}\mathbf{x}_j||_2^2 - \sum_{i,j \in \mathcal{D}} ||\mathbf{R}\mathbf{x}_i - \mathbf{R}\mathbf{x}_j||_2^2. \tag{46}$$

Here $\mathcal{M}$ and $\mathcal{D}$ are the set of "similar" and "dissimilar" instances, respectively. The intuition is to maximize the distances between the dissimilar pairs, and minimize the distances between the similar pairs. Such a term is commonly used in semi-supervised binary coding methods (Wang et al., 2010). We again use the time-frequency alternating optimization procedure of Section 5. For a fixed $\mathbf{r}$, the optimization procedure to update $\mathbf{B}$ is the same. For a fixed $\mathbf{B}$, optimizing $\mathbf{r}$ is done in frequency domain by expanding $J(\mathbf{R})$ as below, with similar techniques used in Section 5.

$$||\mathbf{R}\mathbf{x}_i - \mathbf{R}\mathbf{x}_j||_2^2 = (1/d)||\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j))\tilde{\mathbf{r}}||_2^2. \tag{47}$$

Therefore,

$$J(\mathbf{R}) = (1/d)(\Re(\tilde{\mathbf{r}})^T \mathbf{A}\Re(\tilde{\mathbf{r}}) + \Im(\tilde{\mathbf{r}})^T \mathbf{A}\Im(\tilde{\mathbf{r}})), \tag{48}$$

where $\mathbf{A} = \mathbf{A}_1 + \mathbf{A}_2 - \mathbf{A}_3 - \mathbf{A}_4$, and

$$\mathbf{A}_1 = \sum_{(i,j) \in \mathcal{M}} \Re(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j)))^T \Re(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j))), \tag{49}$$

$$\mathbf{A}_2 = \sum_{(i,j) \in \mathcal{M}} \Im(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j)))^T \Im(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j))), \tag{50}$$

$$\mathbf{A}_3 = \sum_{(i,j) \in \mathcal{D}} \Re(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j)))^T \Re(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j))), \tag{51}$$

$$\mathbf{A}_4 = \sum_{(i,j) \in \mathcal{D}} \Im(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j)))^T \Im(\operatorname{diag}(\mathcal{F}(\mathbf{x}_i) - \mathcal{F}(\mathbf{x}_j))). \tag{52}$$

Hence, the optimization can be carried out as in Section 5, where $\mathbf{M}$ in (34) is simply replaced by $\mathbf{M} + \mu\mathbf{A}$. The semi-supervised extension improves over the non-semi-supervised version by 2% in terms of averaged AUC on the ImageNet-25600 data set.

7. Experiments

To compare the performance of the circulant binary embedding techniques, we conduct experiments on three real-world high-dimensional data sets used by the current state-of-the-art method for generating long binary codes (Gong et al., 2013). The Flickr-25600 data set contains 100K images sampled from a noisy Internet image collection. Each image is represented by a $25,600$ dimensional feature vector. To extract the feature vectors, we use

the publicly available SIFT features, which are densely extracted at three different scales. We cluster the features into 200 centers and then aggregate them into VLAD vectors (Jégou et al., 2010) of 128 × 200 = 25, 600 dimensions. The ImageNet-51200 contains 100k images sampled from 100 random classes from ImageNet (Deng et al., 2009), each represented by a 51, 200 dimensional VLAD vector generated by using 400 cluster centers. The third data set (ImageNet-25600) is another random subset of ImageNet containing 100K images in 25, 600 dimensional space. All the vectors are normalized to be of unit norm.

We compare the performance of the randomized (CBE-rand) and learned (CBE-opt) versions of our circulant embeddings with the current state-of-the-art for high-dimensional data, *i.e.*, bilinear embeddings. We use both the randomized (bilinear-rand) and learned (bilinear-opt) versions. Bilinear embeddings have been shown to perform similarly or better than another promising technique called Product Quantization (Jegou et al., 2011). Finally, we also compare against the binary codes produced by the baseline LSH method (Charikar, 2002), which is still applicable to 25,600 and 51,200 dimensional feature but with much longer running time and much more space. We also show an experiment with relatively low-dimensional feature (2048, with Flickr data) to compare against techniques that perform well for low-dimensional data but do not scale to high-dimensional scenario. Example techniques include ITQ (Gong et al., 2012a), SH (Weiss et al., 2008), SKLSH (Raginsky and Lazebnik, 2009), and AQBC (Gong et al., 2012b).

Following (Gong et al., 2013; Norouzi and Fleet, 2012; Gordo and Perronnin, 2011), we use 10,000 randomly sampled instances for training. We then randomly sample 500 instances, different from the training set as queries. The performance (recall@1-100) is evaluated by averaging the recalls of the query instances. The ground-truth of each query instance is defined as its 10 nearest neighbors based on ℓ_2 distance. For each data set, we conduct two sets of experiments: *fixed-time* where code generation time is fixed and *fixed-bits* where the number of bits is fixed across all techniques. We also show an experiment where the binary codes are used for classification.

The proposed CBE method is found robust to the choice of λ in (31). For example, in the retrieval experiments, the performance difference for $\lambda = 0.1, 1, 10$, is within 0.5%. Therefore, in all the experiments, we simply fix $\lambda = 1$. For the bilinear method, in order to get fast computation, the feature vector is reshaped to a near-square matrix, and the dimension of the two bilinear projection matrices are also chosen to be close to square. Parameters for other techniques are tuned to give the best results on these data sets.

