Generalized Multi-Linear Principal Component Analysis of Binary Tensors

Jakub Mažgút, Peter Tiňo, Mikael Bodén and Hong Yan[§]

28 January 2010

Abstract

Current data processing tasks often involve manipulation of multi-dimensional objects - tensors. In many real world applications such as gait recognition, document analysis or graph mining (with graphs represented by adjacency tensors), the tensors can be constrained to binary values only. To the best of our knowledge at present there is no principled systematic framework for decomposition of binary tensors. To close this gap we propose a generalized multi-linear model for principal component analysis of binary tensors (GML-PCA). In the 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, we constrain the natural parameters of the Bernoulli distributions 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 a an iterative scheme for estimation of the basis tensors and other model parameters via maximum likelihood. We evaluate and compare the proposed GML-PCA technique with an existing real-valued tensor decomposition method (TensorLSI) in two scenarios: (1) in a series of controlled experiments involving synthetic data; (2) on a real world biological dataset of DNA sub-sequences from different functional regions, with sequences represented by binary tensors. The experiments suggest that the GML-PCA model is better suited for modeling binary tensors than its real-valued counterpart - TensorLSI model.

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 many practical problems such as gait [9] and face recognition [10], hyperspectra image processing [14] or text documents analysis [4], the data tensors are specified in a high-dimensional space. 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

[*]J. Mažgut is with the Faculty of Informatics and Information Technologies, Slovak University of Technology, 81219 Bratislava, Slovakia (e-mail: mazgut@fiit.stuba.sk).

[†]P. Tiňo is with the School of Computer Science, The University of Birmingham, Birmingham B15 2TT, UK (e-mail: P.Tino@cs.bham.ac.uk).

[‡]M. Bodén is with the Institute of Molecular Bioscience and School of Information Technology and Electrical Engineering, The University of Queensland, QLD 4072, Australia (e-mail: m.boden@uq.edu.au).

[§]H. Yan is with the Dept. Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong (e-mail: h.yan@cityu.edu.hk).

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 crucially can also increase their interpretability.

Decomposition techniques such as principal component analysis (PCA) [13] were designed to decompose data objects in the form of vectors. For tensor decomposition, the data items need to be first vectorized before the analysis can be applied. Besides 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 [9]. New methods capable of processing multi-dimensional real-valued tensors in their natural structure have been introduced [9, 4]. Such techniques, however, are not suitable for processing binary tensors. Yet, binary tensors arise in many real world applications such as gait recognition [9], document analysis [4] or graph objects represented by adjacency tensors. In this paper we introduce and verify a model based method for binary tensor decomposition that explicitly takes into an account the binary nature of such data.

The paper is organized as follows: Section 2 introduces notation and basic tensor algebra. Section 3 briefly discusses the problem of reduced rank representation of real-valued tensors. A model based formulation for binary data decomposition with iterative update scheme to maximize the model's log-likelihood is presented in sections 4, 5 and 6. Section 7 contains experiments on synthetic and biological datasets. Finally, section 8 summarizes the results and concludes the work.

2 Notation and basic tensor algebra

An Nth order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ can be thought of as an N-dimensional array of real numbers in programming languages. It is addressed by N indices i_n ranging from 1 to I_n, $n = 1, 2, ..., N$. A rank-1 tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ can be obtained as an outer product of N 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 other words, for a particular index setting $(i_1, i_2, ..., i_N) \in \Upsilon = \{1, 2, ..., I_1\} \times \{1, 2, ..., I_2\} \times ... \times \{1, 2, ..., I_N\}$, we have

$$\mathcal{A}_{i_1, i_2, ..., i_N} = \prod_{n=1}^{N} u_{i_n}^{(n)}, \tag{1}$$

where $u_{i_n}^{(n)}$ is the i_n-th component of the vector $\mathbf{u}^{(n)}$. Slightly abusing mathematical notation, we will often write the index N-tuples $(i_1, i_2, ..., i_N) \in \Upsilon$ 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 tensor can be multiplied by a matrix (2nd order tensor) using n-mode products: The n-mode product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ by a matrix $U \in \mathbb{R}^{J \times I_n}$ is a tensor $(\mathcal{A} \times_n U)$ given by

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

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

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 $\mathcal{Q} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$. The expansion of $\mathcal{A}$ can also be written as

