# Nonnegatively Constrained Tensor Network for Classification Problems

Rafał Zdunek, Krzysztof Fonał
Wroclaw University of Science and Technology, Faculty of Electronics,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland
Emails: {rafal.zdunek, krzysztof.fonal}@pwr.edu.pl

***Abstract*—In the era of big data, massive data are multi-modal and highly dimensional, and their processing and analysis is often performed in distributed computing systems, including cloud computing, fog computing as well as ubiquitous network computing. However, a curse of dimensionality strongly limits the usage of fundamental machine learning methods, and hence new state-of-the art models for multi-way data are needed. In this study, we discuss a possibility of using one specific topology of tensor networks for solving multi-label classification problems, under the assumption that the data are gathered in the form of nonnegatively constrained multi-way arrays, and can be represented by a tensor-train model. We propose a new computational algorithm for extracting low-rank and nonnegative 2D features, and show how to apply this algorithm for solving classification problems. The computational experiments demonstrated that our approach outperforms the fundamental and state-of-the art methods for dimensionality reduction and classification problems.**

***Index Terms*—machine learning, tensor networks, big data networks, tensor train, classification**

## I. Introduction

In the network theory, a graph representing either symmetric as well as asymmetric edges between nodes is regarded as a network [1]. Such a model is used in many disciplines of science, particularly in computer science, technology and communication, biology, physics, and sociology. In a wide-range of applications, the networks can be divided into application-oriented groups [2]–[4] that are usually specified by a corresponding adjective or noun adjunct, e.g. such as telecommunication, broadcast, electrical, neural, biology, social, transportation, etc.. The interpretation of network elements (nodes and edges) in each type of network is different. For example, the nodes in a computer network are represented by computers that share resources by data links, while the nodes in a neural networks are neurons that are interconnected by synaptic connections. Topologies of networks can also differ significantly in each type, ranging from simple ones, such as lines, rings, stars, trees, buses, to highly complex with hierarchical levels and various degrees of abstraction.

This study is concerned with an emerging application of the network theory, i.e. a tensor network (TN) [5]. In this type of a network, the nodes are represented by multi-way arrays, often referred to as tensors, which are connected by pair of modes along which the tensors are contracted. The concept of TN was initialized at least in the 70's in quantum physics [6], [7], which resulted in the creation of many topologies of TN, such as the Matrix Product States (MPS) [8], [9], Matrix Product Operators (MPO) [10], Tree Tensor Networks (TTN) [11], Quantics TT (QTT) [12], the Multi-scale Entanglement Renormalization Ansatz (MERA) [13], and Projected Entangled Pair States (PEPS) [14]. Currently it is developed in many areas of computer science, including signal processing, machine learning, and large-scale computations. Cichocki *et al.* published a comprehensive review of TN in a series of research papers [15]–[17], emphasising a potential of TN for big data analytics and large-scale optimization problems. TNs have many important advantageous over traditional matrix and tensor decomposition models that are so popular in multiple disciplines of science. One of them is so-called blessing of dimensionality. It relaxes a curse of dimensionality which intrinsically arises in highly dimensional multi-linear models. An increase in dimensionality results in an exponential grow of the volume of the corresponding high-dimensional space that leads to excessive sparsification of data, and in consequence a decrease in their statistical significance. TNs break down this problem by diminishing the dimensionality due to an increase in their modality (more modes). Moreover, reshaping of high-dimensional multi-way arrays to a longer but low-dimensional tensors is more computationally efficient, and low-rank representations of such objects usually involve much less storage resources.

The reshaped or tensorized data can be represented by many models of TNs. The study presented here is restricted to the Tensor Train (TT) which is known in the quantum physics as MPS. It allows us to decompose a multi-array data into a set of 3-way core tensors connected in the form of a line topology. The TT model was successfully applied in many disciplines of science and engineering, especially in quantum physics [18], [19], chemistry [20], [21], signal and image processing [16], [17], [22]–[24], and machine learning [25]. The core tensors in TT are usually computed with the algorithms that are based on the underlying Singular Value Decomposition (SVD). This approach was proposed by Oseledets [26], and then developed in many papers, e.g. [16], [22]–[25], [27].

In many real-world applications, observed multi-way arrays contain only nonnegative numbers. The example of nonnegative data include images, spectra, probability and frequency matrices, etc. Representing such data with low-rank unconstrained models might be suboptimal because the unconstrained features may mutually suppress when their encoding vectors (coefficients of their linear combinations) also contain negative entries. The nonnegativity constraints imposed onto the model make the output approximations interpretable and with a physical meaning. The nonnegativity constrained low-rank models, such as Nonnegative Matrix Factorization (NMF) [28] or Nonnegative Tensor Factorization (NTF) [29] are well-known in a wide range of applications. A survey of such models and the related algorithms can be found in [30], [31]. To our best knowledge, the nonnegativity constrained TT model has been only used by Lee *et al.* [32] for feature extraction from ERP data, where the underling algorithm for updating core tensors is based on a computational scheme of the Hierarchical Alternating Least-Squares (HALS) algorithm [30]. The Lee's algorithm has only been used for solving clustering problems.

In this study, we propose a new computational framework for updating the nonnegative core tensors in TT with any NMF algorithm, and apply it for solving classification problems in machine learning. The framework is based on a recursive strategy of updating each core tensor in a chain, where all core tensors before and after an updating core are merged to two core subtensors. Moreover, NMF is applied in each recursive step to a matrix that belongs to a class of tall-and-skinny. Hence, it can be easily factorized assuming the distributed computing strategy that was proposed in [33]. Then, we propose the TT-based projection algorithm that allows us to project testing multilinear samples on the subspaces spanned by the core tensors estimated for training samples. The numerical experiments demonstrate high efficiency of the proposed approach.

