Simple Yet Efficient Algorithms for Maximum Inner Product Search via Extreme Order Statistics

Ninh Pham
ninh.pham@auckland.ac.nz
University of Auckland
Auckland, New Zealand

ABSTRACT

We present a novel dimensionality reduction method for the approximate maximum inner product search (MIPS), named *CEOs*, based on the theory of concomitants of extreme order statistics. Utilizing the asymptotic behavior of these concomitants, we show that a few projections associated with the extreme values of the query signature are enough to estimate inner products. This yields a *sublinear* approximate MIPS algorithm with search recall guarantee under a mild condition. The indexing space is exponential but *optimal* for the approximate MIPS on a unit sphere.

To deal with the exponential space complexity, we present practical variants, including *CEOs-TA* and *coCEOs*, that use near-linear indexing space and time. CEOs-TA exploits the threshold algorithm (TA) and provides superior search recalls to LSH-based MIPS solvers. coCEOs is a new data and dimension co-reduction technique that outperforms CEOs-TA and other competitive methods. Empirically, they are simple to implement and achieve at least 100× speedup compared to the bruteforce search while returning top-10 MIPS with recall of at least 90% on many large-scale data sets.

CCS CONCEPTS

• **Information systems** → **Nearest-neighbor search**; • **Theory of computation** → **Nearest neighbor algorithms**.

KEYWORDS

Maximum Inner Product Search; Concomitants of Extreme Order Statistics; Dimensionality Reduction

ACM Reference Format:
Ninh Pham. 2021. Simple Yet Efficient Algorithms for Maximum Inner Product Search via Extreme Order Statistics . In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467345

1 INTRODUCTION

Maximum inner product search (MIPS) is the task of, given a point set $\mathbf{X} \in \mathbb{R}^{d\times n}$ of size n and a query point $\mathbf{q} \in \mathbb{R}^d$, finding the point $\mathbf{p} \in \mathbf{X}$ such that $\mathbf{p} = \arg\max_{\mathbf{x}\in\mathbf{X}} \mathbf{x}^\top \mathbf{q}$. MIPS and its variant top-k MIPS, which finds the top-k largest inner product points with a query, are central tasks in many big data applications, e.g. recommender system [15] and deep learning [23].

Due to the "curse of dimensionality", there is no known algorithms for solving exactly MIPS in truly sublinear time. In fact, if there exists such an algorithm, the Strong Exponential Time Hypothesis (SETH) [13], a fundamental conjecture in computational complexity theory, is wrong [1]. Therefore, recent works study *approximate* variants and deploy efficient approximate MIPS solvers as black-box procedures to speed up large-scale machine learning applications [6, 8, 20, 23].

MIPS applications often arise in large-scale high-dimensional domains and require fast query response. In contextual recommender systems [2], the learning phase (i.e. matrix factorization) cannot be done entirely offline [6]. In other words, the database of items vectors is computed online and hence a large indexing time for approximate MIPS solvers will significantly degrade the system performance. Approximate MIPS can also be used to reduce the computational cost of training and testing deep networks [23]. However, a large indexing space with high latency of updates will deteriorate the overall performance.

Motivated by the computational bottleneck of many MIPS applications in big data, this work addresses the following problem:

If we build a data structure for $\mathbf{X}$ *in* $\tilde{O}(dn)$[1] *time and space, can we have a fast MIPS solver that returns the best search recall?*

We present novel solutions with $\tilde{O}(dn)$ indexing time and space based on the concomitants of extreme order statistics. Our algorithms are simple to implement (few lines of Python codes), run significantly faster, and yield higher search recall than competitive MIPS solvers. Because Gaussian random projection (RP) is a building block of our approach and locality-sensitive hashing (LSH) is our primary alternative for comparison, we review both briefly.

1.1 Gaussian random projections

Given a point set $\mathbf{X} \in \mathbb{R}^{d\times n}$, we generate a Gaussian random matrix $\mathbf{R} \in \mathbb{R}^{l\times d}$ whose elements are randomly sampled from the standard normal distribution $N(0, 1)$. The signature (i.e. projected representation) of $\mathbf{x} \in \mathbf{X}$ is computed by $\mathbf{Rx}$. Since we study MIPS, we state the relative distortion bound of inner product values under random projections (RP) as follows:

Lemma 1.1 (Corollary 3.1 of [14]). *Let* $\theta_{\mathbf{xq}}$ *be the angle between two vectors* $\mathbf{x}, \mathbf{q} \in \mathbb{R}^d$. *Given* $0 < \epsilon < 1$, *we have the following:*

$$\mathbf{Pr}\left[\left|\frac{(\mathbf{Rx})^\top \mathbf{Rq}}{l} - \mathbf{x}^\top\mathbf{q}\right| \geq \epsilon\mathbf{x}^\top\mathbf{q}\right] \leq 2\exp\left(-\frac{l}{8}\epsilon^2\cos^2(\theta_{\mathbf{xq}})\right).$$

[1] Polylogarithmic factors, e.g. $\log d \log n$ is absorbed in the $\tilde{O}$-notation.

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467345

1.2 Locality-sensitive hashing

Locality-sensitive hashing (LSH) [3] is a key algorithmic primitive for similarity search in high dimensions due to the sublinear query time guarantee. Several approaches exploit LSH to obtain sublinear time solutions for approximate MIPS [12, 21, 22, 25]. We will review solutions based on SimHash [7] since we can use SimHash for both similarity estimation and sublinear time search for MIPS.

SimHash. Given a Gaussian random vector $\mathbf{r} \in \mathbb{R}^d$ whose elements are randomly drawn from $N(0,1)$, a SimHash function of $\mathbf{x}$ is $h(\mathbf{x}) = \text{sgn}\left(\mathbf{x}^\top \mathbf{r}\right)$. Denote by $0 \le \theta_{\mathbf{xq}} \le \pi$ the angle between two vectors $\mathbf{x}$ and $\mathbf{q}$, we have

Lemma 1.2.

$$\mathbf{Pr}\left[h(\mathbf{x}) = h(\mathbf{q})\right] = 1 - \frac{\theta_{\mathbf{xq}}}{\pi} = 1 - \frac{\arccos\left(\mathbf{x}^\top \mathbf{q} / \|\mathbf{x}\|\|\mathbf{q}\|\right)}{\pi}.$$

When $\|\mathbf{x}\| = \|\mathbf{q}\| = 1$, arccos() is a monotonically decreasing function of $\mathbf{x}^\top \mathbf{q}$, and hence SimHash is a LSH family for the inner product similarity on a unit sphere. Exploiting the binary representation of SimHash values, we can efficiently estimate the inner product $\mathbf{x}^\top \mathbf{q}$ with the fast Hamming distance computation using built-in functions of compilers.

SimHash-based solutions for MIPS. Since the change of $\|\mathbf{q}\|$ does not affect the result of MIPS, we can assume $\|\mathbf{q}\| = 1$ without loss of generality. Lemma 1.2 shows that the hash collision depends on both $\mathbf{x}^\top \mathbf{q}$ and $\|\mathbf{x}\|$. Therefore, by storing the 2-norm $\|\mathbf{x}\|$, we can use SimHash for estimating $\mathbf{x}^\top \mathbf{q}$. However, the dependency on such 2-norms makes MIPS more challenging to guarantee a sublinear query time. Since inner product is not a metric, SimHash-based solutions have to convert MIPS to nearest neighbor search by applying order preserving transformations to ensure that both data and query are on a unit sphere.

SimpleLSH [21] proposes asymmetric transformations : $\mathbf{q} \mapsto \mathbf{q}' = \{\mathbf{q}, 0\}$ and $\mathbf{x} \mapsto \mathbf{x}' = \{\mathbf{x}/M, \sqrt{1 - \|\mathbf{x}\|^2/M^2}\}$ where M is the maximum 2-norm of all points in $\mathbf{X}$. Since the inner product order is preserved and $\|\mathbf{x}'\| = \|\mathbf{q}'\| = 1$, we can exploit SimHash for both similarity estimation and sublinear time search for MIPS.

