# Feature Fusion via Tensor Network Summation

Giuseppe G. Calvi, Ilia Kisil, Danilo P. Mandic
*Department of Electrical and Electronic Engineering*
*Imperial College London, UK*
{giuseppe.calvi15, i.kisil15, d.mandic}@imperial.ac.uk

***Abstract*—Tensor networks (TNs) have been earning considerable attention as multiway data analysis tools owing to their ability to tackle the curse of dimensionality through the representation of large-scale tensors via smaller-scale interconnections of their intrinsic features. However, despite the obvious benefits, the current treatment of TNs as stand-alone entities does not take full advantage of their underlying structure and the associated feature localization. To this end, we exploit the analogy with feature fusion to propose a rigorous framework for the combination of TNs, with a particular focus on their summation as a natural way of their combination. The proposed framework is shown to allow for feature combination of any number of tensors, as long as their TN representation topologies are isomorphic. Simulations involving multi-class classification of an image dataset show the benefits of the proposed framework.**

***Index Terms*—Sum of tensor networks, Tucker decomposition, classification, feature fusion, graphs**

## I. Introduction

Tensors are multidimensional generalizations of matrices and vectors, and their ability to make efficient use of the inherent structure in multidimensional data in order to perform dimensionality reduction and component extraction makes them a powerful tool in the analysis of Big Data. Owing to their flexibility and a scalable way in which they deal with multi-way data, tensors have found application in a wide range of disciplines, from very theoretical ones, such as physics and numerical analysis [1, 2], to the more practically relevant signal processing applications [3, 4].

The sheer high dimensionality of tensors means that the application of standard numerical methods may be intractable, as in the raw tensor format the required storage memory and number of operations in their manipulation grow exponentially with the tensor order (*curse of dimensionality*) [5]. To overcome this issue, tensor decompositions (TDs) aim to represent tensors in an approximate but much more efficient way through multilinear operations over the latent factors. The most well-known TD approaches are the Canonical Polyadic [6], the Tucker [7], and the Tensor Train decompositions [5] (CPD, TKD, and TT respectively).

Any TD can be considered as a special case of the more general concept of a tensor network (TN), which represents a high order tensor as a set of sparsely interconnected small scale core tensors and the associated factor matrices [4]. In other words, TNs can be viewed as multi-core interconnections of features of the original tensor. The advantages of representing a tensor as a TN are: (i) TNs are perfectly suited to deal with the curse of dimensionality, as a high order tensor, represented as a TN, can be stored on different machines which deal with only the individual cores, (ii) each core may be representative of a specific characteristic of the underlying tensor, thus implying inherent feature extraction from the original data. Despite these advantages, open problems in practical design of TNs include: (i) the choice of TD for a particular application, (ii) minimization of the number of the parameters necessary for a TN representation [5], and (iii) a rigorous framework to combine TNs.

In this work, we address issue (iii), and introduce a TN summation operator for TNs of the same topology. Summation is the most natural way to reduce the number of entities, and we show that a sum of multiple tensors (summands), *in their TN format*, preserves their underlying structure. In this way, the sum of TNs yields another isomorphic TN, the cores of which are a combination of the corresponding cores of the summand TNs. We show that in the TN context, the physical interpretation of summation represents feature fusion of the original tensors in the raw format, however, algorithms for tensor network summation are still in their infancy.

To this end, we introduce a framework for the summation of TNs, achieved by exploiting the block structure of the corresponding original cores. We illuminate that interconnections among the cores in a TN describe the coupling of data structures of the original tensors. This serves as a basis to explore ways to combine the *corresponding individual cores* of two TNs with the same topology in order to obtain a *new TN*, the cores of which carry information jointly present in the individual original cores. This is related to the recently introduced concept of common feature extraction in [8, 9], however, unlike matrices, the proposed framework enables this operation on tensors of any order, with the only condition that the original tensors are represented as isomorphic TNs. Practical advantages are demonstrated through an image classification application based on the ETH-80 dataset [10], whereby every dataset entry is represented as a TN, and their features are combined via the tensor summation framework. This allows us to extract relevant shared information in the original data, which is shown to yield significant advantages in terms of classification rates when used in conjunction with standard machine learning classifiers. The proposed framework therefore both opens up new perspectives on algebraic manipulation of TNs, and represents a first step towards establishing their taxonomy, thus removing the preconception that they have to be treated as stand-alone entities, together with offering new avenues for their applications.

