High-Dimensional Data Learning Based on Tensorial-Singular Space of Tensor Train Cores

Karmouda Ouafae[a]
ouafae.karmouda@univ-lille.fr

Boyer Remy[a]
remy.boyer@univ-lille.fr

Boulanger Jeremie[a]
jeremie.boulanger@univ-lille.fr

[a]University of Lille
CNRS, CRISTAL
Lille, France

***Abstract*—Tensors are multidimensional data structures used to represent many real world data. In the context of supervised learning, Support Vector Machines (SVMs) are known to be very efficient for different classification tasks. In this work, we propose a kernel metric for SVM to deal with non linear classification problems. First, we use the Tensor Train Decomposition (TTD) to decompose a tensor into TT-cores of order three and two matrices. In order to mitigate the problem of non-uniqueness of TTD, we propose a kernel based on the tensorial singular subspaces spanned by TT-cores. The TT-based kernel function proposed is based on the tools of t-Algebra of 3-rd order tensors. We also show that it is possible to use different kernel functions on each TT-core. Numerical experiments on real-world datasets show the competitivity of our approach compared to existing methods and the superiority of our method when dealing with few-sample of high-dimensional inputs.**

***Index Terms*—Tensor Train Decomposition, subspaces, kernel**

I. Introduction

Tensors are seen as multidimensional extensions of matrices represented by multidimensional arrays. The order of a tensor is defined as the number of its dimensions. For example, a scalar is a first-order tensor, a matrix is a 2-order tensors, 3-rd order tensors are represented as cubes, etc.
Different works extended Support Vector Machines (SVMs) to tensor data. For instance, DuSK [1] uses the Canonical Polyadic Decomposition (CPD) to decompose input tensors and proposes a kernel function on the decomposition to treat non linear classification problems. However, this method uses the ALS (Alternating Least Squares) algorithm for computing the CPD that is not guaranteed to converge to a global minimum [2], and because of the ambiguities of the CPD model, this method suffers from low accuracy scores. Signoretto et al. method [3] proposes a Grassmannien tensor-based kernel. Specifically, it defines a kernel function by considering the matrix-based subspaces spanned by factors of the HOSVD (Higher-Order Singular Value Decomposition). Recently, the method KSTTMs [4] proposes a kernelized support tensor train machines. Based on the TTD, the author propose a kernel metric based on kernel mappings on the different fibers of TT-cores. However, since TTD is not unique (cf. sec I-C2), the formula of the kernel function in [4] may suffer from ambiguities. To overcome this, we propose the following methodology. First, we use the Tensor Train Decomposition (TTD) that is one of the simplest tensor network and is able to mitigate the curse of dimensionality [5]. Furthermore, it has proven to be efficient in removing redundant data and provides a compact representation [6]. Next, in order to address the problem of non-unicity of the TTD, our approach consists on defining a kernel function w.r.t the subspaces generated by the cores of TTD using the powerful tools of tensor algebra of 3-rd order tensors proposed in [7], [8]. The superiority of our method is validated through different experiments.
The rest of the paper is organised as follows. We first present some preliminaries that contains some tensors basics, a backgroung in t-Algebra and an overview of SVMs in the tensor case in section I-D. We formulate our proposed method in section II and validate our approach through numerical experiments in section III.

A. Tensors basics

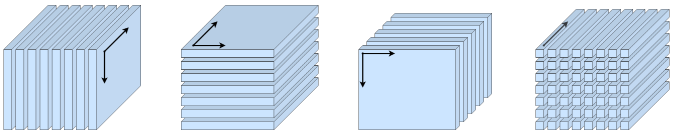
1) Notations and definitions: The notations used through this paper are the following: Vectors, matrices and tensors are represented by x, X and $\mathcal{X}$. The $(i_1, i_2, \cdots, i_Q)$-th entry of the Q-order tensor $\mathcal{X}$ is denoted by $\mathcal{X}(i_1, i_2, \cdots, i_Q)$.
Definition (Inner product): The inner product of two Q-order tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1\times\cdots\times I_Q}$ is defined as:

$$\langle \mathcal{X}, \mathcal{Y}\rangle = \sum_{i_1=1}^{I_1}\sum_{i_2=1}^{I_2}\cdots\sum_{i_Q=1}^{I_Q} \mathcal{X}(i_1,\ldots,i_q)\mathcal{Y}(i_1,\ldots,i_q).$$