It is clear that the inner product values in the transformation space will be scaled down by a factor of M. Furthermore, the top-k inner product values are often very small compared to the vector norms in high dimensions. This means that the MIPS values and the distance gaps between "close" and "far apart" points in the transformation space are arbitrary small. Therefore, we need to use significantly large code length to achieve a reasonable search recall. In addition, the standard (l, L)-parameterized (where l is the number of concatenating hash functions and L is number of hash tables) bucketing algorithm [3] demands a huge space usage (i.e. large L) to guarantee a sublinear query time. In other words, the LSH performance will be degraded in the transformation space.

RangeLSH [25] handles this problem by partitioning the data into several partitions and applying SimpleLSH on each partition. While such distance gaps between "close" and "far apart" points in the LSH framework are slightly improved, the number of hash tables L needs to be scaled proportional to the number of partitions to achieve a reasonable search recall. Nevertheless, the subquadratic time and space cost of building LSH tables in order to guarantee a sublinear query time will be a bottleneck on many big data applications.

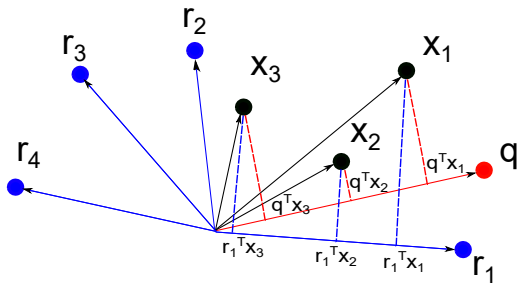

Figure 1: A geometric intuition of CEOs: Among 4 Gaussian random vectors $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_4$, the projections on $\mathbf{r}_1$ and $\mathbf{r}_4$, the closest and furthest vectors to $\mathbf{q}$, preserve the most inner product order and inverse order, respectively. Precomputing inner products on $\mathbf{r}_1$ and $\mathbf{r}_4$ will substantially speedup MIPS.

1.3 Our contribution

The paper presents a novel dimensionality reduction, called *CEOs*, based the theory of concomitants of extreme order statistics for MIPS. While we still apply D Gaussian random projections to construct signatures for data and queries, we only use a small *subset of D projections* to estimate inner products. This subset of size $s \ll D$ corresponds to the random projections associated with the s extreme values (i.e. maximum and minimum) of the query signature.

A geometric intuition is that among D random Gaussian vectors $\mathbf{r}_i, 1 \le i \le D$, the projection closest and furthest to the query $\mathbf{q}$, will preserve the best inner product order. Figure 1 shows that the projection on $\mathbf{r}_1$ preserves the best inner product order. Since $\mathbf{r}_1$ is the closest vector to $\mathbf{q}$, the value $\mathbf{q}^\top \mathbf{r}_1$ is maximum among 4 signature values $\mathbf{q}^\top \mathbf{r}_i, 1 \le i \le 4$. Similarly, we can also use the projection associated with $\mathbf{r}_4$, the furthest one to $\mathbf{q}$, which preserves the best inner product inverse order.

The technical difficulty is to provide a good estimator for $\mathbf{x}^\top \mathbf{q}$ given a small subset of projections associated with the s random vectors closest and furthest to $\mathbf{q}$. We observe that the subset of projections of size $s \ll D$ are the concomitants of extreme sth normal order statistics associated with the query signature. Leveraging the theory of concomitants of extreme order statistics, we provide a surprisingly simple and asymptotically unbiased estimate for $\mathbf{x}^\top \mathbf{q}$. That yields very fast and accurate MIPS solvers.

The key feature of CEOs is that we only need to use a small subset of s projections (e.g. $s = 10$) among the D projections of the data signatures, and the *sign* of these s projections of the query signature (i.e. projections associated with maximum and minimum values) to estimate inner products. This means that we can precompute and rank inner product estimators of all data *before* querying to significantly reduce the query time. For example, in Figure 1, we can precompute and rank $\mathbf{x}_i^\top \mathbf{r}_1, 1 \le i \le 3$, and simply return $\mathbf{x}_1$ when knowing $\mathbf{r}_1$ closest to $\mathbf{q}$.

Under a mild condition, our theoretical analysis shows that we can achieve a *sublinear* query time for MIPS with search recall guarantees given an $O\left((D/s)^s\right)$ indexing space complexity. We show that this exponential indexing space is *optimal*, matching the lower bound of the $(1+\epsilon)$-approximate MIPS on a unit sphere [5].

To trade the query performance for faster indexing, we propose practical CEOs variants, including *CEOs-TA* and *coCEOs*, that build the index in $\tilde{O}(dn)$ time and answer top-k MIPS with very high search recalls. CEOs-TA exploits the threshold algorithm (TA) [10] for solving MIPS. coCEOs is a new data and dimension co-reduction technique and often outperforms CEOs-TA on high search recall requirements. Especially, coCEOs is a budgeted MIPS solver with a parameter $B = o(n)$ that answers top-k MIPS in $o(n)$ time.

Our proposed algorithms are simple to implement. Empirically, on the inferior choice that uses the concomitants of the maximum and minimum order statistics, CEOs requires sublinear index space and outperforms LSH bucket algorithms regarding both efficiency and accuracy. Both CEOs-TA and coCEOs outperform competitive MIPS solvers on top-10 MIPS, and achieve at least 100× speedup compared to the bruteforce search with the search recall at least 90% on many real-world large-scale data sets.

2 THEORY OF CONCOMITANTS OF EXTREME ORDER STATISTICS

Our work studies the extreme behavior of bivariate normal distribution when the number of samples is sufficiently large. In particular, we apply the results of asymptotic theory of concomitants of extreme normal order statistics [9] for approximate MIPS.

2.1 Concomitants of normal order statistics and the connection to random projections

Let $(Q_1, X_1), (Q_2, X_2), \ldots, (Q_D, X_D)$ be D random samples from a bivariate normal distribution $N\left(\mu_Q, \mu_X, \sigma_Q^2, \sigma_X^2, \rho\right)$. We order these samples based on the Q-value. Given the rth order statistic $Q_{(r)}$, the X-value associated with $Q_{(r)}$, denoted by $X_{[r]}$, is called concomitant of the rth order statistic . The seminal work of David and Galambos [9] establishes the following properties of concomitants of normal order statistics.

Lemma 2.1.

$$\mathbf{E}\left[X_{[r]}\right] = \mu_X + \rho\sigma_X \mathbf{E}\left[\frac{Q_{(r)} - \mu_Q}{\sigma_Q}\right],$$

$$\mathbf{Var}\left[X_{[r]}\right] = \sigma_X^2\left(1 - \rho^2\right) + \sigma_X^2\rho^2\,\mathbf{Var}\left[\frac{Q_{(r)} - \mu_Q}{\sigma_Q}\right].$$

Given two vectors $\mathbf{q}, \mathbf{x} \in \mathbb{R}^d$ and any random Gaussian vector $\mathbf{r}_i \in \mathbb{R}^d$, we let $Q_i = \mathbf{q}^\top \mathbf{r}_i$ and $X_i = \mathbf{x}^\top \mathbf{r}_i$. For simplicity, we assume that $\|\mathbf{q}\| = 1$. It is well known that $Q_i \sim N(0,1)$, $X_i \sim N(0, \|\mathbf{x}\|^2)$, and Q_i and X_i are normal bivariates from $N\left(0, 0, 1, \|\mathbf{x}\|^2, \rho\right)$ where $\rho = \mathbf{x}^\top\mathbf{q}/\|\mathbf{x}\|$ [17]. Let $(Q_1, X_1), (Q_2, X_2), \ldots, (Q_D, X_D)$ be D random samples from $N\left(0, 0, 1, \|\mathbf{x}\|^2, \rho\right)$ generated by D Gaussian vectors $\mathbf{r}_i, 1 \le i \le D$. We form the concomitants of normal order statistics by descendingly sorting these pairs based on Q-value.

The theory of concomitants of extreme order statistics studies the asymptotic behavior of the concomitants, e.g. $X_{[1]}$ and $X_{[D]}$, when D goes to infinity. We will discuss how to leverage the asymptotic behavior of these concomitants to estimate $\mathbf{x}^\top\mathbf{q}$.