ISBN 978-90-827970-1-5 

## II. Notation and Background

A tensor of order $N$ is denoted by underlined boldface uppercase letters, $\underline{\mathbf{X}} \in \mathbb{R}^{I_1\times\cdots\times I_N}$, a matrix by boldface uppercase letters, $\mathbf{X} \in \mathbb{R}^{J\times K}$, a vector by boldface lowercase letters, $\mathbf{x} \in \mathbb{R}^{N\times 1}$, and a scalar by italic lowercase letters, $x \in \mathbb{R}$. Subscripts are generally described by indices $n, i, j, k$. An element in an $N$-th order tensor is denoted by $x_{i_1,i_2,\ldots,i_N} = \underline{\mathbf{X}}(i_1, i_2, \ldots, i_N)$. For an $N$-th order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1\times\cdots\times I_n\times\cdots\times I_N}$ and an $M$-th order tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{J_1\times\cdots\times J_m\times\cdots\times J_M}$, with $I_n = J_m$, their $(m, n)$-contraction product is given by $\underline{\mathbf{Z}} = \underline{\mathbf{X}} \times_n^m \underline{\mathbf{Y}}$, where $\underline{\mathbf{Z}}$ is an $(N+M-2)$-th order tensor (for more detail we refer to [11, 12]). By convention $\times_n$ is equivalent to $\times_n^2$, and is referred to as mode-$n$ contraction. The mode-$n$ unfolding of a tensor $\underline{\mathbf{X}}$ rearranges its elements into a matrix, and is expressed as $\mathbf{X}_{(n)}$ (see [11] for more detail). The symbol $\otimes$ denotes the Kronecker product, $\circ$ the outer product, and $||\cdot||$ the Frobenius norm. A TN representation of a tensor $\underline{\mathbf{X}}$ is denoted by a calligraphic bold letter, $\boldsymbol{\mathcal{X}}$. Finally, the operator $\text{vec}(\cdot)$ designates vectorization of a tensor. Examples of implementations can be found in software packages for tensors such as HOTTBOX [13].

### A. Tucker Decomposition

The TKD is analogous to a higher order form of matrix factorization, and decomposes an original tensor $\underline{\mathbf{X}}$ into a core tensor contracted by a factor matrix along each corresponding mode [7]. In the case of a 3-rd order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1\times I_2\times I_3}$, the TKD is expressed as

$$\begin{aligned}\underline{\mathbf{X}} &= \underline{\mathbf{G}} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} + \underline{\mathbf{E}} \\ &= \sum_q^Q \sum_r^R \sum_p^P g_{qrp} \mathbf{a}_q \circ \mathbf{b}_r \circ \mathbf{c}_p + \underline{\mathbf{E}}\end{aligned} \tag{1}$$

where $\underline{\mathbf{G}} \in \mathbb{R}^{Q\times R\times P}$ and $\mathbf{A} \in \mathbb{R}^{I\times Q}, \mathbf{B} \in \mathbb{R}^{J\times R}$, $\mathbf{C} \in \mathbb{R}^{K\times P}$. If $\{Q, R, P\} < \{I_1, I_2, I_3\}$, the TKD is not exact, and a residual $\underline{\mathbf{E}}$ is present. Analogously, a TKD for an $N$-th order tensor is given by

$$\underline{\mathbf{X}} \approx \underline{\mathbf{G}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \cdots \times_N \mathbf{A}^{(N)} \tag{2}$$

For convenience, any tensor expressed in this form will be referred to as "in the TKD format", even though the factors $\mathbf{A}^{(n)}$ may not be necessarily obtained via a TKD. The mode-$n$ unfolding of a tensor in the TKD format is given by

$$\mathbf{X}_{(n)} \approx \mathbf{A}^{(n)} \mathbf{G}_{(n)} \left(\mathbf{A}^{(N)} \otimes \cdots \otimes \mathbf{A}^{(n-1)} \otimes \mathbf{A}^{(n+1)} \otimes \cdots \otimes \mathbf{A}^{(1)}\right)^T \tag{3}$$

### B. Background on Tensor Networks

A decomposition of a tensor into an arrangement of multi-way linked tensors and matrices leads to expressions which often involve numerous contraction products; this can be cumbersome to write and hard to visualize. For this reason, it is becoming common to represent tensors diagrammatically [11], as shown in Fig. 1. An $N$-th order tensor is represented as a node (circle) with as many edges (modes) as the tensor order. In TNs, contractions are designated by linking two common modes, called *contraction modes*, while "dangling" edges are *physical modes* of the represented tensor.

