Extremely Sparse Johnson-Lindenstrauss Transform: From Theory to Algorithm

Rong Yin, Yong Liu, Weiping Wang, Dan Meng

Abstract—Dimension reduction is a fundamental data mining task. However, it has limited applicability in high-dimensional scenarios because of stringent computational requirements. To address these issues, we propose ESE, an extremely sparse Johnson-Lindenstrauss transform, which takes a substantial step in dimension reduction. The projection matrices of ESE are extremely sparse, with only $k$ nonzero elements employing the hash functions, where $k$ is the embedded dimension. Theoretical analysis shows that ESE has a smaller time complexity than the existing projection algorithms and keeps the best accuracy $(1+\varepsilon)$ for the general case, where $0 < \varepsilon \ll 1$. In particular, the optimal statistical accuracy is achieved requiring $\log(n) \log(d)/\varepsilon$ embedded dimension, where $n$ is the number of data, $d$ is the dimension of data. The extensive experiments verify that ESE has a significant advantage in time with satisfactory accuracy, compared to the state-of-the-art dimension reduction algorithms.

Index Terms—dimension reduction; Johnson-Lindenstrauss; embeddings; extremely sparse;

I. INTRODUCTION

For a high-dimensional dataset, directly processing datasets requires high time and storage space. Overcoming these limitations has motivated a variety of dimension reduction approaches to improve time complexity [1]–[7]. In data mining community, dimension reduction, mainly including feature extraction and feature selection, has become an important research and achieved remarkable achievement [8]–[15].

The singular value decomposition (SVD) feature selection [16] and approximate SVD [15] have a solid theoretical guarantee. However, their expensive calculations are not to be underestimated. Subsequently, a series of random projections (RP) algorithms were devised [15], [17], [18]. The elements of the projection matrices are mainly based on $\{-1, 0, -1\}$ or $\{-1, 1\}$ at a certain probability. However, when data dimensions become more and more larger, the time consumed by the matrix multiplication in RP will still be prohibitive. Subsequently, Liu et al. [19] proposes a sparse embeddings algorithm (SE) which builds a sparse embedded matrix. SE has a faster computing speed $O(\varepsilon^{-2}nd)$ with the embedded dimension $O(K/\varepsilon^2)$ than the previous dimension reduction algorithms with satisfactory performance, where $n$ and $d$ are the number and dimension of data, and $0 < \varepsilon \ll 1$. Unfortunately, for large $d$, the cost of matrix multiplication is still relatively large, and its accuracy $(1+\varepsilon)$ is only guaranteed for $K$-means. Makarychev et al. [20] also designed a random projections algorithm based on orthogonal and Gaussian projections, whose time complexity is $O(\varepsilon^{-2}nd\log(K/\varepsilon))$ with the accuracy $(1+\varepsilon)$.

In order to overcome the shortcomings, we propose an extremely sparse Johnson-Lindenstrauss transform ESE, a novel algorithm that, to the best of our knowledge, has the best known theoretical guarantees. ESE is based on hash functions to achieve sparsity. In matrix-matrix product, one only needs to multiply and store the nonzero elements. Therefore, the sparse matrix in ESE can greatly speed up the matrices operations. In detail, with time complexity $O(\varepsilon^{-1}n\log(n)\log(d))$ for fast matrix multiplication, ESE can keep the best accuracy $(1+\varepsilon)$ for general case. Compared to SE [19], ESE reduces the running time by a factor of $O(d/(\varepsilon \log(d) \log(n)))$ and the embedded dimension by a factor of $O(K/\varepsilon^2 \log(d) \log(n)))$. ESE in matrix multiplication is much faster than the state-of-the-art algorithms. Theoretical reasoning and experimental results demonstrate that ESE has a significant advantage over time complexity and keeps the best approximate accuracy.

Section 2 and 3 are the related work and proposed algorithm. Following sections are experiment, proof, conclusions and acknowledgment.