Definition (Tensor Frobenius Norm): The norm of a tensor $\mathcal{X}$ is defined as:

$$||\mathcal{X}||_F = \sqrt{\sum_{i_1=1}^{I_1}\sum_{i_2=1}^{I_2}\cdots\sum_{i_Q=1}^{I_Q} \mathcal{X}^2(i_1,\ldots,i_q)}.$$

Definition (Tensor contraction): The contraction product $\times_q^p$ between two tensors $\mathcal{X}$ and $\mathcal{Y}$ of size $I_1\times\cdots\times I_Q$ and $J_1\times\cdots\times J_P$ respectively with $I_q = J_p$ is a tensor of order $Q+P-2$ such as:



Horizontal: $\mathcal{X}(i,:,:)$ Vertical: $\mathcal{X}(:,j,:)$ Frontal: $\mathcal{X}(:,:,k)$ Tubal: $\mathcal{X}(i,j,:)$

Fig. 1: Slices of a tensor $\mathcal{X}$ of order-3.

$$(\mathcal{X} \times_q^p \mathcal{Y})(i_1,\dots,i_{q-1},i_{q+1},\dots,i_Q,j_1,\dots,j_{p-1},j_{p+1},\dots,j_P)$$
$$= \sum_{k=1}^{I_q} \mathcal{X}(i_1,\dots,i_{q-1},k,i_{q+1},\dots,i_Q)$$
$$\mathcal{Y}(j_1,\dots,j_{p-1},k,j_{q+1},\dots,j_Q).$$

B. Background in t-Algebra

In this section, we present some tensor operations between 3-order tensors proposed in [7], [8].
Slices are defined as two-dimensional sections of a tensor. They are found by fixing all indices but two. A 3-rd order tensor $\mathcal{X}$ has horizontal, lateral and frontal slices denoted by $\mathcal{X}(i,:,;)$, $\mathcal{X}(:,j,;)$ and $\mathcal{X}(:,:,k)$. This later will be denoted as X_k. Tubes are the generalization for higher-order tensors of columns and rows of a matrix. They are found by fixing all indices but one.
Figure 1 shows the different slices and tubes of a 3-order tensor.

A block circulant matrix of a tensor is defined using its frontal slices. The block circulant matrix $circ(\mathcal{X})$ of a 3-rd order tensor $\mathcal{X}$ of size $I_1 \times I_2 \times I_3$ is defined as :

$$circ(\mathcal{X}) = \begin{bmatrix} X_1 & X_{I_3} & X_{I_3-1} & \cdots & X_2 \\ X_2 & X_1 & X_{I_3} & \cdots & X_3 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ X_{I3} & X_{I_3-1} & \ddots & X_2 & X_1 \end{bmatrix}$$

The $MatVec$ operation takes a 3-rd order tensor as input $\mathcal{X}$ of size $I_1 \times I_2 \times I_3$ and returns a block matrix of size $I_1 I_3 \times I_2$:

$$MatVec(\mathcal{X}) = [X_1^T, \cdots, X_{I_3}^T]^T.$$

The $fold$ operation takes back $MatVec(\mathcal{X})$ to its tensor form:

$$fold(MatVec(\mathcal{X})) = \mathcal{X}.$$

Definition (**Block diagonal matrix**): Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$. The block diagonal matrix of $\mathcal{X}$ contains the frontal slices of $\mathcal{X}$ in its diagonal as follows:

$$bdiag(\mathcal{X}) = \begin{bmatrix} X_1 & & & \\ & X_2 & & \\ & & \ddots & \\ & & & X_{I_3} \end{bmatrix}$$