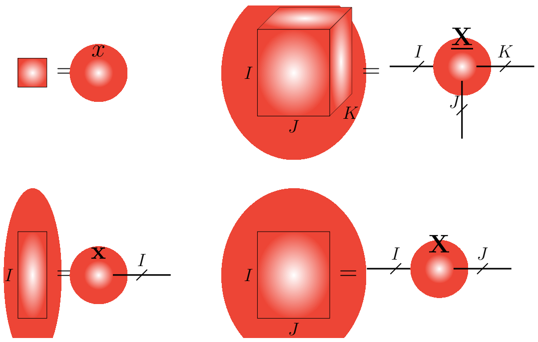

Fig. 1: Building blocks of TNs. Anticlockwise from the top-left: scalar, vector, matrix, and 3-rd order tensor. The edges are referred to as modes, and the associated labels $I, J, K$, indicate their dimensionality.

Two examples of TNs are provided in Fig. 2, whereby any mode which is not a physical mode is a contraction mode. Similarly, nodes connected to one or more physical modes are referred to as *physical nodes*, while the rest are *contraction nodes*. The "shape" of a TN represents its topology, where the concept of topology is the same as that adopted in Graph Theory [14].

**Remark 1**: Each node in a TN, $\boldsymbol{\mathcal{X}}$, can represent either a particular feature of the original tensor $\underline{\mathbf{X}}$ (in case of physical nodes), or a model of how features are combined (in case of contraction nodes).

It is then clear that, a framework for combining TNs is a prerequisite to mixing features of individual cores, which would then allow the extraction of the common features across the individual tensors. Yet, the "feature locality" inherent to the nodes of a TN is of fundamental importance, but is still under-explored. Therefore, the main motivation for this work has been to establish the missing framework for TN summation via a combination of their corresponding cores, thus simultaneously performing a feature fusion.

**Note:** In this work, we refer to any node within a TN as "core" when treating it as a tensor, while within a topological context, we use the term "node".

## III. Sum of Tensor Networks

To establish a framework for the summation of two or more TNs, assume that the individual TNs have: (i) physical modes of equal dimensions, (ii) the same topology. We proceed by providing relevant definitions, the proposed framework, and related mathematical formalism.

**Definition 1.** A **block tensor** is a tensor that is arranged into sub-tensors called **blocks**, that is, its entries are tensors of the same order but not necessarily of the same dimensionality.

**Definition 2.** The **superdiagonal** of a block tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ is the collection of entries $x_{i_1,i_2,\ldots,i_n}$, where $i_1 = i_2 = \cdots = i_n$.

ISBN 978-90-827970-1-5  

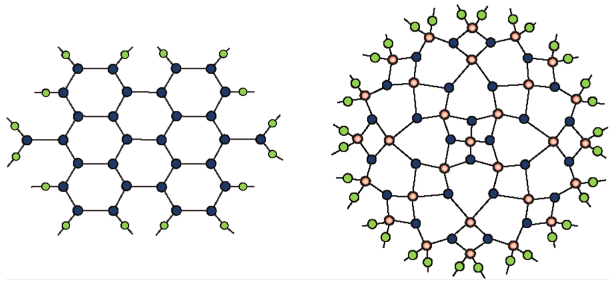

Fig. 2: Examples of TNs (figure adapted from [11]). *Left*: A TN representing a 16-th order tensor, with contraction nodes shown in blue, and physical nodes shown in green. *Right*: A TN representing a 32-th order tensor, with contraction nodes shown in blue and orange, and physical nodes in green.