2.2 Concomitant of the extreme $Q_{(1)}$

Let $X_{[1]}$ be the concomitant of the first (maximum) order statistic $Q_{(1)}$. From Lemma 2.1, we have the following properties of $X_{[1]}$:

Lemma 2.2.

$$\mathbf{E}\left[X_{[1]}\right] = \rho\|\mathbf{x}\|\mathbf{E}\left[Q_{(1)}\right] = \mathbf{x}^\top\mathbf{q}\;\mathbf{E}\left[Q_{(1)}\right],$$

$$\begin{aligned}\mathbf{Var}\left[X_{[1]}\right] &= \|\mathbf{x}\|^2\left(1 - \rho^2\right) + \|\mathbf{x}\|^2\rho^2\,\mathbf{Var}\left[Q_{(1)}\right] \\ &= \|\mathbf{x}\|^2 + \left(\mathbf{x}^\top\mathbf{q}\right)^2\left(\mathbf{Var}\left[Q_{(1)}\right] - 1\right).\end{aligned}$$

We note that $Q_{(1)}$ is the largest variable among D independent standard normal variables. When D is sufficiently large, it is well known that $\mathbf{E}\left[Q_{(1)}\right] \to \sqrt{2\log D}$ and $\mathbf{Var}\left[Q_{(1)}\right] \to 0$, and the rate of convergence is $O\left(1/\log D\right)$ [11]. Using $\xrightarrow{D}$ as a proxy for asymptotic results with a sufficiently large D, we have

Lemma 2.3.

$$\mathbf{E}\left[X_{[1]}\right] \xrightarrow{D} \mathbf{x}^\top\mathbf{q}\sqrt{2\log D}\,,\ \mathbf{Var}\left[X_{[1]}\right] \xrightarrow{D} \|\mathbf{x}\|^2 - \left(\mathbf{x}^\top\mathbf{q}\right)^2.$$

To use $X_{[1]}$ to estimate $\mathbf{x}^\top\mathbf{q}$, we define $Z_{ceo} = X_{[1]}/\sqrt{2\log D}$. It is straightforward that $\mathbf{E}\left[Z_{ceo}\right] \xrightarrow{D} \mathbf{x}^\top\mathbf{q}$ and

$$\mathbf{Var}\left[Z_{ceo}\right] \xrightarrow{D} \left(\|\mathbf{x}\|^2 - \left(\mathbf{x}^\top\mathbf{q}\right)^2\right)/2\log D\,. \tag{1}$$

For notation simplicity, we name the method using $X_{[1]}$ for estimating inner products as *CEOs*.

Comparison with LSH: We now compare the inner product estimators provided by SimHash and CEOs. Define $Z = 1$ if $h(\mathbf{x}) = h(\mathbf{q})$; otherwise 0. By the LSH property, we have $\mathbf{E}\left[Z\right] = 1 - \theta_{\mathbf{xq}}/\pi$ and $\mathbf{Var}\left[Z\right] = \left(\theta_{\mathbf{xq}}/\pi\right)\left(1 - \theta_{\mathbf{xq}}/\pi\right)$. Using the Taylor series with $\arccos x \approx \pi/2 - x$ for $-1 \le x \le 1$, we have:

$$\mathbf{E}\left[Z\right] = 1 - \frac{\theta_{\mathbf{xq}}}{\pi} \approx \frac{1}{2} + \frac{\mathbf{x}^\top\mathbf{q}}{\pi\|\mathbf{x}\|}, \mathbf{Var}\left[Z\right] \approx \frac{1}{4} - \frac{\left(\mathbf{x}^\top\mathbf{q}\right)^2}{\pi^2\|\mathbf{x}\|^2}.$$

Define $Z_{lsh} = (Z - 1/2)\pi\|\mathbf{x}\|$, it is clear that $\mathbf{E}\left[Z_{lsh}\right] \approx \mathbf{x}^\top\mathbf{q}$ and

$$\mathbf{Var}\left[Z_{lsh}\right] = \pi^2\|\mathbf{x}\|^2\mathbf{Var}\left[Z\right] \approx \frac{\pi^2}{4}\|\mathbf{x}\|^2 - \left(\mathbf{x}^\top\mathbf{q}\right)^2. \tag{2}$$

Equations 1 and 2 show that SimHash with $\frac{\pi^2}{2}\log D \approx 5\log D$ bits achieves similar accuracy as CEOs. Since SimpleLSH has to scale down all inner products with a normalization factor M where M is the maximum 2-norm of the data, SimpleLSH needs approximately $5M^2\log D$ bits to have similar accuracy as CEOs.

Comparison with Gaussian RP: The theory of extreme order statistics [9] states that the distribution of $X_{[1]}$ converges in probability to the normal distribution $N\left(\mathbf{x}^\top\mathbf{q}\sqrt{2\log D}, \|\mathbf{x}\|^2 - \left(\mathbf{x}^\top\mathbf{q}\right)^2\right)$ when $D \to \infty$. Hence, $Z_{ceo} = X_{[1]}/\sqrt{2\log D} \xrightarrow{D} N\left(\mathbf{x}^\top\mathbf{q}, \frac{\|\mathbf{x}\|^2 - \left(\mathbf{x}^\top\mathbf{q}\right)^2}{2\log D}\right)$. Applying standard Chernoff bounds [24], we can show that CEOs offers a similar relative error bound as Gaussian RP in Lemma 1.1.

MIPS algorithms: Compared to CEOs, SimHash and Gaussian RP can achieve similar performance for MIPS by using $O\left(\log D\right)$ bits and projections for constructing signatures, respectively. However, after constructing signatures, these approaches have to perform $O\left(n\right)$ inner product estimation, which is clearly a computational bottleneck. In contrast, CEOs does not need this costly estimation step since we can sort all the points based on their projected values in advance. At the query phase, after executing RP to compute the projection index of $Q_{(1)}$, we can simply return the

k point indexes corresponding to the top-k concomitants associated with $Q_{(1)}$ as an approximate top-k MIPS. This key property makes CEOs more efficient than both Gaussian RP and LSH-based solutions on answering approximate MIPS.

2.3 Concomitants of the extreme $Q_{(r)}$

Since Gaussian distribution is symmetric, we can use both $X_{[1]}$ and $-X_{[D]}$ corresponding to the maximum $Q_{(1)}$ and minimum $Q_{(D)}$ order statistics for estimating $\mathbf{x}^\top \mathbf{q}$. One natural question is: "Can we use more concomitants and are they independent so that we can boost the accuracy with the standard Chernoff bounds?".

Indeed, we have a positive answer by investigating the asymptotic independence of concomitants of extreme order statistics. Let s_0 be a fixed positive integer and $D \to \infty$, David and Galambos [9] show that the concomitants $X_{[r]}$ where $1 \le r \le s_0$ are asymptotically independent and normal as follows:

$$X_{[r]} \xrightarrow{D} N\left(\mathbf{x}^\top \mathbf{q}\sqrt{2\log D}, \|\mathbf{x}\|^2 - \left(\mathbf{x}^\top \mathbf{q}\right)^2\right).$$

By the symmetry of Gaussian distribution, we can use the concomitants $X_{[r]}$ and $-X_{[D-r+1]}$ where $1 \le r \le s_0$ to boost the accuracy of the estimator of $\mathbf{x}^\top \mathbf{q}$. We use these $s = 2s_0$ concomitants as a specific dimensionality reduction since $\sum_{r=1}^{s_0}\left(X_{[r]} - X_{[D-r+1]}\right)$ provides an unbiased estimate for $\mathbf{x}^\top \mathbf{q}$.

3 AN OPTIMAL ALGORITHM FOR MIPS

Since we can view the concomitants of extreme order statistics as a specific dimensionality reduction, this section will describe how to exploit key properties of concomitants of extreme sth order statistics to achieve an *optimal* MIPS solver.

We recall our notation that we have $\mathbf{q}, \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, $\|\mathbf{q}\| = 1$, and D Gaussian vectors $\mathbf{r}_i \in \mathbb{R}^d$. We let $Q_i = \mathbf{q}^\top \mathbf{r}_i$, $X_i = \mathbf{x}^\top \mathbf{r}_i$, and $Y_i = \mathbf{y}^\top \mathbf{r}_i$. We have Q_i and X_i are normal bivariates from $N\left(0, 0, 1, \|\mathbf{x}\|^2, \rho_{\mathbf{x}}\right)$ where $\rho_{\mathbf{x}} = \mathbf{x}^\top \mathbf{q}/\|\mathbf{x}\|$. Q_i and Y_i are normal bivariates from $N\left(0, 0, 1, \|\mathbf{y}\|^2, \rho_{\mathbf{y}}\right)$ where $\rho_{\mathbf{y}} = \mathbf{y}^\top \mathbf{q}/\|\mathbf{y}\|$.