Definition (T-product): Let $\mathcal{X}$ be $I_1 \times l \times I_3$ and $\mathcal{Y}$ be $l \times I_2 \times I_3$. Then, the t-product $\mathcal{X} * \mathcal{Y}$ is a tensor of size $I_1 \times I_2 \times I_3$ defined as:

$$\mathcal{X} * \mathcal{Y} = fold(circ(\mathcal{X})MatVec(\mathcal{Y})). \quad (1)$$

It shall be noted that the computation of the t-product in (1) demands a high-computationnal cost. In practice, the t-product is computed using the Discret Fourier Transform (DFT), reader can see [7] for more details.

Definition (T-transpose): For a tensor $\mathcal{X}$ of size $I_1 \times I_2 \times I_3$, its transpose $\mathcal{X}^T$ is a tensor of size $I_2 \times I_1 \times I_3$ obtained by transposing the frontal slices $\mathcal{X}(:,:,k)$ and reversing their order from 2 through I_3.

Definition (Identity tensor): The identity tensor $\mathcal{I}_{IIJ}$ is a tensor whose first frontal slice is the identity matrix I_I and whose all other frontal slices are zeros.

Definition (Tensor inverse) : A tensor $\mathcal{X}$ of size $I \times I \times J$ has an inverse $\mathcal{X}^{-1}$ if:

$$\mathcal{X} * \mathcal{X}^{-1} = \mathcal{X}^{-1} * \mathcal{X} = \mathcal{I}.$$

Definition (Orthogonal tensor): A real tensor of size $I \times I \times J$ is orthogonal if:

$$\mathcal{X}^T * \mathcal{X} = \mathcal{X} * \mathcal{X}^T = \mathcal{I}.$$

Definition (**F-diagonal tensor**): A tensor is f-diagonal if each of its frontal slices is a diagonal matrix.

Definition (Tubal scalar): An element $c \in \mathcal{R}^{1 \times 1 \times n}$ is called a tubal scalar of length n.

Definition (Range): The range of a tensor $\mathcal{X}$ is the t-linear span of the lateral slices of $\mathcal{X}$:

$$span(\mathcal{X}) = \{\ X_1 * c_1 + \cdots + X_{I_3} * c_{I_3}, c_k \in \mathbb{R}^{1 \times 1 \times I_3}\}.$$

Definition (T-svd): Let $\mathcal{X}$ a $I_1 \times I_2 \times I_3$ be a real-tensor. Then, $\mathcal{X}$ can be decomposed as:

$$\mathcal{X} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T,$$

where $\mathcal{U}$ and $\mathcal{V}$ are orthogonal tensors of size $I_1 \times I_1 \times I_3$ and $I_2 \times I_2 \times I_3$ respectively and $\mathcal{S}$ is f-diagonal. Specifically, each frontal slice of $\mathcal{S}$ is diagonal and its diagonal elements are in a decreasing order generalizing the notion of singular values of matrices to tensors. The number of nonzeros singular values of $\mathcal{S}$ defines its tubal rank.
Remark: As proven in [8], if $\mathcal{X} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T$, then, the lateral slices of $\mathcal{U}$ form an orthonormal basis for the range of $\mathcal{X}$ *i.e*, we have:

$$span(\mathcal{X}) = span(\mathcal{U}), \quad (2)$$

and their dimension is equal to the tubal rank of $\mathcal{X}$.

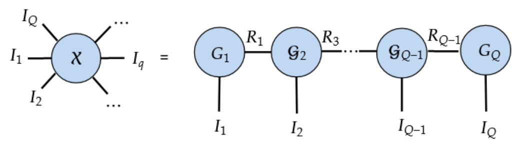


Fig. 2: TTD of a Q-order tensor with TT-ranks $(R_1, \cdots, R_{Q-1})$.

