Generalized Multilinear Model for Dimensionality Reduction of Binary Tensors

Jakub Mažgut[*]
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 3, 842 16 Bratislava, Slovakia
mazgut@fiit.stuba.sk

Abstract
Generalized multilinear model for dimensionality reduction of binary tensors (GMM-DR-BT) is a technique for computing low-rank approximations of multi-dimensional data objects, tensors. The model exposes a latent structure that represents dominant trends in the binary tensorial data while retaining as much information as possible. Recently, there exist several models for computing the low-rank approximations of tensors but to the best of our knowledge at present there is no principled framework for binary tensors. Although the binary tensors occur in many real world applications such as gait recognition, document analysis or graph mining.

In the GMM-DR-BT model formulation, to account for binary nature of the data, each tensor element is modeled by a Bernoulli noise distribution. To extract the dominant trends in the data, the natural parameters of the Bernoulli distributions are constrained by employing the Tucker model to lie in a sub-space spanned by a reduced set of basis (principal) tensors. Bernoulli distribution is a member of exponential family with helpful analytical properties that allow us to derive an iterative scheme for estimation of the basis tensors and other model parameters via maximum likelihood.

Furthermore, we extended the fully unsupervised GMM-DR-BT model to the semi-supervised setting by forcing the model to search for a natural parameter subspace that represents a user specified compromise between modelling the quality and the degree of class separation.

We evaluated and compared the proposed GMM-DR-BT technique with existing real-valued and nonnegative tensor decomposition methods in several experiments involving variety of high-dimensional datasets. The results suggest that the GMM-DR-BT model is better suited for modeling binary tensors than its real-valued and nonnegative counterparts.

Categories and Subject Descriptors
H.2.8 [**Database Management**]: Database Applications —*Data mining*; I.5.1 [**Pattern Recognition**]: Models; I.5.2 [**Pattern Recognition**]: Design Methodology —*Pattern analysis*

Keywords
unsupervised tensor decomposition, Tucker model, binary data, semi-supervised decomposition

1. Introduction
At present an increasing number of data processing tasks involve manipulation of multi-dimensional objects, known also as tensors, where the elements are to be addressed by more than two indices. In other words, the tensors can intuitively be imagined as multidimensional arrays of numbers in programming languages. In many practical problems such as gait [16], hand postures [18] or face recognition [12], hyperspectral image processing [21] and text documents analysis [4], the data tensors are specified in a high-dimensional space where the number of dimensions greatly exceeds the number of samples. Furthermore, these data have usually a large number of redundancy and lie in a subspace of the input space [10]. Applying pattern recognition or machine learning methods directly to such data spaces can result in high computational and memory requirements, as well as poor generalization. To address this "curse of dimensionality" a wide range of decomposition methods have been introduced to compress the data while capturing the 'dominant' trends. Making the learning machines operate on this compressed data space may not only boost their generalization performance but can also increase their interpretability crucially.

Traditional dimensionality reduction and feature extraction methods such as principal component analysis (PCA) [19] were designed to handle data objects in the form of vectors. For a tensorial data decomposition, the data items need to be first vectorized before these methods

[*]Recommended by thesis supervisor: Dr. Peter Tiňo. Defended at Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava on August 30, 2012.

Mažgut, J. Generalized Multilinear Model for Dimensionality Reduction of Binary Tensors. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 4, No. 3 (2012) 35-46

can be applied. Besides the higher computational and memory requirements, the vectorization of data tensors breaks the higher order dependencies presented in the natural data structure that can potentially lead to more compact and useful representations [16]. New methods capable of processing multi-dimensional tensors in their natural structure have been introduced in [4, 16, 17] (for real-valued tensors), [29] (for nonnegative tensors) and [2] (for symmetric tensors). Such techniques, however, are not suitable for processing binary tensors. To the best of our knowledge at present there is no principled systematic framework for decomposition of binary tensors. Yet, binary tensors arise in many real world applications such as gait recognition [16], document analysis [4] or graph objects represented by adjacency tensors. Thus, in the tensor decomposition domain there is still a need for a method that explicitly takes into account the binary nature of such tensorial data.

In this work, we present a novel decomposition method, generalized multilinear model for dimensionality reduction of binary tensors (GMM-DR-BT), which is able to handle binary tensorial data in their natural multidimensional form and which extracts hidden underlying structure of analyzed data in an unsupervised manner. Furthermore, we extended the model to semi-supervised setting by forcing the model to search for a natural parameter subspace which represents a user specified compromise between the modelling quality and the degree of class separation.

2. Notation and Basic Tensor Algebra

Before we present the basic concepts of the domain, let us introduce our notation and necessary basic tensor algebra. The vectors are denoted by lowercase boldface letters (e.g. $\mathbf{u}$), matrices by italic uppercase (e.g. U), and tensors by calligraphic letters (e.g. $\mathcal{A}$). Elements of an N-th order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ are addressed by N indices i_n ranging from 1 to I_n, $n = 1, 2, ..., N$. For convenience, we will often write a particular index setting $(i_1, i_2, ..., i_N) \in \Upsilon = \{1, 2, ..., I_1\} \times \{1, 2, ..., I_2\} \times ... \times \{1, 2, ..., I_N\}$ for a tensor element using vector notation $\mathbf{i} = (i_1, i_2, ..., i_N)$, so that instead of writing $\mathcal{A}_{i_1,i_2,...,i_N}$ we write $\mathcal{A}_{\mathbf{i}}$.

A scalar product of two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is defined as $\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{\mathbf{i} \in \Upsilon} \mathcal{A}_{\mathbf{i}} \cdot \mathcal{B}_{\mathbf{i}}$ and a Frobenius norm of tensor $\mathcal{A}$ is defined as

$$||\mathcal{A}||_F = \langle \mathcal{A}, \mathcal{A} \rangle = \sqrt{\sum_{\mathbf{i} \in \Upsilon} \mathcal{A}_{\mathbf{i}} \cdot \mathcal{A}_{\mathbf{i}}}. \tag{1}$$

An arbitrary tensor can be multiplied by a matrix (2nd-order tensor) using n-mode products. The n-mode product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ by a matrix $U \in \mathbb{R}^{J \times I_n}$ is a tensor $(\mathcal{A} \times_n U)$ given by

$$(\mathcal{A} \times_n U)_{i_1,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} \mathcal{A}_{i_1,\ldots,i_{n-1},i_n,i_{n+1},\ldots,i_N} \cdot U_{j,i_n}, \tag{2}$$

for some $j \in \{1, 2, ..., J\}$.

Another important and widely used procedure is an unfolding of the tensor. Unfolding, also known as the matricization, is a process of reordering the elements of an Nth-order tensor into a matrix along one of the tensor modes. Unfolding of tensor $\mathcal{A}$ along the n-mode is denoted by $\mathcal{A}_{(n)}$. For detailed information with illustrations of various third-order tensor unfolding can be found in [28].

3. Tensor Decomposition Methods

As we mentioned before, the number of data processing tasks that involve manipulation of massive tensorial data has increased enormously over the last few years. This leads to a strong demand for dimensionality reduction or feature extraction techniques that are able to process the tensorial data in their natural multi-dimensional structure. In this work we will focus on tensor decomposition methods based on a reduced-rank representation. Such methods extract the dominant information from the large-scale data by finding a lower dimensional (more compact) representation of the original data with no or little loss of information. Such techniques and representations are used for several purposes from noise removal, data compression, model reduction, and blind source separation to visualization.

The main idea behind the tensor decomposition methods is analogous to the matrix decomposition methods where the given matrix is decomposed as a liner combination of rank-1 matrices. In the tensor domain, the given tensor is decomposed (factorized) as a linear combination of rank-1 tensors. An N-th rank-1 tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ can be obtained as an outer product of N non-zero vectors $\mathbf{u}^{(n)} \in \mathbb{R}^{I_n}$, $n = 1, 2, ..., N$:

$$\mathcal{A} = \mathbf{u}^{(1)} \circ \mathbf{u}^{(2)} \circ ... \circ \mathbf{u}^{(N)}.$$

In the tensor decomposition literature, there are two main concepts how to decompose the given tensor. The concepts are represented by two different models, namely Tucker [26] and PARAFAC [11]. Both models have their origins and have been firstly investigated in the areas of psychometics (e.g. [26]) and chemometrics (e.g. [3]). Nowadays, the models are employed and extended for numerous disciplines from neurosience, social network analysis and web-mining, computer vision and etc.. More details about the actual survey of state-of-the-art applications and extensions can be found in [1].

