Faster accurate sketching for tensor decompositions and tensor networks

Linjian Ma and Edgar Solomonik

Department of Computer Science
University of Illinois Urbana-Champaign

Tensor Networks Reading Group

February 20th, 2024

Presentation overview

Fast and accurate randomized algorithms for low-rank tensor decompositions, NeurIPS 2021

- problem: efficiently sketch the (standard) HOOI algorithm for low-rank Tucker decomposition of sparse tensors
- results: algorithms based on leverage score sampling and TensorSketch; error bounds and experimental analysis

Cost-efficient Gaussian tensor network embeddings for tensor-structured inputs, NeurIPS 2022

- problem: if X is represented by a tensor network, choose a tensor network sketch S to minimize cost of sketching (computing SX)
- results: sufficient condition for JL lemma for any tensor network graph, cost-optimal tensor network sketch under this condition

Tensor

Tensor: multi-dimensional array of data

- Order: number of dimensions of a tensor
- Dimension size: number of elements in each dimension

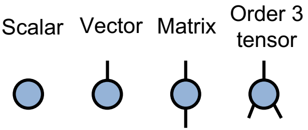
vector	matrix	third order tensor
$\begin{bmatrix}4\\5\end{bmatrix}$	$\begin{bmatrix}1&2\\3&4\end{bmatrix}$	$\begin{bmatrix}1&2\\3&4\end{bmatrix}\begin{bmatrix}5&6\\7&8\end{bmatrix}$

Tensors occur in

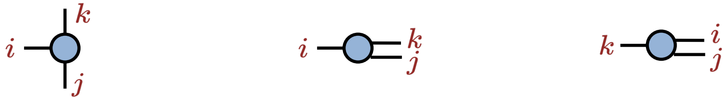
- Data science: image, video, medical data...
- Scientific computing: discretization of high-dimensional functions
- Quantum physics and quantum computing: wavefunction, Hamiltonian, quantum gate

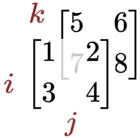
Tensor diagram notation

Tensor diagram: an order N tensor is represented by a vertex with N adjacent edges



Matricization: transform a tensor into a matrix





$$\begin{array}{cc} & k,j \\ i & \begin{bmatrix} 1 & 5 & 2 & 6 \\ 3 & 7 & 4 & 8 \end{bmatrix} \end{array}$$

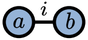
$$\begin{array}{cc} & i,j \\ k & \begin{bmatrix} 1 & 3 & 2 & 4 \\ 5 & 7 & 6 & 8 \end{bmatrix} \end{array}$$

Tensor contraction

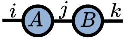
Tensor contraction: summing element products from two tensors over contracted dimensions

A dimension (edge) is contracted if it has no open end

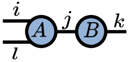
Examples:



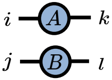
Inner product: $\sum_i a_i b_i$



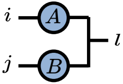
Matrix product : $C_{ik} = \sum_j A_{ij} B_{jk}$



Tensor times matrix: $C_{ilk} = \sum_j A_{ilj} B_{jk}$



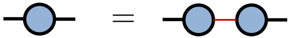
Kronecker/outer product: $T_{ijkl} = A_{ik} B_{jl}$



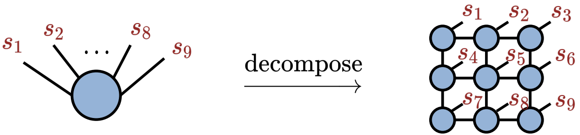
Khatri-Rao product: $T_{ijl} = A_{il} B_{jl}$

Tensor decomposition: break the curse of dimensionality

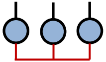
Matrix factorization:



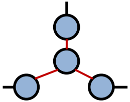
Tensor decomposition: represents a tensor with a (low-rank) tensor network