II. RELATED WORK

Embeddings have become an important tool for dimension reduction. A real gem in this area has been the result of Johnson and Lindenstrauss [26]. The key of JL-projections is the design of projection matrix. At first, the projection matrices are dense or comparatively dense [17], [27]. For example, the entries of the projection matrix obey Gauss distribution or Bernoulli distribution. Due to the high density and projection dimension of the projection matrices, the time complexity of matrix operation is still high. For further reducing the computational requirements, the projection matrix becomes more and more sparse [15], [19], [24], [25], which can speed up the matrices operations to a certain extent. However, when the data dimension is high, the time cost is still costly.
TABLE I
Comparison of the classical algorithms of dimension reduction. The second, third and fourth columns correspond to the number of selected features, time complexity and approximation accuracy. "(K-means)" denotes that the approximation accuracy is obtained by applying the dimension reduction algorithm to K-means learner, rather than its general accuracy. N/A, ε and δ denote "not available", the gap to optimality and the confidence level, respectively.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Dimensions</th>
<th>Time</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP (FOLKLORE)</td>
<td>$O(\log(n)/\varepsilon^2)$</td>
<td>$O(\varepsilon^{-2} n d \log(n)/\log(d))$</td>
<td>$1 + \varepsilon$ (K-means)</td>
</tr>
<tr>
<td>SVD [22]</td>
<td>$K$</td>
<td>$O(nd \min{n,d})$</td>
<td>$2 \ (K\text{-clustering})$</td>
</tr>
<tr>
<td>SVD [23]</td>
<td>$O(K/\varepsilon^2)$</td>
<td>$O(\varepsilon^{-2} n dK/ \log(d))$</td>
<td>$1 + \varepsilon \ (K\text{-clustering})$</td>
</tr>
<tr>
<td>RP [24]</td>
<td>$O(\log(n)/\varepsilon^2)$</td>
<td>$O(\varepsilon^{-2} n d \log^2(n))$</td>
<td>$1 + \varepsilon \ (K\text{-means})$</td>
</tr>
<tr>
<td>RP [15]</td>
<td>$O(K/\varepsilon^2)$</td>
<td>$O(\varepsilon^{-2} n \log(K/\varepsilon))$</td>
<td>$1 + \varepsilon \ (K\text{-means})$</td>
</tr>
<tr>
<td>RP [25]</td>
<td>$O(\log(n)/n)$</td>
<td>$O(\log(n))$</td>
<td>$N/A$</td>
</tr>
<tr>
<td>SE [19]</td>
<td>$O(K/\varepsilon^2)$</td>
<td>$O(\varepsilon^{-2} nd)$</td>
<td>$1 + \varepsilon \ (K\text{-means})$</td>
</tr>
<tr>
<td>RP [20]</td>
<td>$O(\log(K/\varepsilon)/\varepsilon^2)$</td>
<td>$O(\varepsilon^{-2} n d \log(K/\varepsilon))$</td>
<td>$1 + \varepsilon \ (K\text{-means})$</td>
</tr>
<tr>
<td>ESE (This Paper)</td>
<td>$O(\log(n) \log(d)/\varepsilon)$</td>
<td>$O(\varepsilon^{-1} n \log(n) \log(d))$</td>
<td>$1 + \varepsilon$</td>
</tr>
</tbody>
</table>

Inspired by those, we construct an extremely sparse Johnson-Lindenstrauss transform (called ESE), which skillfully uses hash functions to construct the projection matrix to achieve sparsity. By theoretical analysis, compared to the state-of-the-art dimension reduction algorithms, ESE is not only more sparse in the projection matrix but also has a smaller projection dimension $O(\log(n) \log(d)/\varepsilon)$ with the optimal statistical accuracy $(1 + \varepsilon)$. To the best of our knowledge, this is the first time that all these achievements have been achieved. Experimental results and theoretical reasoning verify that ESE has much less time complexity with the satisfactory accuracy. Table I shows the detail of the related algorithms.