3.1 PARAFAC Model

The parallel factor analysis model or simply the PARAFAC model (e.g. [1]), was proposed by Harshman in the seventies [11]. At the same time, Carroll and Chang proposed in [5] independently from Harshman a canonical decomposition (CANDECOMP) model that was considered equivalent to the PARAFAC model. The basic principle of the model is to decompose any given tensor as a sum of a finite number of rank-1 tensors. The rank-1 tensors are called components in the PARAFAC decomposition. In the context of different rank definitions [14], the PARAFAC model tries to find the best rank-R representation.

If the number of components R is equal to the true rank of the analyzed Nth-order tensor $\mathcal{A}$, the decomposition is called *rank decomposition* and is defined as

$$\mathcal{A} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \cdots \circ \mathbf{u}_r^{(N)}, \tag{3}$$

where $R = rank(\mathcal{A})$ and the r-th rank-1 tensor is equal to an outer product of N vectors $(\mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ ... \circ \mathbf{u}_r^{(N)})$. Besides the exact *rank decomposition*, PARAFAC model can also be used to find a lower rank approximation of the given tensor $\mathcal{A}$. If the $R < rank(\mathcal{A})$ then we get an approximation

$$\mathcal{A} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \cdots \circ \mathbf{u}_r^{(N)} + \mathcal{E}, \quad (4)$$

where $\mathcal{E}$ denotes a residue between the real tensor and its approximation.

3.2 Tucker Model

The Tucker decomposition model, also called the three-mode factor analysis, was first introduced by Tucker in 1964 [26]. Tucker proposed only a model for third-order tensors and later in 2000 a group around De Lathauwer [7] proposed a generalization of the model to arbitrary Nth-order tensors. Besides that, the authors showed that the Tucker decomposition is a convincing generalization of the matrix singular value decomposition.

Consider now an orthonormal basis $\{\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, ..., \mathbf{u}_{I_n}^{(n)}\}$ for the n-mode space $\mathbb{R}^{I_n}$. The (column) vectors $\mathbf{u}_k^{(n)}$ can be stored as columns of the basis matrix $U^{(n)} = (\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, ..., \mathbf{u}_{I_n}^{(n)})$. Any tensor $\mathcal{A}$ can be decomposed into the product

$$\mathcal{A} = \mathcal{Q} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 ... \times_N U^{(N)},$$

with expansion coefficients stored in the Nth-order tensor $\mathcal{Q} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$. The expansion of $\mathcal{A}$ can also be written as

$$\mathcal{A} = \sum_{\mathbf{i} \in \Upsilon} \mathcal{Q}_{\mathbf{i}} \cdot (\mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ ... \circ \mathbf{u}_{i_N}^{(N)}). \quad (5)$$

In other words, tensor $\mathcal{A}$ is expressed as a linear combination of $\prod_{n=1}^{N} I_n$ rank-1 basis tensors $(\mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ ... \circ \mathbf{u}_{i_N}^{(N)})$. In addition, from orthonormality of the basis sets, the tensor $\mathcal{Q}$ of expansion coefficients can be obtained as

$$\mathcal{Q} = \mathcal{A} \times_1 (U^{(1)})^T \times_2 (U^{(2)})^T \times_3 ... \times_N (U^{(N)})^T.$$

Note that the orthonormality of basis $\{\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, ..., \mathbf{u}_{I_n}^{(n)}\}$ for the n-mode space $\mathbb{R}^{I_n}$ can be relaxed. It can be easily shown that as long as for each mode $n = 1, 2, ..., N$, the vectors $\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, ..., \mathbf{u}_{I_n}^{(n)}$ are linearly independent, the basis tensors $(\mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ ... \circ \mathbf{u}_{i_N}^{(N)})$, $\mathbf{i} \in \Upsilon$ will be linearly independent as well.

In many practical problems, one can assume that only a reduced set of basis tensors in the expansion (5) is enough to sufficiently approximate the original tensor:

$$\begin{aligned} \mathcal{A} &= \sum_{\mathbf{i} \in K} \mathcal{Q}_{\mathbf{i}} \cdot (\mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ ... \circ \mathbf{u}_{i_N}^{(N)}) + \mathcal{E} \quad (6) \\ &= \mathcal{Q} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 ... \times_N U^{(N)} + \mathcal{E}, \quad (7) \end{aligned}$$

where $K \subset \Upsilon$, and $\mathcal{E}$ denotes a residue between the real tensor and its model approximation. Basis matrices $U^{(n)}$ are called in context of the Tucker decomposition literature also as factors or loadings [20] and the tensor with expansion coefficients as a core tensor of the reduced dimension.

3.3 PARAFAC versus Tucker Model

The main difference between the Tucker and the PARAFAC model is that the Tucker model allows interactions between different components while the PARAFAC does not. The levels of interactions are defined by values of the expansion tensor $\mathcal{Q}$. In fact, the PARAFAC model can be considered a special case of the Tucker model with nonzero elements only on a super-diagonal of the expansion tensor.

Another difference between the models is the uniqueness of decomposition. The PARAFAC model has a unique solution up to permutation and scale. On the other side, the Tucker based decompositions are not unique [14]. We can rotate the expansion coefficient tensor $\mathcal{Q}$ without affecting the fit so long as we apply the inverse rotation to the basis matrices. In order to achieve uniqueness, several constrains such as orthogonality, sparsity, and nonnegativity are imposed on the factor matrices and the expansion coefficient tensor. Detailed information and a brief survey can be found in [14].

There exist many extensions and applications of both the models in various domains. So it is not an easy matter to choose the right one for the desired analysis. In our work, we focus on compact representations (decompositions) of the original data that preserve as much information as possible. According to the results of experiments published in [27], the Tucker decompositions clearly preserve more information than the PARAFAC decompositions. The authors used both the models to decompose, compress and consequently reconstruct data tensors. Then, the reconstructed tensors were compared with the originals using mean square error criterion. While preserving the same compression ratio, the Tucker model had clearly outperformed the PARAFAC model. These results can be explained by higher flexibility of the Tucker model while it allows interactions between the factors. A detailed survey of Tucker based models, their computational methods and applications can be found in (e.g. [1, 14]).

3.4 Extensions of the Tucker Model

The existing Tucker decomposition techniques can be divided into two main groups, namely real-valued and nonnegative tensor decomposition models. There exist several models in each group but in this section we focus only on two essential models to point out the main trends in both groups. An extensive survey of existing models can be found in (e.g. [1, 14, 6]).

3.4.1 Multilinear Principal Component Analysis

The basic PARAFAC and Tucker models were designed to process one data tensor. However, many actual pattern recognition tasks require simultaneously processing of a set of tensors in order to perform reduction to extract the dominant features across all the data samples (tensors). Let's assume that we have a set of N-mode tensors $\mathcal{A}_m$, where $m = 1, \ldots, M$ and that only a reduced number of basis tensors in the expansion (5) is sufficient to approximate all tensors in a given dataset. Then, each data tensor $\mathcal{A}_m$ can be expressed as their linear combination

$$\mathcal{A}_m \approx \sum_{\mathbf{i} \in K} \mathcal{Q}_{m,\mathbf{i}} \cdot (\mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ ... \circ \mathbf{u}_{i_N}^{(N)}), \quad (8)$$

where $K \subset \Upsilon$. In other words, tensors in the given dataset can be found 'close' to the $|K|$-dimensional hyperplane in the tensor space spanned by the basis rank-1 tensors $(\mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ ... \circ \mathbf{u}_{i_N}^{(N)})$, $\mathbf{i} \in K$. So then each tensor $\mathcal{A}_m$ can be represented through expansion coefficients $\mathcal{Q}_{m,\mathbf{i}}$, $\mathbf{i} \in K$.

Based on the principle of simultaneous decomposition of a set of tensors, Lu et al. proposed in [16] a tensor decomposition technique, multilinear principal component analysis (MPCA) for real-valued tensor. The MPCA technique can be considered a generalization of the classical PCA method to arbitrary N-th order tensors. The main objective of MPCA is to find a basis (projection) matrix $U^{(n)}$ for each n-mode space $\mathbb{R}^{I_n}$, where $n = 1, ..., N$, which maximizes the total variance of expansion coefficient tensors (projected data tensors):

$$\Psi_{\mathcal{Q}} = \sum_{m=1}^{M} ||\mathcal{Q}_m - \bar{\mathcal{Q}}||_F, \tag{9}$$

where $\bar{\mathcal{Q}}$ denotes a mean of the expansion coefficient tensors $\mathcal{Q}_m$. This is an analogy with the objective of PCA. The PCA model finds projection that maximizes the variance of the projected data vectors instead of tensors.