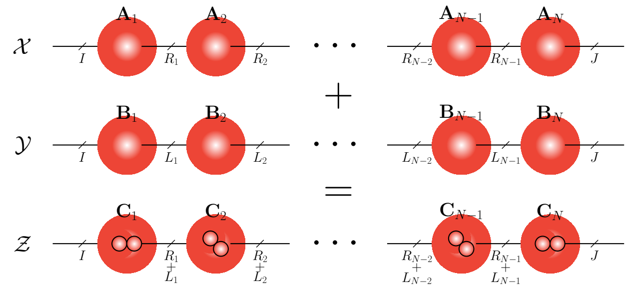


Fig. 3: Topology preservation for the TN summation operator, illustrated for a sum of matrices $\mathbf{X}$ and $\mathbf{Y}$, expressed as a matrix chain. The matrices (nodes) $\mathbf{C}_n$ are composed of the matrices $\mathbf{A}_n$ and $\mathbf{B}_n$, which are arranged according to the sum of TNs framework, either through (i) concatenation or (ii) block-diagonal arrangement.

**Sum of TNs Framework.** Consider two tensors $\underline{\mathbf{X}}, \underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ represented as TNs denoted by $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ which have equivalent topologies, but not necessarily the same dimensionality of contracting modes. The sum $\underline{\mathbf{Z}} = \underline{\mathbf{X}} + \underline{\mathbf{Y}}$ can then be thought of as a new TN, $\boldsymbol{\mathcal{Z}}$, with an equivalent topology to $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$. Its contraction nodes are in the form of a block tensor which is obtained by stacking the corresponding contraction nodes of $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ along its superdiagonal. The physical nodes of $\boldsymbol{\mathcal{Z}}$ are obtained by arranging the corresponding physical nodes of $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ in such a way that the dimensionality of all contracting modes is increased but the dimensionality of the physical modes is kept fixed.

**Proposition 1.** The proposed framework for a sum of TNs is valid for chains of matrices.

*Proof.* Suppose $\mathbf{X} = \mathbf{A}_1\mathbf{A}_2\cdots\mathbf{A}_N$, $\mathbf{Y} = \mathbf{B}_1\mathbf{B}_2\cdots\mathbf{B}_N$, where $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{I \times J}$, and $\mathbf{A}_n, \mathbf{B}_n \in \mathbb{R}^{R_n \times R_{n+1}}$, with $R_0 = I, R_N = J$. Define a new chain of matrices $\mathbf{Z} = \mathbf{C}_1\mathbf{C}_2\cdots\mathbf{C}_N$, where each $\mathbf{C}_n$, $n = 1, \ldots, N$, is an arrangement of $\mathbf{A}_n, \mathbf{B}_n$ according to the proposed framework. By a direct inspection of $\mathbf{Z}$, we have

$$\begin{aligned} \mathbf{Z} &= \mathbf{C}_1\mathbf{C}_2\cdots\mathbf{C}_N \\ &= \begin{bmatrix}\mathbf{A}_1 & \mathbf{B}_1\end{bmatrix}\begin{bmatrix}\mathbf{A}_2 & \mathbf{0}\\ \mathbf{0} & \mathbf{B}_2\end{bmatrix}\cdots\begin{bmatrix}\mathbf{A}_{N-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{B}_{N-1}\end{bmatrix}\begin{bmatrix}\mathbf{A}_N\\ \mathbf{B}_N\end{bmatrix} \\ &= \mathbf{A}_1\mathbf{A}_2\cdots\mathbf{A}_n + \mathbf{B}_1\mathbf{B}_2\cdots\mathbf{B}_n \\ &= \mathbf{X} + \mathbf{Y} \end{aligned} \tag{4}$$

□

Fig. 3 shows a graphical illustration of Proposition 1.

**Remark 2**: The above result for matrices is well-known, and serves here as a specific and straightforward intuition behind the general concept of TN summation.

**Proposition 2.** The TN summation operator is valid for any tensor expressed in the TKD format.

*Proof.* For simplicity, we here provide proof for 3-rd order tensors, but without loss of generality the result holds for sum of tensors of any order. Fig. 4 shows the tensors $\underline{\mathbf{X}}, \underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ in the TKD format, expressed as

$$\begin{aligned} \underline{\mathbf{X}} &= \underline{\mathbf{G}}_x \times_1 \mathbf{A}_x \times_2 \mathbf{B}_x \times_3 \mathbf{C}_x \\ \underline{\mathbf{Y}} &= \underline{\mathbf{G}}_y \times_1 \mathbf{A}_y \times_2 \mathbf{B}_y \times_3 \mathbf{C}_y \end{aligned} \tag{5}$$

with the respective TN representations, $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$. A new TN, $\boldsymbol{\mathcal{Z}}$, is then obtained by combining $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ according to the proposed framework (see Fig. 4), and the so generated tensor, $\underline{\mathbf{Z}}$, can be described by

$$\underline{\mathbf{Z}} = \underline{\mathbf{G}}_z \times_1 \mathbf{A}_z \times_2 \mathbf{B}_z \times_3 \mathbf{C}_z \tag{6}$$

