A Sampling Based Method for Tensor Ring Decomposition[3]

Osman Asif Malik, Stephen Becker

April 22, 2021

Lecturer: Prof. Haim Avron
Student: Osher Arbib

1 Summary

Dimensionality reduction is an essential technique for multiway large-scale data, i.e., tensor. Tensor ring (TR) decomposition has become popular due to its high representation ability and flexibility[1].
In this paper, presented an estimation of usage of the leverage scores to attain a method which has complexity sublinear in the number of input tensor entries. Finally, our algorithms show superior performance in deep learning dataset compression.

2 Preliminaries

2.1 Tensor-Ring Decomposition

Let $\mathscr{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$ be a N-way tensor.
Tensor-Ring Decomposition is a sequence of $\mathcal{G}^{(n)}$ 3-way core tensors:

$$\mathcal{G}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times_n}, \text{ when } R_1 = R_n \tag{1}$$

Tensor reconstruction is given by the definition:

$$\mathscr{X}(i_1, \dots i_N) = TR\left(\mathcal{G}^{(n)}\right) = \sum_{r_1 \dots r_N} \prod_{n=1}^{N} \mathcal{G}^{(n)}\left(r_{n-1}, i_n, r_n\right) \tag{2}$$

Therefore those core-tensors could be estimated by $\arg\min ||TR(G^{(n)}) - \mathscr{X}||_{\mathbb{F}}$. There are two common approaches to this fitting problem - SVD based and alternating least-squares (ALS) based, which is the method this paper is focused. SVD methods are in general faster, however they are suffer higher reconstruction-error.

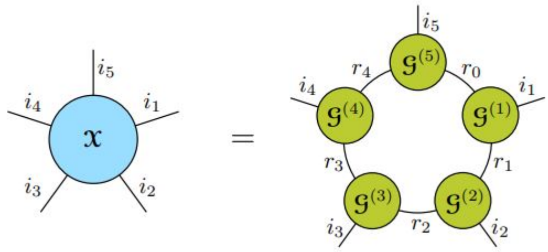


Figure 1: Tensor Ring Decomposition

2.2 ALS Method

subchain tensor $\mathcal{G}^{\neq n} \in \mathbf{R}^{R_n \times \prod_{j\neq n} I_j \times R_{n-1}}$ defined elementwise by:

$$\mathcal{G}^{\neq n} = \sum_{\substack{r_1 \ldots r_{n-1} \\ r_{n+2} \ldots r_N}} \prod_{\substack{j=1 \\ j\neq n}}^{N} \mathcal{G}^{(n)}\left(r_{n-1}, i_n, r_n\right) \tag{3}$$

The fitting objective can be solved iteratively by solving least-squares for each core-tensor as:

$$||TR(G^{(n)}) - \mathscr{X}||_{\mathbb{F}} = ||G^{\neq n}_{[n]} G^{(n)}_{(n)} - \mathscr{X}||_{\mathbb{F}} \tag{4}$$

.

Repeat this process until termination criteria met called ALS-TR (alternating least-squares tensor-ring decomposition). *Note that $X_{[n]}, X_{(n)}$ is classic and mode unfolding's tensor methods*

2.3 Tensor-Ring via Sampling

To reduce the size of least-squares problem in each internal iteration, the paper offers **sketch** each core-tensor by matrix S, and proves that if J is bigger enough [1], then the following holds with probability at-least $1-\delta$ (for $\varepsilon, \delta \in (0,1)$):

$$||S\mathcal{G}^{\neq n}_{[2]} \tilde{Z}^T_{(2)} - S\mathbf{X}^T_{[n]}||_{\mathbb{F}} \leq (1+\varepsilon) \min_{\mathcal{Z}} ||G^{\neq n}_{[n]} G^{(n)}_{(n)} - \mathscr{X}||_{\mathbb{F}} \tag{6}$$