According to the authors of [16], there is no known optimal solution which allows for the simultaneous optimization of the N projection matrices $U^{(n)}$. They used a local optimization procedure to compute one projection matrix at time while the others are kept fixed. Briefly, the projection matrix $U^{(n)}$ consists of the R_n eigenvectors with the largest eigenvalues of a matrix

$$\Phi^{(n)} = \sum_{m=1}^{M} (\mathcal{A}_{m(n)} - \bar{A}_{(n)}) \cdot U_{\Phi^{(n)}} \cdot U_{\Phi^{(n)}}^T \cdot (\mathcal{A}_{m(n)} - \bar{A}_{(n)})^T,$$

where $U_{\Phi^{(n)}} = \big(U^{(n+1)} \otimes U^{(n+2)} \otimes ... \otimes U^{(N)} \otimes U^{(1)} \otimes U^{(2)} ... \otimes U^{(n-1)}\big)$, $\otimes$ denotes a Kronecker product [8] and $A_{m(n)}$ represents unfolding of tensor $\mathcal{A}_m$ along the n-mode, so that $A_{m(n)} \in \mathbb{R}^{I_n \times (I_1 \times ... \times I_{n-1} \times I_{n+1} \times I_N)}$ [7]. The projection matrices are updated sequentially and the whole process is repeated until the cost function (9) converges or the maximum number of iteration is reached. According to [16], altering the ordering of the projection matrix computation did not lead to any significant improvements in practical situations.

3.4.2 Nonnegative Tucker Decomposition

The main motivation behind the development of nonnegative decomposition methods was the problem of performing dimensionality reductions for inherently nonnegative data, e.g. environmental data, chemical concentrations or color intensities [24]. The proposed solution was to represent such data as a linear combination of basis vectors and mixing coefficients, both with nonnegative elements. These nonnegativity constrains which allow only additive not subtractive combinations respect the inherent nonnegativity of analyzed data and avoid physically absurd and uninterpretable factors [25].

The first nonnegative Tucker decomposition algorithms, introduced in [13], use an alternative least square technique to optimize a various global cost functions. The cost functions measure discrepancies between the original data tensor $\mathcal{A}$ and the reduced rank approximation of the data tensor $\hat{\mathcal{A}}$. According to the Tucker decomposition scheme, defined by equation (6), the reduced rank approximation is given by

$$\mathcal{A} \approx \hat{\mathcal{A}} = \mathcal{Q} \times_1 U^{(1)} \times_2 U^{(2)} \cdots \times_N U^{(N)}, \tag{10}$$

where for the nonnegative decomposition, the tensor with expansion coefficients $\mathcal{Q} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}$ and the n-mode basis matrices $U^{(n)}$ are constrained to have only nonnegative elements. Note that, the initial restriction of the Tucker model about orthonormality of column vectors of the basis matrices is relaxed. It can be easily shown that as long as for each mode the vectors are linearly independent, the basis tensors will be linearly independent as well.

The first listed model for the nonnegative Tucker decomposition is based on a minimalization of the widely used Frobenius norm (1), between the data tensor $\mathcal{A}$ and its approximation by the model $\hat{\mathcal{A}}$:

$$D_F(\mathcal{A}||\hat{\mathcal{A}}) = ||\mathcal{A} - \hat{\mathcal{A}}||_F. \tag{11}$$

The model was proposed by Kim et al. in [13] and while it incorporates optimization of the Froberius norm, it performs a nonnegative Tucker decomposition under an assumption of independent and identically distributed Gaussian noise. Other existing models incorporate different cost functions (e.g. KL-divergence, α-divergence) that are suitable for different types of noise.

The authors of [13] derived a multiplicative algorithm for updating the model parameters. The n-mode basis matrices $U^{(n)}$ of the Tucker model are sequentially updated by

$$U^{(n)} \leftarrow U^{(n)} \circledast \frac{A_{(n)}(Q_U^{(n)})^T}{U^{(n)} Q_U^{(n)} (Q_U^{(n)})^T} \tag{12}$$

where

$$Q_U^{(n)} = [\mathcal{Q} \times_1 U^{(1)} \times_2 \cdots \times_{n-1} U^{(n-1)} \times_{n+1} U^{(n+1)} \cdots \times_N U^{(N)}]_{(n)}. \tag{13}$$

Binary operator / represents element-wise division and operator $\circledast$ represents a Hadamard product (element-wise multiplication). While updating the matrices $U^{(n)}$, the rest of parameters are fixed to their current values.

Besides the basis matrices, the core tensor $\mathcal{Q}$ also needs to be updated in each cycle of the iteration while the basis matrices are fixed to their current values. The updating algorithm for the core tensor has a form

$$\mathcal{Q} \leftarrow \mathcal{Q} \circledast \frac{\mathcal{A} \times_1 U^{(1)} ... \times_N (U^{(N)})^T}{\mathcal{Q} \times_1 (U^{(1)})^T U^{(1)} ... \times_N U^{(N)})^T U^{(N)}}. \tag{14}$$

According to the authors, the algorithm was directly derived from a previously nonnegative matrix factorization updating algorithm, so the monotonic convergence analysis which was done for NMF method in [15] can be directly applied also to this extended version of the model for tensors.

4. GMM-DR-BT Model

In the previous sections we summarized the two main groups of tensor decomposition methods, models for real-valued decomposition and models for non-negative decomposition. Such techniques, however, are not suitable for processing binary tensors. Thus, in the tensor decomposition domain there is still need for a method that explicitly takes into account the binary nature of such tensorial data.

To close this gap we propose the generalized multilinear model for dimensionality reduction of binary tensors (GMM-DR-BT). The model is based on an extension of the multilinear Tucker concept (6) for real-valued tensorial data. The extension is analogous to the generalization of linear models for the exponential family of distributions. We use the Tucker model as a multilinear 'predictor' and a logistic function as the link function to link the real-valued multilinear 'predictions' with response variables, in our case binary elements of the data tensors. A probabilistic framework is used to formally define the model and to derive rules for the parameter estimation in analogous manner as did Schein et al. for binary vectorial data in [23].

4.1 The Model

Consider binary Nth-order tensor $\mathcal{A} \in \{0,1\}^{I_1 \times I_2 \times \ldots \times I_N}$. Assume we are given a set of M such tensors $\mathcal{D} = \{\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_M\}$. Each element $\mathcal{A}_{m,\mathbf{i}}$ of the tensor $\mathcal{A}_m$, $m = 1,2,...,M$, is independently generated from a Bernoulli distribution with the mean $\mathcal{P}_{m,\mathbf{i}}$ (all mean parameters for the data are collected in tensor $\mathcal{P} \in [0,1]^{M \times I_1 \times I_2 \times \ldots \times I_N}$ of order $N+1$). Assuming independence among the data tensors, the model likelihood reads

$$\begin{aligned} L(\mathcal{P}) &= \prod_{m=1}^{M} \prod_{\mathbf{i} \in \Upsilon} P(\mathcal{A}_{m,\mathbf{i}} | \mathcal{P}_{m,\mathbf{i}}) \\ &= \prod_{m=1}^{M} \prod_{\mathbf{i} \in \Upsilon} \mathcal{P}_{m,\mathbf{i}}^{\mathcal{A}_{m,\mathbf{i}}} \cdot (1 - \mathcal{P}_{m,\mathbf{i}})^{1-\mathcal{A}_{m,\mathbf{i}}}. \quad (15) \end{aligned}$$

A more detailed model description can be found in the thesis.

Our goal is to find a lower dimensional representation of the binary tensors in $\mathcal{D}$ while still capturing the data distribution well. The mean Bernoulli parameters are confined to the interval $[0,1]$. To solve our problem in an unbounded domain, we rewrite the Bernoulli distribution using the log-odds parameters $\theta_{m,\mathbf{i}} = \log[\mathcal{P}_{m,\mathbf{i}}/(1-\mathcal{P}_{m,\mathbf{i}})]$ and the logistic link function $\sigma(\theta_{m,\mathbf{i}}) = (1 + e^{-\theta_{m,\mathbf{i}}})^{-1} = \mathcal{P}_{m,\mathbf{i}}$. We thus obtain for each data tensor $\mathcal{A}_m$, $m = 1,2,...,M$:

$$P(\mathcal{A}_m | \theta_m) = \prod_{\mathbf{i} \in \Upsilon} \sigma(\theta_{m,\mathbf{i}})^{\mathcal{A}_{m,\mathbf{i}}} \cdot \sigma(-\theta_{m,\mathbf{i}})^{1-\mathcal{A}_{m,\mathbf{i}}}, \quad (16)$$

so the model log-likelihood take the form

$$\begin{aligned} \mathcal{L}(\Theta) &= \log \prod_{m=1}^{M} P(\mathcal{A}_m | \theta_m) \\ &= \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon} \mathcal{A}_{m,\mathbf{i}} \log \sigma(\theta_{m,\mathbf{i}}) + (1 - \mathcal{A}_{m,\mathbf{i}}) \log \sigma(-\theta_{m,\mathbf{i}}), \end{aligned} \quad (17)$$

where we collect all the natural parameters $\theta_{m,\mathbf{i}}$ in a tensor $\Theta \in \mathbb{R}^{M \times I_1 \times I_2 \times \ldots \times I_N}$. Now, $\theta_{m,\mathbf{i}} \in \mathbb{R}$, which allows us to incorporate the real-valued multilinear Tucker model. By using the Tucker model we constrain all the Nth-order parameter tensors θ_m to lie in a subspace spanned by a reduced set of rank-1 basis tensors $(\mathbf{u}_{r_1}^{(1)} \circ \mathbf{u}_{r_2}^{(2)} \circ \ldots \circ \mathbf{u}_{r_N}^{(N)})$, where $r_n \in \{1,2,...,R_n\}$, and $R_n \leq I_n$, $i = 1,2...,N$ The indices $\mathbf{r} = (r_1, r_2, ..., r_N)$ take values from the set $\rho = \{1,2,...,R_1\} \times \{1,2,...,R_2\} \times \ldots \times \{1,2,...,R_N\}$.