III. EXTREMELY SPARSE JOHNSON-LINDENSTRAUSS TRANSFORM

In this section, we characterize the properties of ESE showing it achieves the optimal statistical accuracy, with dramatically reduced computations. This main result is given in Theorem 1. The complexity analysis also follows.

A. Proposed Algorithm

ESE can be described in general using hash-based projection matrix. Let $h$ be hash function from $\{1, \ldots, k\}$ to $\{1, \ldots, d\}$, $h(j) = i$ for $i \in \{1, \ldots, d\}$ with probability of $1/d$, and $\sigma$ be 2-wise independent hash function from $\{1, \ldots, d\}$ to $\{-1, 1\}$, $\sigma(i) = j$ for $j \in \{-1, 1\}$ with probability of $1/2$. Denote the projection matrix by $R \in \mathbb{R}^{d \times k}$, where

$$R_{ij} = \begin{cases} 
\sigma(i), & \text{for } i = h(j), \\
0, & \text{for } i \neq h(j).
\end{cases} \tag{1}$$

The projected dataset can be written as: $X = \sqrt{d/k} XR$, where $X \in \mathbb{R}^{n \times d}$ is the dataset. $R$ has only $k$ nonzero elements, which is a sparse matrix. In the matrix-matrix product, we only need to multiply and store the nonzero elements. The detail of ESE is given in Algorithm 1.

Algorithm 1 Extremely Sparse Johnson-Lindenstrauss Transform (ESE)

Input: Dataset $X \in \mathbb{R}^{n \times d}$.

Output: Embedding dataset $\hat{X} \in \mathbb{R}^{n \times k}$.

1. Set $k = \mathcal{O}\left(\left(2 \log(n) - \log(\delta)\right) \log(d)/\varepsilon\right)$.
2. Build hash function $h: \{1, \ldots, k\} \mapsto \{1, \ldots, d\}$.
3. Build hash function $\sigma: \{1, \ldots, d\} \mapsto \{-1, +1\}$.
4. Build a matrix $R \in \mathbb{R}^{d \times k}$, with $R_{ij} = \sigma(i)$ for $i = h(j)$ and $R_{ij} = 0$ for $i \neq h(j)$.
5. Compute $\hat{X} = \sqrt{\frac{d}{k}} XR$.

B. Complexity Analysis

In ESE, the projection matrix $R$ based on hash functions is extremely sparse. The number of nonzero elements in the matrix $R$ is only $k$. Therefore, the time complexity of the proposed algorithm is $O(nk)$, while for the traditional projection algorithm with the dense matrix is $O(ndk)$. Thus, the proposed ESE is much faster than the traditional projection when $d$ is large.

C. Theoretical Analysis

In this part, we introduce Theorem 1 which is the main result and guarantees the effectiveness of ESE.

Theorem 1: Let $P$ be an arbitrary set of $n$ points in $\mathbb{R}^d$, represented as an $n \times d$ matrix $X$. Given $\varepsilon, \delta \in (0,1)$, let

$$k_0 = \mathcal{O}\left(\frac{(2 \log(n) - \log(\delta)) \log(d)}{\varepsilon}\right). \tag{2}$$

For integer $k \geq k_0$, let $R$ be a $d \times k$ random matrix, which is constructed as in algorithm 1. Let

$$\hat{X} = \sqrt{\frac{d}{k}} XR \tag{3}$$

and let $f: \mathbb{R}^d \rightarrow \mathbb{R}^k$ map the $i$th row of $X$ to the $i$th row of $\hat{X}$. With probability at least $1 - \delta$, for all $u, v \in P$

$$(1 - \varepsilon)\|u - v\|^2 \leq \|f(u) - f(v)\|^2 \leq (1 + \varepsilon)\|u - v\|^2.$$
From a theoretical perspective, we know that this JL-projections allows one to construct a sparse projection matrix $R$ using far fewer random nonzero entries than all previous algorithms. By the sparse projection matrix, we can project data into $k$-dimensional subspace and maintain the Euclidean distance in high accuracy. If the projection dimension $k \geq O(\log(n)\log(d)/\varepsilon)$, the proposed algorithm preserves the optimal statistical accuracy $(1 + \varepsilon)$, which demonstrates that the proposed algorithm is effective.