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

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

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

3 Reduced rank representations of tensors

Several approaches have been proposed for reduced rank representations of tensors [7, 11, 22]. For example, one can assume that a smaller number of basis tensors in the expansion (2) are sufficient to approximate all tensors in a given dataset:

$$\mathcal{A} \approx \sum_{\mathbf{i} \in K} \mathcal{Q}_{\mathbf{i}} \cdot (\mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ ... \circ \mathbf{u}_{i_N}^{(N)}). \tag{3}$$

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

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. If the n-mode space basis are orthonormal, the tensor decomposition is known as the Higher-Order Singular Value Decomposition (HOSVD) [7]. It has to be said that extending matrix (2nd-order tensor) decompositions such as SVD to higher-order tensors is not an easy matter. Familiar concepts such as rank become ambiguous and more complex. However, the main purpose of the decomposition remains unchanged: rewrite a tensor as a sum of rank-1 tensors.

While much work has been done in the context of SVD-style decompositions of real-valued tensors [7], no formalism exists as yet for decomposing binary tensors. Binary tensors occur naturally in many applications where the value $\mathcal{A}_{i_1, i_2, ..., i_N}$ indicates presence or absence of the feature related to the index $(i_1, i_2, ..., i_N)$. For example, in graph theory, a 2nd-order tensor $\mathcal{A}$ (called adjacency matrix) codes a graph by imposing $\mathcal{A}_{i_1, i_2} = 1$ if and only if there is an arc from node i_1 to node i_2, $\mathcal{A}_{i_1, i_2} = 0$ otherwise. We present a framework that is a generalization of the binary probabilistic principal component analysis to tensor data.

4 The model

Consider binary Nth-order tensors $\mathcal{A} \in \{0,1\}^{I_1 \times I_2 \times ... \times I_N}$. 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 assumed to be (independently) Bernoulli distributed with parameter (mean) $p_{m,\mathbf{i}}$:

$$P(\mathcal{A}_{m,\mathbf{i}}|p_{m,\mathbf{i}}) = p_{m,\mathbf{i}}^{\mathcal{A}_{m,\mathbf{i}}} \cdot (1 - p_{m,\mathbf{i}})^{1-\mathcal{A}_{m,\mathbf{i}}}. \tag{4}$$

The distribution can be equivalently parametrized through log-odds (natural parameter)

$$\theta_{m,\mathbf{i}} = \log \frac{p_{m,\mathbf{i}}}{1 - p_{m,\mathbf{i}}},$$

so that the canonical link function linking the natural parameter with the mean is the logistic function

$$p_{m,\mathbf{i}} = \sigma(\theta_{m,\mathbf{i}}) = \frac{1}{1 + e^{-\theta_{m,\mathbf{i}}}}. \tag{5}$$

Note that $1 - p_{m,\mathbf{i}} = \sigma(-\theta_{m,\mathbf{i}})$.

For each data tensor $\mathcal{A}_m$, $m = 1, 2, ..., M$, we have

$$P(\mathcal{A}_m|\theta_m) = \prod_{\mathbf{i} \in \Upsilon} P(\mathcal{A}_{m,\mathbf{i}}|\theta_{m,\mathbf{i}}), \tag{6}$$

where

$$P(\mathcal{A}_{m,\mathbf{i}}|\theta_{m,\mathbf{i}}) = \sigma(\theta_{m,\mathbf{i}})^{\mathcal{A}_{m,\mathbf{i}}} \cdot \sigma(-\theta_{m,\mathbf{i}})^{1-\mathcal{A}_{m,\mathbf{i}}}. \tag{7}$$

We collect all the parameters $\theta_{m,\mathbf{i}}$ in a tensor $\Theta \in \mathbb{R}^{M \times I_1 \times I_2 \times ... \times I_N}$ of order $N + 1$. Assuming the data tensors in $\mathcal{D}$ are independently generated, the model likelihood reads

$$L(\Theta) = \prod_{m=1}^{M} P(\mathcal{A}_m|\theta_m). \tag{8}$$

Using (6), we write the log-likelihood as

$$\mathcal{L}(\Theta) = \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon} \mathcal{A}_{m,\mathbf{i}} \log \sigma(\theta_{m,\mathbf{i}}) + (1 - \mathcal{A}_{m,\mathbf{i}}) \log \sigma(-\theta_{m,\mathbf{i}}). \tag{9}$$

So far the values in the parameter tensor Θ were unconstrained. We constrain all the Nth order parameter tensors $\theta_m \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ (one for each data tensor $\mathcal{A}_m$) to lie in the subspace spanned by the reduced set of basis tensors $(\mathbf{u}_{r_1}^{(1)} \circ \mathbf{u}_{r_2}^{(2)} \circ ... \circ \mathbf{u}_{r_N}^{(N)})$, 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 $\rho = \{1, 2, ..., R_1\} \times \{1, 2, ..., R_2\} \times ... \times \{1, 2, ..., R_N\}$.