Furthermore, we allow for an Nth-order bias tensor $\Delta \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, so that the parameter tensors θ_m are constrained onto an affine space. Using (6) we get

$$\theta_{m,\mathbf{i}} = \sum_{\mathbf{r} \in \rho} \mathcal{Q}_{m,\mathbf{r}} \cdot \prod_{n=1}^{N} u_{r_n,i_n}^{(n)} + \Delta_{\mathbf{i}}, \quad (18)$$

and the log-likelihood is evaluated as

$$\begin{aligned} \mathcal{L} = & \\ & \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon} \mathcal{A}_{m,\mathbf{i}} \log \sigma \left(\sum_{\mathbf{r} \in \rho} \mathcal{Q}_{m,\mathbf{r}} \cdot \prod_{n=1}^{N} u_{r_n,i_n}^{(n)} + \Delta_{\mathbf{i}} \right) + \\ & (1 - \mathcal{A}_{m,\mathbf{i}}) \log \sigma \left(- \sum_{\mathbf{r} \in \rho} \mathcal{Q}_{m,\mathbf{r}} \cdot \prod_{n=1}^{N} u_{r_n,i_n}^{(n)} - \Delta_{\mathbf{i}} \right) (19) \end{aligned}$$

4.2 Parameter Estimation

To get analytical parameter updates in the maximum likelihood framework, we use the trick of [23] and take advantage of the fact that while the log-likelihood (19) of the constrained model is not convex in the parameters, it is convex in any of these parameters, if the others are fixed. The trick is to derive the analytical updates from a lower bound on the log-likehood using

$$\log \sigma(\hat{\theta}) \geq -\log 2 + \frac{\hat{\theta}}{2} - \log\cosh\left(\frac{\theta}{2}\right) - (\hat{\theta}^2 - \theta^2)\, \frac{\tanh \frac{\theta}{2}}{4\theta}, \quad (20)$$

where θ stands for the current value of individual natural parameters $\theta_{m,\mathbf{i}}$ of Bernoulli noise models $P(\mathcal{A}_{m,\mathbf{i}} | \theta_{m,\mathbf{i}})$ and $\hat{\theta}$ stands for the future estimate of the parameters, given the current parameter values. This leads to an iterative scheme where the model parameters are fitted alternating between the least square updates for basis tensors $\mathbf{u}_{r_n}^{(n)}$, expansion coefficients $Q_{m,\mathbf{r}}$ and bias tensor Δ. While one set of parameters is updated, the others are held fixed. This procedure is repeated until the log-likelihood converges to a desired degree of precision. The updates rules lead to monotonic increase in the log-likelihood.

Derivation of the parameter updates is rather involved and (due to space limitations) we refer the interested reader to the thesis, where details of the derivations as well as the complete update formulas can be found. In the following text, we present just the final updating formulas.

4.2.1 Updates for n-mode Space Basis

Holding the bias tensor Δ and the expansion coefficients $\mathcal{Q}_{m,\mathbf{r}}$, $m = 1,2,...,M$, $\mathbf{r} \in \rho$ fixed, we obtain a update rule for the n-mode space basis $\{\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, ..., \mathbf{u}_{R_n}^{(n)}\}$. For each n-mode and its coordinate $j \in \{1,2,\ldots,I_n\}$, the basis vectors are updated by solving linear system:

$$\sum_{t=1}^{R_n} u_{t,j}^{(n)}\, \mathcal{K}_{q,t,j}^{(n)} = \mathcal{S}_{q,j}^{(n)}, \quad (21)$$

where

$$\begin{aligned} \mathcal{S}_{q,j}^{(n)} = \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon_{-n}} & \big(2\mathcal{A}_{m,[\mathbf{i},j|n]} - 1 \\ & - T_{m,[\mathbf{i},j|n]} \Delta_{[\mathbf{i},j|n]}\big)\, \mathcal{B}_{m,\mathbf{i},q}^{(n)}, \end{aligned} \quad (22)$$

$$\mathcal{K}^{(n)}_{q,t,j} = \sum_{m=1}^{M} \sum_{\mathbf{r} \in \rho_{-n}} \mathcal{Q}_{m,[\mathbf{r},t|n]} \times \sum_{\mathbf{i} \in \Upsilon_{-n}} T_{m,[\mathbf{i},j|n]} \, \mathcal{B}^{(n)}_{m,\mathbf{i},q} \prod_{s=1,s\neq n}^{N} u^{(s)}_{r_s,i_s}, \quad (23)$$

$$\mathcal{B}^{(n)}_{m,\mathbf{i},q} = \sum_{\mathbf{r} \in \rho_{-n}} \mathcal{Q}_{m,[\mathbf{r},q|n]} \cdot \prod_{s=1,s\neq n}^{N} u^{(s)}_{r_s,i_s}, \quad (24)$$

T denotes $(\tanh \frac{\theta_{m,\mathbf{i}}}{2})/\theta_{m,\mathbf{i}}$, and $q = 1, 2, \ldots, R_n$. Note that the updates for coefficients of basis vectors for different mode n and its coordinates $j \in \{1, 2, \ldots, I_n\}$ are conveniently decoupled.

4.2.2 Updates for Expansion Coefficients

When updating the expansion coefficients $\mathcal{Q}_{m,\mathbf{r}}$, the bias tensor Δ and the basis sets $\{\mathbf{u}^{(n)}_1, \mathbf{u}^{(n)}_2, ..., \mathbf{u}^{(n)}_{R_n}\}$ for all n modes $n = 1, 2, ..., N$ are kept fixed to their current values. The update rule for expansion coefficients $\mathcal{Q}_{m,\mathbf{r}}$ of the m-th input tensor $\mathcal{A}_m$ can be obtained by solving a set of linear equations

$$\mathcal{T}_{\mathbf{v},m} = \sum_{\mathbf{r} \in \rho} \mathcal{P}_{\mathbf{v},\mathbf{r},m} \; \mathcal{Q}_{m,\mathbf{r}}, \quad (25)$$

where

$$\mathcal{T}_{\mathbf{v},m} = \sum_{\mathbf{i} \in \Upsilon} (2\mathcal{A}_{m,\mathbf{i}} - 1 - T_{m,\mathbf{i}} \, \Delta_{\mathbf{i}}) \; C_{\mathbf{v},\mathbf{i}}, \quad (26)$$

$$\mathcal{P}_{\mathbf{v},\mathbf{r},m} = \sum_{\mathbf{i} \in \Upsilon} T_{m,\mathbf{i}} \; C_{\mathbf{v},\mathbf{i}} \; C_{\mathbf{r},\mathbf{i}}, \quad (27)$$

$C_{\mathbf{r},\mathbf{i}}$ denotes $\prod_{n=1}^{N} u^{(n)}_{r_n,i_n}$ and $\mathbf{v} \in \rho$ is a basis index. In terms of these equations, the expansion coefficients updates for different input tensors are conveniently decoupled.