7.1 Computational Time

When generating k-bit code for d-dimensional data, the full projection, bilinear projection, and circulant projection methods have time complexity $O(kd)$, $O(\sqrt{k}d)$, and $O(d \log d)$, respectively. We compare the computational time in Table 2 on a fixed hardware. Based on our implementation, the computational time of the above three methods can be roughly characterized as $d^2 : d\sqrt{d} : 5d \log d$. Note that faster implementation of FFT algorithms will lead to better computational time for CBE by further reducing the constant factor. Due to the small storage requirement $\mathcal{O}(d)$, and the wide availability of highly optimized FFT libraries, CBE is also suitable for implementation on GPU. Our preliminary tests based on

d	Full projection	Bilinear projection	Circulant projection
2^{15}	5.44×10^2	2.85	1.11
2^{17}	-	1.91×10^1	4.23
2^{20} (1M)	-	3.76×10^2	3.77×10^1
2^{24}	-	1.22×10^4	8.10×10^2
2^{27} (100M)	-	2.68×10^5	8.15×10^3

Table 2: Computational time (ms) of full projection (LSH, ITQ, SH *etc.*), bilinear projection (Bilinear), and circulant projection (CBE). The time is based on a single 2.9GHz CPU core. The error is within 10%. An empty cell indicates that the memory needed for that method is larger than the machine limit of 24GB.

GPU shows up to 20 times speedup compared with CPU. In this paper, for fair comparison, we use same CPU based implementation for all the methods.
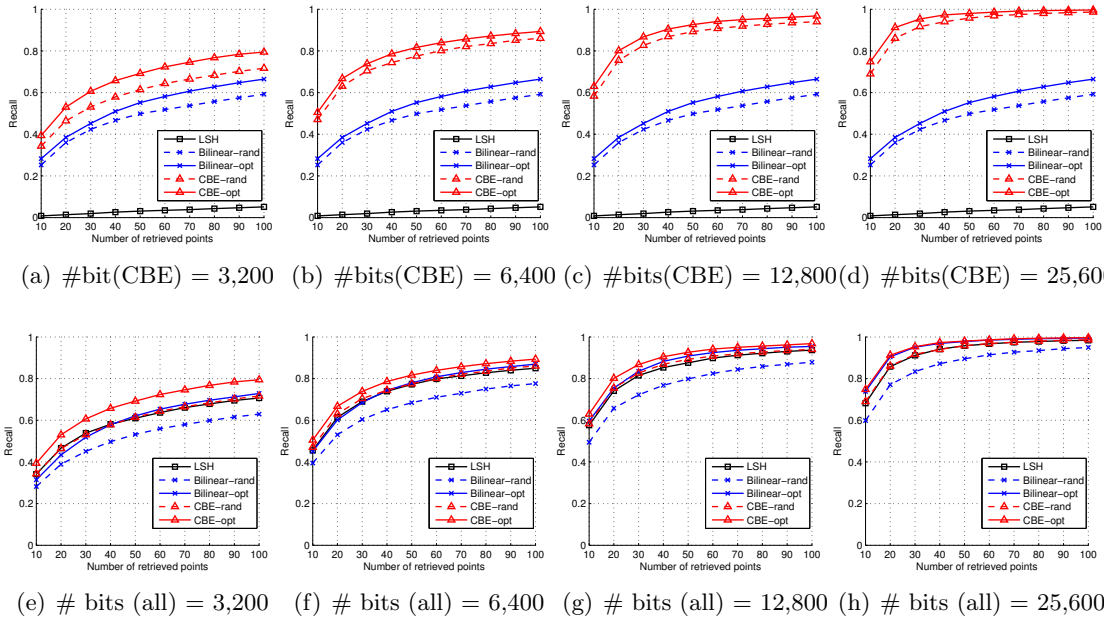

(a) #bit(CBE) = 3,200 (b) #bits(CBE) = 6,400 (c) #bits(CBE) = 12,800 (d) #bits(CBE) = 25,600

(e) # bits (all) = 3,200 (f) # bits (all) = 6,400 (g) # bits (all) = 12,800 (h) # bits (all) = 25,600

Figure 1: Recall on Flickr-25600. The standard deviation is within 1%. **First Row**: Fixed time. "# bits" is the number of bits of CBE. Other methods are using fewer bits to make their computational time identical to CBE. **Second Row**: Fixed number of bits. CBE-opt/CBE-rand are 2-3 times faster than Bilinear-opt/Bilinear-rand, and hundreds of times faster than LSH.
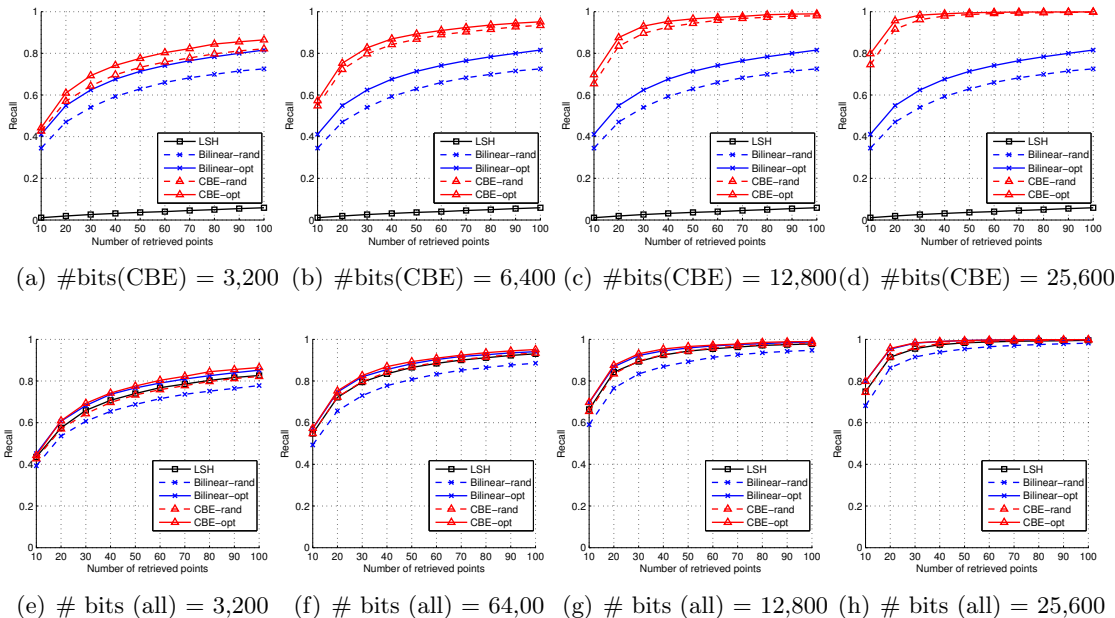
19

(a) #bits(CBE) = 3,200 (b) #bits(CBE) = 6,400 (c) #bits(CBE) = 12,800(d) #bits(CBE) = 25,600

(e) # bits (all) = 3,200 (f) # bits (all) = 64,00 (g) # bits (all) = 12,800 (h) # bits (all) = 25,600

Figure 2: Recall on ImageNet-25600. The standard deviation is within 1%. **First Row**: Fixed time. "# bits" is the number of bits of CBE. Other methods are using fewer bits to make their computational time identical to CBE. **Second Row**: Fixed number of bits. CBE-opt/CBE-rand are 2-3 times faster than Bilinear-opt/Bilinear-rand, and hundreds of times faster than LSH.