Let $X_{[r]}$ and $Y_{[r]}$ be the concomitants of $Q_{(r)}$ in D random samples from the bivariate normal distribution $N\left(0, 0, 1, \|\mathbf{x}\|^2, \rho_{\mathbf{x}}\right)$ and $N\left(0, 0, 1, \|\mathbf{y}\|^2, \rho_{\mathbf{y}}\right)$, respectively. We note that $X_{[r]}$ and $Y_{[r]}$ are not independent due to the link to $Q_{(r)}$.

For simplicity, we only consider $X_{[1]}$ and $Y_{[1]}$ and denote $\rho_{[1]}$ be the correlation between $X_{[1]}$ and $Y_{[1]}$. Let $\tau_1 = \mathbf{x}^\top \mathbf{q}$, $\tau_2 = \mathbf{y}^\top \mathbf{q}$ and assume $\tau_1 > \tau_2$. By the asymptotic property of concomitants of extreme order statistics, we have:

$$X_{[1]} \xrightarrow{D} N\left(\tau_1\sqrt{2\log D}, \|\mathbf{x}\|^2 - \tau_1^2\right),$$
$$Y_{[1]} \xrightarrow{D} N\left(\tau_2\sqrt{2\log D}, \|\mathbf{y}\|^2 - \tau_2^2\right).$$

Applying the standard Chernoff bound [24] for the Gaussian variable $Y_{[1]} - X_{[1]}$, we have:

$$\mathbf{Pr}\left[Y_{[1]} - X_{[1]} \ge 0\right] \le D^{-(\tau_1-\tau_2)^2/\left(\sqrt{\|\mathbf{x}\|^2-\tau_1^2}+\sqrt{\|\mathbf{y}\|^2-\tau_2^2}\right)^2}. \quad (3)$$

Let $\alpha_{\mathbf{y}} = (\tau_1 - \tau_2)/\left(\sqrt{\|\mathbf{x}\|^2 - \tau_1^2} + \sqrt{\|\mathbf{y}\|^2 - \tau_2^2}\right) > 0$. When $\alpha_{\mathbf{y}}^2$ is not very small and D is sufficiently large, $X_{[1]}$ is ranked higher than $Y_{[1]}$ with high probability in the sorted list associated with $Q_{(1)}$. This observation is the key property which makes CEOs more efficient than competitive MIPS solvers.

Due to the asymptotic independence and normal distribution of $s = 2s_0$ concomitants $X_{[r]}$ and $-X_{[D-r+1]}$ where $1 \le r \le s_0$, we can use the sum of these s concomitants as an inner product estimate. Since the variance of the average of $2s_0$ independent random variables is decreased by a factor of $1/2s_0$, we have

$$\mathbf{Pr}\left[\sum_{r=1}^{s_0}\left(Y_{[r]} - Y_{[D-r+1]}\right) \ge \sum_{r=1}^{s_0}\left(X_{[r]} - X_{[D-r+1]}\right)\right] \le D^{-2s_0\alpha_{\mathbf{y}}^2}. \quad (4)$$

Assume that $\alpha_{\mathbf{y}}^2$ is not very small, i.e. $\alpha_{\mathbf{y}}^2 \ge c/s_0$ where $c > 1$. Given a sufficiently large n, $D = n^{1/c}$ suffices for asymptotic properties of concomitants associated with the sth order statistics. With $D = n^{1/c}$, the probability in Equation 4 is bounded by $1/n^2$. Applying the union bound, we state our main result as follows:

Theorem 3.1. *Assume $\mathbf{x}$ is the top-1 MIPS for the query $\mathbf{q}$. Let $\alpha_{\mathbf{y}} = \left(\mathbf{x}^\top \mathbf{q} - \mathbf{y}^\top \mathbf{q}\right)/\left(\sqrt{\|\mathbf{x}\|^2 - (\mathbf{x}^\top \mathbf{q})^2} + \sqrt{\|\mathbf{y}\|^2 - (\mathbf{y}^\top \mathbf{q})^2}\right)$ for any $\mathbf{y} \in \mathbf{X} \setminus \mathbf{x}$. Given a sufficiently large n and a constant $c > 1$, we assume that $\alpha_{\mathbf{y}}^2 \ge c/s_0$ for all $\mathbf{y} \in \mathbf{X} \setminus \mathbf{x}$. By choosing $D = n^{1/c}$, for all $\mathbf{y} \in \mathbf{X} \setminus \mathbf{x}$, we have:*

$$\mathbf{Pr}\left[\sum_{r=1}^{s_0}\left(X_{[r]} - X_{[D-r+1]}\right) \ge \sum_{r=1}^{s_0}\left(Y_{[r]} - Y_{[D-r+1]}\right)\right] \ge 1 - 1/n.$$

3.1 A sublinear top-k MIPS solver: CEOs

Since the estimation does not use the values of the query signature (except the signature order), after performing Gaussian RP on the data points, we can precompute and sort *all* possible combinations of s concomitants before querying. This leads to an algorithm with $O(1)$ index lookup with recall guarantees, as shown in Algorithm 1.

Sublinear time: Define $\alpha_* = \arg\min_{\mathbf{y}\in\mathbf{X}\setminus\mathbf{x}} \alpha_{\mathbf{y}}$. When $\alpha_*^2 = c/s_0$ for any $c > 1$, Theorem 3.1 shows that CEOs has a sublinear query time due to the sublinear cost of random projections. However, the downside is an exponential indexing space. For a fixed k, building the index takes $O\left((D/s)^s\, n\right)$ time and $O\left((D/s)^s\right)$ space since we have $\binom{D}{s_0}$ different sets I, and each of I has $\binom{D-s_0}{s_0}$ different sets J.

When $D = n^{1/c}$ and $\alpha_*^2 = c/s_0$, CEOs has an exponential indexing space $O\left(n^{s/c}\right) = O\left(n^{1/\alpha_*^2}\right)$. While the value α_* is data and query dependent, we will show that this exponential indexing space is *optimal* for approximate MIPS on a unit sphere.

3.2 Optimality of CEOs on a unit sphere

Since $\arg\max_{\mathbf{x}\in\mathbf{X}} \mathbf{x}^\top \mathbf{q} = \arg\min_{\mathbf{x}\in\mathbf{X}} \|\mathbf{x} - \mathbf{q}\|$ on a unit sphere, we investigate Theorem 3.1 for a decision version of the approximate nearest neighbor search (ANNS) over the Euclidean space to show the optimality in the space usage. For any radius r, the decision version for a data structure that answers $(1+\epsilon)$-ANNS is as follows:

- If there is $\mathbf{x} \in \mathbf{X}$ such that $\|\mathbf{x} - \mathbf{q}\| \le r$, answer YES.
- If there is no $\mathbf{x} \in \mathbf{X}$ such that $\|\mathbf{x} - \mathbf{q}\| \le (1+\epsilon)r$, answer NO.

Andoni et al. [5] analyze the lower bound of the decision problem given a constant probes to the data structure and ignoring all preprocessing time on the Euclidean space. In particular, any algorithm uses a constant number of probes to the data structure must

Algorithm 1 CEOs

function INDEXING(data matrix $\mathbf{X}_{d\times n}$, random Gaussian matrix $\mathbf{R}_{D\times d}$)
  Up-project $\mathbf{X}$ into D dimensions by computing $\mathbf{X}' = \mathbf{R}\mathbf{X}$
  For each pair of sets I and J, each containing distinct s_0 projection indexes among D up-projections and $I \cap J = \emptyset$, partially sort $\sum_{i\in I} \mathbf{X}'_i - \sum_{j\in J} \mathbf{X}'_j$ to add top-k indexes with the largest values to the list $\mathcal{L}_{IJ}$
  return $O\left((D/s)^s\right)$ lists $\mathcal{L}_{IJ}$ where I and J correspond to concomitants of the s_0 maximum and minimum order statistics
end function

function QUERYING(query point $\mathbf{q} \in \mathbb{R}^d$, data matrix $\mathbf{X}_{d\times n}$, random Gaussian matrix $\mathbf{R}_{D\times d}$, our index)
  Up-project $\mathbf{q}$ into D dimensions by computing $\mathbf{q}' = \mathbf{R}\mathbf{q}$
  Compute the sets I and J, each of s_0 projection indexes corresponding to s_0 maximum and minimum values of $\mathbf{q}'$
  return The top-k points from the list $\mathcal{L}_{IJ}$
end function

use $n^{\Omega(1/\epsilon^2)}$ space. Since a high search recall demands very small ϵ, this result presents a fundamental difficulty for any practical data structures that can answer the nearest neighbor search very accurate and fast.