4.2.3 Updates for the Bias Tensor

As before, holding the expansion coefficients $\mathcal{Q}_{m,\mathbf{r}}$, $m = 1, 2, ..., M$, $\mathbf{r} \in \rho$, and the basis sets $\{\mathbf{u}^{(n)}_1, \mathbf{u}^{(n)}_2, ..., \mathbf{u}^{(n)}_{R_n}\}$ for all n modes $n = 1, 2, ..., N$ fixed, we obtain a simple update rule for the bias tensor:

$$\Delta_{\mathbf{j}} = \frac{\sum_{m=1}^{M} 2\mathcal{A}_{m,\mathbf{j}} - 1 - T_{m,\mathbf{j}} \cdot \sum_{\mathbf{r} \in \rho} \mathcal{Q}_{m,\mathbf{r}} \; C_{\mathbf{r},\mathbf{j}}}{\sum_{m=1}^{M} T_{m,\mathbf{j}}}. \quad (28)$$

4.2.4 Decomposing Unseen Binary Tensors

Note that our model is not generative, however, it is straightforward to find expansion coefficients for an Nth-order tensor $\mathcal{A}' \in \{0,1\}^{I_1 \times I_2 \times \ldots \times I_N}$ not included in the training set $\mathcal{D}$. One simply needs to solve for expansion coefficients in the natural parameter space, given that the parameters are confined onto the affine subspace of the tensor parameter space found in the training phase. Recall that the affine subspace is determined by the bias tensor Δ and the basis sets $\{\mathbf{u}^{(n)}_1, \mathbf{u}^{(n)}_2, ..., \mathbf{u}^{(n)}_{R_n}\}$, one for each n mode, $n = 1, 2, ..., N$. These are kept fixed.

The likelihood (19) to be maximized with respect to the expansion coefficients stored in tensor $\mathcal{Q}$ reads

$$\mathcal{L}(\mathcal{Q}; \mathcal{A}') = \sum_{\mathbf{i},\ s.t.\ \mathcal{A}'_{\mathbf{i}}=1} \log \sigma \left(\sum_{\mathbf{r} \in \rho} \mathcal{Q}_{\mathbf{r}} \; C_{\mathbf{r},\mathbf{i}} + \Delta_{\mathbf{i}} \right) + \sum_{\mathbf{i},\ s.t.\ \mathcal{A}'_{\mathbf{i}}=0} \log \sigma \left(- \sum_{\mathbf{r} \in \rho} \mathcal{Q}_{\mathbf{r}} \; C_{\mathbf{r},\mathbf{i}} - \Delta_{\mathbf{i}} \right). \quad (29)$$

Any optimization technique can be used. The quantities $C_{\mathbf{r},\mathbf{i}}$ and $\Delta_{\mathbf{i}}$ are constants given by the trained model. The tensor $\mathcal{Q}$ can be initialized by first finding the closest data tensor from the training dataset $\mathcal{D}$ to $\mathcal{A}'$ in the Hamming distance sense,

$$m(\mathcal{A}') = \operatorname*{arg\,min}_{m=1,2,\ldots,M} \sum_{\mathbf{i} \in \Upsilon} |\mathcal{A}'_{\mathbf{i}} - \mathcal{A}_{m,\mathbf{i}}|,$$

and then setting the initial value of $\mathcal{Q}$ to the expansion coefficient tensor of $\mathcal{A}_{m(\mathcal{A}')}$.

When using gradient ascent,

$$\mathcal{Q}_{\mathbf{v}} \leftarrow \mathcal{Q}_{\mathbf{v}} + \eta \; \frac{\partial \; \mathcal{L}(\mathcal{Q}; \mathcal{A}')}{\partial \; \mathcal{Q}_{\mathbf{v}}},$$

the updates take the form

$$\mathcal{Q}_{\mathbf{v}} \leftarrow \mathcal{Q}_{\mathbf{v}} + \eta \sum_{\mathbf{i} \in \Upsilon} C_{\mathbf{v},\mathbf{i}} \left[\mathcal{A}'_{\mathbf{i}} - \sigma \left(\sum_{\mathbf{r} \in \rho} \mathcal{Q}_{\mathbf{r}} \; C_{\mathbf{r},\mathbf{i}} + \Delta_{\mathbf{i}} \right) \right], \quad (30)$$

where $\eta > 0$.

5. Experiments

In this section, we compare our proposed generalized multilinear model for dimensionality reduction of binary tensors (GMM-DR-BT) with several existing tensor decomposition methods. We evaluated how well their compact representations preserve the information. The examined models are used to compress and subsequently reconstruct the data to measure a discrepancy between original and reconstructed data. To achieve a fair comparison, we employ two real-valued and two nonnegative tensor decomposition models, namely tensor latent semantic indexing model [1] (TensorLSI) [4], multilinear principal component analysis model (MPCA) [16], nonnegative tucker decomposition model employing least square error function and similar model employing KL-divergence [13].

5.1 Outline of the Experiments

To get a proper comparison of our proposed model with other existing models, we tested the ability of compression on three different datasets which have data tensors with different orders, complexity of underlying structure and sparsity. On each dataset, all the models were used to find principal subspaces spanned by different number of basis tensors or vectors using a portion of data samples assigned for training.

After training the models, tensors that were not included in the training (hold-out set) were "compressed" by projecting them onto the principal subspace thus obtaining their low dimensional representations (projections).

[1]TensorLSI model is only capable to process 2nd-order tensors (matrices).

In order to evaluate the amount of preserved information, the compressed representations were reconstructed back into the original binary tensor space. Note that since the models we consider represent binary tensors through continuous values in $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, a straightforward deterministic reconstruction in the binary space is not appropriate. Therefore we used the area under the ROC curve (AUC) designed to compare different real-valued predictions of binary data to evaluate the amount of preserved information in compressed representations of the original data.

For a fair comparison, the decomposition methods are compared based on the AUC with respect to the number of free parameters. In general, the free parameters correspond to the basis vectors and the offset (bias tensor). For TensorLSI and MPCA the offset represents the mean tensor (used to center the data); for GMM-DR-BT it represents the bias tensor; the NTD models do not have bias vector/tensor and centering of the data is not appropriate considering the nonnegativity constraint. For this reason we exclude centering of the data and the bias tensor from the models in this type of an experiment. If we exclude the offset from the models, for GMM-DR-BT, MPCA and NTD models, the number of free parameters is equal to $\sum_{n=1}^{N} R_n \cdot I_n$. TensorLSI with R basis tensors has $R \cdot \sum_{n=1}^{N} \min\{R, I_n\}$ free parameters.

5.2 Synthetic Data

In order to evaluate the ability of the models to find compact data representations, we generated 5 datasets of 3rd-order binary tensors of size $(I_1, I_2, I_3) = (15, 15, 15)$. Each dataset has 4,500 tensors and was sampled from underlying subspaces of the natural parameter space spanned by 30 randomly generated linearly independent basis tensors. The detailed description of the generating process can be found in the thesis.

From each dataset we hold out one-third $(1,500)$ binary tensors as a test set and let the models find the latent subspace on the remaining $(3,000)$ tensors (training set). The performance of the examined models to compress and subsequently reconstruct the sets of synthetic tensors by calculating the mean and standard deviation of AUC values across all the 5 test sets of binary tensors is summarized in figure 1. As could be easily seen from the figure, GMM-DR-BT model outperforms all the real-valued and nonnegative counterpart models.

5.3 DNA Sequences

The DNA sequence dataset includes almost 62,000 DNA subsequences. In brief, each sequence is represented by a 2nd-order binary tensor (matrix) with 31 rows and 250 columns. Rows represent short subsequences, terms, that are widespread sequences of nucleotides that have or may have a biological significance. Columns represent positions in the DNA sequence and binary elements indicate the presence/absence of a term in the sequence at a given position. Detailed information about the dataset and the DNA sequence representation by a binary tensor can be found in the thesis.

From the dataset of 62,000 sequences we randomly sampled out 5 groups, each of 4,500 sequences. On each group, all examined decomposition models were used to find a latent subspace spanned by different number of basis tensors using 3,000 of the sequences (binary matrices). The hold-out sets of sequences were projected onto the latent space and then reconstructed back into the original tensor space to measure the discrepancy between projected and original data by the AUC. The procedure is similar ty the one used in the previous experiment with synthetic data.

Reconstruction results in terms of AUC for the different dimensionality of the latent spaces are shown in figure 2. Our proposed GMM-DR-BT model clearly outperforms other decomposition models except the case of the smallest latent subspace where TensorLSI model achieved the best accuracy in term of AUC.

Furthermore, the subsequences from the dataset originate from two different functional regions of genomic sequences. In the thesis we illustrate how well can our GMM-DR-BT method reveal biologically meaningful dominant trends and we closely analyzed the topographic organization of DNA sequences in the latent space. Several well known biological characteristics were pointed out by our GMM-DR-BT analysis.

5.4 USF Gait Challenge Dataset

The dataset of gait silhouette video sequences, USF HumanID "Gait Challenge", was created by Sarkar et al. [22] and is considered as a benchmark set for gait recognition systems. Each data sample represents a person walking in elliptical paths in front of a camera. In our experiment we used the dataset version 1.7 which was binarized by [16] to get the binary silhouettes. The dataset consists of 731 binary gait samples of size (32x22x10).