We further allow for an N-th order bias tensor $\Delta \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$, so that the parameter tensors θ_m are constrained onto an affine space. Using (2) we get

$$\theta_m = \sum_{\mathbf{r} \in \rho} \mathcal{Q}_{m,\mathbf{r}} \cdot (\mathbf{u}_{r_1}^{(1)} \circ \mathbf{u}_{r_2}^{(2)} \circ ... \circ \mathbf{u}_{r_N}^{(N)}) + \Delta \tag{10}$$

so that by

$$\begin{aligned}\theta_{m,\mathbf{i}} &= \sum_{\mathbf{r}\in\rho} \mathcal{Q}_{m,\mathbf{r}} \cdot (\mathbf{u}^{(1)}_{r_1} \circ \mathbf{u}^{(2)}_{r_2} \circ ... \circ \mathbf{u}^{(N)}_{r_N})_{\mathbf{i}} + \Delta_{\mathbf{i}} \\ &= \sum_{\mathbf{r}\in\rho} \mathcal{Q}_{m,\mathbf{r}} \cdot \prod_{n=1}^{N} u^{(n)}_{r_n,i_n} + \Delta_{\mathbf{i}}, \end{aligned} \tag{11}$$

the log-likelihood is evaluated as

$$\begin{aligned}\mathcal{L} = \sum_{m=1}^{M}\sum_{\mathbf{i}\in\Upsilon} \quad & \mathcal{A}_{m,\mathbf{i}} \quad \log\sigma\left(\sum_{\mathbf{r}\in\rho} \mathcal{Q}_{m,\mathbf{r}} \cdot \prod_{n=1}^{N} u^{(n)}_{r_n,i_n} + \Delta_{\mathbf{i}}\right) + \\ & (1-\mathcal{A}_{m,\mathbf{i}}) \quad \log\sigma\left(-\sum_{\mathbf{r}\in\rho} \mathcal{Q}_{m,\mathbf{r}} \cdot \prod_{n=1}^{N} u^{(n)}_{r_n,i_n} - \Delta_{\mathbf{i}}\right). \end{aligned} \tag{12}$$

5 Parameter estimation

To get analytical parameter updates, we use the the trick of [17] and take advantage of the fact that while the log-likelihood (12) is not convex in the parameters, it is convex in any parameter, if the others are kept fixed. This leads to an iterative estimation scheme detailed below.

The analytical updates will be derived from a lower bound on the log-likelihood (12) using [17]:

$$\log\sigma(\hat{\theta}) \geq -\log 2 + \frac{\hat{\theta}}{2} - \log\cosh\left(\frac{\theta}{2}\right) - (\hat{\theta}^2 - \theta^2)\ \frac{\tanh\frac{\theta}{2}}{4\theta}, \tag{13}$$

where θ stands for the current value of individual natural parameters $\theta_{m,\mathbf{i}}$ of the Bernoulli noise models $P(\mathcal{A}_{m,\mathbf{i}}|\theta_{m,\mathbf{i}})$ and $\hat{\theta}$ stands for the future estimate of the parameter, given the current parameter values. Hence, from (9) we obtain[1]

$$\begin{aligned}\mathcal{L}(\hat{\Theta}) &= \sum_{m=1}^{M}\sum_{\mathbf{i}\in\Upsilon} \mathcal{A}_{m,\mathbf{i}} \log\sigma(\hat{\theta}_{m,\mathbf{i}}) + (1-\mathcal{A}_{m,\mathbf{i}}) \log\sigma(-\hat{\theta}_{m,\mathbf{i}}) \\ &\geq \sum_{m=1}^{M}\sum_{\mathbf{i}\in\Upsilon} \mathcal{A}_{m,\mathbf{i}} \left[-\log 2 + \frac{\hat{\theta}_{m,\mathbf{i}}}{2} - \log\cosh\left(\frac{\theta_{m,\mathbf{i}}}{2}\right) - (\hat{\theta}^2_{m,\mathbf{i}} - \theta^2_{m,\mathbf{i}})\ \frac{\tanh\frac{\theta_{m,\mathbf{i}}}{2}}{4\theta_{m,\mathbf{i}}}\right] \\ &+ (1-\mathcal{A}_{m,\mathbf{i}}) \left[-\log 2 - \frac{\hat{\theta}_{m,\mathbf{i}}}{2} - \log\cosh\left(\frac{\theta_{m,\mathbf{i}}}{2}\right) - (\hat{\theta}^2_{m,\mathbf{i}} - \theta^2_{m,\mathbf{i}})\ \frac{\tanh\frac{\theta_{m,\mathbf{i}}}{2}}{4\theta_{m,\mathbf{i}}}\right] \quad (14) \\ &= H(\hat{\Theta},\Theta). \quad (15)\end{aligned}$$

