Practical Leverage-Based Sampling for Low-Rank Tensor Decomposition

Tamara G. Kolda  Brett Larsen
Sandia National Labs  Stanford University

SIAM Annual Meeting 2020
Development in Machine Learning: Foundations and Applications
July 8, 2020

Tensors Come From Many Applications

- **Chemometrics**: Emission x Excitation x Samples (Fluorescence Spectroscopy)
- **Neuroscience**: Neuron x Time x Trial
- **Criminology**: Day x Hour x Location x Crime (Chicago Crime Reports)
- **Machine Learning**: Multivariate Gaussian Mixture Models Higher-Order Moments
- **Transportation**: Pickup x Dropoff x Time (Taxis)
- **Sports**: Player x Statistic x Season (Basketball)
- **Cyber-Traffic**: IP x IP x Port x Time
- **Social Network**: Person x Person x Time x Interaction-Type
- **Signal Processing**: Sensor x Frequency x Time
- **Trending Co-occurrence**: Term A x Term B x Time
Tensors Come From Many Applications

- **Chemometrics**: Emission x Excitation x Samples (Fluorescence Spectroscopy)
- **Neuroscience**: Neuron x Time x Trial
- **Criminology**: Day x Hour x Location x Crime (Chicago Crime Reports)
- **Machine Learning**: Multivariate Gaussian Mixture Models
- **Transportation**: Pickup x Dropoff x Time (Taxis)
- **Sports**: Player x Statistic x Season (Basketball)
- **Cyber-Traffic**: IP x IP x Port x Time
- **Social Network**: Person x Person x Time x Interaction-Type
- **Signal Processing**: Sensor x Frequency x Time
- **Trending Co-occurrence**: Term A x Term B x Time

Tensor Decomposition Finds Patterns in Massive Data (Unsupervised Learning)
Tensor Decomposition Identifies Factors

\[ \mathbf{X} = \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d+1}} \]

Data

\[ \mathbf{M} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_{d+1}] \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d+1}} \]

CP Model

Sum of \( r \) Outer Product Tensors

\[ \mathbf{A}_k \in \mathbb{R}^{n_k \times r} \]

Factor Matrices

\[ \mathbf{x}_i = x(i_1, i_2, \ldots, i_{d+1}) \]

\[ m_i = m(i_1, i_2, \ldots, i_{d+1}) = \sum_{j=1}^{r} \prod_{k=1}^{d+1} a_k(i_k, j) \]

Model Rank
Example Sparse Multiway Data: Reddit

- Reddit is an American social news aggregator, web content rating, and discussion website
  - A “subreddit” is a discussion forum on a particular topic
- Tensor obtained from frost.io (http://frostt.io/tensors/reddit-2015/)
  - Built from reddit comments posted in the year 2015
  - Users and words with less than 5 entries have been removed

Reddit Tensor

- 8 million users
- 200 thousand subreddits
- 8 million words

4.7 billion non-zeros (10^{-8}\%)
106 gigabytes

\[ x(i, j, k) = \log(1 + \text{the number of times user } i \text{ used word } j \text{ in subreddit } k) \]

Used a rank \( r = 25 \) decomposition

Interpreting Reddit Components

Reddit Tensor

Component 6

Compute rank $r=25$

Model

variance explained $\approx 6\%$

Relative Weights of 25 Components

Largest of 200K entries, in absolute value

Largest entries, in absolute value

Largest of 8M entries, in absolute value

Top Subreddits

Color-coded by overall frequency

Top Words

Color-coded by overall frequency

Computing $\text{rank } r = 25$

Model variance explained $\approx 6\%$

Relative Weights of 25 Components

Top Subreddits

Color-coded by overall frequency

Top Words

Color-coded by overall frequency

7/8/2020

Kolda - AN08 - Developments in ML
Example Reddit Components Include Rare Words Apropos to High-Scoring Reddits

Component #6: International News

Component #8: Relationships
Example Reddit Components Include Rare Words Apropos to High-Scoring Reds