7.2 Retrieval

The recalls of different methods are compared on the three data sets, shown in Figure 1 – 3. The top row in each figure shows the performance of different methods when the code generation time for all the methods is kept the same as that of CBE. For a fixed time, the proposed CBE yields much better recall than other methods. Even CBE-rand outperforms LSH and Bilinear code by a large margin. The second row compares the performance for different techniques with codes of same length. In this case, the performance of CBE-rand is almost identical to LSH even though it is hundreds of time faster. This is consistent with our analysis in Section 4. Moreover, CBE-opt/CBE-rand outperform Bilinear-opt/Bilinear-rand in addition to being 2-3 times faster.

There exist several techniques that do not scale to high-dimensional case. To compare our method with those, we conduct experiments with fixed number of bits on a relatively low-dimensional data set (Flickr-2048), constructed by randomly sampling 2,048 dimensions of Flickr-25600. As shown in Figure 4, though CBE is not designed for such scenario, the CBE-opt performs better or equivalent to other techniques except ITQ which scales very poorly with d ($\mathcal{O}(d^3)$). Moreover, as the number of bits increases, the gap between ITQ and CBE becomes much smaller suggesting that the performance of ITQ is not expected to be better than CBE even if one could run ITQ on high-dimensional data.
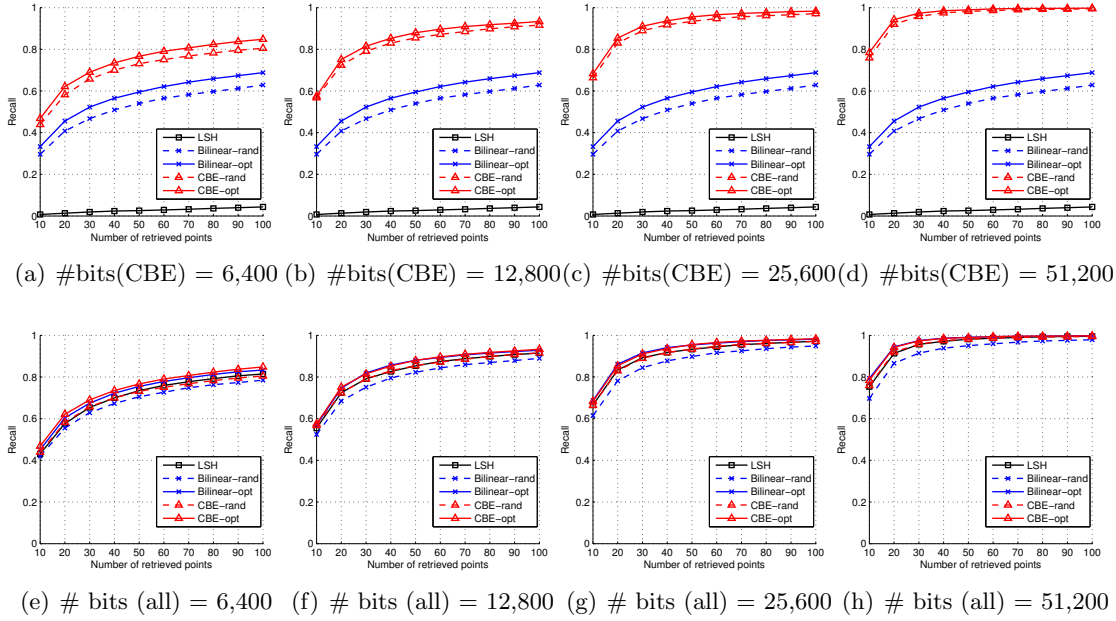
(a) #bits(CBE) = 6,400 (b) #bits(CBE) = 12,800(c) #bits(CBE) = 25,600(d) #bits(CBE) = 51,200

(e) # bits (all) = 6,400 (f) # bits (all) = 12,800 (g) # bits (all) = 25,600 (h) # bits (all) = 51,200

Figure 3: Recall on ImageNet-51200. The standard deviation is within 1%. **First Row**: Fixed time. "# bits" is the number of bits of CBE. Other methods are using fewer bits to make their computational time identical to CBE. **Second Row**: Fixed number of bits. CBE-opt/CBE-rand are 2-3 times faster than Bilinear-opt/Bilinear-rand, and hundreds of times faster than LSH.

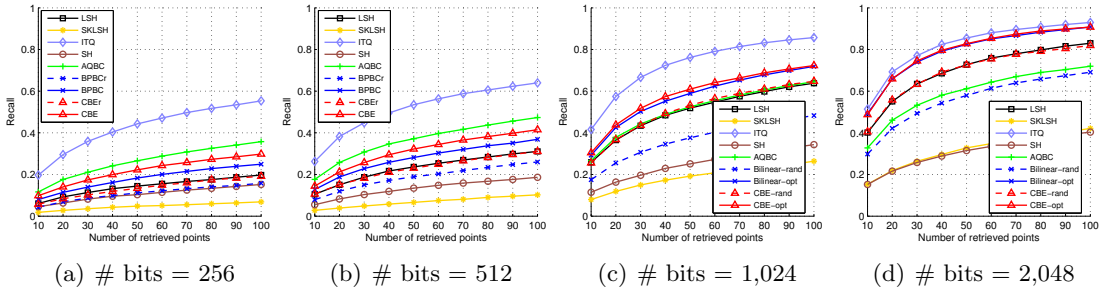(a) # bits = 256   (b) # bits = 512   (c) # bits = 1,024   (d) # bits = 2,048

Figure 4: Performance comparison on relatively low-dimensional data (Flickr-2048) with fixed number of bits. CBE gives comparable performance to the state-of-the-art even on low-dimensional data as the number of bits is increased. However, these other methods do not scale to very high-dimensional data setting which is the main focus of this work.