An upper bound for $(1+\epsilon)$-ANNS over the Hamming space is proposed in [16] that uses $n^{O(1/\epsilon^2)}$ space. Our work complements this result by showing a data structure which achieves a constant index lookups and uses $n^{O(1/\epsilon^2)}$ over the Euclidean space on a unit sphere. We show it by deriving and bounding the value $\alpha_{\mathbf{y}}$ where $\|\mathbf{x}-\mathbf{q}\| = r$ and $\|\mathbf{y}-\mathbf{q}\| = (1+\epsilon)r$ as follows.

$$\begin{aligned}\alpha_{\mathbf{y}} &= \frac{\mathbf{x}^\top\mathbf{q} - \mathbf{y}^\top\mathbf{q}}{\sqrt{1-(\mathbf{x}^\top\mathbf{q})^2} + \sqrt{1-(\mathbf{y}^\top\mathbf{q})^2}} \\ &= \frac{\left(1-r^2/2\right) - \left(1-(1+\epsilon)^2 r^2/2\right)}{\sqrt{1-(1-r^2/2)^2} + \sqrt{1-\left(1-(1+\epsilon)^2 r^2/2\right)^2}} \\ &= \frac{\epsilon\,(\epsilon+2)}{\sqrt{4/r^2-1} + (1+\epsilon)\sqrt{4/r^2-(1+\epsilon)^2}}\end{aligned}$$

Hence,

$$\begin{aligned}\frac{\epsilon\,(\epsilon+2)}{(2+\epsilon)\sqrt{4/r^2-1}} \le \alpha_{\mathbf{y}} \le \frac{\epsilon\,(\epsilon+2)}{(2+\epsilon)\sqrt{4/r^2-(1+\epsilon)^2}} \\ \frac{\epsilon}{\sqrt{4/r^2-1}} \le \alpha_{\mathbf{y}} \le \frac{\epsilon}{\sqrt{4/r^2-(1+\epsilon)^2}}\end{aligned}$$

On a unit sphere and for a fixed but not too small radius r, we have $\alpha_* = \Theta(\epsilon)$. This means that CEOs can answer the decision version of $(1+\epsilon)$-ANNS with the Euclidean distance on a unit sphere using $n^{O(1/\epsilon^2)}$ space and the query requires a constant index lookups (ignoring the RP cost). This matches the lower bound on the space usage provided by [5].

4 PRACTICAL APPROXIMATE MIPS SOLVERS

Even though the mild condition of Theorem 3.1 holds and CEOs has sublinear query time, the exponential space and time complexity for building the index is a bottleneck for many big data applications. Before describing several practical CEOs variants that trade the query performance for indexing efficiency, we make some notes on practical settings for the CEOs variants.

First, while the asymptotic behavior of concomitants requires a significanly large D, we observe that the setting $D = O(d)$ suffices in our high-dimensional real-world data sets. Second, we consider a post-processing where we have a *constant* budget of $b > k$ inner product computations to boost the top-k MIPS accuracy.

Our improvements of CEOs are based on the observation that we can implement it as a specific dimensionality reduction. This approach, called *CEOs-Est*, aggregates the concomitants of extreme sth order statistics, i.e. $\sum_{r=1}^{s_0}\left(X_{[r]} - X_{[D-r+1]}\right)$. While CEOs-Est estimates n inner products in $O(n)$ time, it still runs significantly faster than both Gaussian RP and bruteforce search. Especially, we can use CEOs-Est to calculate the search performance of CEOs *before* implementing it.

4.1 Sublinear CEOs for $s = 1, 2$

Despite of the exponential space complexity, we can still use CEOs for small values of s, e.g. $s = 1, 2$. That leads to *1CEOs* and *2CEOs* variants. 1CEOs builds the index in $O(dDn)$ time. It uses $O(dD)$ space since each of D dimensions stores $b = O(1)$ points in d dimensions. The query time is dominated by the RP cost $O(dD)$. When $D = o(n)$, 1CEOs answers MIPS in both sublinear space and time. 2CEOs shares the same query time as 1CEOs but the indexing complexity is quadratic in D.

Equation 3 indicates that, given a sufficiently large D, $X_{[1]}$ corresponding to the top-k MIPS tends to be ranked at the top positions on the sorted list corresponding to $Q_{(1)}$. By increasing b inner products in post-processing, we can achieve higher accuracy of top-k MIPS since we allow larger gap $Y_{[1]} - X_{[1]}$. Our empirical results show that 1CEOs and 2CEOs outperform the sublinear LSH bucket algorithms on both indexing and querying for approximate MIPS.

4.2 Improvement with Threshold Algorithm

We observe that CEOs-Est estimates inner product values by the sum of s positive concomitants and extracts top-b indexes with the largest estimates for post-processing. Therefore, ones can speed up CEOs-Est by using the well-known threshold algorithm (TA) [10]. The pseudo-code of Algorithm *CEOs-TA* that exploits the TA algorithm for speeding up CEOs-Est can be seen in the supplement.

Complexity: CEOs-TA builds the index in $O(dDn + Dn\log n)$ time, which is slightly larger than $O(dDn)$ time of CEOs-Est, but uses the same $O(Dn + dn)$ space. In practice, CEOs-TA provides the same accuracy as CEOs-Est due to the same computation of top-b estimators for post-processing, but runs 5 – 10 times faster, especially when s and b are small. [2]

4.3 Improvement with a co-reduction

While CEOs-TA can speed up CEOs-Est given a similar indexing time and space complexity, we could not bound its running time. Equation 3 shows that we do not need to keep the whole data in the sorted list associated with the dimension of $Q_{(r)}$ for answering MIPS. Exploiting this property, we propose *coCEOs*, a co-reduction

[2] We can also use CEOs-TA to speed up the index construction of CEOs.

Algorithm 2 coCEOs

function INDEXING(data matrix $\mathbf{X}_{d\times n}$, random Gaussian matrix $\mathbf{R}_{D\times d}$, m)
  Up-project $\mathbf{X}$ into D dimensions by computing $\mathbf{X}' = \mathbf{R}\mathbf{X}$
  For each projection $i \in [D]$, partially sort $\mathbf{X}'_i$ to add top-m indexes with the largest and smallest values to the lists $\mathcal{L}_i$ and $\mathcal{S}_i$, respectively
  return $2D$ lists $\mathcal{L}_i$ and $\mathcal{S}_i$ where $i \in [D]$ as our index
end function

procedure QUERYING(query point $\mathbf{q} \in \mathbb{R}^d$, data matrix $\mathbf{X}_{d\times n}$, random Gaussian matrix $\mathbf{R}_{D\times d}$, our index)
  Up-project $\mathbf{q}$ into D dimensions by computing $\mathbf{q}' = \mathbf{R}\mathbf{q}$
  Compute the sets I and J, each of s_0 projection indexes corresponding to s_0 maximum and minimum values of $\mathbf{q}'$
  For each $i \in I$ and $j \in J$, scan all points in lists $\mathcal{L}_i$ and $\mathcal{S}_j$, and update their partial estimates in the histogram
  Retrieve top-b points with the largest partial estimates
  Post-processing: Compute b inner products from these points and return top-k points with the largest value
end procedure

method that keeps a fraction of data in our index and uses a small number of concomitants for answering MIPS. Moreover, coCEOs can govern the query time given a budget computation.