We next show that $\underline{\mathbf{Z}} = \underline{\mathbf{X}} + \underline{\mathbf{Y}}$, based on the mode-1 unfolding, however, note that the same procedure can be applied to any mode. Define $\mathbf{A}_z = \begin{bmatrix}\mathbf{A}_x & \mathbf{A}_y\end{bmatrix}$, and $\underline{\mathbf{G}}_z$ as an arrangement of $\underline{\mathbf{G}}_x$ and $\underline{\mathbf{G}}_y$ along the superdiagonal of $\underline{\mathbf{G}}_z$, and consider

$$\begin{aligned} \mathbf{K}_1 &= (\mathbf{C}_x \otimes \mathbf{B}_x)^T \\ \mathbf{K}_2 &= (\mathbf{C}_y \otimes \mathbf{B}_y)^T \end{aligned} \tag{7}$$

Upon performing the mode-1 unfolding of $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$ according to (3), and combining the matrices $\mathbf{X}_{(1)}, \mathbf{Y}_{(1)}$ in their TN formats, then from Proposition 1 the resulting matrix, $\mathbf{Z}_*$, can be expressed as

$$\mathbf{Z}_* = \begin{bmatrix}\mathbf{A}_x & \mathbf{A}_y\end{bmatrix}\begin{bmatrix}\mathbf{G}_{x(1)} & \mathbf{0}\\ \mathbf{0} & \mathbf{G}_{y(1)}\end{bmatrix}\begin{bmatrix}\mathbf{K}_1 & \mathbf{K}_2\end{bmatrix}^T \tag{8}$$

Therefore, in order to prove Proposition 2 it is sufficient to show that $\mathbf{Z}_{(1)} = \mathbf{Z}_*$, where $\mathbf{Z}_{(1)}$ is the mode-1 unfolding of $\underline{\mathbf{Z}}$, represented as in (6). To this end, consider

$$\begin{aligned} \mathbf{Z}_{(1)} &= \mathbf{A}_z\mathbf{G}_{z(1)}(\mathbf{C}_z \otimes \mathbf{B}_z)^T \\ &= \begin{bmatrix}\mathbf{A}_x & \mathbf{A}_y\end{bmatrix}\mathbf{G}_{z(1)}\left(\begin{bmatrix}\mathbf{C}_x & \mathbf{C}_y\end{bmatrix} \otimes \begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix}\right)^T \end{aligned} \tag{9}$$

For convenience, denote $\underline{\mathbf{X}}, \underline{\mathbf{Y}}, \underline{\mathbf{G}}_x, \underline{\mathbf{G}}_y \in \mathbb{R}^{2\times 2\times 2}$, and define $\underline{\mathbf{G}}_\alpha(:,:,j) = \hat{\mathbf{G}}_\alpha(j)$, where $\alpha \in \{x, y\}$. Hence,

$$\mathbf{G}_{z(1)} = \begin{bmatrix}\hat{\mathbf{G}}_x(1) & \mathbf{0} & \hat{\mathbf{G}}_x(2) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \hat{\mathbf{G}}_y(1) & \mathbf{0} & \hat{\mathbf{G}}_y(2)\end{bmatrix} \tag{10}$$

ISBN 978-90-827970-1-5 © EURASIP 2018

Without loss of generality, assume the concrete values

$$\mathbf{C}_x = \begin{bmatrix} 1 & 2 \\ 5 & 6 \end{bmatrix} \qquad \mathbf{C}_y = \begin{bmatrix} 3 & 4 \\ 7 & 8 \end{bmatrix} \tag{11}$$

to give

$$\begin{aligned} &\begin{bmatrix}\mathbf{C}_x & \mathbf{C}_y\end{bmatrix} \otimes \begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} = \\ &= \begin{bmatrix} 1\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} & 2\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} & 3\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} & 4\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} \\ 5\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} & 6\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} & 7\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} & 8\begin{bmatrix}\mathbf{B}_x & \mathbf{B}_y\end{bmatrix} \end{bmatrix} \\ &= \mathbf{U} \end{aligned} \tag{12}$$

