A practical, fully parallel implementation of the (H-)Tucker decomposition via randomization

Martina Iannacito*[1], Davide Palitta[2], and Sascha Portaro[2]

[1]Università di Bologna – Italy
[2]Università di Bologna – Italy

Abstract

Nowadays, the Tucker decomposition is a widely used tensor method for compressing and revealing hidden patterns in data while preserving its multiway nature. The possible application fields include deep neural networks, image classification and recognition, computer vision, recommender systems, signal processing, etc.
The Tucker decomposition factorizes an order-d tensor as the product of d factor matrices and an order-d tensor of smaller size. Each factor matrix tries to capture most of the information of the column space of the corresponding tensor matricization, whose column number grows exponentially with d. Classical Tucker decomposition algorithms form the d tensor matricizations and factorize them with SVD. The high computational costs and memory requirements of these deterministic algorithms motivate the search for more efficient methods. Randomization approaches are frequently employed to decrease computational costs, boosting algorithm performance.

We present a new Tucker decomposition technique combining tensor fiber sampling and randomized sketching techniques. In particular, the fiber sampling overcomes the computational costs of forming the tensor matricizations, while the sketching improves the approximation of the tensor matricization column space. These modifications are based on recent results in matrix randomization; in our framework, they decrease the computational time and memory requirements. Under the assumption of an existing low-rank structure, the approximation errors produced by our variant are comparable to those of existing randomized Tucker methods.
Starting from our randomized Tucker decomposition, we introduce the same modifications in the Hierarchical Tucker (H-Tucker) decomposition. This method relies on a binary tree structure to factorize tensors, forming and decomposing a tensor matricization for each tree node. Since the number of nodes is usually larger than d, the computational benefits of our randomized H-Tucker variant are even more appreciable. To enhance further computational efficiency, we develop a randomized H-Tucker parallel variant, thanks to the naturally parallelizable structure of the algorithm. The randomized changes reduce the inter-node communication costs and the local memory requirements.

*Speaker