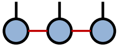
Canonical polyadic (CP) decomposition



Tucker decomposition



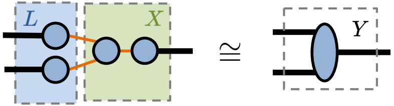
Tensor train decomposition



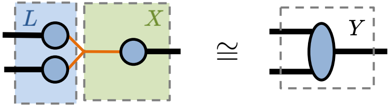
(Rank-constrained) linear least squares with tensor networks

$$\min_{X,\mathrm{rank}(X)\leq R} \|\, L \;\; X \; - \; Y \,\|_F$$

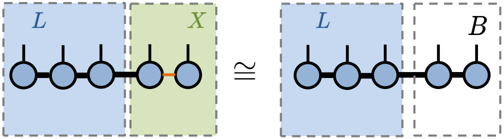
Tucker decomposition



CP decomposition



Tensor train truncation



Sketching for linear least squares

Sketching: randomly project a data L to low dimensional spaces

$$L \longrightarrow SL$$

- $L \in \mathbb{R}^{s \times n}$, $S \in \mathbb{R}^{m \times s}$ with the sketch size $m \ll s$
- S is a random matrix (called embedding)

Sketching for linear least squares

Sketching: randomly project a data L to low dimensional spaces

$$L \longrightarrow SL$$

- $L \in \mathbb{R}^{s\times n}$, $S \in \mathbb{R}^{m\times s}$ with the sketch size $m \ll s$
- S is a random matrix (called embedding)

Standard LLS:

$$X^* = \underset{X}{\operatorname{argmin}} \|LX - Y\|_F$$

Sketched LLS:

$$\hat{X} = \underset{X}{\operatorname{argmin}} \|SLX - SY\|_F$$

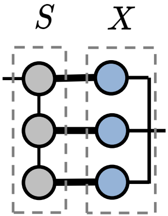
- Gaussian random matrix is standard for embedding
- Sparse embedding[1] can be used when L, Y are sparse (computing SL only costs $\text{nnz}(L)$)

[1] Charikar et al, Finding frequent items in data streams, 2002

Sketching general tensor networks

Problem: Find a tensor network embedding S for the tensor network X, so that

- The embedding is (ϵ, δ)-accurate
- The sketch size (number of rows of S) is low
- Asymptotic cost to compute SX is minimized



An (oblivious) embedding $S \in \mathbb{R}^{m \times s}$ is (ϵ, δ)-accurate if[1]

$$\Pr\left[\left|\frac{\|Sx\|_2 - \|x\|_2}{\|x\|_2}\right| > \epsilon\right] \leq \delta \quad \text{for any } x \in \mathbb{R}^s$$

[1]Woodruff, Sketching as a tool for numerical linear algebra, 2014

Outline: sketching for tensor networks

$$\min_X \|LX - Y\|_F \quad \rightarrow \quad \min_X \|SLX - SY\|_F$$

Sketching for low-rank Tucker decomposition of large and sparse tensors

- L is a Kronecker product of matrices and has orthonormal columns
- A new sketch size upper bound on the problem
- Reach at least 98% of the standard algorithm's accuracy with better cost

Outline: sketching for tensor networks

$$\min_X \|LX - Y\|_F \quad \rightarrow \quad \min_X \|SLX - SY\|_F$$

Sketching for low-rank Tucker decomposition of large and sparse tensors

- L is a Kronecker product of matrices and has orthonormal columns
- A new sketch size upper bound on the problem
- Reach at least 98% of the standard algorithm's accuracy with better cost

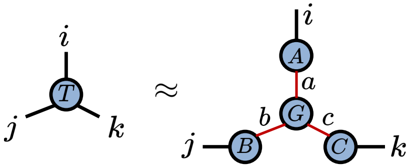
A cost-efficient algorithm to sketch arbitrary tensor network

- L has arbitrary tensor network structure
- Find accurate and cost-optimal embeddings S
- Asymptotically faster than previous works for CP decomposition

Alternating least squares for Tucker decomposition

Tucker decomposition

$$\min_{G,A,B,C} \sum_{i,j,k} \left(T_{ijk} - \sum_{a,b,c} G_{abc} A_{ia} B_{jb} C_{kc} \right)^2$$