Upon substituting $\mathbf{G}_{z(1)}$ and $\mathbf{U}$ into (9) and making use of the sparse nature of (10), we arrive at

$$\begin{aligned} &\mathbf{Z}_{(1)} = \mathbf{A}_z \mathbf{G}_{z(1)} \mathbf{U}^T \\ &= \begin{bmatrix}\mathbf{A}_x & \mathbf{A}_y\end{bmatrix} \begin{bmatrix} \hat{\mathbf{G}}_x(1) & \hat{\mathbf{G}}_x(2) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \hat{\mathbf{G}}_y(1) & \hat{\mathbf{G}}_y(2) \end{bmatrix} \begin{bmatrix} 1\mathbf{B}_x^T & 5\mathbf{B}_x^T \\ 2\mathbf{B}_x^T & 6\mathbf{B}_x^T \\ 3\mathbf{B}_y^T & 7\mathbf{B}_y^T \\ 4\mathbf{B}_y^T & 8\mathbf{B}_y^T \end{bmatrix} \\ &= \begin{bmatrix}\mathbf{A}_x & \mathbf{A}_y\end{bmatrix} \begin{bmatrix} \mathbf{G}_{x(1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_{y(1)} \end{bmatrix} \begin{bmatrix} \mathbf{K}_1^T \\ \mathbf{K}_2^T \end{bmatrix} \\ &= \begin{bmatrix}\mathbf{A}_x & \mathbf{A}_y\end{bmatrix} \begin{bmatrix} \mathbf{G}_{x(1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_{y(1)} \end{bmatrix} \begin{bmatrix}\mathbf{K}_1 & \mathbf{K}_2\end{bmatrix}^T = \mathbf{Z}_* \end{aligned} \tag{13}$$

□

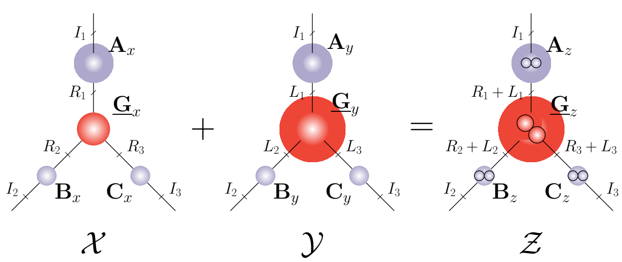


Fig. 4: Topology of a sum of tensors $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$ in the TKD format, to give $\boldsymbol{\mathcal{Z}} = \boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{Y}}$.

## IV. Experimental Results

The TN summation operator was validated through a practical example of feature fusion in image classification. We considered the benchmark ETH-80 dataset, which consists of 3280 images, composed of 8 classes, with 10 objects per class and 41 images per object. For our simulations, the images were rescaled to $32 \times 32$ pixels. Given the RGB format of the considered images, the dataset is conveniently represented through 3-rd order tensors $\underline{\mathbf{X}}_m \in \mathbb{R}^{32\times 32\times 3}, m = 1, \ldots, M$ (where $M = 3280$). Simulations were performed in our software package for tensors: HOTTBOX [13].

Fig. 5 shows that for each image, $\underline{\mathbf{X}}_m$, in the training set, upon performing TKD, the size of the corresponding core tensors $\underline{\mathbf{G}}_m \in \mathbb{R}^{R_1\times R_2\times R_3}$ was set to $R_1 = R_2 = R_3 = 3$. These values were empirically found to offer a good approximation of the original images, while at the same time maintaining a small size core tensor. To emphasize the features of each image without affecting the TKD approximation, each core tensor $\underline{\mathbf{G}}_m$ was normalized to unit norm and the factor matrices $\{\mathbf{A}_m, \mathbf{B}_m, \mathbf{C}_m\}$ were scaled by $\eta_m^{1/3}$, where $\eta_m = ||\underline{\mathbf{G}}_m||$. Through TN summation, the scaled factor matrices $\{\mathbf{A}_m, \mathbf{B}_m, \mathbf{C}_m\}$ were concatenated into the matrices $\{\mathbf{A}_c, \mathbf{B}_c, \mathbf{C}_c\}$ and the corresponding core tensors into $\underline{\mathbf{G}}_c$. Since our goal was to examine the mixture of actual features, we focused on the factor matrices $\{\mathbf{A}_c, \mathbf{B}_c, \mathbf{C}_c\}$, onto which an SVD was subsequently applied. For dimensionality reduction, the first $\{R_1, R_2, R_3\}$ singular vectors were retained (we refer to this operator as tSVD(·)), to yield matrices $\{\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t\}$. Finally, for each image, $\underline{\mathbf{X}}_m$, a new core tensor was computed as