Definition (Projector): $\mathcal{P}$ is a projector if $\mathcal{P}^2 = \mathcal{P} * \mathcal{P} = \mathcal{P}$. If $\mathcal{X} \in span(P)$, then $P * \mathcal{X} = \mathcal{X}$.

Definition (Pseudo-inverse): For a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ whose frontal slices are full column rank with $I_2 < I_1$, we define its pseudo-inverse as :

$$\mathcal{X}^\dagger = (\mathcal{X}^T * \mathcal{X})^{-1} * \mathcal{X}^T.$$

In this case, $P = \mathcal{X} * \mathcal{X}^\dagger$ is an orthogonal projector onto the range of $\mathcal{X}$.

C. Tensor train decomposition (TTD)

TTD is one of the simplest tensor network and is able to mitigate the curse of dimensionality [6]. It decomposes a Q-order tensor into a graph (train) connected of $(Q-2)$ tensors of order 3 and 2 matrices. TTD is represented graphically in Figure 2. The nodes of this graph represent a tensor whose order is denoted by the number of edges. The number beside the edges is the TT-rank and the connection between two tensors correponds to the contraction product.

1) Definition: A tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_Q}$ admits a TTD with TT-ranks $(R_1, \ldots, R_{Q-1})$ if it can be expressed as:

$$\mathcal{X}(i_1, \ldots, i_Q) = \sum_{r_1=1}^{R_1} \ldots \sum_{r_2=1}^{R_2} \sum_{r_{q-1}=1}^{R_{Q-1}} G_1(i_1, r_1) \mathcal{G}_2(r_1, i_2, r_2) \ldots \mathcal{G}_{Q-1}(r_{Q-2}, i_{Q-1}, r_{Q-1}) G_Q(r_{Q-1}, i_Q),$$

where $\mathcal{G}_q \in \mathbb{R}^{R_{q-1} \times I_q \times R_q}$ for $2 \le q \le Q-1$, $G_1 \in \mathbb{R}^{I_1 \times R_1}$ and $G_Q^T \in \mathbb{R}^{I_Q \times R_{Q-1}}$ are the TT-cores. In the tensor format, TTD can be formulated using the contraction product between TT-cores as follows:

$$\mathcal{X} = G_1 \times_2^1 \mathcal{G}_2 \times_3^1 \mathcal{G}_3 \times_4^1 \cdots \times_{Q-1}^1 \mathcal{G}_{Q-1} \times_Q^1 G_Q. \quad (3)$$

Recall that the number of entries in the raw data Q-order cubic (I) tensor grows exponentially with respect to the tensor order. This means that to store $\mathcal{X}$ directly, $O(I^Q)$ entries needs to be stored. However, if only the decomposition of $\mathcal{X}$ is stored instead, *i.e*, its TTD, only $O(QIR^2)$ is required (as each core has IR^2 components. This clearly shows that TTD mitigates the curse of dimensionality.

2) Non uniqueness of TTD: It shall be noted that TTD is not unique [9]. In fact, $\mathcal{X}$ can be written in a TTD format using different cores than those in eq. (3) as follows:

$$\mathcal{X} = A_1 \times_2^1 \mathcal{A}_2 \times_3^1 \mathcal{A}_3 \times_4^1 \cdots \times_{Q-1}^1 \mathcal{A}_{Q-1} \times_Q^1 A_Q, \quad (4)$$

where,

$$A_1 = G_1 M_1^{-1}$$
$$A_Q = M_{Q-1}^{-1} G_Q$$
$$\mathcal{A}_q = M_{q-1} \times_2^1 \mathcal{G}_q \times_3^1 M_q$$

where M_q are nonsingular matrices of dimension $R_q \times R_q$. In order to treat all the TT-cores of a tensor in the same manner, we add a third dimension to the matrices in the TTD that will be equal to 1. Hence, all TT-cores will be denoted by a calligraphic notation $\mathcal{G}_q$ for $q \in \{1, \ldots, Q\}$.
In the next section, we will review the problem formulation of the extension of the standard SVMs to the tensorial case a as well as the use of kernel methods to deal with non linear classification problems.