IV. EXPERIMENT

K-means is a widely recognized algorithm and reflects the distance relationship between data. ESE reflects that the distance relationship between the projected data is still approximately the same as the original one. Therefore, experiments analyze the performance of ESE based on K-means, and compare it with state-of-the-art algorithms: 1) SVD: A classic dimension reduction method; 2) RP: Choosing the representative one [15]; 3) SE$^1$: the state-of-the-art random projections method [19]; 4) K-means$^2$, on the machine with 32 cores (2.40GHz) and 64 GB of RAM.

The experiments are conducted on 8 real-world datasets. See Table II for details. ORL and TDT2 are from website$^5$, and the rest are from website$^6$. On each dataset, 70% is used for training and 30% for testing, and we normalize them. The number of clusters and classes is the same.

A. Evaluation Metrics

In order to avoid contingency, every kind of experiment is repeated 30 times. Under different $k$, we compute the projection time (in seconds), and then logarize it to get Fig. 2. After dimension reduction, run all algorithms on a standard $K$-means to get clustering accuracy (Fig. 1).

The accuracy is defined as $\text{Accuracy} = \frac{\sum_{n=1}^{n} \mu(\hat{y}, \text{map}(y))}{n}$, where $y$ is the ground truth label of the $i$th datum and $\hat{y}$ corresponds to the derived label. $\mu(p, q)$ is the delta function

where $\mu(p, q) = 1$ if $p = q$ and $\mu(p, q) = 0$ otherwise. $\text{map}(\cdot)$ is the best mapping function that matches the true labels and the derived ones. The greater the accuracy, the better.

B. Experimental Results

1) Fig. 1 shows that ESE is at the same level of accuracy as the best. The accuracy of SE and RP fluctuate on different datasets. SVD maintains high accuracy on most datasets, which is almost the same as ESE. The higher the projection dimension is, the higher the clustering accuracy is. This verifies our theoretical results.

2) Fig. 2 shows ESE has a significant advantage over projection time and even achieves several hundred orders of magnitude faster than RP and others. The larger the dimension of the dataset, the more obvious the time advantage of ESE is. For example, on the higher dimension TDT2, news20 and Rcv1-binary datasets, ESE reduces the time cost by a factor of 400 compared to SE and 22,000 compared to RP. With the increase of the projection dimension $k$, the projection time increases in every algorithm. These are consistent with the empirical results and our theoretical analysis.

V. PROOF

Let $\text{x} \cdot \text{y}$ denote the inner product of vectors $\text{x}, \text{y}$. To simplify notation we will work with a matrix $R$ scaled by $d$. As a result, to get $\hat{X}$ we need to scale $XR$ by $1/d$. Different columns of $R$ are independent of each other. After scaling, the entry of $R$ is $R_{ij}$, where $R_{ij} = dr_{ij}$, $c_j$ denotes the $j$th column of $R$, $\hat{x}_i$ and $x_i$ denote the $i$th row of $\hat{X}$ and $X$, namely the $i$th datum. Therefore, $\hat{x}_i = \frac{1}{\sqrt{2d}}(x_1 \cdot c_1, ... , x_i \cdot c_i)$. We replace $x_i$ with $\text{x}_i$.

Lemma 1: For all $h \in [0, 1/(2d)]$, all $d \geq 1$ and all unit vectors $\text{x}, \text{y}$.

$$E(\text{exp}(hQ_1(\text{x})^2)) \leq \frac{1}{\sqrt{1 - 2hd}},$$

and

$$E(Q(\text{x})^4) \leq 3d^2.$$  \hspace{1cm} (5)

Proof According to Lemma 2, we know $E(Q(\text{x})^4) \leq E(T^4)$, while $E(T^4) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \exp(-\lambda^2/2)(\lambda^4d^2) d\lambda = 3d^2$. The following will prove (4).