We also conduct additional experiments to compare CBE with the more recent Hadamard-based algorithms. The first algorithm we consider generates the binary code using the Fast
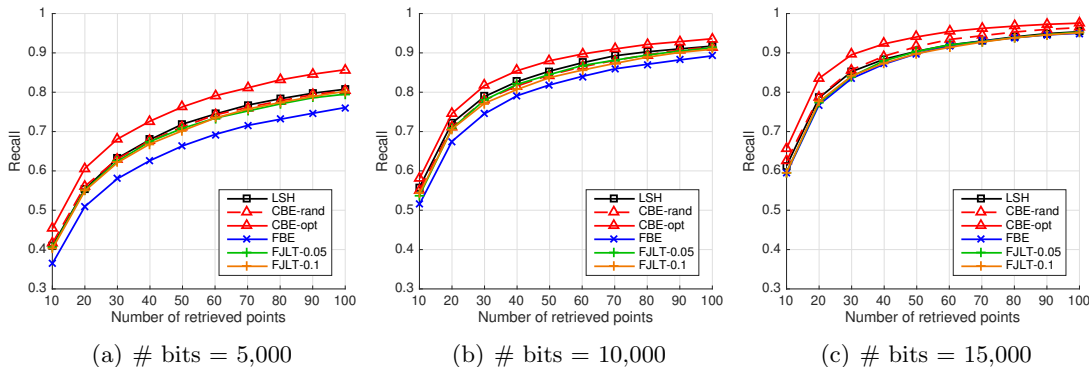
Figure 5: Recall on the Flickr-25600 data set. The methods compared are CBE-rand, CBE-opt, Fast Binary Embedding (FBE), Fast Johnson-Lindenstruss Transformation (FJLT) based methods and LSH. We follow the detailed setting of the FBE paper(Yi et al., 2015). In FJLT-p, p represents the percentage of nonzero elements in the sparse Gaussian matrix.

Johnson-Lindenstrss Transformation (FJLT). Similar to the circulant projection, FJLT has been used in dimensionality reduction (Ailon and Chazelle, 2006), deep neural networks (Yang et al., 2015), and kernel approximation (Le et al., 2013). Here, the binary code of $\mathbf{x} \in \mathbb{R}^d$ is generated as

$$h(\mathbf{x}) = \text{sign}(\mathbf{PHDx}), \tag{53}$$

where $\mathbf{P} \in \mathbb{R}^{k \times d}$ is a sparse matrix with the nonzeros entries generated iid from the standard distribution. $\mathbf{H} \in \mathbb{R}^{d \times d}$ is the Hadamard matrix, and $\mathbf{D} \in \mathbb{R}^{d \times d}$ is a diagonal matrix with random signs. Although the Hadamard transformation has computational complexity $\mathcal{O}(d \log d)$ (the multiplication with $\mathbf{H}$), this method is often slower than CBE due to the sparse Gaussian projection step (i.e., multiplication by $\mathbf{P}$).

The second method we compare with is Fast Binary Embedding (FBE). It is a theoretically sound method recently proposed in (Yi et al., 2015). FBE generates binary bits using a partial Walsh-Hadamard matrix and a set of partial Gaussian Toeplitz matrices. The method can achieve the optimal measurement complexity $\mathcal{O}(\frac{1}{\delta^2} \log N)$. Different from CBE, FBE uses multiple Toeplitz matrices each of which has more degrees of freedom than the circulant matrix ($2d$ vs. d), and their theory does not apply to the more structured CBE case. We follow the parameters settings in (Yi et al., 2015) (Flickr-25600 data set, with the number of bits 5,000, 10,000 and 15,000). Note that under the setting, FBE is at least a few times slower than CBE due to the use of multiple Toeplitz projections. Figure 5 shows the retrieval performance. Based on the experiments, in addition to being much faster than FBE and FJLT, CBE-rand provides comparable or even better performance. Another advantage of CBE is that the framework permits data-dependent optimization to further improve the performance. In all the experiments, CBE-opt achieves the highest recall by a large margin compared to other methods.

Original	LSH	Bilinear-opt	CBE-opt
25.59±0.33	23.49±0.24	24.02±0.35	24.55 ±0.30

Table 3: Multiclass classification accuracy (%) on binary coded ImageNet-25600. The binary codes of same dimensionality are 32 times more space efficient than the original features (single-float).

7.3 Classification

Besides retrieval, we also test the binary codes for classification. The advantage is to save on storage, allowing even large scale data sets to fit in memory (Li et al., 2011; Sánchez and Perronnin, 2011). We follow the asymmetric setting of (Sánchez and Perronnin, 2011) by training linear SVM on binary code $\text{sign}(\mathbf{Rx})$, and testing on the original $\mathbf{Rx}$. Empirically, this has been shown to give better accuracy than the symmetric procedure. We use ImageNet-25600, with randomly sampled 100 images per category for training, 50 for validation and 50 for testing. The code dimension is set as 25,600. As shown in Table 3, CBE, which has much faster computation, does not show any performance degradation compared with LSH or bilinear codes in classification task.

8. Conclusion

We proposed a method of binary embedding for high-dimensional data. Central to our framework is to use a type of highly structured matrix, the circulant matrix, to perform the linear projection. The proposed method has time complexity $\mathcal{O}(d \log d)$ and space complexity $\mathcal{O}(d)$, while showing no performance degradation on real-world data compared with more expensive approaches ($\mathcal{O}(d^2)$ or $\mathcal{O}(d^{1.5})$). The parameters of the method can be randomly generated, where interesting theoretical analysis was carried out to show that the angle preserving quality can be as good as LSH. The parameters can also be learned based on training data with an efficient optimization algorithm.