D. Support Vector Machines in the tensorial case

Let us consider a binary classification problem where the dataset is composed of M Q-order tensors $\mathcal{X}_m \in \mathbb{R}^{I_1 \times \cdots \times I_Q}$ labeled with $y_m \in \{-1, 1\}$. This classification problem is said to be linearly separable if its decision function has the following form:

$$f(\mathcal{X}) = sgn\left(\langle \mathcal{W}, \mathcal{X} \rangle + b\right), \quad (5)$$

where $\mathcal{W} \in \mathbb{R}^{I_1 \times \ldots \times I_Q}$ is a Q-order tensor of weights and b is the bias and where "sgn" denotes the sign function.
In order to estimate $\mathcal{W}$ and b, a generalization of the SVM formulation problem was introduced in several works such as in [1] as follows:

$$\begin{cases} \min\limits_{\mathcal{W}, b, \xi} \dfrac{1}{2} ||\mathcal{W}||_F^2 + C \sum\limits_{m=1}^{M} \xi_m, \\ \text{subject to } y_m(\langle \mathcal{W}, \mathcal{X}_m \rangle + b) \ge 1 - \xi_m, \\ \xi_m \ge 0, 1 \le m \le M, \end{cases} \quad (6)$$

where ξ_m is the error of the m-th training example of the dataset and C is the trade-off between misclassification error and the classification margin.
When the dataset is non linearly separable, the previous formulation of SVM is no longer adapted, hence the idea of kernel methods is to represent the dataset in a high-dimensional pre-Hilbert space usually named the *feature space* in which the dataset is linearly separable. Following the scheme of the kernel trick for conventional SVMs, the projection of data in the feature space is done via an implicit feature map:

$$\Phi : \mathcal{X}_m \to \Phi(\mathcal{X}_m) \in \mathbb{R}^{H_1 \times \cdots \times H_Q}.$$

The constraints in eq. (6) expressed in the feature space can be written as [1]:

$$y_m(\langle \mathcal{W}, \Phi(\mathcal{X}_m) \rangle + b) \ge 1 - \xi_m. \quad (7)$$

After solving the dual problem of eq. (6) in the feature space taking account the constraints in eq. (7), the decision function eq. (5) can be expressed as follows [1] :

$$f(\mathcal{X}) = \text{sgn}\left(\sum_{m=1}^{M} \alpha_m y_m \langle \Phi(\mathcal{X}_m), \Phi(\mathcal{X}) \rangle + b\right), \quad (8)$$

where α_m is the m-th Lagrangian variable.
In general, the feature map Φ is unknown, only the inner product of the projection of the couple $(\Phi(\mathcal{X}_m), \Phi(\mathcal{X}_n))$ is required which is given by $k(\mathcal{X}_m, \mathcal{X}_n)$ where k is a symmetric positive definite function k named kernel function. Therefore, the decision function eq. (8) can be expressed using the kernel function as:

$$f(\mathcal{X}) = \operatorname{sgn}\left(\sum_{m=1}^{M} \alpha_m y_m k(\mathcal{X}_m, \mathcal{X}) + b\right). \tag{9}$$

In the next section, we demonstrate the TT-based feature mapping approach.