$$\mathring{\underline{\mathbf{G}}}_m = \underline{\mathbf{X}}_m \times_1 \mathbf{A}_t^T \times_2 \mathbf{B}_t^T \times_3 \mathbf{C}_t^T \tag{14}$$

The vectorized versions $\text{vec}(\mathring{\underline{\mathbf{G}}}_m), m = 1, \ldots, M$, were then fed to standard machine learning classifiers for performance evaluation. During the testing stage, for each new element $\underline{\mathbf{X}}_*$, $\mathring{\underline{\mathbf{G}}}_*$ was computed via (14) using the $\{\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t\}$ matrices obtained during training, and subsequently vectorized.

**Remark 2:** The physical meaning of equation (14) is a projection of the raw images onto the feature space that is common to the whole dataset, which greatly reduces the risk of overfitting, hence ensuring that $\mathring{\underline{\mathbf{G}}}_m$ can be used for classification purposes.

**Algorithm 1.** Sum of TNs for image classification

```
Input: Dataset {X_m}_{m=1}^M, core size {R_1, R_2, R_3}

Initialize {A_c, B_c, C_c}
for each element m in dataset do
    X_m = G_m ×_1 ×A_m ×_2 B_m ×_3 C_m
    η_m = ||G_m||
    A_c = [A_c  η_m^{1/3} A_m]
    B_c = [B_c  η_m^{1/3} B_m]
    C_c = [C_c  η_m^{1/3} C_m]
end for
A_t = tSVD(A_c, R_1)
B_t = tSVD(B_c, R_2)
C_t = tSVD(C_c, R_3)
for each element m in dataset do
    G̊_m = X_m ×_1 A_t^T ×_2 B_t^T ×_3 C_t^T
end for
Train classifier on {vec(G̊_m)}_{m=1}^M.
```

The procedure outlined in Algorithm 1 was applied to the ETH-80 dataset by randomly selecting 75% of the available images to serve as training data. The classifiers used were: (i) SVM, which employed an RBF kernel and a one-vs-one approach for multi-class classification, (ii) kNN with the number of neighbours (Euclidean distance) set to three, (iii) a Multilayer Perceptron NN (MLP) with two hidden layers and a ReLU activation function, and (iv) a Random Forest (RF) with 10 trees, the nodes of which were expanded until