For any real-valued random variable $U$ and for all $h$ such that $E(\text{exp}(hU^2))$ is bounded. According to the Monotone Convexity Theorem (MCT), we get the formula $E(\text{exp}(hU^2)) = E\left(\sum_{k=0}^{\infty} \frac{(hU^2)^k}{k!}\right) = \sum_{k=0}^{\infty} \frac{h^k}{k!} E(U^{2k})$.

Here we obtain:

$$E(\text{exp}(hT^2)) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \exp(-\lambda^2/2) \exp(h\lambda^2d) d\lambda = \frac{1}{\sqrt{1 - 2hd}} \sum_{k=0}^{\infty} \frac{h^k}{k!} E(T^{2k})$$ \hspace{1cm} (6)

$$\geq \sum_{k=0}^{\infty} \frac{h^k}{k!} E(Q(\text{x})^{2k}) = E(\text{exp}(hQ(\text{x})^2)).$$
For convergence, we take $h \in [0, 1/(2d)]$ and apply the MCT in (6). Therefore, we have $\mathbb{E}(\exp(hQ(x)^2)) \leq \frac{1}{1 - 2sh}$ for $h \in [0, 1/(2d)]$.

**Lemma 2:** Let $T \sim \mathcal{N}(0, d)$. For every unit vector $x \in \mathbb{R}^d$, all $d \geq 1$ and all $k \in \mathbb{N}_+$, we have
\[
\mathbb{E}(Q(x)^{2k}) \leq \mathbb{E}(T^{2k}).
\] (7)

**Proof**  Our strategy for giving bounds on the moments of $Q(x)$ will be to determine a “worst-case” unit vector $w$ and bound the moments of $Q(w)$. Let $w = \frac{1}{\sqrt{d}} (1, \ldots, 1)$. For any vector $x$, $Q(x) = Q_1(x) = x \cdot c_1$, where $c_1 = d(r_{11}, \ldots, r_{d1})$. If $x = (x_1, \ldots, x_d)$ is such that $x_i^2 = x_j^2$ for all $i, j$, then by symmetry, $Q(x)$ and $Q(w)$ are identically distributed and this Lemma holds trivially. Otherwise, we can assume without loss of generality that $x_1^2 \neq x_2^2$ and consider the “more balanced” unit vector $\theta = (c, c, x_3, \ldots, x_d)$, where $c = \sqrt{(x_1^2 + x_2^2)/2}$.

We express $\mathbb{E}(Q(x)^{2k})$ as a sum of averages over $r_{11}, r_{21}$ and apply (8) in Lemma 3 to get that each term (average) in the sum. More precisely,
\[
\mathbb{E}(Q(x)^{2k}) = d^{2k} \sum_M \mathbb{E} \left( (M + x_1r_{11} + x_2r_{21})^{2k} \right) \cdot \mathbb{P} \left[ \sum_{i=3}^d x_i r_{i1} = M \cdot d \right]
\leq d^{2k} \sum_M \mathbb{E} \left( (M + cr_{11} + cr_{21})^{2k} \right) \cdot \mathbb{P} \left[ \sum_{i=3}^d x_i r_{i1} = M \cdot d \right]
= \mathbb{E}(Q(\theta)^{2k}).
\]

Applying this argument repeatedly yields the lemma, as $\theta$ eventually becomes $w$, we obtain $\mathbb{E}(Q(x)^{2k}) \leq \mathbb{E}(Q(w)^{2k})$. The following we will prove $\mathbb{E}(Q(w)^{2k}) \leq \mathbb{E}(T^{2k})$.  

**Fig. 1.** Clustering accuracy and different $k$ of various algorithms on ORL, gisette, TDT2, news20, Rcv1-binary, smallNORB, cifar10 and SVHN datasets.