Component #11: Sports
Tensor Decomposition Identifies Factors

\[ \mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d+1}} \]

\[ \mathbf{M} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_{d+1}] \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d+1}} \]

\[ m_i = m(i_1, i_2, \ldots, i_{d+1}) = \sum_{j=1}^{r} \prod_{k=1}^{d+1} a_{kj}(i_k, j) \]

Key Idea: Alternate among the \( d \) factor matrices, fixing all but that one and solving. Each subproblem is linear least squares.
Prototypical CP Least Squares Problem has Khatri-Rao Product (KRP) Structure

\[ \min_B \| Z B^T - X^T \|^2 \]

\[ N \gg r, n \]

Linking back to mode-\((d+1)\) least squares subproblem

\[ N = \prod_{k=1}^{d} n_k \]

\[ n = n_{d+1} \]

\[ Z \in \mathbb{R}^{N \times r} \]

\[ B^T \in \mathbb{R}^{r \times n} \]

Khatri-Rao Product (KRP) Structure

Unknown

May Be Very Sparse

\[ B = A_{d+1} \]

\[ X^T \in \mathbb{R}^{N \times n} \]

\[ Z = A_d \odot \cdots \odot A_1 \]

\[ X = X_{(d+1)} \]

- KRP costs \( O(Nr) \) to form
- System costs \( O(Nnr^2) \) to solve
- KRP structure
  - Cost reduced to \( O(Nnr) \)
- KRP structure + data sparse
  - Cost reduced to \( O(r \text{ nnz}(X)) \)
Structure of Khatri-Rao Product (KRP): Hadamard Combinations of Rows of Inputs

KRP of $d$ Matrices:  
$$Z = A_d \odot \cdots \odot A_1$$

Each row of KRP is Hadamard product of specific rows in Factor Matrices:

$$Z(i,:) = A_1(i_1,:) \ast \cdots \ast A_d(i_d,:)$$

where

$$i = (n_{d-1} \cdots n_1)(i_d - 1) + (n_{d-2} \cdots n_1)(i_{d-1} - 1) + \cdots + n_1(i_2 - 1) + i_1 \quad \in [N]$$

1-1 Correspondence between linear index and multi index:

$$i \in [N] \Leftrightarrow (i_1, \ldots, i_d) \in [n_1] \otimes \cdots \otimes [n_d]$$

Number of columns is the same in all input matrices, but number of rows varies.
Ingredient #1: Sample Subset of Rows in Overdetermined Least Squares System

\[ \min_B \| Z B^T - X^T \|^2 \]

\[ Z \in \mathbb{R}^{N \times r} \quad B^T \in \mathbb{R}^{r \times n} \quad X^T \in \mathbb{R}^{N \times n} \]

Khatri-Rao Product (KRP) Structure

Unknown

May Be Very Sparse

\[ N \gg r, n \]

\[ \min_B \| \Omega Z B^T - \Omega X^T \|^2 \]

\[ \Omega Z \in \mathbb{R}^{s \times r} \quad B^T \in \mathbb{R}^{r \times n} \quad \Omega X^T \in \mathbb{R}^{s \times n} \]

Sampled KRP

Unknown

Sampled Data

Complexity reduced from \( O(Nnr) \) to \( O(snr^2) \)

Key surveys:
M. W. Mahoney, *Randomized Algorithms for Matrices and Data*, 2011;

How sample so that solution of sampled problem yields something close to the optimal residual of the original problem?
Pick a single random index $\xi$ with probability $p_{\xi}$

Choose

$$\Omega = \begin{bmatrix} 0 & \cdots & 0 & \frac{1}{\sqrt{p_\xi}} & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{N \times 1}$$

Then (assuming all $p_i$ positive) the sampled the sampled residual equals true residual in expectation:

$$\mathbb{E}\|\Omega ZB^T - \Omega X^T\|^2 = \sum_{i=1}^{N} p_i \left( \left\| \frac{1}{\sqrt{p_i}} Z(i,:)B^T - \frac{1}{\sqrt{p_i}} X^T(i,:) \right\|^2 \right) = \|ZB^T - X^T\|^2$$

Pick $s$ random indices $\xi_j$ (with replacement) such that $P(\xi_j = i) = p_i$.