For our experiment we randomly divided the samples into 5 disjoin groups and performed a 5-fold cross validation to get the amount of preserved information in the reconstructed tensors. As in the previous experiments, we used the AUC for the evaluation. Results are summarized in figure 3 and once again our GMM-DR-BT model clearly outperforms the other tensor models.

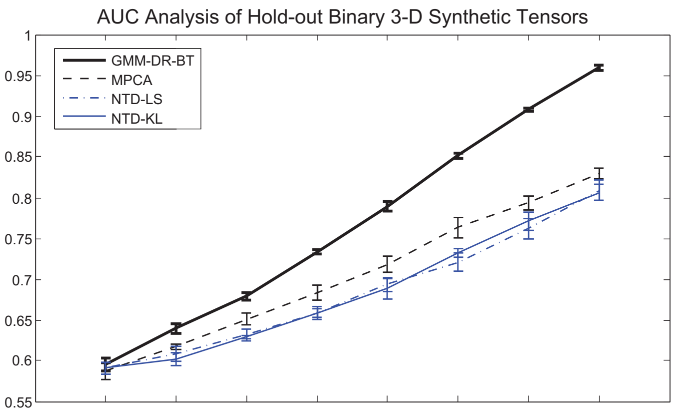
6. Semi-Supervised Extension of GMM-DR-BT

So far we considered the GMM-DR-BT model to be only an unsupervised dimensionality reduction method for binary tensors. However, many problems in the machine learning domain involve decompositions which to certain degree preserve the label information provided for some data items. Such decomposition methods that aim to benefit from both labeled and unlabeled data and make a compromise between the quality of data representation and the degree of class separation are called semi-supervised decomposition methods.

In this section we propose an extension of our GMM-DR-BT model to the semi-supervised setting by forcing the model to search for a natural parameter subspace that represents a user specified compromise between the modelling quality and the degree of class separation. We do so by extending the model likelihood function with a separability measure of projected data samples from different classes.

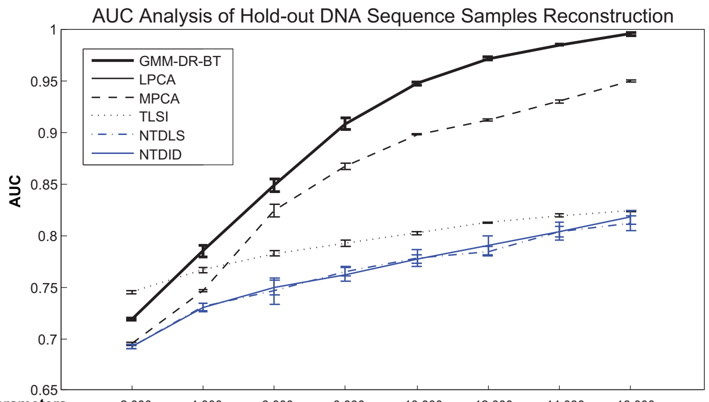
6.1 The Model

To enforce class separability of data items living in a metric space, Globerson and Roweis introduce a distribution



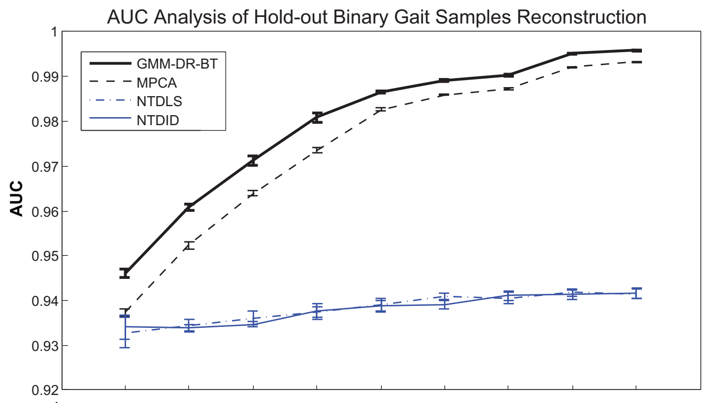
Number of free parameters	=45	=90	=135	=180	=225	=270	=315	=360
GMM-DR-BT (num. of basis vectors)	1x1x1	2x2x2	3x3x3	4x4x4	5x5x5	6x6x6	7x7x7	8x8x8
MPCA (num. of basis vectors)	1x1x1	2x2x2	3x3x3	4x4x4	5x5x5	6x6x6	7x7x7	8x8x8
NTD-LS (num. of basis vectors)	1x1x1	2x2x2	3x3x3	4x4x4	5x5x5	6x6x6	7x7x7	8x8x8
NTD-KL (num. of basis vectors)	1x1x1	2x2x2	3x3x3	4x4x4	5x5x5	6x6x6	7x7x7	8x8x8

Figure 1: AUC analysis of hold-out 3rd-order synthetic binary tensor reconstructions obtained by the models using different number of free parameters among 5 different sets of binary tensors. Table under the plot describes model settings for particular number of free parameters.



Number of free parameters	≈ 2,000	≈ 4,000	≈ 6,000	≈ 8,000	≈ 10,000	≈ 12,000	≈ 14,000	≈ 16,000
GMM-DR-BT (num. of row x column v.)	1 x 8	2 x 16	3 x 24	4 x 32	5 x 40	6 x 48	7 x 56	8 x 64
MPCA (num. of row x column vectors)	1 x 8	2 x 16	3 x 24	4 x 32	5 x 40	6 x 48	7 x 56	8 x 64
NTD-LS (num. of row x column vectors)	1 x 8	2 x 16	3 x 24	4 x 32	5 x 40	6 x 48	7 x 56	8 x 64
NTD-KL (num. of row x column vectors)	1 x 8	2 x 16	3 x 24	4 x 32	5 x 40	6 x 48	7 x 56	8 x 64
TensorLSI (num. of basis tensors)	7	15	22	29	37	45	53	61

Figure 2: AUC analysis of DNA sequence hold-out 2nd-order binary tensor reconstructions obtained by the models using different number of free parameters among 5 different sets of binary tensors. Table under the plot describes model settings for particular number of free parameters.



Number of free parameters	=96	=150	=204	=246	=300	=354	=408	=450	=504
GMM-DR-BT (num. of basis vectors)	2x1x1	3x2x1	4x3x1	5x3x2	6x4x2	7x5x2	8x6x2	9x6x3	10x7x3
MPCA (num. of basis vectors)	2x1x1	3x2x1	4x3x1	5x3x2	6x4x2	7x5x2	8x6x2	9x6x3	10x7x3
NTD-LS (num. of basis vectors)	2x1x1	3x2x1	4x3x1	5x3x2	6x4x2	7x5x2	8x6x2	9x6x3	10x7x3
NTD-KL (num. of basis vectors)	2x1x1	3x2x1	4x3x1	5x3x2	6x4x2	7x5x2	8x6x2	9x6x3	10x7x3

Figure 3: AUC analysis of hold-out 3rd-order binary gait tensor reconstructions obtained by the models using different number of free parameters among 5 different sets of binary tensors. Table under the plot describes model settings for particular number of free parameters.

over data items l, given a single data point m [9]:

$$p(l|m) = \frac{e^{-d(m,l)}}{\sum_{k \neq m} e^{-d(m,k)}} \qquad m \neq l, \tag{31}$$

where $d(m,l)$ is the distance between the points m and l. Loosely speaking, given a particular data item m, under $p(l|m)$ we are more likely to pick data points closer to m than the more distant ones. In the ideal situation, where all points in the same class are collapsed to a single point and infinitely far from points of different classes, the conditional distributions (31) would become "bi-level" distributions [9]:

$$p_0(l|m) \propto \begin{cases} 1 & y_m = y_l \\ 0 & y_m \neq y_l, \end{cases} \tag{32}$$

where y_m denotes a class label of data point m. In [9], maximal class separation under a given data model is achieved by tuning the model parameters so that the class divergence, $\sum_m \mathrm{KL}[p_0(\cdot|m)||p(\cdot|m)]$, is minimized. Minimizing $\sum_m \mathrm{KL}[p_0(\cdot|m)||p(\cdot|m)]$ is equivalent to maximizing

$$\sum_m \frac{1}{c(y_m)-1} \sum_{\substack{l:y_l=y_m \\ l\neq m}} \log\ p(l|m) = \tag{33}$$

$$\sum_m \frac{1}{c(y_m)-1} \left(\sum_{\substack{l:y_l=y_m \\ l\neq m}} -d(m,l) - \log \sum_{\substack{k \\ k\neq m}} e^{-d(m,k)} \right) \tag{34}$$

where $c(y_m)$ denotes a number of points in class y_m.