Practically, the S matrix is never explicitly constructed, but realization of index vector is sampled to construct the matrices $\mathcal{G}^{\neq n}, \tilde{\mathbf{X}}$:

$$\mathcal{G}^{\neq n}(:, i, :) = \mathcal{G}^{(n+1)}(:, i_{n+1}, :) \ldots \mathcal{G}^{(N)}(:, i_N, :) \mathcal{G}^{(1)}(:, i_1, :) \ldots \mathcal{G}^{(n-1)}(:, i_{n-1}, :) \tag{7}$$

[1] The proper condition is:

$$J > \max\left(\frac{16}{3(\sqrt{2}-1)^2} \ln \frac{4R_{n-1}R_n}{\delta}, \frac{4}{\varepsilon\delta}\right) \prod_{j=1}^{N} R_j^2 \tag{5}$$

.

if **leverage score** $l_i = ||U(i,:)||_2^2$, [2], then the sampling probability $p^{(n)}(i) = \frac{l_{i_n}(G_{(2)}^{(n)})}{rank(G_{(2)}^{(n)})}$

2.4 TR-SVD[6]

Using randomized projection P on the data-tensor $\mathcal{X}$, for each core we can find out sequentially the SVD of the inverse power-iteration. Then compute the remainder tensor to find the next core-tensor and so-on. [3]

3 Complexity Analysis

Given I tensor-rank, N-tensor dimension (i.e.-$\mathscr{X} \in \mathbb{R}^{\prod_N I_i}$) , R- tensor-core dimension (i.e. $\mathcal{G}_i \in \mathbb{R}^{R_{i-1} \times I \times R_i}$, and J is number-of sampled entities from $\mathscr{X}$. Number of ALS iterations denotes by $iters$.

Method	Complexity
TR-ALS	$NIR^2 + \#iterNI^NR^2$
TR-SVD-rand	I$^N R^2$
TR-ALS-sampled	NIR^4 + $\#iterNIJR^2$

4 Methods

4.1 The relative Error and Convergence

The relative error is defined by $\frac{||X - \tilde{X}||_{\mathbf{F}}}{||X||_{\mathbf{F}}}$
convergence is defined by $\frac{rel\_error - \_rel\_error}{prev\_rel\_error} \leq \mathcal{TH}$

4.2 J finding

Those experiments done by running the TRALS firstly, then finding the best J by linear-search, starting from $J_{init} = 2R^2$ (due to the theoretical lower bound), until the relative error is too close the TR-ALS one.

4.3 Deep Learning Model Accuracy

Which is not part of the original paper, but inspired by [6] (which mentioned TR may use for compressing deep learning dataset), we show that TR methods, especially ALS-based can use as training-set for deep learning algorithm.
"coil-100"[4] is a dataset contained 100 classes of 72 images (taken from different POV), is used as tiny-benchmark for classification CNN networks (modern deep learning architecture used also for images, which used the spatial information). We compare the classification accuracy $\frac{TP}{total}$ on test/validation when the training we used once the original dataset then by reconstruction of ALS-TR method, with a small CNN[2]

[2] U is the $|r|$ left-singular of $\mathcal{G}_{[}^{[\backslash]} \in]$

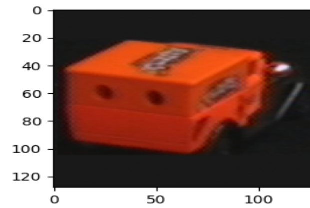
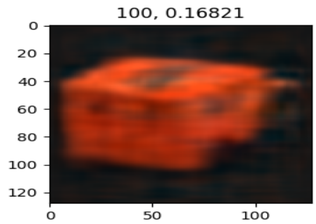
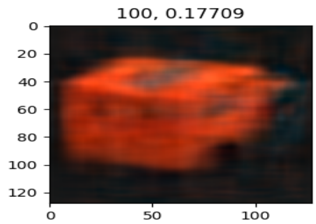
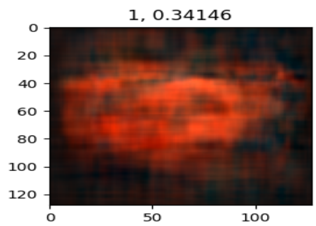
[3] this method is not explained in the paper at all, but mentioned as alternative method. I inspired by the paper's author implementation of this methodgithub:TR-SVD

4.4 Sampling method

The main issue of this paper, in orientation of our course, is the using of **leverage scoring** and **sketching** method in order to find the best entries in tensor-core to resample in order to "contained the most information", improve the convergence time (*iterations* and computation time), while keeping the accuracy of ALS methods.