Appendix A. Proofs of the Technical Lemmas

A.1 Proof of Lemma 6

For convenience, define $\mathbf{u}^{\perp} = \mathbf{u} - \Pi\mathbf{u}$, and similarly define $\mathbf{v}^{\perp}$. From our earlier observation about independence, we have that

$$\mathbb{E}\left[\left(\frac{1 - \text{sign}(\mathbf{r}^T\mathbf{a})\,\text{sign}(\mathbf{r}^T\mathbf{b})}{2} - \frac{\theta}{\pi}\right)\left(\frac{1 - \text{sign}(\mathbf{r}^T\mathbf{u}^{\perp})\,\text{sign}(\mathbf{r}^T\mathbf{v}^{\perp})}{2} - \frac{\theta}{\pi}\right)\right] = 0. \quad (54)$$

Because the LHS is equal to the product of the expectations, and the first term is 0. Thus the quantity we wish to bound is

$$\mathbb{E}\left[\left(\frac{1 - \text{sign}(\mathbf{r}^T\mathbf{a})\,\text{sign}(\mathbf{r}^T\mathbf{b})}{2} - \frac{\theta}{\pi}\right)\left(\frac{\text{sign}(\mathbf{r}^T\mathbf{u})\,\text{sign}(\mathbf{r}^T\mathbf{v}) - \text{sign}(\mathbf{r}^T\mathbf{u}^{\perp})\,\text{sign}(\mathbf{r}^T\mathbf{v}^{\perp})}{2}\right)\right].$$

Now by using the fact that $\mathbb{E}[XY] \leq \mathbb{E}[|X||Y|]$, together with the observation that the quantity $|(1 - \text{sign}(\mathbf{r}^T\mathbf{a})\,\text{sign}(\mathbf{r}^T\mathbf{b}))/2 - \theta/\pi|$ is at most 2, we can bound the above by

$$\mathbb{E}\left[\left|\,\text{sign}(\mathbf{r}^T\mathbf{u})\,\text{sign}(\mathbf{r}^T\mathbf{v}) - \text{sign}(\mathbf{r}^T\mathbf{u}^\perp)\,\text{sign}(\mathbf{r}^T\mathbf{v}^\perp)\right|\right]. \tag{55}$$

This is equal to

$$2\,\text{Pr}[\text{sign}(\mathbf{r}^T\mathbf{u})\,\text{sign}(\mathbf{r}^T\mathbf{v}) \neq \text{sign}(\mathbf{r}^T\mathbf{u}^\perp)\,\text{sign}(\mathbf{r}^T\mathbf{v}^\perp)], \tag{56}$$

since the term in the expectation is 2 if the product of signs is different, and 0 otherwise. To bound this, we first observe that for any two unit vectors $\mathbf{x}, \mathbf{y}$ with $\angle(\mathbf{x}, \mathbf{y}) \leq \epsilon$, we have $\text{Pr}[\text{sign}(\mathbf{r}^T\mathbf{x}) \neq \text{sign}(\mathbf{r}^T\mathbf{y})] \leq \epsilon/\pi$. We can use this to say that

$$\text{Pr}[\text{sign}(\mathbf{r}^T\mathbf{u}) \neq \text{sign}(\mathbf{r}^T\mathbf{u}^\perp)] = \frac{\angle(\mathbf{u}, \mathbf{u}^\perp)}{\pi}. \tag{57}$$

This angle can be bounded in our case by $(\pi/2) \cdot \delta$ by basic geometry.[10] Thus by a union bound, we have that

$$\text{Pr}[\left(\text{sign}(\mathbf{r}^T\mathbf{u}) \neq \text{sign}(\mathbf{r}^T\mathbf{u}^\perp)\right) \vee \left(\text{sign}(\mathbf{r}^T\mathbf{v}) \neq \text{sign}(\mathbf{r}^T\mathbf{v}^\perp)\right)] \leq \delta. \tag{58}$$

This completes the proof.

A.2 Proof of Lemma 7

Denoting the ith entry of $\mathbf{p}$ by p_i (so also for $\mathbf{q}$), we have that

$$S := \langle \mathbf{D}\mathbf{p}, s_{\to t}(\mathbf{D}\mathbf{q})\rangle = \sum_{i=0}^{d-1} \sigma_i \sigma_{i+t} p_i q_{i+t}. \tag{59}$$

We note that $\mathbb{E}[S] = 0$, by linearity of expectation (as $t > 0$, $\mathbb{E}[\sigma_i\sigma_{i+t}] = 0$), thus the lemma is essentially a tail bound on S. While we can appeal to standard tail bounds for quadratic forms of sub-Gaussian random variables (e.g. Hansen-Wright (Rudelson and Vershynin, 2013)), we give below a simple argument. Let us define

$$f(\sigma_0, \sigma_1, \ldots, \sigma_{d-1}) = \sum_{i=0}^{d-1} p_i q_{i+t} \sigma_i \sigma_{i+t}. \tag{60}$$

We will view f as being obtained from a martingale as follows. Define

$$Q_i := f(\sigma_0, \sigma_1, \ldots, \sigma_i, 0, \ldots, 0) - f(\sigma_0, \sigma_1, \ldots, \sigma_{i-1}, 0, \ldots, 0). \tag{61}$$

In this notation, we have $S = Q_0 + Q_1 + \cdots + Q_{d-1}$.

We have the martingale property that $\mathbb{E}[Q_i|Q_0, Q_1, \ldots, Q_{i-1}] = 0$ for all i, (because σ_i is ± 1 with equal probability). Further, we have the *bounded difference* property, i.e., $|Q_i| \leq |p_i q_{i+t}| + |p_{i-t}q_i|$. This implies that

$$|Q_i|^2 \leq 2(p_i^2 q_{i+t}^2 + p_{i-t}^2 q_i^2). \tag{62}$$

---

10. $\mathbf{u}$ is a unit vector, and $\mathbf{u}^\perp + \Pi\mathbf{u} = \mathbf{u}$, and $\|\Pi\mathbf{u}\| \leq \delta$, so the angle is at most $\sin^{-1}(\delta)$.