Any two natural parameter tensors θ_m and θ_l living in the tensor subspace represent tensors of Bernoulli distributions $P(\mathcal{A}_m|\theta_m)$ and $P(\mathcal{A}_l|\theta_l)$ given by (16). The distance between those Bernoulli tensors is quantified by the symmetric KL divergence $D(m,l)$

$$D(m,l) = \sum_{\mathbf{i}\in\Upsilon} \left(\frac{\mathrm{KL}[\mathcal{P}_{m,\mathbf{i}} \,||\, \mathcal{P}_{l,\mathbf{i}}] + \mathrm{KL}[\mathcal{P}_{l,\mathbf{i}} \,||\, \mathcal{P}_{m,\mathbf{i}}]}{2} \right), \tag{35}$$

where KL divergence between two Bernoulli distributions defined by their means is equal to

$$\mathrm{KL}[\mathcal{P}_{m,\mathbf{i}} \,||\, \mathcal{P}_{l,\mathbf{i}}] = \sum_{x\in\{0,1\}} P(x|\mathcal{P}_{m,\mathbf{i}}) \log \frac{P(x|\mathcal{P}_{m,\mathbf{i}})}{P(x|\mathcal{P}_{l,\mathbf{i}})} \tag{36}$$

and $\mathcal{P}_{m,\mathbf{i}} = \sigma(\theta_{m,\mathbf{i}}) = \left(1 + e^{-\theta_{m,\mathbf{i}}}\right)^{-1}$.

Using $D(m,l)$ as a metric on the tensor subspace of the Bernoulli natural parameters, (31) becomes

$$p(l|m) = \frac{e^{-D(m,l)}}{\sum_{k\neq m} e^{-D(m,k)}} \qquad m \neq l. \tag{37}$$

Given a subset of data tensors $\mathcal{D}_\ell \subset \mathcal{D}$ with class labels, the degree of projected class separation is quantified by (see (34))

$$\mathcal{F}(\mathcal{D}_\ell, \mathbf{y}) =$$

$$\sum_{m\in\mathcal{D}_\ell} \frac{1}{c(y_m)-1} \left(\sum_{\substack{l:y_l=y_m \\ l\neq m}} -D(m,l) - \log \sum_{\substack{k\in\mathcal{D}_\ell \\ k\neq m}} e^{-D(m,k)} \right), \tag{38}$$

where $\mathbf{y}$ is an $|\mathcal{D}_\ell|$-dimensional vector that contains labels for each data tensor in $\mathcal{D}_\ell$.

We aim to find tensor basis that simultaneously maximizes log-likelihood (17) of all training tensors and the degree of projected class separation (38),

$$\mathcal{L}(\mathcal{D}) + \beta\ \mathcal{F}(\mathcal{D}_\ell, \mathbf{y}), \tag{39}$$

where $\beta > 0$ is a regularization constant controlling the trade-off between data representation and separation.

To fit tensor basis, any optimalization technique can be used to maximize (39). We derived the update rules by using a basic gradient ascent method:

$$\mathbf{u}^{(n)}_{q,j} \leftarrow \mathbf{u}^{(n)}_{q,j} + \eta \left(\frac{\partial\mathcal{L}(\mathcal{D})}{\partial\mathbf{u}^{(n)}_{q,j}} + \beta\ \frac{\partial\mathcal{F}(\mathcal{D}_\ell,\mathbf{y})}{\partial\mathbf{u}^{(n)}_{q,j}} \right), \tag{40}$$

$$\Delta_{\mathbf{j}} \leftarrow \Delta_{\mathbf{j}} + \eta \left(\frac{\partial\mathcal{L}(\mathcal{D})}{\partial\Delta_{\mathbf{j}}} + \beta\ \frac{\partial\mathcal{F}(\mathcal{D}_\ell,\mathbf{y})}{\partial\Delta_{\mathbf{j}}} \right). \tag{41}$$

After each update cycle through the training set (updates of the projection space), the expansion coefficients (projections) of the data tensors were calculated as described in section 4.2. A detailed derivation of the final updating formulas can be found in the thesis.

6.2 Experiments

To illustrate the workings of the semi-supervised tensor basis selection, we employed the proposed model to analyze the previously mentioned USF gait challenge dataset [22]. For this experiment, we selected 4 persons (classes) from the dataset that had the highest number of samples (14). This gave us dataset of 4 classes with total of 56 binary tensors of size (32x22x10). We split the 56 tensors into two equal sized disjoin sets, training and hold-out set. Each set contains 7 samples from each class.

The training set of tensors was used to train the models by finding the tensor basis and projecting the training tensors into that basis. After training the models, tensors that were not included in training (hold-out set) were "compressed" by projecting them onto the principal subspace by (30). The hold-out set was used to verify whether the subspace found using the training set represents any global trends in the data.

The model setting (number of basis vectors for each mode) was set to find 4 principal tensors obtained as outer product of 2 1st-order, 2 2nd-order and 1 4rd-order vectors. So each training or hold-out tensor is represented by a 4-dimensional vector of expansion coefficients. To visualize the distribution of such representations, we used principal component analysis (PCA) and projected the real-valued 4-dimensional expansion vectors onto a two-dimensional space defined by the two leading principal vectors.

The results are presented in figure 4. Plots in the first and second columns correspond to the training and hold-out sets, respectively. The first row represents a model with randomly chosen basis vectors for each mode. The second row corresponds to the completely unsupervised setting ($\beta = 0$). The third and fourth rows represent two different settings of class separation enforcement, $\beta = 5$ and $\beta = 20$, respectively. There is a certain degree of natural class separation visible in the tensor subspace found

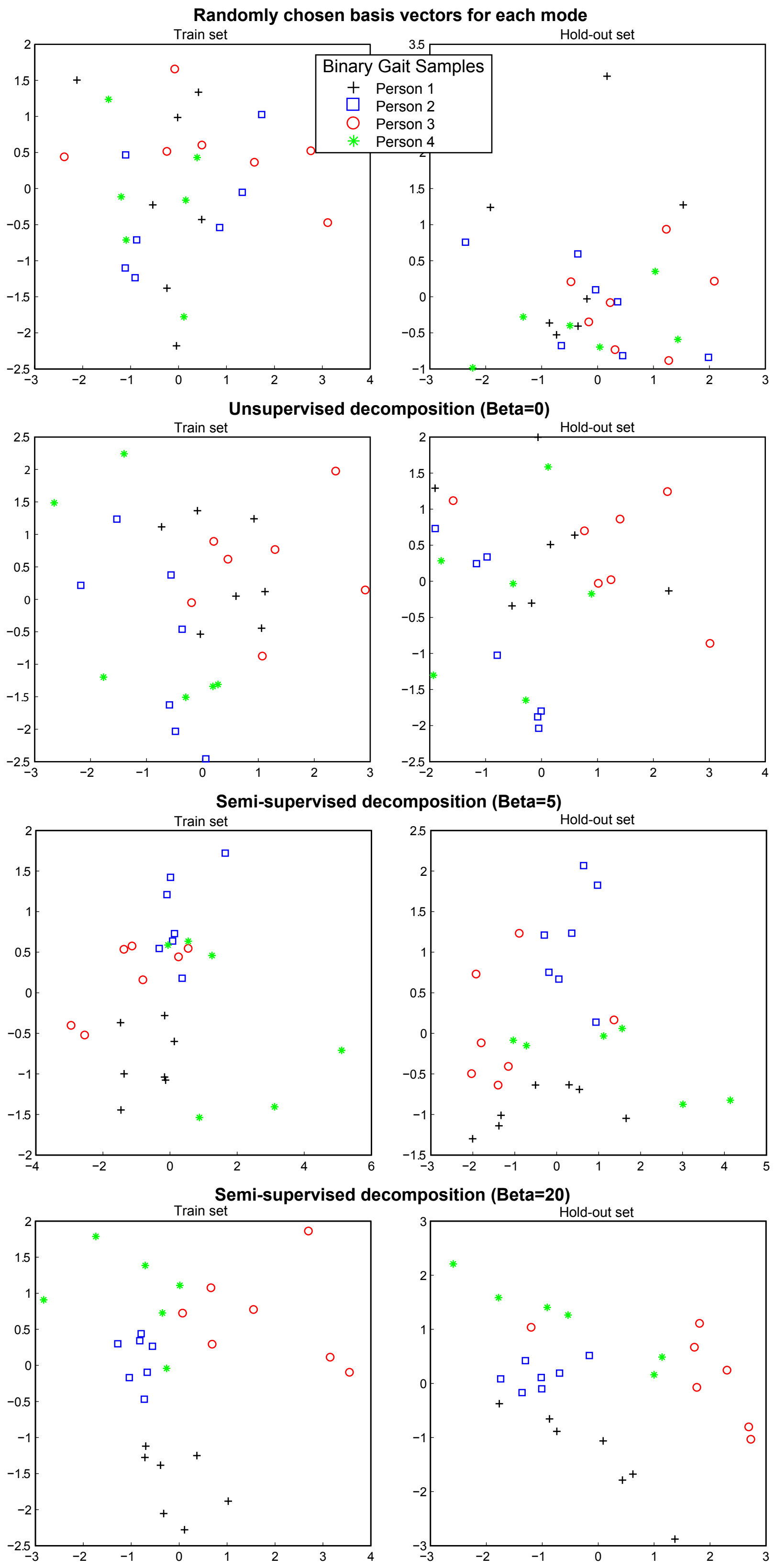


Figure 4: Two-dimensional PCA projections from 4 different tensor spaces of training and hold-out sets of binary gait tensors. The first row represents a tensor model with randomly chosen basis vectors for each mode. The second row corresponds to the completely unsupervised setting ($\beta = 0$). The third and forth rows represents two different settings of class separation enforcement, $\beta = 5$ and $\beta = 20$, respectively.