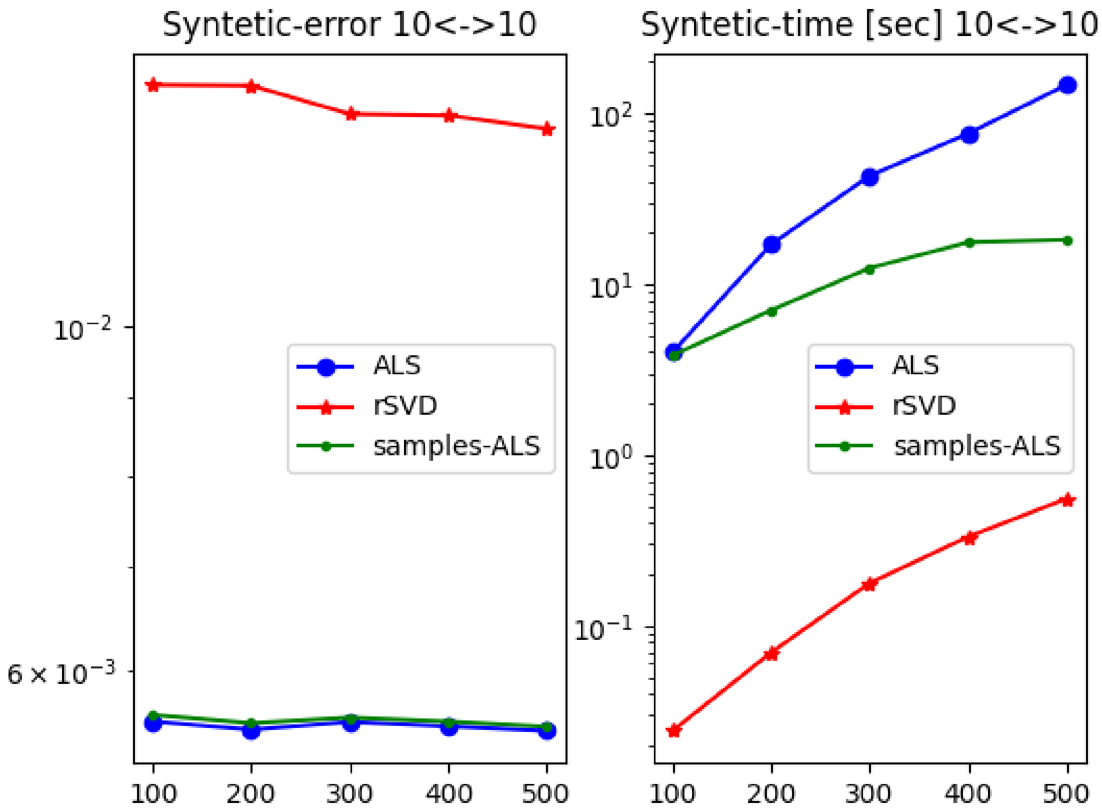
4.5 Real Data

4.5.1 Method Comparison

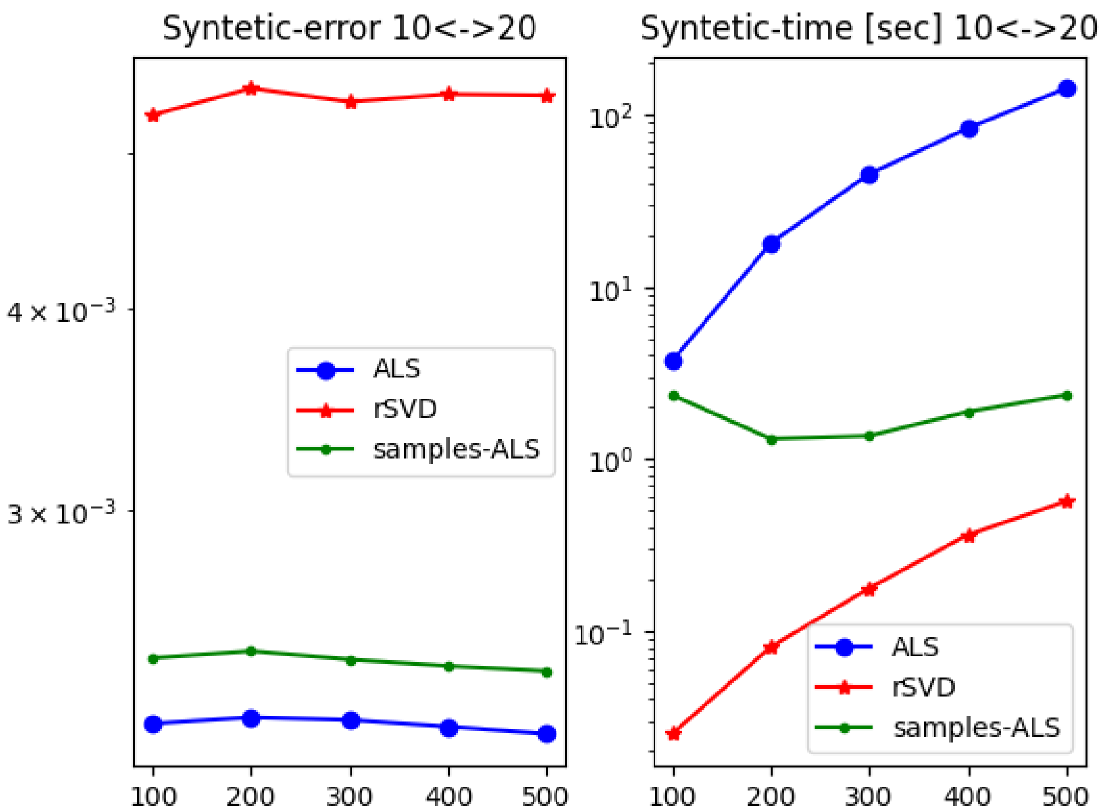
	Red Truck		Tabby Cat		
	relative error	time [sec]	relative error	time [sec]	
TR-ALS	0.168	186.47	0.134	1647.918	
TR-ALS-sampled	0.177	14.018	0.1426	106.805	
SVD-Rand	0.342	0.052	0.1979	1.037	

4.6 Synthetic Dataset

We generate synthetic data, using three 3-way tensor used has core-tensor (generated by Gaussian-random $R' \times I \times R'$ tensors, one element is peaked and set to "large value"(20)).
This experiment is repeated twice, firstly with true dimension reduction hyper-parameters $R = R' = 10$ then with $R'! = R = 20$, the purpose is to describe the ALS methods are converged with low relative-error while SVD methods are faster but much less accurate.



,



,

As we can see, the ALS method are in general more accurate than the SVD one, on the other hand, vanilla TR-ALS is dramatically slower than SVD methods, we can see sketching-based ALS enjoy both advantages.