Choose $\Omega \in \mathbb{R}^{s \times N}$ such that

$$\omega(j, i) = \begin{cases} \frac{1}{\sqrt{sp_i}} & \text{if } \xi_j = i \\ 0 & \text{otherwise} \end{cases}$$

Each row has a single nonzero!

Then, as before, we have:

$$\mathbb{E}\|\Omega ZB^T - \Omega X^T\|^2 = \|ZB^T - X^T\|^2$$

**Ingredient #3: Use Factor Matrix Leverage Scores for Sampling Probabilities (Main Thm)**

Given linear system: \[ \| Z B^T - X^T \|^2 \text{ with } Z = A_d \odot \cdots \odot A_1 \in \mathbb{R}^{N \times r}, X^T \in \mathbb{R}^{n \times N} \]

Define sampling probabilities: \[ p_i = \frac{1}{s^d} \prod_{k=1}^{d} \ell_{ik}(A_k) \text{ for all } i \in [N] \]

And random sampling matrix: Pick a \( s \) random indices \( \xi_j \) such that \( P(\xi_j = i) = p_i \) and define \( \Omega \in \mathbb{R}^{s \times N} \) with \( \omega(j,i) = \begin{cases} \frac{1}{\sqrt{s p_i}} & \text{if } \xi_j = i \\ 0 & \text{otherwise} \end{cases} \)

Solve sampled problem: \[ \tilde{B}_* \equiv \arg \min_{B \in \mathbb{R}^{r \times n}} \| \Omega Z B^T - \Omega X \|_F^2 \]

Get probabilistic error bound: With probability \( 1 - \delta \) for \( \delta \in (0,1) \), we have \[ \| Z \tilde{B}_*^T - X^T \|_F^2 \leq (1 + O(\epsilon)) \| Z B_*^T - X^T \|_F^2 \]

when number of samples satisfies: \[ s = O(r^d \log(n/\delta)/\epsilon^2) \]

Leverage Scores \[ \ell_{ik}(A_k) = \| Q_k(i_k,:) \|_2 \] where \( Q_k \) is orthonormal basis for column space of \( A_k \)

1-1 Correspondence between linear index and multi index: \( i \in [N] \Leftrightarrow (i_1, \ldots, i_d) \in [n_1] \odot \cdots \odot [n_d] \)
**Ingredient #4: Bound Leverage Scores**

**Ingredient #5: Efficient Sampling**

KRP: \( Z = A_d \odot \cdots \odot A_1 \)

**Upper Bound on Leverage Score**

**Lemma** (Cheng et al., NIPS 2016; Battaglino et al., SIMAX 2018):

\[
\ell_i(Z) \leq \prod_{k=1}^{d} \ell_{i_k}(A_k)
\]

Recall probability of sampling row \( i \)

\[
p_i = \frac{1}{r^d} \prod_{k=1}^{d} \ell_{i_k}(A_k)
\]

Too expensive to calculate \( O(Nr^2) \)

Cheap to calculate individual leverage scores \( O(r^2 \sum_k n_k) \)

1-1 Correspondence between linear index and multi index:

\[ i \in [N] \Leftrightarrow (i_1, \ldots, i_d) \in [n_1] \otimes \cdots \otimes [n_d] \]

But still don’t want to consider all \( N \) possible combinations corresponding to all rows of \( Z \)!
Ingredient #4: Bound Leverage Scores
Ingredient #5: Efficient Sampling

KRP: $Z = A_d \odot \cdots \odot A_1$

$A_1 \in \mathbb{R}^{n_1 \times r}$

$A_2 \in \mathbb{R}^{n_2 \times r}$

$\vdots$

$A_d \in \mathbb{R}^{n_d \times r}$

$\Omega Z \in \mathbb{R}^{8 \times r}$

Row $i$ of $Z$

Sampled KRP

Upper Bound on Leverage Score

Lemma (Cheng et al., NIPS 2016; Battaglino et al., SIMAX 2018):

$$\ell_i(Z) \leq \prod_{k=1}^{d} \ell_{i_k}(A_k)$$

Cheap to calculate individual leverage scores
$O(r^2 \sum_k n_k)$

Too expensive to calculate
$O(Nr^2)$

Recall probability of sampling row $i$

$$p_i \equiv \frac{1}{r^d} \prod_{k=1}^{d} \ell_{i_k}(A_k)$$

But still don’t want to consider all $N$ possible combinations corresponding to all rows of $Z$!