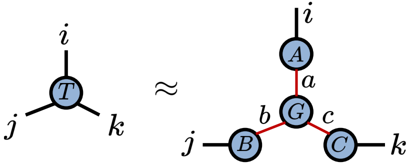
- $T \in \mathbb{R}^{s \times s \times s}$, $X \in \mathbb{R}^{R \times R \times R}$
- $A, B, C \in \mathbb{R}^{s \times R}$ with orthonormal columns, $R < s$

Alternating least squares for Tucker decomposition

Tucker decomposition

$$\min_{G,A,B,C} \sum_{i,j,k} \left(T_{ijk} - \sum_{a,b,c} G_{abc} A_{ia} B_{jb} C_{kc} \right)^2$$

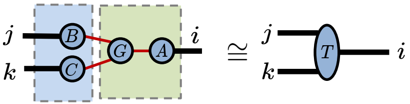
- $T \in \mathbb{R}^{s \times s \times s}$, $X \in \mathbb{R}^{R \times R \times R}$
- $A, B, C \in \mathbb{R}^{s \times R}$ with orthonormal columns, $R < s$



Higher order orthogonal iteration (HOOI)[1]

- Costs $\Omega(\text{nnz}(T)R)$ for arbitrary tensor order
- Fast convergence (usually in around 10 iterations)

$$\min_{X,\text{rank}(X) \le R} || \ L \ \ X \ - \ Y \ ||_F$$

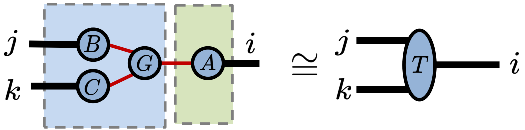


[1] Lathauwer et al, On the best rank-1 and rank-$(R_1, R_2, \ldots, R_n)$ approximation of higher-order tensors, SIMAX 2000

Sketching for Tucker decomposition: previous work

Sketch alternating unconstrained least squares (AULS)[1]

- Advantage: cost with t iterations is $O\left(\mathrm{nnz}(T) + t\left(sR^5 + R^7\right)\right)$
- Disadvantage: not an orthogonal iteration and has slow convergence



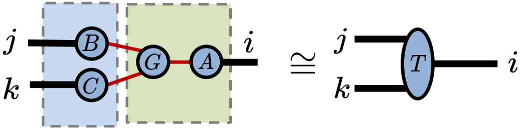
Apply sketching on high-order SVD[2]

- Apply randomized SVD on matricizations of T
- Disadvantages: accuracy lower than HOOI and costs $\Omega(\mathrm{nnz}(T)R)$

[1] Malik and Becker, Low-rank tucker decomposition of large tensors using Tensorsketch, NeurIPS 2018
[2] Ahmadi-Asl et al, Randomized algorithms for computation of Tucker decomposition and HOSVD, IEEE Access 2021

Sketched HOOI for Tucker decomposition

$$\min_{X,\text{rank}(X)\leq R} || \; L \;\; X \; - \; Y \; ||_F$$



HOOI: solve and truncate

$$X^* \leftarrow \underset{X}{\text{argmin}} \|LX - Y\|_F^2$$

$$X_R^* \leftarrow \text{rank-}R \text{ approximation of } X^*$$

$$GA \leftarrow X_R^*$$

Sketched HOOI: sketch, solve and truncate

$$\hat{X} \leftarrow \underset{X}{\text{argmin}} \|SLX - SY\|_F^2$$

$$\hat{X}_R \leftarrow \text{rank-}R \text{ approximation of } \hat{X}$$

$$\hat{G}\hat{A} \leftarrow \hat{X}_R$$

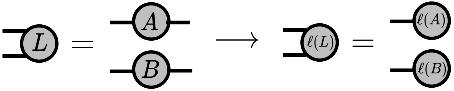
Sketched HOOI for Tucker decomposition

We use efficient embeddings S for solving $\min_X \|SLX - SY\|_F^2$

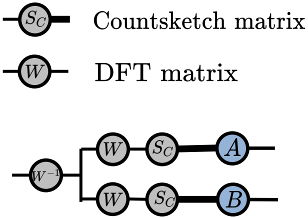
- L is a Kronecker product of factor matrices and changes over iterations
- Y is a matricization of the input tensor and can be sparse