Since one projection can associate with the maximum or minimum order statistics, coCEOs keeps both top-m point indexes with the largest and smallest values. Hence, coCEOs can be seen as a compressed data structure that keeps the top-$2m$ points on each projection $i \in [D]$ for a fixed parameter m. When the query comes, we will choose the smallest or largest top-m indexes depending on the rank of $Q_{(r)}$. We observe that the larger inner product the point has, the more frequent it should be listed on these top-$2m$ points, and hence the larger estimate it has. Therefore, coCEOs will compute *partial* estimates of inner products of the points in the index, as shown in Algorithm 2 .

Complexity: While the indexing space is $O(mD + dn)$, the construction time is $O(dDn + Dn\log m)$. By using a hash table to maintain the histogram of partial estimates, the query time of coCEOs is $O(dD + ms)$. Particularly, coCEOs requires linear space but still runs in sublinear time when $ms = o(n)$.

Practical setting: While CEOs-Est sums the s projections associated with $Q_{(r)}$ for n points, coCEOs computes partial estimates for the points that are likely to be the top-k MIPS. Hence, coCEOs can exploit a larger number of concomitants of extreme statistics compared to CEOs-Est. Let $s' > s$ be the number of concomitants used by coCEOs. In order to govern the running time of coCEOs, we set a budget of B computations. coCEOs selects the top $m = B/s'$ values from each of s' sorted lists associated with $Q_{(r)}$ for aggregation. Hence, coCEOs has $O(B)$ query time and we can use it on the budgeted MIPS setting [26].

4.4 Make CEOs variants more practical

For CEOs variants, both indexing and querying requires Gaussian RP, which takes $O(dD)$ time for one point. This cost is significant and often dominates the query time when d is large. Fortunately, there are several approaches to simulate the Gaussian RP. We will use the Structured Spinners [4] that exploit the fast Hadamard transform to simulate Gaussian RP.

In particular, we generate 3 random diagonal matrices $\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3$ whose values are randomly chosen in $\{+1, -1\}$. The Gaussian RP $\mathbf{R}\mathbf{x}$ can be simulated by the mapping $\mathbf{x} \mapsto \mathbf{H}\mathbf{D}_3\mathbf{H}\mathbf{D}_2\mathbf{H}\mathbf{D}_1\mathbf{x}$ where $\mathbf{H}$ is the Hadamard matrix. With the fast Hadamard transform, the Gaussian RP can be simulated in $O(D\log D)$ time, which will not dominate the query time of the CEOs variants. [3]

5 EXPERIMENT

We implement CEOs variants and other competitors in C++ [4] using -O3 optimization and conduct experiments on a 2.80 GHz core i5-8400 32GB of RAM with a single CPU. We also release a few lines of Python and Matlab codes [5] to demonstrate the simplicity of our methods. We present empirical evaluations on top-k MIPS to verify our claims, including:

(1) CEOs provides very high MIPS recall using a small number of concomitants s, which is consistent with the theory of concomitants of extreme order statistics.
(2) On inferior choice for $s = 1$ and 2, 1CEOs and 2CEOs outperform the LSH bucket algorithms on indexing and querying regarding both space and time.
(3) Both CEOs-TA and coCEOs outperform LSH-based estimation from top-k MIPS solvers on many real-world data sets.
(4) coCEOs is faster than CEOs-TA and its indexing time is faster than ipNSW, a recent graph-based MIPS solver.

We measure the average query time per query in ms of each method. We use $P@b$ = |Retrieved top-$b \cap$ True top-10|/10 to measure the search recall due to the post-processing with b inner product computations. We consider top-10 MIPS and hence when $b = 10$, we do not need post-processing. Other empirical results with different values of k are in the supplement.

5.1 MIPS solvers and data sets

We implement CEOs variants, including (1) *1CEOs* and *2CEOs* as sublinear MIPS solvers, (2) practical variants including *CEOs-Est*, *CEOs-TA*, and *coCEOs* which have $\tilde{O}(dn)$ indexing time and space. All CEOs variants use $D = 2^{\lfloor \log d \rfloor + 1} = O(d)$ to exploit the fast Structured Spinners. We implement recent LSH-based schemes, including *SimpleLSH* [21] and *RangeLSH* [25]. We compare with *dWedge* [18] for budgeted MIPS since it outperforms GreedyMIPS [26]. [6] While our main competitors are LSH-based methods with theoretical guarantees, we also compare CEOs variants with *ipNSW* [19] [7], a

[3] We add up additional zero coordinates if D is not a power of 2.
[4] https://github.com/NinhPham/MIPS
[5] https://drive.google.com/file/d/1cALMtc8u2027rRXc4XR14n4wBSsbrBSD
[6] Our C++ program has the GreedyMIPS implementation.
[7] https://github.com/stanis-morozov/ip-nsw

Table 1: Overview of the data sets

	Cifar10	Nuswide	Yahoo	Msong	Gist	Imagenet	Tiny5m
d	3072	500	300	420	960	150	384
n	49K	270K	625K	1M	1M	2.3M	5M

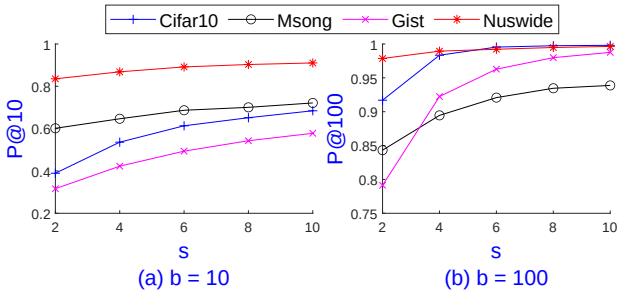

Figure 2: Top-10 MIPS accuracy of CEOs on several data sets with $D = 2^{\lfloor \log d \rfloor + 1}$, varying s, and (a) $b = 10$ and (b) $b = 100$.

Delaunay graph approximation based method that requires $O(dn^2)$ indexing time and does not offer any search recall guarantee. We implement the bruteforce search and LSH hash evaluations with the Eigen-3.3.4 library [8] for fast C++ matrix-vector multiplication.

We conduct experiments on many real-world large-scale data sets, as shown in Table 1. We randomly extract 1000 points (e.g. 1000 user vectors for Yahoo) to form the query set. All randomized results are the average of 5 runs of the algorithms.

5.2 Search recall of CEOs

This subsection shows the search recall of CEOs on top-10 MIPS by implementing CEOs-Est. We observe that the setting $D = 2^{\lfloor \log d \rfloor + 1}$ satisfies $D \approx n^{1/2}$ on our data sets. Figure 2 presents the accuracy $P@b$ of top-10 MIPS when varying s on Cifar10, Msong, Gist, and Nuswide. It is clear that increasing s leads to a substantial rise of accuracy. When $s = 10, b = 100$, CEOs achieves over 90% accuracy on Msong and especially reaches nearly perfect recall on the others. The results confirm the reliability of our theoretical analysis for the sublinear CEOs since the settings are similar to the requirements of Theorem 3.1 i.e. $D = n^{1/c}$ for any $c > 1$.

Given the setting where $D = O(d), s = 2$, and $b = 100$, CEOs in practice uses $O(d^2 + nd)$ space and $O(nd^2)$ time complexity to build the index and achieves at least 80% recall with $O(d \log d)$ RP cost and only 100 inner product computations on these data sets.

5.3 Comparison on sublinear algorithms

This subsection compares the performance of sublinear CEOs, including *1CEOs* and *2CEOs*, with the (l, L)-parameterized LSH bucket algorithms, including *SimpleLSH* and *RangeLSH*. Given b inner product computations in post-processing, CEOs can simply choose top-b points with the largest inner product estimates in the list $\mathcal{L}_{IJ}$. However, it is impossible to tune LSH parameters to return the best b candidates for each query.

Given a fixed number of hash tables L, the number of concatenating hash functions l will control the number of collisions and therefore the candidate set size. The LSH query complexity is dominated by the hash computation $O(dlL)$. For LSH parameter settings, we first fix $L = 512$ as suggested in previous work [21, 22] and select the parameter l for the *highest* search recall given a fixed $b = 100$ candidates in post-processing. RangeLSH uses $p = 4$ partitions and each partition has L/p hash tables. We observe that increasing p will decrease the performance.

Due to the similar results, we report the representative results on Gist on Table 2. More empirical results on Yahoo and detailed explanations on the choice of parameter settings of LSH-based methods can be found in the supplement.