1-1 Correspondence between linear index and multi index:

$i \in [N] \Leftrightarrow (i_1, \ldots, i_d) \in [n_1] \otimes \cdots \otimes [n_d]$
Ingredient #6: Combine Repeated Rows

Problem: Concentrated sampling probabilities identify a few key rows but can lead to many repeats!

Least Squares Problems from Real-world Tensor Data Sets

Example 1: \( N = 3.2 \times 10^{12}, s = 2^{17}, \tau = \frac{1}{s} = 8 \times 10^{-6} \)
\[ D = \{ i : p_i > \tau \}, |D| \approx 15000, \sum_{i \in D} p_i = 0.51 \]

Example 2: \( N = 8.7 \times 10^{12}, s = 2^{17}, \tau = \frac{1}{s} = 8 \times 10^{-6} \)
\[ D = \{ i : p_i > \tau \}, |D| \approx 10000, \sum_{i \in D} p_i = 0.41 \]

Example 3: \( N = 8.6 \times 10^{12}, s = 2^{17}, \tau = \frac{1}{s} = 8 \times 10^{-6} \)
\[ D = \{ i : p_i > \tau \}, |D| \approx 7000, \sum_{i \in D} p_i = 0.25 \]

Combining repeat rows ⇒ 2-20X speedup
Ingredient #7: Hybrid Deterministic and Randomly-Sampled Rows

Deterministic Rows

\[ \mathcal{D}_\tau = \{ i \in [N] \mid p_i \geq \tau \} \]

\[ s_{\text{det}} = |\mathcal{D}_\tau| \]

\[ p_{\text{det}} = \sum_{i \in \mathcal{D}_\tau} p_i \]

for \( i \in \mathcal{D}_\tau \) do
  add row \( A_1(i_1,:) \ast \cdots \ast A_d(i_d,:) \)
end for

Random Rows

\[ s_{\text{rnd}} = s - s_{\text{det}} \]

for \( j = 1 \ldots, s_{\text{rnd}} \) do
  repeat
    for \( k = 1 \ldots, d \) do
      \( i_k \leftarrow \text{multi}(\ell(A_k)/r) \)
    end for
  repeat until \( i \notin \mathcal{D}_\tau \)
  \( \omega \leftarrow \sqrt{(1 - p_{\text{det}})/(s_{\text{rnd}} p_i)} \)
  add row \( \omega (A_1(i_1,:) \ast \cdots \ast A_d(i_d,:)) \)
end for

for \( i \in [N] \) do
  \( (i_1, \ldots, i_d) \in [n_1] \otimes \cdots \otimes [n_d] \)
end for

1-1 Correspondence between linear index and multi index:
Ingredient #8: Find All High-Probability Rows without Computing All Probabilities

• Recall

\[ p_i \equiv \frac{1}{\tau^d} \prod_{k=1}^{d} \ell_i (A_k) \]

• For given tolerance \( \tau > 1/N \), define the set of deterministic rows to include

\[ D_\tau = \{ i \in [N] \mid p_i \geq \tau \} \]

- Compute without computing all \( p_i \) values
- A few high leverage scores means all the others are necessarily low!
- Use bounding procedure to eliminate most options
- Compute products of at most a top few leverage scores in each mode

Sorted Leverages Scores (Descending)

\[ n_1 \]
\[ n_2 \]
\[ n_3 \]