Thus we can use Azuma's inequality to conclude that for any $\gamma > 0$,

$$\Pr[|\sum_i Q_i - \mathbb{E}[\sum_i Q_i]| > \gamma] < e^{-\frac{\gamma^2}{2 \cdot \sum_i 2(p_i^2 q_{i+t}^2 + p_{i-t}^2 q_i^2)}} = e^{-\frac{\gamma^2}{8 \sum_i^2 p_i^2 q_{i+t}^2}}. \tag{63}$$

We can now use the fact that $\sum_i p_i^2 q_{i+t}^2 \leq \rho^2 \sum_i p_i^2 = \rho^2$ (since $\|\mathbf{p}\|_2 = 1$ and $\|\mathbf{q}\|_\infty \leq \rho$). This establishes the lemma.

A.3 Proof of Lemma 9

First, using Lemma 7 we have, for any $i \neq j$ and $c > 0$,

$$\Pr[|\langle X_i, Y_j \rangle| > c] < e^{-c^2/8\rho^2}. \tag{64}$$

We have a similar bound for $\Pr[|\langle X_i, X_j \rangle| > c]$. Thus by setting $c = 4\rho\sqrt{\log(k/\delta)}$ (δ as in the statement of the lemma), we can take a union bound over all k^2 choices of $i \neq j$ and conclude that w.p. at least $1 - \delta$, we have

$$\max_{i \neq j}\{|\langle X_i, X_j \rangle|, |\langle X_i, Y_j \rangle|\} < 4\rho\sqrt{\log(k/\delta)}. \tag{65}$$

We now prove that whenever Eq. (65) holds, we obtain (γ, k) orthogonality for the desired γ. Let us start with a basic fact in linear algebra.

Lemma 11 *Let A be an $d \times k$ matrix with $\sigma_k(A) \geq \tau$, for some parameter τ. Then any unit vector in the column span of A can be written as $\sum_i \alpha_i A_i$, with $\sum_i \alpha_i^2 \leq 1/\tau^2$.*

Proof By the definition of σ_k, we have that for any α_i, $\|\sum_i \alpha_i A_i\|_2^2 \geq \tau^2 \left(\sum_i \alpha_i^2\right)$. Thus for any unit vector $\sum_i \alpha_i A_i$, we have $\sum_i \alpha_i^2 \leq 1/\tau^2$. ∎

Now let B be the $d \times 2k$ matrix whose columns are $X_1, Y_1, X_2, Y_2, \ldots, X_k, Y_k$ in that order. Consider the entries of $B^T B$. Since X_i, Y_i are unit vectors, the diagonals are all 1. The $(2i-1, 2i)$th and $(2i, 2i-1)$th entries are exactly $\cos\theta$, because the angle between X_i, Y_i is θ. The rest of the entries are of magnitude $< \eta := 4\rho\sqrt{\log(k/\delta)}$.

Thus if we consider $M = B^T B - I$ (diagonal removed from $B^T B$), we have $-(\cos\theta + k\eta)I \preceq M \preceq (\cos\theta + k\eta)I$ (diagonal dominance). Thus we conclude that $B^T B$ has all its eigenvalues $\geq 1 - \cos\theta - k\eta$. Since $\theta \in (0, \pi/2)$, we can use the standard inequality $\cos\theta < 1 - \theta^2/2$ to conclude that the eigenvalues are $\geq \theta^2/2 - k\eta$. Now by our assumption on ρ, we have that $k\eta < \theta^2/4$. This implies that all the eigenvalues are $\geq \theta^2/4$.

Thus we have $\sigma_{2k}^2(B) \geq \theta^2/4$. We prove now that this lets us obtain a decomposition that helps us prove (γ, k)-orthogonality. A crucial observation is the following.

Lemma 12 *The projection of X_i onto $span\{X_1, Y_1, X_2, Y_2, \ldots, X_{i-1}, Y_{i-1}\}$ has length at most $\frac{2\eta\sqrt{2k}}{\theta}$.*

Proof Let $\mathcal{S}$ denote $span\{X_1, Y_1, \ldots, X_{i-1}, Y_{i-1}\}$. By definition, the squared of the length of projection is equal to $\max\{\langle y, X_i \rangle^2 \mid y \in \mathcal{S} \text{ and } \|y\|_2 = 1\}$ (this is how the projection onto a subspace can be defined).

To bound this, consider any unit vector $y \in \mathcal{S}$, and suppose we write it as $\sum_{j<i} \alpha_j X_j + \beta_j Y_j$. Let B' be the matrix that has columns $X_j, Y_j, j < i$. Then it is straightforward to see that $\sigma_{2(i-1)}(B') \geq \sigma_{2k}(B) \geq \theta/2$. Thus Claim 11 implies that $\sum_{j<i} \alpha_j^2 + \beta_j^2 \leq 4/\theta^2$. This means that

$$\langle X_i, y \rangle^2 = \big(\sum_{j<i} \alpha_j \langle X_j, X_i \rangle + \beta_j \langle Y_j, X_i \rangle \big)^2 \tag{66}$$

$$\leq \big(\sum_{j<i} \alpha_j^2 + \beta_j^2 \big) \big(\sum_{j<i} \langle X_j, X_i \rangle^2 + \langle Y_j, X_i \rangle^2 \big) \tag{67}$$

$$\leq \frac{4}{\theta^2} \cdot (2i - 2)\eta^2. \tag{68}$$

(In the first step, we used Cauchy-Schwartz.) Taking square roots now gives the claim. ∎

Now we perform the following procedure on the vectors (it is essentially Gram-Schmidt orthonormalization, with the slight twist that we deal with X_i, Y_i together):
1. Initialize: $\mathbf{u}_1 = X_1, \quad \mathbf{e}_1 = 0, \quad \mathbf{v}_1 = Y_1, \quad \mathbf{f}_1 = 0$.
2. For $i = 2, \ldots, k$, we set $\mathbf{u}_i, \mathbf{v}_i$ to be the projections of X_i, Y_i (respectively) orthogonal to $\mathrm{span}\{X_1, Y_1, \ldots, X_{i-1}, Y_{i-1}\}$. Set $\mathbf{e}_i = X_i - \mathbf{u}_i$ and $\mathbf{f}_i = Y_i - \mathbf{v}_i$.