II. Proposed method

Our approach consists on defining a tensorial kernel using TTD based on sub-kernels defined on TT-cores. One can think about using traditional kernels on the vectorized version of TT-cores. However, as TTD is not unique (cf. section I-C2), this can badly affects the performance of the classification task. For example, comparing two tensors via their non-unique decomposition will lead to compare cores that are not similar, *e.g* in eq. (4) where $\mathcal{G}_q$ and $\mathcal{A}_q$ are quite different. In order to mitigate this issue, we consider learning on the subspaces they generate as $\mathcal{G}_q$ and $\mathcal{A}_q$ span the same subspaces and is therefore more robust.
In the following, we demonstrate the proposed TT-based feature mapping approach. We consider in our approach mapping the subspace $span(\mathcal{G}_q)$ of each TT-core $\mathcal{G}_q$ in the tensor feature space using a mapping Φ_q. This mapping is also unknown and similarly to Φ, is mainly used for technical consideration. It will be removed via a kernel trick, using only kernels in the next equations. Next, we represent our approach to compute the inner product between TTDs in the feature space using inner products of projectors of TT-cores.
Let $\mathcal{X}, \mathcal{X}' \in \mathbb{R}^{I_1 \times \cdots \times I_Q}$ with $\{\mathcal{G}_q\}_{q \in \{1,\ldots,Q\}}$ and $\{\mathcal{G}'_q\}_{q \in \{1,\ldots,Q\}}$ being respectively the sets of TT-cores of $\mathcal{X}$ and $\mathcal{X}'$. Assume $\Phi(\mathcal{X}), \Phi(\mathcal{X}') \in \mathbb{R}^{H_1 \times \cdots \times H_Q}$. Their inner product can be computed as :

$$\langle \Phi(\mathcal{X}), \Phi(\mathcal{X}') \rangle = \prod_{q=1}^{Q} \langle \Phi_q(\operatorname{span}(\mathcal{G}_q)), \Phi_q(\operatorname{span}(\mathcal{G}'_q)) \rangle, \tag{10}$$

From the kernel trick we have,

$$\langle \Phi_q(\operatorname{span}(\mathcal{G}_q)), \Phi_q(\operatorname{span}(\mathcal{G}'_q)) \rangle = k_q(\operatorname{span}(\mathcal{G}_q), \operatorname{span}(\mathcal{G}'_q)), \tag{11}$$

where k_q can be any kernel function used for a standard SVM such as a Gaussian kernel, polynomial kernel, linear kernel, *etc*.
Combining eq. (10) and eq. (11), we obtain the corresponding TT-based kernel function:

$$k(\mathcal{X}, \mathcal{X}') = \prod_{q=1}^{Q} k_q(\operatorname{span}(\mathcal{G}_q), \operatorname{span}(\mathcal{G}'_q)) \tag{12}$$

Dataset	DuSK	K-STTM	Our approach
UCF11	$0.54(10^{-2})$	$0.93(10^{-2})$	$\mathbf{0.96(10^{-2})}$
Extended	$0.33(10^{-2})$	$\mathbf{1.0(0)}$	$\mathbf{1.0(0)}$
Faces96	$0.33(10^{-2})$	$0.9(10^{-2})$	$\mathbf{1.0(0)}$

TABLE I: Average accuracy scores of different models on different datasets: mean(standard deviation)

TT-ranks	$[1,1,1,1,1]$	$[1,2,1,1,1]$	$[1,3,2,3,1]$	$[1,3,3,3,1]$
UCF11	$0.96(10^{-2})$	$0.93(10^{-2})$	$0.87(10^{-2})$	$0.81(10^{-2})$
Extended	1.0(0)	1.0(0)	1.0(0)	$0.96(10^{-2})$

TABLE II: Accuracy scores of our approach on different datsets w.r.t different values of TT ranks.