The remainder of this paper is organised as follows. In Section II, we present the fundamentals for tensor decomposition models and review the fundamental models of tensor networks. Section III contains our contribution, together with pseudo-codes of the proposed algorithms. The experimental results are described in Section IV. Finally, the conclusions are drawn in Section V.

## II. Tensor network models

In this section, we give preliminaries on the selected tensor operations and briefly describe the selected models of tensor networks . The notation for tensors and related operations used in this study are as followed. Calligraphic uppercase letters (e.g. $\mathcal{X}$) denotes tensors, boldface uppercase ones (e.g. $\boldsymbol{X}$) – matrices; lowercase boldface ones (e.g. $\boldsymbol{x}$) – vectors, and not bold letters are scalars. For the $N$-way array or $N$-th order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$, $x_{i_1,\ldots,i_N}$ denotes the $(i_1, i_2, \ldots, i_N)$-th element, $\boldsymbol{x}_j$ stands for the $j$-th column of $\boldsymbol{X}$. The symbol $||\cdot||_F$ denotes the Frobenius norm. The set of nonnegative real numbers is denoted by $\mathbb{R}_+$. For a matrix, $\boldsymbol{X} \in \mathbb{R}_+^{I \times J}$ or $\boldsymbol{X} \geq 0$ means that all elements in $\boldsymbol{X}$ are nonnegative.

**Mode-$n$ unfolding:** Unfolding along the $n$-th mode, also known as the mode-$n$ matricization, reshapes a $N$-way array $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_N}$, into a matrix $\boldsymbol{X}_{(n)} \in \mathbb{R}^{I_n \times \prod_{p \neq n} I_p}$ by mapping any tensor element $x_{i_1,\ldots,i_N}$ to a matrix element $x_{i_n,j}$, where $j = 1 + \sum_{\substack{k=1 \\ k \neq n}}^{N} (i_k - 1) j_k$ with $j_k = \prod_{\substack{m=1 \\ m \neq n}}^{k-1} I_m$. The unfolding can also be done in other ways. The mode-$\{n\}$ canonical matricization reshapes a tensor $\mathcal{X}$ to a matrix $\boldsymbol{X}_{<n>} \in \mathbb{R}^{\prod_{p=1}^{n} I_p \times \prod_{r=n+1}^{N} I_r}$ by mapping tensor element $x_{i_1,\ldots,i_N}$ to matrix element $x_{i,j}$, where $i = 1 + \sum_{p=1}^{n} (i_p - 1) \prod_{m=1}^{p-1} I_m$ and $j = 1 + \sum_{r>n}^{N} (i_r - 1) \prod_{m=n+1}^{r-1} I_m$.

**Tensor contraction:** Tensor contraction is a fundamental operation in tensor networks that links tensors (nodes) into a network. It basically expressed a multiplication operation along the selected modes of two tensors, and it might be considered as a generalization of a matrix multiplication. The tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ can be linked along its $n$-th mode with the tensor $\mathcal{Y} \in \mathbb{R}^{J_1 \times J_2 \times \ldots \times J_M}$ along its $m$-th mode, provided that $I_n = J_m$. The modes $\{n, m\}$ is a pair of contracting modes, i.e the modes along which the tensors are multiplied. The mode-$\binom{m}{n}$ product of $\mathcal{X}$ and $\mathcal{Y}$ gives the tensor $\mathcal{Z} = \mathcal{X} \times_n^m \mathcal{Y} \in \mathbb{R}^{I_1 \times \ldots \times I_{n-1} \times I_{n+1} \times \ldots \times I_N \times J_1 \times \ldots \times J_{m-1} \times J_{m+1} \times \ldots \times J_M}$ whose elements are calculated as:

$$z_{i_1,\ldots,i_{n-1},i_{n+1},\ldots,i_N,j_1,\ldots,j_{m-1},j_{m+1},\ldots,j_M} = = \sum_{i_n=1}^{I_n} x_{i_1,\ldots,i_{n-1},i_n,i_{n+1},\ldots,i_N} y_{j_1,\ldots,j_{m-1},i_n,j_{m+1},\ldots,j_M}.$$

For easier presentation, super- or sub-index $(m, n)$ can be omitted with assumption that $\times_N^1$ is a default configuration. For example: $\mathcal{X} \times_n \boldsymbol{Y} = \mathcal{X} \times_n^1 \boldsymbol{Y}$, and $\boldsymbol{X}\boldsymbol{Y} = \boldsymbol{X} \times_2^1 \boldsymbol{Y}$. The mode-$\binom{1}{N}$ product is a special case for TT and is denoted by the operator $\bullet$, i.e. $\mathcal{X} \bullet \mathcal{Y} = \mathcal{X} \times_N^1 \mathcal{Y}$.

**Tensor train model:** The TT model of the tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_N}$ is formulated as follows:

$$\begin{aligned} \mathcal{X} &= \sum_{j_1=1}^{J_1} \sum_{j_2=1}^{J_2} \cdots \sum_{j_{N-1}=1}^{J_{N-1}} \mathcal{X}_1(1,:,j_1) \circ \mathcal{X}_2(j_1,:,j_2) \\ &\quad \circ \ldots \circ \mathcal{X}_N(j_{N-1},:,1) \\ &= \mathcal{X}_1 \bullet \mathcal{X}_2 \bullet \ldots \bullet \mathcal{X}_N, \end{aligned} \tag{1}$$

where $\forall n : \mathcal{X}_n \in \mathbb{R}^{J_{n-1} \times I_n \times J_n}$ is the $n$-th core tensor, and the symbol $\circ$ stands for the outer product. The set $\{J_0, J_1, \ldots, J_{N-1}, J_N\}$ determine the TT-ranks, where $J_0 = J_N = 1$. The cores $\mathcal{X}_1 \in \mathbb{R}^{I_1 \times J_1}$ and $\mathcal{X}_N \in \mathbb{R}^{J_{N-1} \times I_N}$ can be regarded as matrices. Note that the $n$-th core tensor is linked with the $(n+1)$-th core tensor only along one mode, i.e. the last mode of $\mathcal{X}_n$ is contracted with the first mode of $\mathcal{X}_{n+1}$ for $n = 1, \ldots, N-1$. In this way, $\mathcal{X}$, modelled by a chain of core tensors, resembles a topology of a line network, shown in Fig. 1. Each intermediate circle, which represents a node being