ISBN 978-90-827970-1-5 © EURASIP 2018 

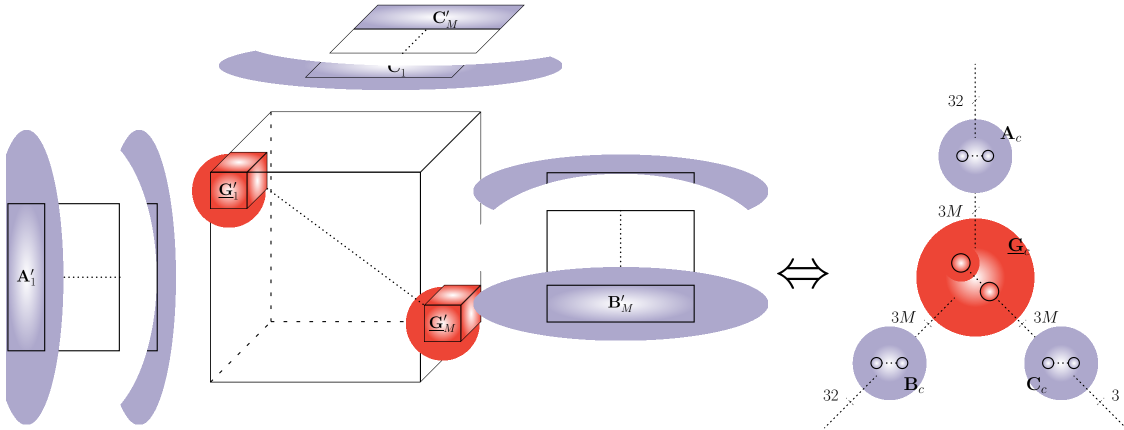

Fig. 5: An example of TN summation for the TKD representations of the images in the training set, where $\{\mathbf{A}'_m, \mathbf{B}'_m, \mathbf{C}'_m\} = \eta_m^{1/3}\{\mathbf{A}_m, \mathbf{B}_m, \mathbf{C}_m\}$ and $\underline{\mathbf{G}}'_m = \underline{\mathbf{G}}_m/\eta_m$ *Left*: A TKD arrangement of matrices and core tensors. *Right*: Sum of the Tucker factors represented in the TN format. Notice that upon TN summation the dimensions of the physical modes remain unchanged.

TABLE I: Classification rates for the ETH-80 dataset, in %.

| | SVM | kNN | MLP | RF |
|---|---|---|---|---|
| Original images | 89.84% | 88.31% | 85.43% | 83.33% |
| Tensor stacking | 92.94% | 88.55% | 88.28% | **89.75%** |
| Sum of TNs | **94.81%** | **90.51%** | **89.23%** | 86.20% |

all leaves were pure. Each classifier was trained based on the data features obtained via the proposed TN summation, as well as on the original data. For a fair comparison, the features extracted via another tensor method were also employed, which involves stacking the images into a higher-order tensor (see [15] for details), referred to as "tensor stacking". The classification rates were computed as an average of 50 realizations and the results are summarized in TABLE I. Observe the benefits of the proposed method for the SVM, kNN and MLP based classifiers. In particular, with TN summation, the SVM classifier attained an improvement of about 2% over tensor stacking, and almost 5% with respect to a direct usage on the original data.

## V. Conclusion

We have introduced a mathematical formalism behind the sum of tensor networks (TNs), and have validated this operator for both chains of matrices (2-nd order tensors) and for general tensors in the Tucker format. The TN summation has been achieved through a block arrangement of the original cores of two or more TNs, to yield a new TN whereby the features of the individual summands are embedded in its cores. Through an analogy with feature fusion, we have proposed a new algorithm for image classification, which rests solely upon the proposed sum of TNs. Tests on the ETH-80 dataset have demonstrated that the mixture of features obtained via tensor network summation can significantly enhance classification rates of standard machine learning classifiers.

## References

[1] T. G. Kolda, "Orthogonal tensor decompositions," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 1, pp. 243–255, 2001.
[2] L. de Lathauwer, B. D. Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.
[3] P. Comon, "Tensors: A brief introduction," *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 44–53, 2014.
[4] A. Cichocki, D. P. Mandic, A. H. Phan, C. F. Caiafa, G. Zhou, Q. Zhao, and L. D. Lathauwer, "Tensor decompositions for signal processing applications," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015.
[5] V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.
[6] R. Bro, "PARAFAC. Tutorial and applications," *Chemometrics and Intelligent Laboratory Systems*, vol. 38, no. 2, pp. 149–171, 1997.
[7] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.
[8] G. Zhou, A. Cichocki, Y. Zhang, and D. Mandic, "Group component analysis for multiblock data: Common and individual feature extraction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 11, pp. 2426–2439, 2016.
[9] I. Kisil, G. G. Calvi, and D. P. Mandic, "Common and individual feature extraction using tensor decompositions: A remedy for the curse of dimensionality?" in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, p. TBA.
[10] B. Leibe and B. Schiele, "Analyzing appearance and contour based methods for object categorization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2003, pp. 409–415.
[11] A. Cichocki, N. Lee, I. 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.
[12] A. Cichocki, A.-H. Phan, Q. Zhao, N. Lee, I. 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.
[13] I. Kisil, A. Moniri, G. G. Calvi, B. Scalzo Dees, and D. P. Mandic, "HOTTBOX: Higher Order Tensor ToolBOX," https://github.com/hottbox.
[14] O. Morris, M. Lee, and A. Constantinides, "A unified method for segmentation and edge detection using graph theory," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 11, 1986, pp. 2051–2054.
[15] A. H. Phan and A. Cichocki, "Tensor decompositions for feature extraction and classification of high dimensional datasets," *Nonlinear theory and its applications, IEICE*, vol. 1, no. 1, pp. 37–68, 2010.

ISBN 978-90-827970-1-5 © EURASIP 2018