A popular choice for k_q that gives rise to a positive definite kernel is given by:

$$k_q(\operatorname{span}(\mathcal{G}_q), \operatorname{span}(\mathcal{G}'_q)) = \exp\left(-\gamma \sin^2(\theta_q)\right), \tag{13}$$

where $\gamma > 0$ and θ_q is the principal angle between $\operatorname{span}(\mathcal{G}_q)$ and $\operatorname{span}(\mathcal{G}'_q)$. It should be noted that despite θ_q being the geodesic distance in the Grassman manifold between the two subspaces, the expression $\sin(\theta_q)$ is considered instead, making the kernel k_q definite positive (therefore, SVM methods can be used for classification) [10]. Readers can refer to [10] for explicit ways of computing the principal angles. In our case, it is possible to directly use the projectors. For that, we first compute the t-svd of $\mathcal{G}_q$ and $\mathcal{G}'_q$ as follows:

$$\mathcal{G}_q = \mathcal{U}_q * \mathcal{S}_q * \mathcal{V}_q^T,$$
$$\mathcal{G}'_q = \mathcal{U}'_q * \mathcal{S}'_q * \mathcal{V}'^T_q,$$

In this work, we choose to use a gausssian kernel for subkernels k_q. In this case, the final expression of the kernel proposed in our approach is given by:

$$k(\mathcal{X}, \mathcal{Y}) = \prod_{q=1}^{Q} \exp\left(-\gamma \left|\left|\mathcal{U}_q * \mathcal{U}_q^T - \mathcal{U}'_q * \mathcal{U}'^T_q\right|\right|_F^2\right). \tag{14}$$

III. Experiments

We evaluate our approach on different real-world datasets and compare it to three state of the art methods as a baseline. It shall be noted that the SVM approach presented in the previous section is defined for a binary classification problem. In the case of multiclass problem, a *one-vs-rest* approach is utilised.

A. Datasets

- **UCF11 dataset**: This dataset [11] contains 1600 video clips belonging to 11 human actions such as: *diving*, *trampoling jumping*, *walking*, *shooting*... We consider the

TT-ranks	$[1,3,3,1,1]$	$[1,3,2,1,1]$	$[1,3,1,1,1]$	$[1,2,3,2,1]$
KSTTM	$0.6(10^{-2})$	$0.9(10^{-2})$	$0.4(10^{-2})$	$0.5(10^{-2})$
Our approach	$\mathbf{1.0(0)}$	$\mathbf{1.0(0)}$	$\mathbf{1.0(0)})$	$\mathbf{1.0(0)}$

TABLE III: Accuracy scores of KSTTM and our approach on the Faces96 dataset w.r.t different TT-ranks.

Method	DuSK	KSTTM	our approach
UCF11	1879	**130**	148
Extended	279	**18**	34.9
Faces96	0.4	**0.32**	**0.32**

TABLE IV: Computationnal time of different methods on the three real-world datasets considered.

first 240 frames from each clip video to have a consistent size between all samples. The resolution of each RGB frame is 320×240. These clip videos can be interpreted as tensors of order 4 with dimensions $240 \times 240 \times 320 \times 3$. A total of 109 tensors with these dimensions are present in each of these 11 classes.

- **Extended Yale dataset B**: This dataset [12] contains 28 human subjects. For each subject, there are 576 images of size 480×640 taken under 9 poses. Each pose is taken under 64 different illuminations. Hence, each subject is represented by a tensor of size $9 \times 480 \times 640 \times 64$.
- The **Faces96** dataset. This dataset contains 2261 images in JPG format of 119 persons. This images are of size $196 \times 196 \times 3$ and taken under different positions from the camera. In our experiments, we consider the first 16 positions from each subject so that each subject is represented by a tensor of size $16 \times 196 \times 196 \times 3$.