Leverage score sampling

- Sample each row of L based on the leverage score vector $\ell(L)$



Tensorsketch: tensorized Countsketch[1]



[1] Pham and Pagh, Fast and scalable polynomial kernels via explicit feature maps, KDD 2013

Sketched HOOI for Tucker decomposition

We derive sketch size bounds so that

$$\left\|L\hat{X}_R - Y\right\|_F^2 \leq (1 + O(\epsilon)) \left\|LX_R^* - Y\right\|_F^2$$

- $X_R^*, \hat{X}_R$: optimal and the sketched solution
- We apply Mirsky's inequality[1] to bound change in singular values of the sketched L
- Sketch size upper bound is at most $O(1/\epsilon)$ times that for unconstrained LS

[1] Mirsky, The Quarterly journal of mathematics, 1960

Sketched HOOI for Tucker decomposition

We derive sketch size bounds so that

$$\left\|L\hat{X}_R - Y\right\|_F^2 \leq (1 + O(\epsilon)) \left\|LX_R^* - Y\right\|_F^2$$

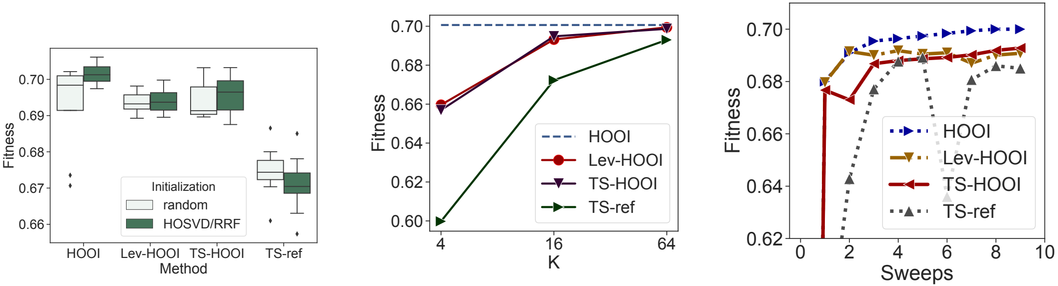
- $X_R^*, \hat{X}_R$: optimal and the sketched solution
- We apply Mirsky's inequality[1] to bound change in singular values of the sketched L
- Sketch size upper bound is at most $O(1/\epsilon)$ times that for unconstrained LS

Algorithm performs well in experiments

- Sketched HOOI converges to at least 98% of the accuracy of standard HOOI
- With leverage score sampling, cost with t iterations is $O\left(\mathrm{nnz}(T) + t\left(sR^3 + R^6\right)\right)$

[1]Mirsky, The Quarterly journal of mathematics, 1960

Experiments: tensors with spiked signal



(a) 5 sweeps, sample size $16R^2$ (b) 5 sweeps, sample size KR^2 (c) sample size $16R^2$

- $T = T_0 + \sum_{i=1}^{5} \lambda_i a_i \circ b_i \circ c_i$, each a_i, b_i, c_i has unit 2-norm, $\lambda_i = 3\frac{\|T_0\|_F}{i^{1.5}}$
- Leading low-rank components obey the power-law distribution
- Tensor size $200 \times 200 \times 200$, $R = 5$
- TS-ref: sketched AULS with TensorSketch

Sketching general tensor networks

Goal: accurately and efficiently sketch arbitrary tensor network structure

Sketching general tensor networks

Previous work:

- Kronecker product embedding[1]: inefficient in computational cost
- Tree embedding (e.g. tensor train)[1,2]: efficient for specific data (Kronecker product, tensor train), but efficiency unclear for general tensor network data