The important observation is that for any i, we have

$$\mathrm{span}\{\mathbf{u}_j, \mathbf{v}_j : j < i\} = \mathrm{span}\{\mathbf{u}_j, \mathbf{v}_j, \mathbf{e}_j, \mathbf{f}_j : j < i\} = \mathrm{span}\{X_j, Y_j : j < i\}. \tag{69}$$

This is because by definition, $\mathbf{e}_i, \mathbf{f}_i \in \mathrm{span}\{X_j, Y_j : j < i\}$ for all i. Thus we have that $\mathbf{u}_i$ and $\mathbf{v}_i$ satisfy the first condition in Definition 8. It just remains to analyze the lengths. Now we can use Claim 12 to conclude that

$$\|\mathbf{e}_i\|_2^2, \|\mathbf{f}_i\|_2^2 < \frac{8k\eta^2}{\theta^2} = \frac{128 \cdot k\rho^2 \log(k/\delta)}{\theta^2}. \tag{70}$$

Once again, we use the bound on ρ to conclude that this quantity is at most 16ρ. This completes the proof of Lemma 9, with $\gamma = 4\sqrt{\rho}$.

A.4 Proof of Lemma 10

We start with a simple claim about the angle between $\mathbf{u}_i$ and $\mathbf{v}_i$.

Lemma 13 *For all i, we have $\angle(\mathbf{u}_i, \mathbf{v}_i) \in (\theta - \pi\gamma, \theta + \pi\gamma)$.*

Proof The angle between X_i and $\mathbf{u}_i$ is at most $\sin^{-1}(\gamma) < (\pi/2)\gamma$. So also, the angle between Y_i and $\mathbf{v}_i$ is at most $(\pi/2)\gamma$. Thus the angle between $\mathbf{u}_i, \mathbf{v}_i$ is in the interval $(\theta - \pi\gamma, \theta + \pi\gamma)$ (by triangle inequality for the geodesic distance). ∎

Let $\eta > 0$ be a parameter we will fix later (it will be a constant times $\gamma\sqrt{\log(k/\delta)}$). For all i, we define the following events:

$$E_i : \quad \min\{\langle \mathbf{r}, \mathbf{u}_i \rangle, \langle \mathbf{r}, \mathbf{v}_i \rangle\} < \eta \tag{71}$$

$$F_i : \quad \neg E_i \text{ and } \mathrm{sign} \langle \mathbf{r}, \mathbf{u}_i \rangle \neq \mathrm{sign} \langle \mathbf{r}, \mathbf{v}_i \rangle \tag{72}$$

The following claim now follows easily.

Lemma 14 *For any i, we have*

$$\Pr[E_i] \leq 2\eta, \tag{73}$$

$$\Pr[F_i] \in \left(\frac{\theta}{\pi} - \pi\gamma - 2\eta, \frac{\theta}{\pi}\right) \tag{74}$$

Proof The first inequality follows from the small ball probability of a univariate Gaussian (since $\langle \mathbf{r}, \mathbf{u}_i \rangle$ is a Gaussian of unit variance), and the second follows from Claim 13 and (73). ∎

We will set η to be larger than $\pi\gamma$, so the RHS in (74) can be replaced with $(\theta/\pi - 3\eta, \theta/\pi)$. Furthermore, the events above for a given i depend *only* on the projection of $\mathbf{r}$ to $\mathrm{span}\{\mathbf{u}_i, \mathbf{v}_i\}$; thus they are independent for different i. Let us abuse notation slightly and denote by E_i also the indicator random variable for the event E_i (so also F_i). Then by standard Chernoff bounds, we have for any $\tau > 0$,

$$\Pr\left[\sum_i E_i \geq 2k\eta + k\tau\right] < e^{-\frac{k\tau^2}{4\eta + \tau}}, \tag{75}$$

$$\Pr\left[\sum_i F_i \notin \left(\frac{k\theta}{\pi} - 3k\eta - k\tau, \frac{k\theta}{\pi} + k\tau\right)\right] < 2e^{-\frac{k\tau^2}{\theta + \tau}}. \tag{76}$$

Finally let H denote the event:

$$\max_i\{|\langle \mathbf{r}, \mathbf{e}_i \rangle|, |\langle \mathbf{r}, \mathbf{f}_i \rangle|\} \geq \eta. \tag{77}$$

For any i, since $\|\mathbf{e}_i\| < \gamma$, we have $\Pr[|\langle \mathbf{r}, \mathbf{e}_i \rangle| > t\gamma] \leq e^{-t^2/2}$. We can use the same bound with $\mathbf{f}_i$, and take a union bound over all i, to conclude that $\Pr[H] \leq 2k \cdot e^{-\eta^2/2\gamma^2}$.

Let us call a choice of $\mathbf{r}$ *good* if neither of the events in (75)-(76) above occur, and additionally H does not occur. Clearly, the probability of an $\mathbf{r}$ being good is at least $1 - \delta$, provided τ and η are chosen such that the RHS of the tail bounds above are all made $\leq \delta/4$.

Before setting these values, we note that for a good $\mathbf{r}$,

$$\frac{1}{k}\sum_i \mathbf{1}\{\mathrm{sign}\langle \mathbf{r}, X_i \rangle \neq \mathrm{sign}\langle \mathbf{r}, Y_i \rangle\} \in \left(\frac{\theta}{\pi} - 3\eta - \tau, \frac{\theta}{\pi} + 2\eta + 2\tau\right). \tag{78}$$

This is because whenever $F_i \wedge \neg H$ occurs, we have $\mathrm{sign}\langle \mathbf{r}, X_i \rangle \neq \mathrm{sign}\langle \mathbf{r}, Y_i \rangle$, and thus the LHS above is at least $\frac{\theta}{\pi} - 3\eta - \tau$. Also if we have $\neg H$, then the only way we can have $\mathrm{sign}\langle \mathbf{r}, X_i \rangle \neq \mathrm{sign}\langle \mathbf{r}, Y_i \rangle$ is if either F_i occurs, or if E_i occurs (in the latter case, it is not necessary that the signs are unequal). Thus we can upper bound the LHS by $\frac{\theta}{\pi} + 2\eta + 2\tau$.