in the unsupervised manner, without using any class label information. Random subspace position completely fails to discriminate between the four classes. Further imposition of pressure for more class separation yields tensor basis giving strong improvement in the class distribution over the completely unsupervised case as could be seen in the plots of the third and forth rows. More experiments with different datasets can be found in the thesis.

7. Conclusions and Contributions

Current data processing tasks often involve manipulation of binary tensors. However, a principled systematic framework for decomposition of binary tensors was missing. We closed this gap by introducing a generalized multilinear model for dimensionality reduction of binary tensors - GMM-DR-BT. The model is based on the Tucker model concept and could be considered a generalization of the existing logistic principal component analysis model (LPCA) for binary vectorial data decomposition. A probabilistic framework is used to derive the update rules for model parameters. To account for binary nature of the data, each tensor element is modeled by a Bernoulli noise distribution. To extract the dominant trends in the data, we constrain the natural parameters of the Bernoulli distributions to lie in a sub-space spanned by a reduced set of basis (principal) tensors. We derived a simple closed form iterative scheme for parameter estimation.

In the experiments involving synthetic and real-world data sets, we have shown that our GMM-DR-BT model is better suited for modeling binary tensors than the existing real-valued and non-negative tensor decomposition counterparts. Examined models were used to compress and subsequently reconstruct data to measure the discrepancy between original and reconstructed data. Our proposed model clearly outperformed the other models.

We have also investigated the ability of the tensor based models for unsupervised analysis of DNA sub-sequences (represented as binary tensors) from different functional regions. The detailed description and results from this experiment can be found in the thesis and had confirmed the conclusion from the previous experiments that our GMM-DR-BT model is better suited for modeling binary tensors than the existing counterparts.

Furthermore, we extended our GMM-DR-BT model to the semi-supervised setting by forcing the model to search for a natural parameter subspace that represents a user specified compromise between the modeling quality and the degree of class separation. The results from analyzing the binary gait samples showed that implying a combined pressure for modeling quality and class separation yields an improvement in the class distribution over the completely unsupervised case.

Acknowledgements. This work was partially supported by the Scientific Grant Agency of Slovak Republic, grant No. VG1/0508/09 and by the Slovak Research and Development Agency under the contract No. APVV-0208-10.

References

[1] E. Acar and B. Yener. Unsupervised multiway data analysis: A literature survey. *IEEE Trans. Knowl. Data En.*, 21(1):6 –20, Jan. 2009.

[2] J. Brachat, P. Comon, B. Mourrain, and E. Tsigaridas. Symmetric tensor decomposition. *Linear Algebra Appl.*, In Press, Corrected Proof, 2010.

[3] R. Bro. PARAFAC. tutorial and applications. *Chemometrics and Intelligent Laboratory Systems*, 38(2):149 – 171, 1997.

[4] D. Cai, X. He, and J. Han. Tensor space model for document analysis. In *Proc. 29th Annu. ACM SIGIR Int. Conf. Research and Development in Information Retrieval*, pages 625–626, Seatlle, WA, Aug. 2006.

[5] J. Carroll and C. J. Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition. *Psychometrika*, pages 283–319, 1970.

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

[7] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Applicat.*, 21(4):1253–1278, 2000.

[8] H. Eves. *Elementary Matrix Theory*. Dover Publications, April 1980.

[9] A. Globerson and S. Roweis. Metric learning by collapsing classes. *Advances in Neural Information Processing Systems*, 18:451–458, 2006.

[10] L. Haiping, K. N. Plataniotis, and A. N. Venetsanopoulos. A survey of multilinear subspace learning for tensor data. *Pattern Recognition*, 44(7):1540–1551, Jul 2011.

[11] R. A. Harshman. Foundations of the PARAFPAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.

[12] K. Jia and S. Gong. Multi-modal tensor face for simultaneous super-resolution and recognition. In *10th IEEE Int. Conf. Computer Vision*, volume 2, pages 1683–1690, Beijing, Oct. 2005.

[13] Y. D. Kim and S. Choi. Nonnegative tucker decomposition. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2007.

[14] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 2008. to appear (accepted June 2008).

[15] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. *Advences in Neural Information Processing Systems*, 13, 2001.

[16] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. MPCA: Multilinear principal component analysis of tensor objects. *IEEE Trans. Neural Netw.*, 19:18–39, Jan. 2008.

[17] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. Uncorrelated multilinear discriminant analysis with regularization and aggregation for tensor object recognition. *IEEE Trans. Neural Netw.*, 20:103–123, Jan. 2009.

[18] C. Nolker and H. Ritter. Visual recognition of continuous hand postures. *IEEE Trans. Neural Netw.*, 13:983–994, July 2002.

[19] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901.

[20] A. Phan and A. Cichocki. Tensor decompositions for feature extraction and classification of high dimensional datasets. In *Nonlinear Theory and Its Applications, IEICE (invited paper)*, October 2010.

[21] N. Renard and S. Bourennane. An ICA-based multilinear algebra tools for dimensionality reduction in hyperspectral imagery. In *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, volume 4, pages 1345–1348, Las Vegas, NV, Apr. 2008.

[22] S. Sarkar, P. J. Phillips, Z. Liu, I. R. Vega, P. Grother, and K. W. Bowyer. The human id gait challenge problem: Data sets, performance, and analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:162–177, 2005.

[23] A. Schein, L. Saul, and L. Ungar. A generalized linear model for principal component analysis of binary data. In *9th Int. Workshop Artificial Intelligence and Statistics*, Key West, FL, Jan. 2003.

[24] S. Sra and D. I. S. Nonnegative matrix approximation: Algorithms and applications. Technical Report TR-06-27, Dept. of Computer Sciences, University of Texas at Austin, Austin, TX 78712, USA, June 2006.

[25] J. A. Tropp. Literature Survey: Non-negative matrix factorization. Technical report, Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX 78712, USA, March 2009.

[26] L. R. Tucker. The extension of factor analysis to three-dimensional matrices. In H. Gulliksen and N. Frederiksen, editors, *Contributions to mathematical psychology.*, pages 110–127. Holt, Rinehart and Winston, New York, 1964.

[27] H. Wang and N. Ahuja. Rank-r approximation of tensors: Using image-as-matrix representation. In *Computer Vision and Pattern Recognition*, pages 346–353, 2005.

[28] H. Wang and N. Ahuja. A tensor approximation approach to dimensionality reduction. *Int. J. Comput. Vision*, 76:217–229, March 2008.

[29] S. Zafeiriou. Discriminant nonnegative tensor factorization algorithms. *IEEE Trans. Neural Netw.*, 20:217–235, Feb. 2009.

Selected Papers by the Author

J. Mažgut, P. Tiňo, M. Bóden, H. Yan. Multilinear Decomposition and Topographic Mapping of Binary Tensors. In *Artificial Neural Networks – ICANN 2010*, LNCS 6352, pages 317–326, Thessaloniki, Greece, 2010. Springer.

J. Mažgut, M. Paulinyová, P. Tiňo. Using Dimensionality Reduction Method for Binary Data to Questionnaire Analysis. In Z. Kotásek, et al. (Eds.), *MEMICS 2011*, LNCS 7119, pages 146–154, Lednice, Czech Republic, 2011. Springer.

J. Mažgut. Analyzing DNA Sequences Based on Oligomer-Position Information. In *5th Student Research Conference in Informatics and Information Technologies Bratislava*, pages 186–193. Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, 2009.

J. Mažgut. The Analysis of Binary Data. In *6th Student Research Conference in Informatics and Information Technologies Bratislava*, pages 499–505. Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, 2010.