[1] Ahle et al, Oblivious sketching of high-degree polynomial kernels, SODA 2020
[2] Rakhshan and Rabusseau, Tensorized random projections, AISTATS 2020

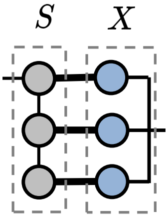
Sketching general tensor networks

Previous work:

- Kronecker product embedding[1]: inefficient in computational cost
- Tree embedding (e.g. tensor train)[1,2]: efficient for specific data (Kronecker product, tensor train), but efficiency unclear for general tensor network data

Assumptions throughout our analysis:

- Multiply $A, B \in \mathbb{R}^{n \times n}$ has a cost of $O(n^3)$
- S is a Gaussian tensor network defined on graphs
- Each dimension to be sketched has large size

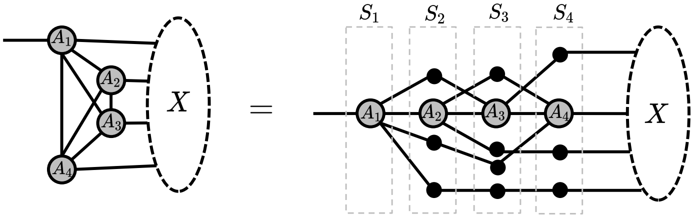


[1] Ahle et al, Oblivious sketching of high-degree polynomial kernels, SODA 2020
[2] Rakhshan and Rabusseau, Tensorized random projections, AISTATS 2020

Sufficient condition for (ϵ, δ)-accurate embedding

The embedding is accurate if we can rewrite $S = S_1 \cdots S_N$ and

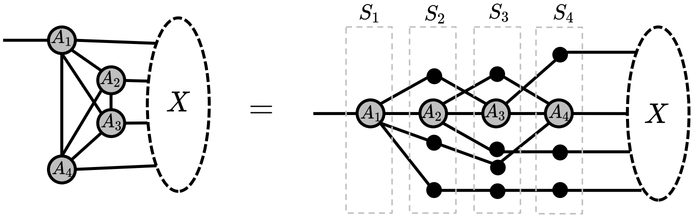
- S_i is the Kronecker product of A_i (a Gaussian random matrix) and identity matrices
- A_i has row size $\Omega(N \log(1/\delta)/\epsilon^2)$



Sufficient condition for (ϵ, δ)-accurate embedding

The embedding is accurate if we can rewrite $S = S_1 \cdots S_N$ and

- S_i is the Kronecker product of A_i (a Gaussian random matrix) and identity matrices
- A_i has row size $\Omega(N \log(1/\delta)/\epsilon^2)$

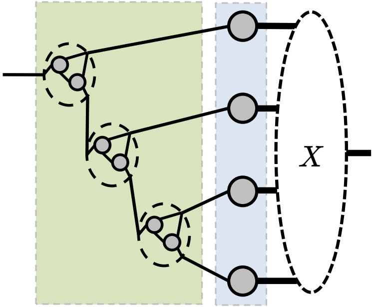


Two key prior results used in the proof[1]

- If A_i is (ϵ, δ)-accurate, so is the Kronecker product between A_i and identity matrices
- If $S_1, \ldots, S_N$ are $(\epsilon/\sqrt{N}, \delta)$-accurate, $S_1 \cdots S_N$ is $(O(\epsilon), \delta)$-accurate

[1] Ahle et al, Oblivious sketching of high-degree polynomial kernels, SODA 2020

A sketching algorithm with efficient computational cost and sketch size



Embedding containing a Kronecker product embedding + binary tree of gadgets

Each small gadget sketches the product of two tensors

- Each gadget contains a pair of tensors
- Dimension sizes in each gadget are chosen based on data tensors to minimize cost
- Can reduce cost by $O(\sqrt{m})$ compared to containing one tensor