**Fig. 2.** Projection time and different $k$ of various algorithms on ORL, gisette, TDT2, news20, Rcv1-binary, smallNORB, cifar10 and SVHN datasets.
\[ T \sim N(0, d), \quad T = \sqrt{d} \tilde{T} \quad \text{and} \quad \tilde{T} \sim N(0, 1). \quad Q(w) = w - c_1. \] We have \( Y \in \{-1, +1\} \) represents the nonzero elements in the first column of \( R \). Therefore, we get \( Q(w) = \sqrt{d}Y \).

For every \( k \in N_+ \), we observe \( E(Q(w)^{2k}) = (\sqrt{d})^{2k} E(Y^{2k}) \), and \( E(T^{2k}) = (\sqrt{d})^{2k} E(T^{2k}) \). Because \( Y \in \{-1, +1\} \), we have \( E(Y^{2k}) = 1 \). According to the well-known fact, we have the \( (2k) \)-th moment of \( N(0, 1) \) is \((2k-1)!! = (2^k k)! \geq 1\). So we get this lemma.

**Lemma 3:** Let \( r_1, r_2 \) be any two numbers in \( \{r_{ij}\} \). For any \( a, b \in R \), let \( c = (a^2 + b^2)/2 \). Then \( \forall M \in R \) and \( k \in N_+ \),

\[ E((M + ar_1 + br_2)^{2k}) \leq E((M + cr_1 + cr_2)^{2k}). \quad (8) \]

**Proof** We have the following representation, \( W = E((M + cr_1 + cr_2)^{2k}) - E((M + ar_1 + br_2)^{2k}) = W_1 + W_2. \) Because \( R_{ij} = dr_{ij} \) and \( r_1, r_2 \) belong to \( \{r_{ij}\} \), we have \( r_i \in \{0, +1, -1\} \). According to analysis we know \( P(r_i = 0) > P(r_i = \pm 1) \), and \( P(r_i = +1) = P(r_i = -1) \).

If \( a^2 = b^2 \), then \( a = \pm b \) and \( c = \pm h \) satisfy equality. If \( a^2 \neq b^2 \), we have the following according to \( W_1 \) and \( W_2 \).

1. \( W_1 \) is the value under the condition of \( |r_{ij}| = |r_{ij}| \).

2. \( W_2 \) is the value under the condition of \( |r_{ij}| \neq |r_{ij}| \).

For any \( u, v \in \{-1, 0, +1\} \) and \( |u| \neq |v| \), we remember \( P(r_1 = u, r_2 = v) = P_{2uv} \). So, \( W_2 = E_{u,v,-u,-v}P_{2uv}W_{2uv} \), where

\[ W_{2uv} = (M + uc + vc)^{2k} - (M + ua + vb)^{2k} + (M + uc - vc)^{2k} - (M + ua - vb)^{2k} + (M - uc + vc)^{2k} - (M - ua + vb)^{2k} + (M - uc - vc)^{2k} - (M - ua - vb)^{2k}. \]

Since \( a^2 \neq b^2 \) we can use the binomial theorem to expand every term \( M^{2k} \) in \( W_1 \) and get \( W_1 = 2M^{2k} + \sum_{i=0}^{2k} \binom{2k}{i} M^{2k-i} D_i \), where \( D_i = (a^2 - b^2)^i \). We have \( (a^2 - b^2)^i \geq (a - b)^i \) implying \( W_1 = 2M^{2k} + \sum_{i=0}^{2k} \binom{2k}{i} M^{2k-i} D_i \geq 0. \)

Since \( \sum_{i=0}^{2k} \binom{2k}{i} M^{2k-i} D_i \geq 0 \), we can get \( W_1 \geq 0. \)