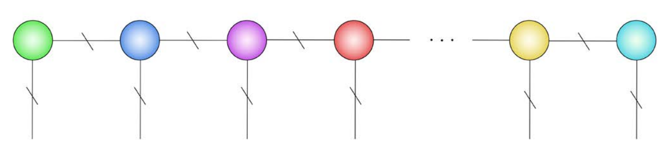

Fig. 1. Topology of the TT network

a core tensor, has three links that represent the contraction modes. The first and last circle have only two links, which means that they denote matrices.

**Hierarchical Tucker model:** In the Hierarchical Tucker (HT) model, the tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_N}$ with $N = 2^d$, where $d = 1, 2, \ldots$, is modeled by a multi-level binary tree structure which is shown in Fig. 2. HT is equivalent to TTN [11] in quantum physics. The fundamental Tucker model is expressed by: $\mathcal{X} = \mathcal{G} \times_1 \boldsymbol{U}^{(1)} \times_2 \ldots \times_N \boldsymbol{U}^{(N)}$, where $\mathcal{G} \in \mathbb{R}^{J_1 \times \ldots \times J_N}$ is the core tensor, and $\boldsymbol{U}^{(n)} \in \mathbb{R}^{I_n \times J_n}$ are factor matrices of ranks $(J_1, \ldots, J_N)$ (the circles with digit "0" in Fig. 2). The HT expands the core tensor $\mathcal{G}$ with a multi-level binary tree structure composed of 3-way core tensors (nodes), except for the top level. Each core tensor merges two modes at a given level. The nodes at the lowest level are marked by circles with digit "1", the top level (the circle with the letter "$L$") is represented by a matrix.

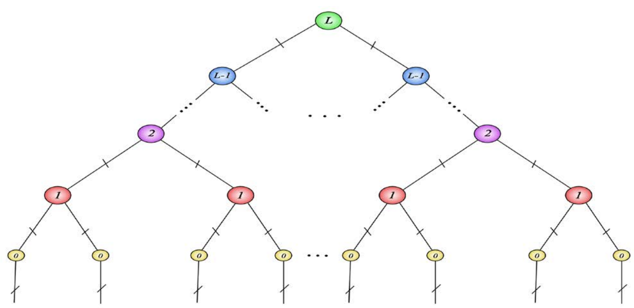


Fig. 2. Topology of the HT network

Table I lists the storage complexity of the following tensor decomposition models: CANDECOMP/PARAFAC (CP), standard Tucker, HT, and TT, under the assumption that $I_1 = I_2 = \ldots = I_N = I$ and $J_1 = J_2 = \ldots = J_N = J$.

TABLE I
STORAGE COMPLEXITIES OF TENSOR DECOMPOSITION MODELS

| ***Model*** | ***Storage complexity*** |
|---|---|
| Full tensor format | $\mathcal{O}(I^N)$ |
| CP | $\mathcal{O}(NIJ)$ |
| Tucker | $\mathcal{O}(NIJ + J^N)$ |
| HT | $\mathcal{O}(NIJ + NJ^3)$ |
| TT/MPS | $\mathcal{O}(NIJ^2)$ |
| TT/MPO | $\mathcal{O}(NI^2J^2)$ |
| QTT | $\mathcal{O}(NJ^2 \log_q(I))$ |

## III. ALGORITHM

The most frequently-used algorithms for updating the core tensors in the model (1) is based on the SVD of the recursively reshaped TT model [16], [22]–[27]. In this study, we propose to update the core tensors with NMF due to its intrinsic nonnegativity constraints and our assumption that all the core tensors should contain nonnegative numbers. Let $\mathcal{X} \in \mathbb{R}_+^{I_1 \times \ldots \times I_N}$ be a nonnegative tensor (with all nonnegative entries) that can be represented by the model (1), which can be rewritten as follows:

$$\begin{aligned} \mathcal{X} &= \mathcal{X}_1 \bullet \mathcal{X}_2 \bullet \ldots \bullet \mathcal{X}_N \\ &= \mathcal{X}_{<n} \bullet \mathcal{X}_n \bullet \mathcal{X}_{>n} \qquad (2) \\ &= \mathcal{X}_{<n} \bullet \mathcal{X}_{\geq n} \qquad (3) \\ &= \mathcal{X}_{\leq n} \bullet \mathcal{X}_{>n}, \qquad (4) \end{aligned}$$

where $\mathcal{X}_{<n} \in \mathbb{R}_+^{I_1 \times \ldots \times I_{n-1} \times J_{n-1}}$, $\mathcal{X}_n \in \mathbb{R}_+^{J_{n-1} \times I_n \times J_n}$, $\mathcal{X}_{>n} \in \mathbb{R}_+^{J_n \times I_{n+1} \times \ldots \times I_N}$, $\mathcal{X}_{\geq n} \in \mathbb{R}_+^{J_{n-1} \times I_n \times \ldots \times I_N}$, and $\mathcal{X}_{\leq n} \in \mathbb{R}_+^{I_1 \times \ldots \times I_n \times J_n}$.