Analysis of the algorithm

c: asymptotic sketching cost for our algorithm

c_{opt}: optimal asymptotic sketching cost under the embedding sufficient condition

m: sketch size

Input data tensor network structure	Optimality of the algorithm
General hypergraph	$c = O(\sqrt{m} \cdot c_{\text{opt}})$
General graph	$c = O(m^{0.375} \cdot c_{\text{opt}})$
Each data tensor has a dimension to be sketched (e.g. Kronecker product, tensor train)	$c = c_{\text{opt}}$

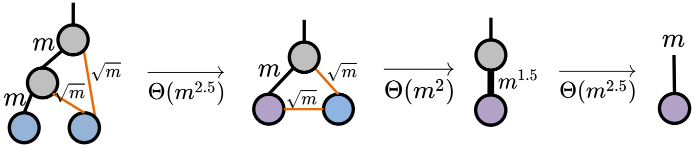
Analysis of the algorithm

Lower bound analysis

- When the data contains 2 tensors, sketching lower bound can be derived
- Kronecker product case: when the data has two vectors with size m (sketch size), the sketching computational cost is $\Omega(m^{2.5})$
- When the data has more tensors, for a given contraction path the lower bound is the sum of two-tensor-contraction lower bounds

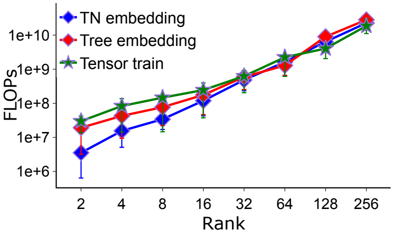
Algorithm design

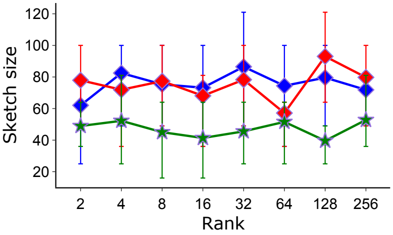
- For the 2-tensor data, can design embedding attaining the lower bound



- For the data with more tensors, we can derive the optimal way to sketch using the two-tensor scheme for a given contraction path
- We can try all data contraction paths to get the optimal sketching path

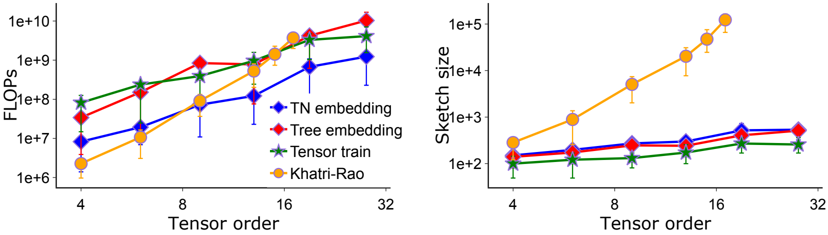
Experiments: sketching a tensor train data





- Input tensor train: order 6, each dimension size $s = 500$ with varying rank
- TN embedding: Kronecker product + a binary tree of small gadgets
- Tree embedding: Kronecker product + a binary tree tensor network
- Sketching error is within 0.1
- Our TN embedding achieves the best asymptotic cost for all tensor train ranks

Experiments: sketching a Kronecker product data



- Input data: each dimension size $s = 1000$ with varying number of orders
- Sketching error is within 0.1
- Our TN embedding achieves the best asymptotic cost
- TN, tree, and tensor train embeddings have efficient sketch size

Application: CP decomposition with alternating least squares

Algorithm for CP-ALS	per-iteration cost	preparation cost
standard ALS	$\Theta(s^N R)$	/
leverage score sampling[1]	$\tilde{\Theta}(N(R^{N+1} + sR^N)/\epsilon^2)$	$\Theta(s^N)$
recursive leverage score sampling[2]	$\tilde{\Theta}(N^2(R^4 + NsR^3/\epsilon)/\delta)$	$\Theta(s^N)$
Our algorithm	$\tilde{\Theta}(N^2(N^{1.5}R^{3.5}/\epsilon^3 + sR^2)/\epsilon^2)$	$\Theta(s^N m)$