For any \( u, v \in \{-1, 0, +1\} \) and \( |u| \neq |v| \), we remember \( P(r_1 = u, r_2 = v) = P_{2uv} \). So, \( W_2 = E_{u,v,-u,-v}P_{2uv}W_{2uv} \), where

\[ W_{2uv} = (M + uc + vc)^{2k} - (M + uc + vb)^{2k} + (M + uc - vc)^{2k} - (M + uc - vb)^{2k} + (M - uc + vc)^{2k} - (M - uc + vb)^{2k} + (M - uc - vc)^{2k} - (M - uc - vb)^{2k}. \]

Since \( a^2 \neq b^2 \) we can use the binomial theorem to expand every term \( M^{2k} \) in \( W_2 \) and get \( W_2 = \sum_{i=0}^{2k} \binom{2k}{i} M^{2k-i} D_i \), where \( D_i = (a^2 + b^2)^i \). We have \( (a^2 + b^2)^i \geq (a + b)^i \) implying \( W_2 \geq 0. \)

The proof for the case is similar to the above. According to [17], we get \( W_2 \geq 0. \) To sum up, we get \( W \geq 0. \)

**Proof of Theorem 1**

For the proof of Theorem 1, we firstly prove the following inequalities:

For any \( \varepsilon > 0 \) and any unit vector \( x \in \mathbb{R}^d \),

\[ P[S(x) > (1 + \varepsilon)kd] < \exp \left( -\frac{k}{2} \left( \varepsilon^2/2 - \varepsilon^3/3 \right) \right), \quad (9) \]

\[ P[S(x) < (1 - \varepsilon)kd] < \exp \left( -\frac{k}{2} \left( \varepsilon^2/2 - \varepsilon^3/3 \right) \right). \quad (10) \]

We start with the upper tail.

Note that for an arbitrary vector \( x \), \( S = S(x) = \sum_{j=1}^{k} (x \cdot e_j)^2 = \sum_{j=1}^{k} Q_j^2(x) \). For arbitrary \( h > 0 \), according to Markov’s inequality we get \( P[S > (1 + \varepsilon)kd] = P \left[ \exp(hS) > \exp \left( h(1 + \varepsilon)kd \right) \right] < \left( \exp(hS) \right) \exp \left( -h(1 + \varepsilon)kd \right) \). Since \( \{Q_j\}_{j=1}^{b} \) are i.i.d, we have \( P[S > (1 + \varepsilon)kd] = \left( \prod_{j=1}^{b} \exp(hQ_j^2) \right)^k \).

To optimize the bound, this gives \( h = \frac{1}{2} \frac{\varepsilon}{(1 + \varepsilon)\log_a} < \frac{1}{2}. \)

Taking (4) to (11). Thus, for any \(\varepsilon > 0 \), one can see that

\[ \left( \exp(hQ_j^2) \right)^k \exp \left( -h(1 + \varepsilon)kd \right) \]

\[ < \left( \frac{1}{\sqrt{1 - 2hd}} \right)^k \exp \left( -h(1 + \varepsilon)kd \right) \]

\[ = \left( 1 - \frac{\varepsilon}{(1 + \varepsilon) \log_a} \right)^{-k/2} \exp \left( -\frac{k}{2} \left( \varepsilon^2/2 - \varepsilon^3/3 \right) \right). \quad (11) \]

The proof of lower bound in (10) is similar to (9).

For arbitrary \( h > 0 \), we get that for any \( \varepsilon > 0 \),

\[ P[S < (1 - \varepsilon)kd] = P \left[ \exp(hS) < \exp \left( h(1 - \varepsilon)kd \right) \right] < \left( \exp(-hQ_j^2) \right)^k \exp \left( h(1 - \varepsilon)kd \right) \].