Note that any entry of the model (3) is expressed by: $x_{i_1,\ldots,i_N} = \sum_{j_{n-1}=1}^{J_{n-1}} (\mathcal{X}_{<n})_{i_1,\ldots,i_{n-1},j_{n-1}} (\mathcal{X}_{\geq n})_{j_{n-1},i_n,\ldots,i_N}$. The mode-$\{n-1\}$ canonical matricization of (3) leads to:

$$\boldsymbol{M}_{n-1} = (\mathcal{X}_{<n} \bullet \mathcal{X}_{\geq n})_{<n-1>} \in \mathbb{R}_+^{I_1 \cdot \ldots \cdot I_{n-1} \times I_n \cdot \ldots \cdot I_N}. \qquad (5)$$

Applying NMF with rank $J_{n-1}$ to (5), we get:

$$\boldsymbol{M}_{n-1} = \boldsymbol{F}_{n-1} \boldsymbol{Z}_{n-1}, \qquad (6)$$

where $\boldsymbol{F}_{n-1} \in \mathbb{R}_+^{I_1 \cdot \ldots \cdot I_{n-1} \times J_{n-1}}$ and $\boldsymbol{Z}_{n-1} \in \mathbb{R}_+^{J_{n-1} \times I_n \cdot \ldots \cdot I_N}$. Then the matrices $\boldsymbol{F}_{n-1}$ and $\boldsymbol{Z}_{n-1}$ can be reshaped to the tensors

$$\mathcal{X}_{<n} = \texttt{reshape}(\boldsymbol{F}_{n-1}, [I_1, \ldots, I_{n-1}, J_{n-1}]), \qquad (7)$$

and

$$\mathcal{X}_{\geq n} = \texttt{reshape}(\boldsymbol{Z}_{n-1}, [J_{n-1}, I_n, \ldots, I_N]). \qquad (8)$$

Similarly, the mode-$\{n\}$ canonical matricization of $\mathcal{X}$ gives the matrix $\boldsymbol{M}_n = \mathcal{X}_{<n>} \in \mathbb{R}_+^{I_1 \cdot \ldots \cdot I_n \times I_{n+1} \cdot \ldots \cdot I_N}$. Factorizing it with NMF at rank $J_n$, we obtain: $\boldsymbol{M}_n = \boldsymbol{F}_n \boldsymbol{Z}_n$, where $\boldsymbol{F}_n \in \mathbb{R}_+^{I_1 \cdot \ldots \cdot I_n \times J_n}$ and $\boldsymbol{Z}_n \in \mathbb{R}_+^{J_n \times I_{n+1} \cdot \ldots \cdot I_N}$. Note that $\boldsymbol{F}_n$ and $\boldsymbol{Z}_n$ can be easily reshaped to the tensors of the model (4). Thus:

$$\mathcal{X}_{\leq n} = \texttt{reshape}(\boldsymbol{F}_n, [I_1, \ldots, I_n, J_n]), \qquad (9)$$

$$\mathcal{X}_{>n} = \texttt{reshape}(\boldsymbol{Z}_n, [J_n, I_{n+1}, \ldots, I_N]). \qquad (10)$$

To estimate the core tensors in (1), NMF can be applied recursively to reshaped matrices. For $n = 1$, we have: $\boldsymbol{M}_1 = \mathcal{X}_{<1>} = \mathcal{X}_{(1)} \in \mathbb{R}_+^{I_1 \times I_2 \cdot \ldots \cdot I_N}$. Applying NMF with rank $J_1$ to $\boldsymbol{M}_1$, we have $\boldsymbol{M}_1 = \boldsymbol{F}_1 \boldsymbol{Z}_1$, where $\boldsymbol{F}_1 \in \mathbb{R}_+^{I_1 \times J_1}$ and $\boldsymbol{Z}_1 \in \mathbb{R}_+^{J_1 \times I_2 \cdot \ldots \cdot I_N}$. Note that $\boldsymbol{F}_1 = \mathcal{X}_{\leq 1} = \mathcal{X}_1 \in \mathbb{R}^{J_0 \times I_1 \times J_1}$, assuming that $J_0 = 1$, and $\boldsymbol{Z}_1$ can be reshaped to: $\mathcal{X}_{>1} = \texttt{reshape}(\boldsymbol{Z}_1, [J_1, I_2, \ldots, I_N])$. In the next step ($n = 2$), the mode-$\{2\}$ canonical matricization is applied to $\mathcal{X}_{>1}$, which gives: $\tilde{\boldsymbol{M}}_2 = (\mathcal{X}_{>1})_{<2>} \in \mathbb{R}_+^{J_1 I_2 \times I_3 \cdot \ldots \cdot I_N}$, and NMF of $\tilde{\boldsymbol{M}}_2$ at rank $J_2$ gives us the factors $\tilde{\boldsymbol{F}}_2 \in \mathbb{R}_+^{J_1 I_2 \times J_2}$ and $\tilde{\boldsymbol{Z}}_2 \in \mathbb{R}_+^{J_2 \times I_3 \cdot \ldots \cdot I_N}$. Then, $\mathcal{X}_2$ can be obtained from $\tilde{\boldsymbol{F}}_2$ by the reshaping: $\mathcal{X}_2 = \texttt{reshape}(\tilde{\boldsymbol{F}}_2, [J_1, I_2, J_3])$, and $\mathcal{X}_{>2} = \texttt{reshape}(\tilde{\boldsymbol{Z}}_2, [J_2, I_3, \ldots, I_N])$. For $n = 3$, $\tilde{\boldsymbol{M}}_3 = (\mathcal{X}_{>2})_{<2>} \in \mathbb{R}_+^{J_2 I_3 \times I_4 \cdot \ldots \cdot I_N}$, and the whole procedure repeats.

In the above recursive procedure, the core tensors are estimated from the left side ($\mathcal{X}_1$) to the right side ($\mathcal{X}_N$). Note that the opposite way of estimation is also possible.