1-1 Correspondence between linear index and multi index:

\[ i \in [N] \iff (i_1, \ldots, i_d) \in [n_1] \otimes \cdots \otimes [n_d] \]
Ingredient #9: Efficiently Extract RHS from (Sparse) Unfolded Data Tensor

\[ \min_B \| Z B^T - X^T \|^2 \]

\[ \min_B \| \Omega Z B^T - \Omega X^T \|^2 \]

- Never form \( X^T \) explicitly
- Precompute linear indices for every nonzero and every mode
- Results is sparse RHS

Similar in spirit to ideas for dense tensors in Battaglino et al., SIMAX 2018
Numerical Results
Single Least Squares Problem with $N = 46M$ rows, $r = 10$ columns, $n = 183$ right-hand sides

\[ \Omega Z \in \mathbb{R}^{s \times r}, \quad B^T \in \mathbb{R}^{r \times n}, \quad \Omega X^T \in \mathbb{R}^{s \times n} \]

\[
\begin{align*}
\tilde{B}_* & \equiv \arg \min_{B \in \mathbb{R}^r} \| \Omega Z B^T - \Omega X^T \|_2^2 \\
B_* & \equiv \arg \min_{B \in \mathbb{R}^r} \| Z B^T - X^T \|_2^2
\end{align*}
\]
Over 9X Speed-up for Amazon Tensor with 1.7 Billion Nonzeros

Amazon Tensor: 4.8M x 1.8M x 1.8M Amazon Tensor with 1.7B nonzeros.
Rank r = 25 CP decomposition

Best Fit = 0.3380
47 minutes per run
(9X faster)

Best Fit = 0.3396
7.5 hours per run

Best Fit = 0.3397
43 minutes per run
(10X faster)
Over 12X Speed-up for Reddit Tensor with 4.6 Billion Nonzeros (106 GB)

Amazon Tensor: 8.2M x 0.2M x 8.1M Reddit Tensor with 4.7B nonzeros.
Rank r = 25 CP decomposition

Best Fit = 0.0589
8 hours per run
(12X faster)

Best Fit = 0.0593
96 hours per run
Conclusions & Future Work

- How to make CP tensor decomposition faster for large-scale sparse tensors? Matrix sketching
- How to avoid repeated samples? Combine repeat rows or deterministically include high-probability rows
- How to efficiently sample? Sample independently from each factor matrix to build KRP
- How to extract data for RHS from data tensor? Pre-compute linear indices for tensor fibers
- Overall result: Order-of-magnitude speed-ups
- Many open problems: How to pick # samples (per mode even), deterministic threshold, robust stopping conditions, sampling based on data as well as KRP, parallelization of method, etc.

Contact Info: Brett bwlarSEN@stanford.edu, Tammy tgkolda@sandia.gov

Choose $\Phi$ so that all leverage scores of $\Phi \mathbf{Z}$ approximately equal, then uniform sampling yields $\beta \approx 1$

- “Uniformize” the leverage scores per Mahoney
- Fast Johnson-Lindenstrauss Transform (FJLT) uses random rows of matrix transformed by FFT and Rademacher diagonal
- FJLT cost per iteration: $O(rN \log N)$

**Gaining Efficiency for KRP matrices**

- Transform individual factor matrices before forming $\mathbf{Z}$
- Sample rows of $\mathbf{Z}$ implicitly
- Kronecker Fast Johnson-Lindenstrauss Transform (KFJLT)
- Special handling of right-hand side with preprocessing costs
- KFJLT cost per iteration: $O(r \sum \sigma_k n_k \log n_k + s r^2)$

**References**

Deterministic Can Account for Substantial Portion of Probability

Single Least Squares Problem with $N = 46M$ rows, $r = 10$ columns, $n = 183$ right-hand sides

Difference to True Residual

Percent Above $\tau \left( \frac{s_{\text{det}}}{s} \right)$

$p_{\text{det}}$
Some Trade-off Between Accuracy and Expense for Deterministic

Random Mode: 1, Solution Factors

Hybrid Mode: 1, Solution Factors
Uber Tensor: 183 x 24 x 1140 x 1717 Uber Tensor with 3M nonzeros (0.038% dense).
Rank r = 25 CP decomposition