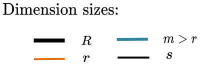
- When performing a low-rank CP decomposition with $s \gg R^{1.5}$, our algorithm is $\Theta(NR\epsilon/\delta) = \Omega(NR)$ times better than the recursive leverage score sampling
- Larger preparation cost is needed
- Sparse tensor network embeddings based on CountSketch and sampling[3] can be used to reduce the preparation cost and have better dependency on ϵ, N

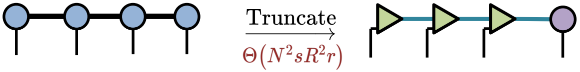
[1] Larsen and Kolda, Practical leverage-based sampling for low-rank tensor decomposition, SIMAX 2022
[2] Malik, More Efficient Sampling for Tensor Decomposition With Worst-Case Guarantees, ICML 2022
[3] Bharadwaj et al, Fast exact leverage score sampling from Khatri-Rao products with applications to tensor decomposition, Neurips 2023.

Application: truncation of high-rank tensor train

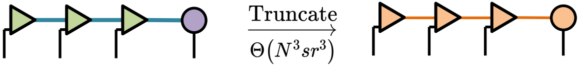
Standard tensor train truncation algorithms have a cost of $\Theta(NsR^3)$, we can achieve a better cost of $\Theta(NsR^2(Nr))$ using sketching when $R > Nr$

- Use randomized range finder with TN embedding to reduce the bond dimension to $m = \Theta(Nr)$





- Use the standard truncation algorithm to reduce the bond dimension from $m = \Theta(Nr)$ to r



Application: truncation of high-rank tensor train

- Analysis assumes each physical dimension size s greater than the output tensor train rank r
- When $s < r$, using our embedding without the Kronecker product part can still yield a cost of $\Theta(N^2 s R^2 r)$
- The cost $\Theta(N^2 s R^2 r)$ attains the asymptotic cost lower bound under the embedding sufficient condition
- Previous work[1] uses tensor train embedding on tensor train truncation, and our analysis shows it yields the same asymptotic cost and is also efficient

[1] Daas et al, Randomized algorithms for rounding in the Tensor-Train format, SISC 2023

Conclusion

Sketching for low-rank Tucker decomposition of large and sparse tensors

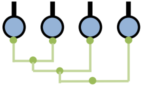
- Accurately sketch rank-constrained linear least squares problem arising in Tucker ALS
- Reach at least 98% of the standard algorithm's accuracy with input-sparsity cost

A cost-efficient algorithm to sketch arbitrary tensor network

- Seek cost-optimal accurate embeddings for a given tensor network-structured input data
- Achieve asymptotically faster sketching algorithms for low-rank tensor network approximations

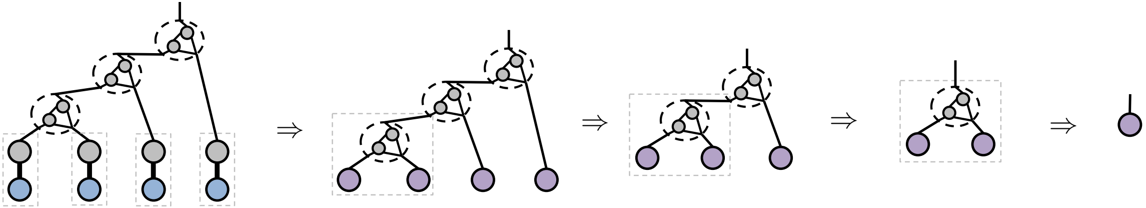
Backup slides

Example: sketching Kronecker product data



Consider contracting an input Kronecker product from left to the right

Sketching contraction path as follows



Our algorithm reduces cost by up to $O(\sqrt{m})$ for the same accuracy compared to using tree embeddings[1]

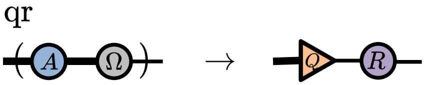
[1] Ahle et al, Oblivious sketching of high-degree polynomial kernels, SODA 2020