Starting from $n = N$ and using the model (3), we have $\mathcal{X} = \mathcal{X}_{<N} \bullet \mathcal{X}_{\geq N}$. Following (5), we have: $\boldsymbol{M}_{N-1} = \mathcal{X}_{<N-1>} \in \mathbb{R}_+^{I_1 \cdot \ldots \cdot I_{N-1} \times I_N}$. For the right-left updating, NMF is applied to $\boldsymbol{M}_{N-1}^T$, which gives: $\boldsymbol{M}_{N-1}^T = \bar{\boldsymbol{F}}_{N-1}\bar{\boldsymbol{Z}}_{N-1}$, where $\bar{\boldsymbol{F}}_{N-1} \in \mathbb{R}_+^{I_N \times J_{N-1}}$, $\bar{\boldsymbol{Z}}_{N-1} \in \mathbb{R}_+^{J_{N-1} \times I_1 \cdot \ldots \cdot I_{N-1}}$, and $\mathcal{X}_N = \bar{\boldsymbol{F}}_{N-1}^T$. In the next step, $(\bar{\boldsymbol{Z}}_{N-1})^T$ is reshaped to the tensor $\mathcal{X}_{<N} \in \mathbb{R}_+^{I_1 \times \ldots \times I_{N-1} \times J_{N-1}}$, and $\bar{\boldsymbol{M}}_{N-2} = (\mathcal{X}_{<N})_{<N-2>} \in \mathbb{R}_+^{I_1 \cdot \ldots \cdot I_{N-2} \times I_{N-1} J_{N-1}}$. Next, NMF is applied to $(\bar{\boldsymbol{M}}_{N-2})^T$ at rank $J_{N-1}$, and the recursive procedure repeats until $n = 1$.

The multi-way samples are be ordered along an arbitrary mode but to have a higher flexibility in selecting higher ranks, we proposed it was an interior mode. If the samples are ordered along the first or last mode, then the modes of training and testing tensors should be permuted in such a way that the samples are represented by the $n$-th mode, where $1 < n < N$. Following the concept of two-side recursive updates that was proposed in [34], both recursive procedures can be applied, e.g. according to the model (3). In the forward direction (from left to right), the core tensors $\{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_{n-1}\}$ are estimated while the core tensors $\{\mathcal{X}_N, \mathcal{X}_{N-1}, \ldots, \mathcal{X}_n\}$ are estimated in the backward direction (from right to left). The complete procedure is expressed by Algorithm 1.

**Algorithm 1 NMF-TT**

**Input :** $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ – $N$-way array of multilinear training samples, $[J_1, \ldots, J_{N-1}]$ – TT-ranks
**Output:** $\{\mathcal{X}_n\}$ - estimated core tensors

$J_0 = 1$
$\boldsymbol{M} = \mathcal{X}_{(1)}$ % Mode-1 unfolding
**for** $i = 1, \ldots, n-1$ **do**
  Compute: $\boldsymbol{M} = \boldsymbol{F}\boldsymbol{Z}$ `// NMF at rank` $J_i$
  $\mathcal{X}_i = \texttt{reshape}(\boldsymbol{F}, [J_{i-1}, I_i, J_i])$
  $\boldsymbol{M} = \texttt{reshape}(\boldsymbol{Z}, [J_i I_{i+1}, \prod_{p=i+2}^{N} I_p])$
**end**
$\boldsymbol{M} \leftarrow \left(\texttt{reshape}(\boldsymbol{M}, [J_{n-1}\prod_{p=n}^{N} I_p, I_{N+1}])\right)^T$
$J_{N+1} = 1$
**for** $i = N+1, \ldots, n$ **do**
  Compute $\boldsymbol{M} = \boldsymbol{F}\boldsymbol{Z}$ `// NMF at rank` $J_i$
  $\mathcal{X}_i = \texttt{reshape}(\boldsymbol{F}^T, [J_{i-1}, I_i, J_i])$
  $\boldsymbol{M} = \texttt{reshape}(\boldsymbol{Z}, [J_{n-1}\prod_{p=n}^{i-1} I_p, J_{i-1} I_{i-1}])$
**end**
$\mathcal{X}_n = \texttt{reshape}(\boldsymbol{M}, [J_{n-1}, I_n, J_n])$

*Remark 1:* Assuming the computational complexity $\mathcal{O}(IJT)$ for NMF of $\boldsymbol{Y} \in \mathbb{R}_+^{I \times T}$ given the rank $J$, then we have $\mathcal{O}(\prod_{r=1}^{i} J_r \prod_{p=i}^{N} I_p)$ for any $i$-th step in Algorithm 1. Thus, the computational complexity for all steps can be roughly approximated by $\mathcal{O}(\sum_{i=1}^{N}\prod_{r=1}^{i} J_r \prod_{p=i}^{N} I_p)$.

Algorithm 1 extracts the multilinear features from the multilinear training samples, stored in $\mathcal{X}$. The core tensor $\mathcal{X}_n \in \mathbb{R}_+^{J_{n-1} \times I_n \times J_n}$ contains $I_n$ 2D features ($J_{n-1} \times J_n$) that are extracted from the training samples. The similar features need to be extracted from testing samples. Let $\mathcal{Y} \in \mathbb{R}_+^{I_1 \times \ldots \times I_N}$ contain $I_n$ testing multilinear samples ordered along its $n$-th mode. Note that $I_n$ for $\mathcal{Y}$ can be different than $I_n$ for $\mathcal{X}$. Assuming testing samples come from the same observation space as training samples, we have:

$$\mathcal{Y} = \mathcal{X}_{<n} \bullet \hat{\mathcal{X}}_n \bullet \mathcal{X}_{>n}, \tag{11}$$

where $\mathcal{X}_{<n}$ and $\mathcal{X}_{>n}$ are extracted from $\mathcal{X}$ (training samples), and $\hat{\mathcal{X}}_n \in \mathbb{R}_+^{J_{n-1} \times I_n \times J_n}$ contains 2D features of testing samples. The core tensor $\hat{\mathcal{X}}_n$ can therefore be estimated by projecting the samples in $\mathcal{Y}$ onto tensor subspaces spanned by the core tensors $\{\mathcal{X}_1, \ldots, \mathcal{X}_{n-1}, \mathcal{X}_n, \ldots, \mathcal{X}_N\}$. The projection procedure is described by Algorithm 2.