Table 2: Comparison of indexing and querying between 1CEOs, 2CEOs, SimpleLSH ($l = 24$) and RangeLSH ($l = 16, p = 4$) on Gist when fixing $b = 100$ and $L = 512$. Bruteforce needs 417 ms per query and stores data in 7.1 GB.

Algorithms	Index		Query	
	Space	Time	$P@100$	Time
1CEOs	**0.8 GB**	**1.3 mins**	51%	**2.3 ms**
2CEOs	7.5 GB	1.2 hours	**80%**	**2.3 ms**
SimpleLSH_{24}	9.1 GB	1.5 hours	35%	8.1 ms
RangeLSH_{16}	9.1 GB	1.1 hours	46%	5.9 ms

It is clear that 1CEOs and 2CEOs outperform LSH schemes regarding both accuracy and efficiency on indexing and querying. While 2CEOs gives higher search recalls than 1CEOs, its indexing time and space complexity are more significant. Regarding the index, since the data set itself requires 7.1GB, 2CEOs uses only 0.4GB extra storage compared to 2GB of LSH schemes. Because 1CEOs only keeps $b = 100$ points on each of $D = 1024$ projections, its index size is approximately 10% of the data size.

Regarding querying, we observe that the hash computation takes 70% of the query time of LSH schemes, whereas 97% of the query time of 1CEOs and 2CEOs are used for computing $b = 100$ inner products. We observe that 1CEOs with $b = 500$ achieves 80% search recall as 2CEOs but needs 11 ms for each query. Note that the accuracy of 2CEOs is consistent with that of CEOs-Est with $s = 2$, as shown in Figure 2 (b). In other words, ones can foresee the performance of CEOs by implementing and testing CEOs-Est in just few minutes. In contrast, we spent a day to find the best parameter settings for LSH schemes.

5.4 Comparison on estimation algorithms

This subsection shows experiments comparing the performance of CEOs variants to dWedge and LSH code estimation schemes on Gist, Msong, Yahoo, and Tiny5m. While CEOs-Est and LSH code schemes are dimensionality reduction methods, dWedge, CEOs-TA and coCEOs compute partial inner products as estimates to break the $O(n)$ barrier. We note that Gaussian RP often suffers lower accuracy than these algorithms so we do not report it here. [9]

Regarding parameter settings, CEOs-Est and CEOs-TA use $s = 10$, hence share the same accuracy. LSH schemes use the code length $l = 128$ and the `_builtin_popcount` function of compilers for Hamming distance computation. We observe that LSH schemes with $l = 128$ have similar query time as CEOs-Est.

While CEOs-Est and LSH schemes require $O(n)$ times for MIPS, CEOs-TA often runs much faster since it does not have to estimate n inner products. To show the sublinear query time, we will compare CEOs-TA with coCEOs and dWedge using $B = n/100$.

[8] http://eigen.tuxfamily.org/index.php?title=Main_Page

[9] Our C++ program has the Gaussian RP implementation.

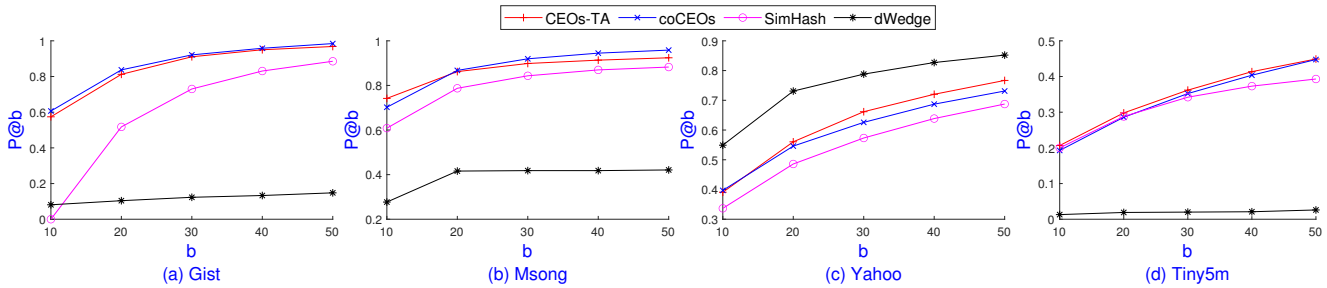

Figure 3: Comparison of top-10 MIPS accuracy between CEOs-TA, coCEOs, SimHash, and dWedge with $D = 2^{\lfloor \log d \rfloor + 1}$, $s = 10$ for CEOs-TA and $s' = 20, B = n/100$ for coCEOs, and varying b on: (a) Gist, (b) Msong, (c) Yahoo, and (d) Tiny5m.

dWedge shares the same query time with coCEOs due to the same mechanism: using B computations to compute partial estimates of inner products. For coCEOs, we set $s' = 20$ since it can use a larger number of extreme order statistics.

Figure 3 shows the accuracy $P@b$ for a wide range of b inner products in post-processing of CEOs-TA, coCEOs, SimHash, dWedge on Gist, Msong, Yahoo, and Tiny5m. It is clear that CEOs variants consistently outperform SimHash on these data sets. Though dWedge only shows advantages on Yahoo, coCEOs can achieve the same recall by $s' = 50$. With $b = 50$, CEOs variants achieve at least 90% recall on Gist and Msong whereas SimHash gives just above 80%. CEOs variants can provide 45% recall on Tiny5m whereas SimHash achieves approximately 35%. On Tiny5m, we can set $b = 100$ and $s = 20$ to increase $P@b$ of CEOs-TA to 80% while SimHash still suffers from very low accuracy. Empirical results on different k values and other LSH schemes are shown in the supplement.

Table 3: Comparison of indexing and querying with $b = 50$ in post-processing between CEOs-Est, CEOs-TA, coCEOs, SimHash, SimpleLSH, RangeLSH ($p = 4$) on Gist. Bruteforce needs 417 ms per query and stores data in 7.1 GB.

Algorithms	Index		Query	
	Space	Time	$P@50$	Time
CEOs-Est	7.6 GB	72 s	**97%**	9 ms
CEOs-TA	14.7 GB	315 s	**97%**	**2.8 ms**
coCEOs	7.5 GB	176 s	**98%**	**2.2 ms**
SimHash_{128}	7.1 GB	30 s	89%	16.7 ms
SimpleLSH_{128}	7.1 GB	30 s	73%	11.3 ms
RangeLSH_{128}	7.1 GB	53 s	76%	15.4 ms

Table 4: Comparison of querying performance between CEOs-TA, coCEOs, and SimHash on Yahoo and Imagenet. Bruteforce uses 74 ms and 179 ms per query for Yahoo and Imagenet, respectively.

Dataset	Querying	SimHash	CEOs-TA	coCEOs
Yahoo	$P@100$	80%	**90%**	**86%**
	Time	10.5 ms	7.3 ms	**0.64 ms**
Imagenet	$P@500$	87%	**96%**	**91%**
	Time	88 ms	–	**28 ms**

Table 3 shows a detailed comparison of CEOs and LSH schemes on indexing and querying with $b = 50$ on Gist. Though CEOs variants have worse indexing space and time (just in minutes) due to the random projections, they outperform LSH schemes regarding the querying performance. CEOs-TA and coCEOs achieve nearly perfect recall with more than 150× speedup compared to bruteforce.

Though CEOs-TA uses more time and space to construct the index, CEOs-TA and coCEOs achieve almost the same search recall and speedup. Empirically, we observe that CEOs-TA needs a few thousands of inner product estimates on these data sets. This cost is nearly the same cost of computing partial inner products with $B = n/100$ of coCEOs and dWedge. Therefore, coCEOs and CEOs-TA share the same query time. For example, they achieve more than 90% accuracy with at least 150× speedup compared to bruteforce on Msong and Nuswide, but we do not report in details here.

5.5 coCEOs and CEOs-TA on high search recalls

We observe that CEOs-TA needs larger values of s and b to achieve significantly high search recalls on some data sets. On Tiny5m, it requires $b = 100$, $s = 40$ to achieve 90% recall. However, this setting incurs a high query time. In contrast, coCEOs with $b = 500$, $s' = 40$ achieves 90% accuracy with at least 100× speedup whereas SimHash 128 bits gives only above 50% accuracy with just 16× speedup. On the settings of large b and s, CEOs-TA's performance is deteriorated and even outperformed by CEOs-Est. This subsection shows that coCEOs is superior to CEOs-TA for high search recalls.