Randomized SVD using sketching

Given a matrix $A \in \mathbb{R}^{m \times n}$, find a rank-$r$ approximation with $r \ll m, n$ in the SVD form

Randomized range finder[1]

- Generate a random embedding matrix $\Omega \in \mathbb{R}^{n \times \Theta(r)}$
- $Q, R \leftarrow \mathrm{qr}(A\Omega)$, so $Q \in \mathbb{R}^{m \times \Theta(r)}$

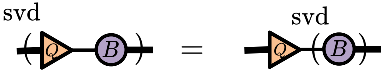


Dimensionality reduction

- $B \leftarrow Q^T A$

SVD on the low-rank matrix QB

- $Q_B, \Sigma, V_B^T \leftarrow \mathrm{svd}(B)$
- Return QQ_B, Σ, V_B^T



[1] Nathan, Martinsson, and Tropp, Finding structure with randomness, SIAM review 2011

Sketched rank-constrained linear least squares problem

Proof sketch: when S is a $(1/2, \delta, \epsilon)$-accurate sketching matrix

$$\|LP_R - Y\|_F^2 = \left\|Y^\perp\right\|_F^2 + \underbrace{\|P_R - P_{\text{opt}}\|_F^2}_{\text{low rank truncation error}}$$

$$\left\|L\widehat{P}_R - Y\right\|_F^2 = \left\|Y^\perp\right\|_F^2 + \underbrace{\left\|\widehat{P}_{\text{opt}} - P_{\text{opt}}\right\|_F^2}_{\text{sketched least squares error}} + \underbrace{\left\|\widehat{P}_{\text{R}} - \widehat{P}_{\text{opt}}\right\|_F^2 + 2\left\langle \widehat{P}_{\text{R}} - \widehat{P}_{\text{opt}}, \widehat{P}_{\text{opt}} - P_{\text{opt}} \right\rangle_F}_{\text{sketched low rank truncation error}}$$

Sketched rank-constrained linear least squares problem

Proof sketch: when S is a $(1/2, \delta, \epsilon)$-accurate sketching matrix

$$\|LP_R - Y\|_F^2 = \left\|Y^\perp\right\|_F^2 + \underbrace{\|P_R - P_{\text{opt}}\|_F^2}_{\text{low rank truncation error}}$$

$$\left\|L\widehat{P}_R - Y\right\|_F^2 = \left\|Y^\perp\right\|_F^2 + \underbrace{\left\|\widehat{P}_{\text{opt}} - P_{\text{opt}}\right\|_F^2}_{\text{sketched least squares error}} + \underbrace{\left\|\widehat{P}_{\text{R}} - \widehat{P}_{\text{opt}}\right\|_F^2 + 2\left\langle \widehat{P}_{\text{R}} - \widehat{P}_{\text{opt}}, \widehat{P}_{\text{opt}} - P_{\text{opt}}\right\rangle_F}_{\text{sketched low rank truncation error}}$$

- $\left\|\widehat{P}_{\text{opt}} - P_{\text{opt}}\right\|_F^2 = O(\epsilon^2)\left\|Y^\perp\right\|_F^2$ (Drineas et al., Numerische mathematik 2011)
- $\left\|\widehat{P}_{\text{R}} - \widehat{P}_{\text{opt}}\right\|_F^2 = \|P_R - P_{\text{opt}}\|_F^2 + O(\epsilon)\|LP_R - Y\|_F^2$ (Mirsky's inequality, (Mirsky, The Quarterly journal of mathematics, 1960))
- $\left\langle \widehat{P}_{\text{R}} - \widehat{P}_{\text{opt}}, \widehat{P}_{\text{opt}} - P_{\text{opt}}\right\rangle_F = O(\epsilon)\|LP_R - Y\|_F^2$ (Mirsky's inequality)

Application: randomized hierarchical SVD for tensor train truncation

Use randomized range finder to reduce the bond dimension to $m = \Theta(Nr)$