**Algorithm 2 Tensor projection**

**Input :** $\mathcal{Y} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ – testing multilinear samples, $\{\mathcal{X}_n\}$ - estimated training core tensors, $[J_1, \ldots, J_{N-1}]$ – TT-ranks
**Output:** $\hat{\mathcal{X}}_n$

$J_0 = 1$
$\boldsymbol{M} = \mathcal{Y}_{(1)}$ % Mode-1 unfolding
**for** $i = 1, \ldots, n-1$ **do**
  $\boldsymbol{Q} = \texttt{reshape}(\mathcal{X}_i, [J_{i-1} I_i, J_i])$
  $\boldsymbol{M} = \texttt{reshape}(\boldsymbol{Q}^T \boldsymbol{M}, [J_i I_{i+1}, \prod_{p=i+2}^{N+1} I_p])$
**end**
$\boldsymbol{M} \leftarrow \texttt{reshape}(\boldsymbol{M}, [J_{n-1}\prod_{p=n}^{N} I_p, I_{N+1}])$
$J_{N+1} = 1$
**for** $i = N+1, \ldots, n$ **do**
  $\boldsymbol{Q} = \texttt{reshape}(\mathcal{X}_i, [J_{i-1} I_i, J_i])$
  $\boldsymbol{M} \leftarrow \texttt{reshape}(\boldsymbol{M}\boldsymbol{Q}^T, [J_{n-1}\prod_{p=n}^{i-2} I_p, J_{i-1} I_{i-1}])$
**end**
$\hat{\mathcal{X}}_n = \texttt{reshape}(\boldsymbol{M}, [J_{n-1}, I_n, J_n])$

## IV. Experiments

This section presents the results of numerical experiments that were carried out on various well-known datasets. Due to the limited space, we present only the results of applying the proposed algorithm to dimensionality reduction of original high-dimensional samples in classification problems. We compared it with the fundamental and most efficient NMF algorithms that are also free from discriminative constraints and used for solving similar classification problems. Obviously, the discriminant constraints are used in the classifier that is applied to the results provided by the discussed methods.

### A. Data sets

The following datasets were used for the tests:

- **COIL-100**: This data set[1] contains RGB color images of 100 objects at varying object poses, covering a full-degree rotation with 5 degree interval, which gives 72 images per one class. Hence, the whole dataset contains 7200 images of $128 \times 128$ pixel resolution.

[1] http://www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php

- **MNIST**: MNIST[2] contains grey-scale images of 70 000 hand-written digits, divided into two groups: 60 000 for training and 10 000 for testing. The resolution of each image is $20 \times 20$ pixels. The digits are size-normalized and centered in a fixed-size image.
- **ORL**: ORL contains 400 frontal facial images of 40 people (10 pictures per person). The pictures [3] were taken at different times, varying the lighting gradient, facial expressions (open / closed eyes, smiling / not smiling) and facial occlusions (glasses / no glasses). All the images have a dark homogeneous background with the subjects in an upright, frontal position. The resolution of each image is $112 \times 92$ pixels.

### B. Setup

The proposed NMF-TT algorithm (Algorithm 1) can be used with various NMF algorithms. A survey of them can be found in [30], [31]. One of the fundamental ones is the multiplicative algorithm that minimizes the Euclidean distance using the alternating optimization scheme. It was proposed by Lee and Seung in [28], and in this study we will refer to it as MUE-NMF. Its update rules are simple to implement and guarantee a monotonic convergence, however, they are terrible slowly convergent. One of the most efficient algorithms for NMF is the fast HALS which is an improved version [35] of the basic HALS [36]. In this study, we used the fast HALS both in Algorithm 1 (denoted by the acronym: HALS-TT) as well as directly, i.e. for a comparison with HALS-TT.

All the algorithms were coded in Matlab 2016a and run on the workstation equipped with CPU Intel Core i7-7700 (4 cores, 8 threads), 16GB RAM, 512 GB SSD, and under Linux Ubuntu.

Since the alternating optimization is intrinsically non-convex, NMF algorithms may suffer from various indeterminacies, which means that the results might be initializer-dependent. To minimize the risk of a suboptimal solution, each algorithm (including HALS-TT) was validated with 10 Monte Carlo (MC) runs. In each MC run, a new initializer was generated from a uniformly distribution, and the outcome results are averaged. Each algorithm was terminated when its normalized residual error drops below the threshold of $10^{-5}$.

The output tensors $\mathcal{X}_n$ (Algorithm 1) and $\hat{\mathcal{X}}_n$ (Algorithm 2) were yielded to a classifier. We tested different classifiers, such as $k$-NN (Nearest Neighbors), Support Vector Machine (SVM), and the best results are obtained if the Linear Discriminant Analysis (LDA) is combined with the 1-NN classifier.

### C. Results

The analyzed datasets are partitioned according to the 5-fold Cross Validation (CV), and the classification results are evaluated with the Misclassification Rate (MCR) measure – the `crossval` function in Matlab. The results averaged over CV folds and MC runs are presented in Fig. 3.