According to the distribution characteristics of \( R \) in (3), we know \( E(R_{ij}) = E(r_{ij}) = 0 \), thus, we have \( E(R_{ij}) = \frac{\varepsilon}{(1 + \varepsilon) \log_a} = \frac{1}{d} \) and \( E(Q_j) = E(d \sum_{i=1}^{d} x_i r_{ij}) = dE \left( \sum_{i=1}^{d} x_i r_{ij} \right) = d \sum_{i=1}^{d} x_i E(r_{ij}) = 0. \)

So, we obtain that

\[ E(Q_j^2) = \left( \sum_{i=1}^{d} x_i r_{ij} \right)^2 \]

\[ = d^2 E \left( \sum_{i=1}^{d} x_i r_{ij} \right)^2 \]

\[ = d^2 \left[ \sum_{i=1}^{d} x_i^2 E(r_{ij}^2) + \sum_{i=1}^{d} \sum_{m=1}^{d} 2x_i x_m E(r_{ij} E(r_{mj})) \right] \]

\[ = d \|x\|^2 = d \]

Let us expand \( \exp(-hQ_j^2) \) to get

\[ P[S(x) < (1 - \varepsilon)kd] = \left( \frac{1}{1 - hQ_j^2} \right)^k \exp \left( h(1 - \varepsilon)kd \right) \]

\[ = \left( 1 - hE(Q_j^2) + \frac{h^2}{2} E(Q_j^4) \right)^k \exp \left( h(1 - \varepsilon)kd \right). \quad (13) \]
Substituting (12) and (5) for (13), we get (15). Taking \( h = \frac{1}{16} \cdot \frac{1}{|1+e| \log d} \) which is not optimal but is still “good enough” and a series of expansion, we get (16):

\[
\left(1 - h\mathbb{E}(Q^2_i) + \frac{h^2}{2}\mathbb{E}(Q^2_i)^k\right) \exp\left(h(1 - e)kd\right)
\leq \left(1 - hd + \frac{3}{2}(hd)^2\right)^k \exp\left(h(1 - e)kd\right)
\leq \left(1 - \frac{e}{(1 + e) \log d}\right)^{-k/2} \exp\left(-\frac{-k\epsilon}{2 \log d}\right).
\]

Here, we complete the proof of upper and lower bounds.

As mentioned above, we have \( \|f(x)\|^2 = S \times T^2 \). Let

\[
2 \times \left(1 - \frac{e}{(1 + e) \log d}\right)^{-k/2} \exp\left(-\frac{-k\epsilon}{2 \log d}\right) \leq 2\delta/n^2,
\]

we obtain

\[
k_0 = \mathcal{O}\left((2 \log(n) - \log d) \log d/e\right).
\]

Combining (9) and (10), for each of the \( \binom{n}{2} \) pairs \( u, v \in P \), the squared norm of the vector \( u - v \) is maintained within a factor of \( 1 + \epsilon \). Therefore, if for some family \( r_{ij} \) as (3) we can get that for some \( \delta > 0 \) and any fixed vector \( x \in \mathbb{R}^d \), \( P_i((1 - \epsilon)\|x\|^2 \leq \|f(x)\|^2 \leq (1 + \epsilon)\|x\|^2 \geq 1 - 2\delta/n^2 \), then the probability of not getting right results is bounded by \( \binom{n}{2} \times 2\delta/n^2 < \delta \). \( \square \)

VI. CONCLUSIONS

Theoretical analysis shows that the proposed ESE has the minimum time complexity \( \mathcal{O}(\epsilon^{-1} n \log(n) \log(d)) \) in fast matrix multiplication and keeps the best accuracy \( (1 + \epsilon) \) for the general case, compared with the state-of-the-art embedded algorithms. Experimental results demonstrate that ESE has a significant advantage over time complexity than other dimension reduction algorithms with satisfactory accuracy.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China (No.61703396, No.61676293, No.62076234), the CCF-Tencent Open Fund, the Youth Innovation Promotion Association CAS, the Excellent Talent Introduction of Institute of Information Engineering of CAS (No. Y7Z0111107), Beijing Outstanding Young Scientist Program (NO. BJJWZYJH01201910002009), and the Beijing Municipal Science and Technology Project (No. Z191100007119002).

REFERENCES