We observe that CEOs-TA requires $s = 20, b = 100, D = 512$ and $s = 50, b = 500, D = 1024$ to achieve 90% accuracy on Yahoo and Imagenet, respectively. coCEOs uses $B = n/100$, $s' = 2s = 40$ on Yahoo and $B = n$, $s' = 2s = 100$ on Imagenet, respectively. Since the SimHash-based estimation shares the same query time with other LSH schemes but achieves higher search recall (see Table 3), we present only SimHash's results with $l = 128$ and $l = 512$ bits code on Yahoo and Imagenet, respectively.

Table 4 shows a detailed comparison of these methods on the querying process on Yahoo and Imagenet. coCEOs outperforms SimHash regarding both search recall and query time. It is 16× and 3× faster than SimHash, and provides 86% and 91% recalls on Yahoo and Imagenet, respectively. Although CEOs-TA achieves the highest recalls with at least 90% on both data sets, it is even slower than bruteforce on Imagenet with $d = 128$. We note that we can increase the coCEOs's accuracy by increasing b. For example,

Table 5: Comparison of indexing time, query time and search recall between coCEOs and ipNSW on Yahoo, Gist, and Imagenet.

Dataset	coCEOs			ipNSW		
	Index	Query	Recall	Index	Query	Recall
Yahoo	**1 min**	**0.6 ms**	**87%**	1.8 h	0.8 ms	66%
Gist	**3 mins**	**1.2 ms**	95%	2.6 h	1.6 ms	**98%**
Imagenet	**17 mins**	9.7 ms	**75%**	1.7 h	**2.3 ms**	71%

coCEOs with $b = 150$ on Yahoo has 90% search recall and still offer at least 100× speedup compared to bruteforce.

5.6 Comparison to ipNSW

This subsection shows that coCEOs can achieve similar querying performance as ipNSW [19], an approximation of Delaunay graph-based method. ipNSW has a similar indexing space as coCEOs but requires significantly higher indexing time due to the graph construction. Besides several internal parameters that controlling the performance, ipNSW has two main parameters to govern the indexing time, including M to govern the sparse neighborhood of the graph and $efConstruction$ to control the recall of the greedy search procedure. The query time of ipNSW depends on $efSearch$, which controls the backtracking steps to avoid stucking into a local minimum caused by the greedy search algorithm. Our experiments use $efConstruction = 100$, $M = 32$. We need $efSearch = 128$ on Imagenet and $efSearch = 10$ on the others for reasonable recalls.

We observe that different budget B of coCEOs requires different values of s' to achieve high search recalls and fast query time. To match the search recalls or query time of ipNSW, we set $B = n/\{1000, 100, 10\}$, $s' = \{40, 40, 30\}$, $b = \{50, 100, 500\}$, $D = 1024$ for Gist, Yahoo, and Imagenet, respectively.

Table 5 shows that coCEOs achieves very competitive query performance, regarding the search recall and query time compared to ipNSW. Especially, coCEOs requires minutes to construct indexes while ipNSW needs hours due to the quadratic inner product computations on constructing the graph.

Since we use the released source code of ipNSW, there might be implementation differences, such as libraries, cache locality, and optimization strategies. Hence, we compare the number of inner products needed for different methods. Since the cost of computing partial inner products is bounded by $2B$, the computational cost of coCEOs is bounded by $2B/d + b$ inner product computations, which are $\{142, 70, 3620\}$ on Yahoo, Gist, and Imagenet, respectively. These numbers are significantly less than that of ipNSW, i.e. $\{904, 663, 4065\}$ inner products for the corresponding data sets. Therefore, we believe that coCEOs can run significantly faster with more optimization strategies.

6 CONCLUSIONS

The paper proposes CEOs, a novel dimensionality reduction for top-k MIPS based on the theory of concomitants of extreme order statistics. Theoretically, we show that CEOs is an optimal dimensionality reduction on a unit sphere. Under a mild condition of data and query distribution, CEOs leads to a sublinear query time with search recall guarantees. Empirically, we propose two variants, including CEOs-TA and coCEOs, using $\tilde{O}(dn)$ indexing space and time. These light-weight indexing CEOs schemes are competitive with recent advanced MIPS solvers. They achieve more than 100× speedup compared to the bruteforce search while returning top-10 MIPS with search recall at least 90% on many large-scale data sets.

We note that the optimality of CEOs on a unit sphere is not in a full range, i.e. all inner product values should not be very close to 1. An open question is whether CEOs is optimal in a full range of inner product values or in the whole space given any order-preserving transformations.

ACKNOWLEDGMENTS

We thank Rasmus Pagh for pointing to the lower bound papers.

REFERENCES

[1] A. Abboud, A. Rubinstein, and R. R. Williams. Distributed PCP theorems for hardness of approximation in P. In *FOCS*, pages 25–36, 2017.
[2] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In *Recommender Systems Handbook*, pages 191–226. Springer, 2015.
[3] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 2008.
[4] A. Andoni, P. Indyk, T. Laarhoven, I. P. Razenshteyn, and L. Schmidt. Practical and optimal LSH for angular distance. In *NIPS*, pages 1225–1233, 2015.
[5] A. Andoni, P. Indyk, and M. Pătraşcu. On the optimality of the dimensionality reduction method. In *FOCS*, pages 449–458, 2006.
[6] Y. Bachrach, Y. Finkelstein, R. Gilad-Bachrach, L. Katzir, N. Koenigstein, N. Nice, and U. Paquet. Speeding up the Xbox recommender system using a euclidean transformation for inner-product spaces. In *RecSys*, pages 257–264, 2014.
[7] M. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, 2002.
[8] B. Chen, Y. Xu, and A. Shrivastava. Fast and accurate stochastic gradient estimation. In *NeurIPS*, pages 12339–12349, 2019.
[9] H. A. David and J. Galambos. The asymptotic theory of concomitants of order statistics. *Journal of Applied Probability*, 11(4):762–770, 1974.
[10] R. Fagin, A. Lotem, and M. Naor. Optimal aggregation algorithms for middleware. *J. Comput. Syst. Sci.*, 66(4):614–656, 2003.
[11] P. Hall. On the rate of convergence of normal extremes. *Journal of Applied Probability*, 16(2):433–439, 1979.
[12] Q. Huang, G. Ma, J. Feng, Q. Fang, and A. K. H. Tung. Accurate and fast asymmetric locality-sensitive hashing scheme for maximum inner product search. In *KDD*, pages 1561–1570, 2018.
[13] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4), 2001.
[14] A. Kabán. Improved bounds on the dot product under random projection and random sign projection. In *SIGKDD*, pages 487–496, 2015.
[15] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.
[16] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *SIAM J. Comput.*, 30(2):457–474, 2000.
[17] P. Li, T. Hastie, and K. W. Church. Very sparse random projections. In *SIGKDD*, pages 287–296, 2006.
[18] S. S. Lorenzen and N. Pham. Revisiting wedge sampling for budgeted maximum inner product search. In *ECML/PKDD*, 2020.
[19] S. Morozov and A. Babenko. Non-metric similarity graphs for maximum inner product search. In *NeurIPS*, pages 4726–4735, 2018.
[20] S. Mussmann and S. Ermon. Learning and inference via maximum inner product search. In *ICML*, pages 2587–2596, 2016.
[21] B. Neyshabur and N. Srebro. On symmetric and asymmetric LSHs for inner product search. In *ICML*, pages 1926–1934, 2015.
[22] A. Shrivastava and P. Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In *NIPS*, pages 2321–2329, 2014.
[23] R. Spring and A. Shrivastava. Scalable and sustainable deep learning via randomized hashing. In *KDD*, pages 445–454, 2017.
[24] M. J. Wainwright. *Basic tail and concentration bounds*, pages 21–57. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019.
[25] X. Yan, J. Li, X. Dai, H. Chen, and J. Cheng. Norm-ranging LSH for maximum inner product search. In *NeurIPS*, pages 2956–2965, 2018.
[26] H. Yu, C. Hsieh, Q. Lei, and I. S. Dhillon. A greedy approach for budgeted maximum inner product search. In *NIPS*, pages 5459–5468, 2017.