[2]http://yann.lecun.com/exdb/mnist/

[3]http://people.cs.uchicago.edu/∼dinoj/vis/orl/

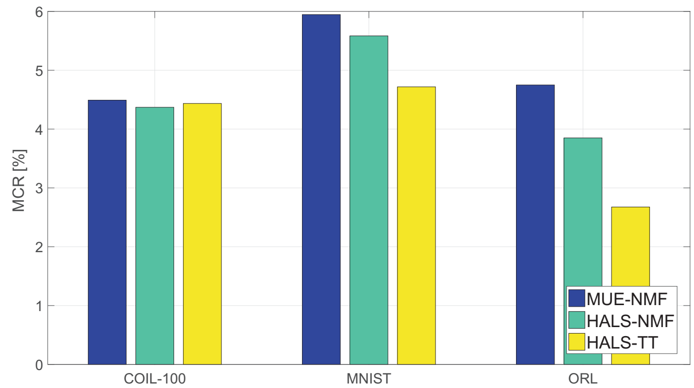


Fig. 3. Misclassification rates (MCR) obtained for classification of the tested datasets (COIL-100, MNIST, ORL) with the algorithms: MUE-NMF, HALS-NMF, and HALS-TT.

Additionally, the Hinton diagram of the confusion matrix obtained for classification of the ORL dataset is illustrated in Fig. 4.

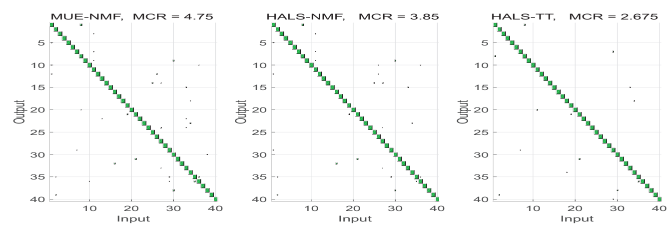


Fig. 4. Hinton diagrams of the confusion matrices obtained for classification of the ORL dataset with the algorithms: MUE-NMF, HALS-NMF, and HALS-TT.

### D. Discussion

The results shown in Fig. 3 confirm that the proposed NMF-TT algorithm outperforms the fundamental and the most efficient NMF methods in terms of the accuracy of classification. For the MNIST and ORL datasets, HALS-TT is considerably better than the other ones. For the COIL-100, all the algorithms give roughly similar accuracy results, and HALS-TT is ranked the second best. Additionally, the Hinton diagram in Fig. 4 indicate that all classes from the ORL dataset are recognized correctly, and misclassification errors are uniformly cross-class dissipated with the lowest mean-value of MCR for HALS-TT.

The results also demonstrate a potential of NMF-TT for solving large-scale classification problems, especially when analyzed samples are represented by high-dimensional multi-way arrays. Note that the matrix $\boldsymbol{M}$ in Algorithm 1 belongs to a class of tall-and-skinny matrices, i.e. it has much more columns than rows. In such a case, the MapReduce computational paradigm can be applied. In the mapping phase, the matrix $\boldsymbol{M}$ can be partitioned into column-blocks (chunks), and each block can be proceeded concurrently by multiple nodes in the distributed computational environment. In the reducing phase, the features extracted from each block can be aggregated into a much smaller matrix which can be shared and proceeded by any NMF algorithm on a single node. The details on this idea can be found in [33].

## V. Conclusions

In this study, we discuss a possibility of imposing nonnegativity constraints onto the TT model of tensor networks, and proposed a new computational algorithm for updating the nonnegative core tensors in the nonnegative TT model. The algorithm is based on recursive updates of core tensors with a general framework of NMF, which reflects its flexibility and scalability for processing large-scale data. It has been successfully applied for solving classification problems but its range of applications is not only limited to such problems. We believe that it can find useful applications in many disciplines of science, especially where other tensor network models are used. It can also be used for low-rank nonnegative data approximations, especially in image completion problems, recommendation systems, distance prediction problems, etc. These aspects will be studied in our future research.

## Acknowledgment

This work was partially supported by the statutory grant, no. 401/0034/18. The other part is supported by the grant "Mloda Kadra", no. 0402/0116/18 from the Ministry of Science and Higher Eduction.

## References

[1] E. Estrada and P. A. Knight, *First Course in Network Theory*. Oxford University Press, 2015.

[2] S. Dorogovtsev and J. Mendes, *Evolution of Networks: From Biological Nets to the Internet and WWW*. Oxford University Press, 2003.

[3] N. Ahmed, H. Rahman, and M. Hussain, "A comparison of 802.11ah and 802.15.4 for iot," *ICT Express*, vol. 2, no. 3, pp. 100–102, 2016, special Issue on ICT Convergence in the Internet of Things (IoT).

[4] B. Liu, Z. Yan, and C. W. Chen, "Medium access control for wireless body area networks with qos provisioning and energy efficient design," *IEEE Transactions on Mobile Computing*, vol. 16, no. 2, pp. 422–434, Feb 2017.

[5] S.-J. Ran, E. Tirrito, C. Peng, X. Chen, G. Su, and M. Lewenstein, "Review of Tensor Network Contraction Approaches," *arXiv e-prints*, p. arXiv:1708.09213, Aug 2017.

[6] R. Penrose, "Applications of negative dimensional tensors," in *Combinatorial Mathematics and its Applications*. Academic Press, 1971.

[7] K. G. Wilson, "The renormalization group: Critical phenomena and the kondo problem," *Rev. Mod. Phys.*, vol. 47, p. 773, 1975.

[8] S. Östlund and S. Rommer, "Thermodynamic limit of density matrix renormalization," *Phys. Rev. Lett.*, vol. 75, pp. 3537–3540, Nov 1995.

[9] S. R. White, "Density matrix formulation for quantum renormalization groups," *Phys. Rev. Lett.*, vol. 69, no. 19, pp. 2863–2866, 1992.

[10] F. Verstraete, J. J. García-Ripoll, and J. I. Cirac, "Matrix product density operators: Simulation of finite–temperature and dissipative systems," *Phys. Rev. Lett.*, vol. 93, no. 20, p. 207204, 2004.