4.7 DL Classification

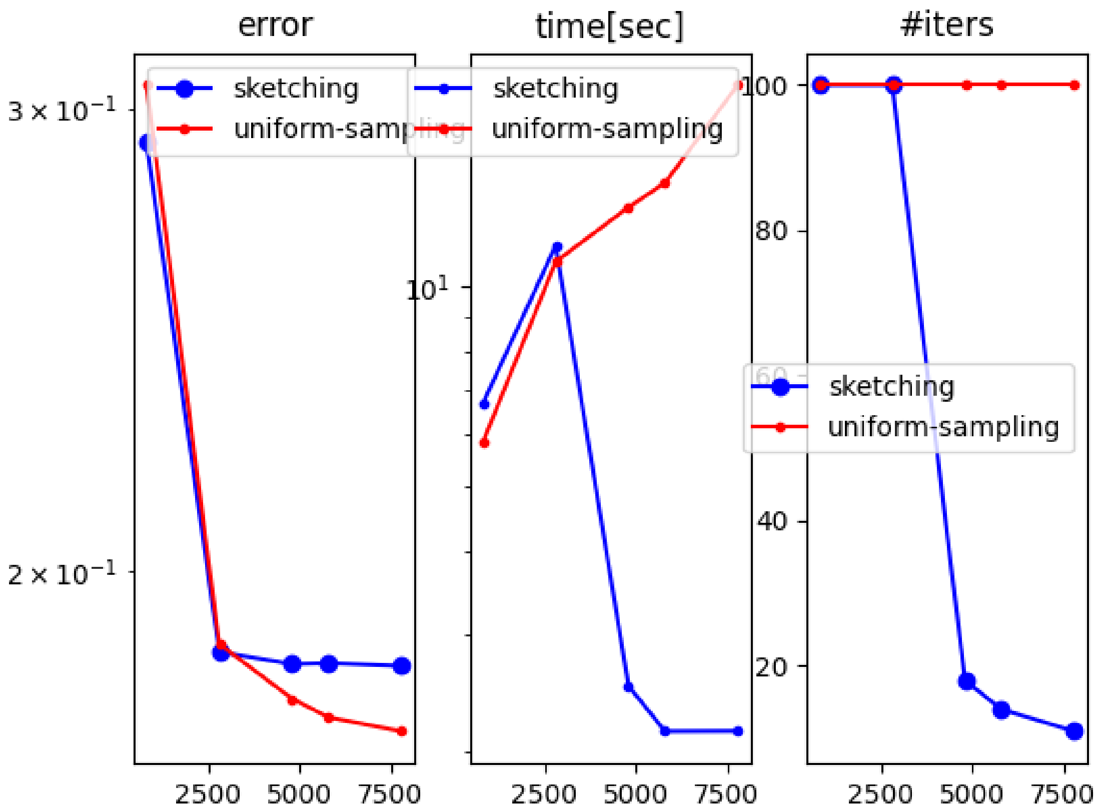
Dataset	val accuracy	test accuracy
coil-100	0.9549	0.9465
coil-100-rTR-ALS	0.9681	0.96041

We can see that sketching-based SVD method is outperform the network performance when the test-set is the regular images while the training-set is based on sampled-TR-ALS reconstruction. This may be an evidence about the power of this method, but it remains for further research.

4.8 Sampling method

We compare ALS with sampling with uniform distribution against leverage-score based Note this comparison done on real-data ("coil100/Red Truck"), three parameters are take into account: the relative-error, number of iterations and computation time (including the leverage score itself, done by

SVD, and repeated for each iteration).



As we can see, when the sample size J is large enough, the leverage-score method has a significant advantage against the uniform-distribution.

5 Conclusion

The sampling TR-ALS is (almost) accurate as TR-ALS for **real** and **synthetic** dataset.
As expected, ALS methods are more accurate, on synthetic and real dataset, and the speed of paper's method is comparable with SVD vanilla method (reported by the authors) and even SOTA[6].
We saw that the restoration of ALS methods are usable for DL missions.
And we saw the sampling method based on $leverage - score$ is important factor in the paper goals achievement.
The algorithms implemented in python, using the popular framework CPU-version of pytorch[5].

References

[1] Salman Ahmadi-Asl et al. "Randomized algorithms for fast computation of low rank tensor ring model". In: *Machine Learning: Science and Technology* 2.1 (Dec. 2020), p. 011001. DOI: 10.1088/2632-2153/abad87. URL: https://doi.org/10.1088/2632-2153/abad87.

[2] Sergio Alves. *Image Classification with COIL-100 Dataset in PyTorch*. June 2020. URL: https://medium.com/@sergioalves94/image-classification-with-coil-100-dataset-in-pytorch-b5a9c4bbe1db.

[3] Osman Asif Malik and Stephen Becker. *A Sampling Based Method for Tensor Ring Decomposition*. 2020. arXiv: 2010.08581 [math.NA].

[4] Nayar and H. Murase. *Columbia Object Image Library: COIL-100*. Tech. rep. CUCS-006-96. Department of Computer Science, Columbia University, Feb. 1996.

[5] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[6] Longhao Yuan et al. "Randomized tensor ring decomposition and its application to large-scale data reconstruction". In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 2127–2131.