Let us now set the values of η and τ. From the above, we need to ensure:

$$\frac{k\tau^2}{4\eta + \tau} \geq \log(4/\delta), \quad \frac{k\tau^2}{\theta + \tau} \geq \log(8/\delta), \quad \text{and} \quad \frac{\eta^2}{2\gamma^2} \geq \log(4k/\delta). \tag{79}$$

Thus we set $\eta = 2\gamma\sqrt{\log(4k/\delta)}$, and

$$\tau \geq \max\left\{\frac{2\log(8/\delta)}{k}, \sqrt{\frac{2\theta\log(8/\delta)}{k}}, \sqrt{\frac{8\eta\log(4/\delta)}{k}}\right\}. \tag{80}$$

For the above inequality to hold, it suffices to set

$$\tau \geq \frac{8\log(1/\delta)}{\sqrt{k}}. \tag{81}$$

This gives the desired bound on the deviation in the angle.

References

Dimitris Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 2003.

Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *ACM Symposium on Theory of Computing*, 2006.

Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *ACM Symposium on Theory of Computing*, 2002.

Yu Cheng, Felix Xinnan Yu, Rogerio Feris, Sanjiv Kumar, Alok Choudhary, and Shih-Fu Chang. An exploration of parameter redundancy in deep networks with circulant projections. In *IEEE International Conference on Computer Vision*, 2015.

Anna Choromanska, Krzysztof Choromanski, Mariusz Bojarski, Tony Jebara, Sanjiv Kumar, and Yann LeCun. Binary embeddings with structured hashed projections. In *International Conference on Machine Learning*, 2016.

Anirban Dasgupta, Ravi Kumar, and Tamás Sarlós. Fast locality-sensitive hashing. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2011.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Peter Frankl and Hiroshi Maehara. The johnson-lindenstrauss lemma and the sphericity of some graphs. *Journal of Combinatorial Theory, Series B*, 44(3):355–362, 1988.

Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (99):1, 2012a.

Yunchao Gong, Sanjiv Kumar, Vishal Verma, and Svetlana Lazebnik. Angular quantization-based binary codes for fast similarity search. In *Advances in Neural Information Processing Systems*, 2012b.

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

Albert Gordo and Florent Perronnin. Asymmetric distances for binary embeddings. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

Robert M Gray. *Toeplitz and circulant matrices: A review.* Now Pub, 2006.

Aicke Hinrichs and Jan Vybíral. Johnson-Lindenstrauss lemma for circulant matrices. *Random Structures & Algorithms*, 39(3):391–398, 2011.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *ACM Symposium on Theory of Computing*, 1998.

Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (1):117–128, 2011.

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

Brian Kulis and Trevor Darrell. Learning to hash with binary reconstructive embeddings. In *Advances in Neural Information Processing Systems*, 2009.

Kasper Green Larsen and Jelani Nelson. Optimality of the Johnson-Lindenstrauss Lemma. *arXiv:1609.02094*, 2016.

Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood – approximating kernel expansions in loglinear time. In *International Conference on Machine Learning*, 2013.

Ping Li, Anshumali Shrivastava, Joshua Moore, and Arnd Christian Konig. Hashing algorithms for large-scale learning. In *Advances in Neural Information Processing Systems*, 2011.

Edo Liberty, Nir Ailon, and Amit Singer. Dense fast random projections and lean walsh transforms. *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 512–522, 2008.

Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Hashing with graphs. In *International Conference on Machine Learning*, 2011.

Jiří Matoušek. On variants of the Johnson–Lindenstrauss lemma. *Random Structures & Algorithms*, 33(2):142–156, 2008.

Mohammad Norouzi and David Fleet. Minimal loss hashing for compact binary codes. In *International Conference on Machine Learning*, 2012.

Mohammad Norouzi, David Fleet, and Ruslan Salakhutdinov. Hamming distance metric learning. In *Advances in Neural Information Processing Systems*, 2012.

Alan V Oppenheim, Ronald W Schafer, John R Buck, et al. *Discrete-time signal processing*, volume 5. Prentice Hall Upper Saddle River, 1999.

Maxim Raginsky and Svetlana Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *Advances in Neural Information Processing Systems*, 2009.

Mark Rudelson and Roman Vershynin. Hanson-wright inequality and sub-gaussian concentration. *Electron. Commun. Probab*, 18(0), 2013.

Jorge Sánchez and Florent Perronnin. High-dimensional signature compression for large-scale image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

Jan Vybíral. A variant of the Johnson–Lindenstrauss lemma for circulant matrices. *Journal of Functional Analysis*, 260(4):1096–1105, 2011.

Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Sequential projection learning for hashing with compact codes. In *International Conference on Machine Learning*, 2010.

Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems*, 2008.

Zichao Yang, Marcin Moczulski, Misha Denil, Nando de Freitas, Alex Smola, Le Song, and Ziyu Wang. Deep fried convnets. In *IEEE International Conference on Computer Vision*, 2015.

Xinyang Yi, Constantine Caramanis, and Eric Price. Binary embedding: Fundamental limits and fast algorithm. In *International Conference on Machine Learning*, 2015.

Felix Xinnan Yu, S. Kumar, Y. Gong, and S.-F. Chang. Circulant binary embedding. In *International Conference on Machine Learning*, 2014.

Felix Xinnan Yu, Sanjiv Kumar, Henry Rowley, and Shih-Fu Chang. Compact nonlinear maps and circulant extensions. *arXiv preprint arXiv:1503.03893*, 2015.

Hui Zhang and Lizhi Cheng. New bounds for circulant Johnson-Lindenstrauss embeddings. *arXiv preprint arXiv:1308.6339*, 2013.