[11] Y.-Y. Shi, L.-M. Duan, and G. Vidal, "Classical simulation of quantum many-body systems with a tree tensor network," *Phys. Rev. A*, vol. 74, no. 2, p. 022320, 2006.

[12] I. V. Oseledets, "Approximation of $2^d \times 2^d$ matrices using tensor decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 31, no. 4, pp. 2130–2145, Jun. 2010.

[13] G. Vidal, "Entanglement renormalization," *Phys. Rev. Lett.*, vol. 99, p. 220405, Nov 2007.

[14] F. Verstraete, M. M. Wolf, D. Perez-Garcia, and J. I. Cirac, "Criticality, the area law, and the computational power of projected entangled pair states," *Phys. Rev. Lett.*, vol. 96, p. 220601, Jun 2006.

[15] A. Cichocki, "Era of big data processing: A new approach via tensor networks and tensor decompositions," *CoRR*, vol. abs/1403.2048, 2014. [Online]. Available: http://arxiv.org/abs/1403.2048

[16] A. Cichocki, N. Lee, I. V. Oseledets, A. H. Phan, Q. Zhao, and D. P. Mandic, "Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions," *Foundations and Trends in Machine Learning*, vol. 9, no. 4-5, pp. 249–429, 2016.

[17] A. Cichocki, A. H. Phan, Q. Zhao, N. Lee, I. V. Oseledets, M. Sugiyama, and D. P. Mandic, "Tensor networks for dimensionality reduction and large-scale optimization: Part 2 applications and future perspectives," *Foundations and Trends in Machine Learning*, vol. 9, no. 6, pp. 431–673, 2017.

[18] B. P. Lanyon, C. Maier, M. Holzäpfel, T. Baumgratz, C. Hempel, P. Jurcevic, I. Dhand, A. S. Buyskikh, A. J. Daley, M. Cramer, M. B. Plenio, R. Blatt, and C. F. Roos, "Efficient tomography of a quantum many-body system," *Nature Physics*, vol. 13, p. 1158?1162, July 2017.

[19] T. Huckle and K. Waldherr, "Computations in quantum tensor networks," Institut für Informatik, TUM, XVIII Householder Symposium 2011, Lake Tahoe, Jul. 2011.

[20] V. A. Kazeev, M. Khammash, M. Nip, and C. Schwab, "Direct solution of the chemical master equation using quantized tensor trains," *PLoS Computational Biology*, vol. 10, no. 3, 2014.

[21] V. Khoromskaia and B. N. Khoromskij, "Tensor numerical methods in quantum chemistry: from hartreefock to excitation energies," *Phys. Chem. Chem. Phys.*, vol. 17, pp. 31 491–31 509, 2015.

[22] A. H. Phan, A. Cichocki, A. Uschmajew, P. Tichavský, G. Luta, and D. P. Mandic, "Tensor networks for latent variable analysis. part I: algorithms for tensor train decomposition," *CoRR*, vol. abs/1609.09230, 2016.

[23] J. A. Bengua, H. N. Phien, H. D. Tuan, and M. N. Do, "Efficient tensor completion for color image and video recovery: Low-rank tensor train," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2466–2479, May 2017.

[24] C. Y. Ko, K. Batselier, W. Yu, and N. Wong, "Fast and accurate tensor completion with tensor trains: A system identification approach," *CoRR*, vol. abs/1804.06128, 2018. [Online]. Available: http://arxiv.org/abs/1804.06128

[25] W. Wang, V. Aggarwal, and S. Aeron, "Tensor train neighborhood preserving embedding," *IEEE Trans. Signal Processing*, vol. 66, no. 10, pp. 2724–2732, 2018.

[26] I. V. Oseledets, "Tensor-train decomposition," *SIAM J. Sci. Comput.*, vol. 33, no. 5, pp. 2295–2317, 2011.

[27] L. Grasedyck and S. Krämer, "Stable ALS Approximation in the TT-Format for Rank-Adaptive Tensor Completion," *arXiv e-prints*, p. arXiv:1701.08045, 2017.

[28] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, 1999.

[29] A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proc. of the 22-th International Conference on Machine Learning*, Bonn, Germany, 2005.

[30] A. Cichocki, R. Zdunek, A. H. Phan, and S.-I. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley and Sons, 2009.

[31] Y.-X. Wang and Y.-J. Zhang, "Nonnegative matrix factorization: A comprehensive review," *IEEE Trans. on Knowl. and Data Eng.*, vol. 25, no. 6, pp. 1336–1353, 2013.

[32] N. Lee, A.-H. Phan, F. Cong, and A. Cichocki, "Nonnegative tensor train decompositions for multi-domain feature extraction and clustering," in *Neural Information Processing*, A. Hirose, S. Ozawa, K. Doya, K. Ikeda, M. Lee, and D. Liu, Eds. Cham: Springer International Publishing, 2016, pp. 87–95.

[33] R. Zdunek and K. Fonal, "Distributed geometric nonnegative matrix factorization and hierarchical alternating least squares-based nonnegative tensor factorization with the mapreduce paradigm," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 17, 2018.

[34] K. Fonal and R. Zdunek, "Distributed and randomized tensor train decomposition for feature extraction," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019.

[35] A. Phan and A. Cichocki, "Multi-way nonnegative tensor factorization using fast hierarchical alternating least squares algorithm (HALS)," in *Proc. of The 2008 International Symposium on Nonlinear Theory and its Applications*, Budapest, Hungary, 2008.

[36] A. Cichocki, R. Zdunek, and S.-I. Amari, "Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization," in *Independent Component Analysis and Signal Separation*, ser. LNCS. Springer Berlin / Heidelberg, 2007, vol. 4666, pp. 169–176, iCA 2007.