B. Classification performance

- Table I shows the classification results of different models. Here, a grid search has been realized on possible ranks, from [1,1,1,1] to [3,3,3,3]. Only the best scores are presented here. This scores are obtained by training on 50% of data and testing on the rest. This procedure is repeated 5 times and the mean scores with standard deviation are reported in Table I. We notice that the DuSK method has low accuracy scores due to its failure to manage the non uniqueness of CPD and finding a good rank. KSTTM achieves comparable results with our approach whereas our approach achieves the best performance.
- In order to see the influence of the TT ranks on the accuracy scores of our approach, we test different values in Table II. We notice that our approach achieves high accuracy scores on different values of TT-ranks. Hence, we can use small TT-ranks for reduction of the calculation costs. However, we remark that KSTTM is very sensitive to the choice of TT-ranks. This is clearly observed in Table III. In fact, for various values of TT-ranks, KSTTM does not achieve good accuracy scores.
- Table IV shows time computation of different models on different datasets. DuSK method is very costly because of the ALS (alternating least squares) algorithm. KSTTM reaches good results in terms of time computation. Our approach achieves reasonnable results in terms of complexity while being robust to the choice of TT-ranks.

IV. Conclusion

In the context of supervised learning of higher-order tensors, we have proposed in this work a new extension of SVMs to high-order tensors that deals with non linear classification problems. Based on the Tensor Train Decomposition (TTD), we have defined a new way to derive a kernel function on the tensor space and have shown that we can use different kernel functions on each TT-core. In order to overcome the non-unicity of TTD, our kernel function is defined using the subspaces generated by TT-cores. This is realised using tools of t-Algebra for 3-rd order tensors. Finally, we have shown that our approach achieves better performance on different real-world datasets considered.

References

[1] L.He, X.Kong, P.S.Yu, A.B.Ragin, Z.Hao, and X.Yang, "Dusk: A dual structure-preserving kernel for supervised tensor learning with applications to neuroimages," SIAM International Conference on Data Mining, 2014.

[2] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," SIAM Review, vol. 51, no. 3, pp. 455–500, September 2009.

[3] M.Signoretto, L. Lieven, and J.A.K.Suykens, "A kernel-based framework to tensorial data analysis," Neural networks : the official journal of the International Neural Network Society, vol. 24 8, pp. 861–74, 2011.

[4] C.Chen, K.Batselier, W.Yu, and N.Wong, "Kernelized support tensor train machines," Pattern Recognition, vol. 122, p. 108337, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320321005173

[5] R.Orús, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states," Annals of Physics, vol. 349, pp. 117 – 158, 2014. [Online]. Available: hp://www.sciencedirect.com/science/article/pii/S0003491614001596

[6] I.V.Oseledets and E.E.Tyrtyshnikov, "Breaking the curse of dimensionality, or how to use svd in many dimensions," SIAM Journal on Scientific Computing, vol. 31, no. 5, pp. 3744–3759, 2009. [Online]. Available: https://doi.org/10.1137/090748330

[7] M.E.Kilmer and C.D.Martin, "Factorization strategies for third-order tensors," Linear Algebra and its Applications, vol. 435, no. 3, pp. 641–658, 2011, special Issue: Dedication to Pete Stewart on the occasion of his 70th birthday. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0024379510004830

[8] M.E.Kilmer, K.Braman, H.Karen, H.Ning, and R.Hoover, "Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging," SIAM Journal on Matrix Analysis and Applications, vol. 34, 01 2013.

[9] Y.Zniyed, "Breaking the curse of dimensionality based on tensor train : models and algorithms," Ph.D. dissertation, Université Paris-Saclay, 2019, thèse de doctorat dirigée par R.Boyer, Traitement du signal et des images Université Paris-Saclay (ComUE) 2019. [Online]. Available: http://www.theses.fr/2019SACLS330

[10] S.Jayasumanaand, R.Harley, M.Salzmann, H.Li, , and M.Harandi, "Kernel methods on riemannian manifolds with gaussian rbf kernels," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 12, pp. 2464–2477, 2015.

[11] J.Liu, J.Luo, and M.Shah, "Recognizing realistic actions from videos "in the wild"," in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1996–2003.

[12] A.S.Georghiades, P.N.Belhumeur, and D.J.Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," IEEE Trans. Pattern Anal. Mach. Intelligence, vol. 23, no. 6, pp. 643–660, 2001.