Denote $(\tanh\frac{\theta_{m,\mathbf{i}}}{2})/\theta_{m,\mathbf{i}}$ by $T_{m,\mathbf{i}}$. Grouping together constant terms in (14) leads to

$$H(\hat{\Theta},\Theta) = Const + \sum_{m=1}^{M}\sum_{\mathbf{i}\in\Upsilon} \left[\hat{\theta}_{m,\mathbf{i}}\left(\mathcal{A}_{m,\mathbf{i}} - \frac{1}{2}\right) - \frac{T_{m,\mathbf{i}}}{4}\ \hat{\theta}^2_{m,\mathbf{i}}\right]. \tag{16}$$

[1] θ's are fixed current values of the parameters and should be treated as constants

Note that $H(\hat{\Theta}, \Theta) = \mathcal{L}(\hat{\Theta})$ only if $\hat{\Theta} = \Theta$. Therefore by choosing $\hat{\Theta}$ that maximizes $H(\hat{\Theta}, \Theta)$ we guarantee $\mathcal{L}(\hat{\Theta}) \geq H(\hat{\Theta}, \Theta) \geq H(\Theta, \Theta) = \mathcal{L}(\Theta)$ [17].

We are now ready to constrain the Bernoulli parameters to be optimized (see (11)):

$$\hat{\theta}_{m,\mathbf{i}} = \sum_{\mathbf{r} \in \rho} \mathcal{Q}_{m,\mathbf{r}} \cdot \prod_{n=1}^{N} u^{(n)}_{r_n, i_n} + \Delta_{\mathbf{i}}. \tag{17}$$

We will update the model parameters so as to maximize

$$\mathcal{H} = \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon} \mathcal{H}_{m,\mathbf{i}}, \tag{18}$$

where

$$\mathcal{H}_{m,\mathbf{i}} = \left(\mathcal{A}_{m,\mathbf{i}} - \frac{1}{2} \right) \hat{\theta}_{m,\mathbf{i}} - \frac{T_{m,\mathbf{i}}}{4} \, \hat{\theta}^2_{m,\mathbf{i}}, \tag{19}$$

with $\hat{\theta}_{m,\mathbf{i}}$ given by (17).

5.1 Updates for n-mode space basis

When updating the n-mode space basis $\{\mathbf{u}^{(n)}_1, \mathbf{u}^{(n)}_2, ..., \mathbf{u}^{(n)}_{R_n}\}$, the bias tensor Δ and the expansion coefficients $\mathcal{Q}_{m,\mathbf{r}}$, $m = 1, 2, ...M$, $\mathbf{r} \in \rho$, are kept fixed to their current values.

For $n = 1, 2, ..., N$, define

$$\Upsilon_{-n} = \{1, 2, ..., I_1\} \times ... \times \{1, 2, ..., I_{n-1}\} \times \{1\} \times \{1, 2, ..., I_{n+1}\} \times ... \times \{1, 2, ..., I_N\}, \tag{20}$$

with obvious interpretation in the boundary cases. Given $\mathbf{i} \in \Upsilon_{-n}$ and an n-mode index $j \in \{1, 2, ..., I_n\}$, the index N-touple $(i_1, ..., i_{n-1}, j, i_{n+1}, ..., i_N)$ formed by inserting j at the nth place of $\mathbf{i}$ is denoted by $[\mathbf{i}, j|n]$.

In order to evaluate

$$\frac{\partial \, \mathcal{H}}{\partial \, u^{(n)}_{q,j}}, \quad q = 1, 2, ..., R_n, \; j = 1, 2, ..., I_n,$$

we realize that $u^{(n)}_{q,j}$ is involved in expressing all $\hat{\theta}_{m,[\mathbf{i},j|n]}$, $m = 1, 2, ..., M$, with $\mathbf{i} \in \Upsilon_{-n}$. Therefore,

$$\frac{\partial \, \mathcal{H}}{\partial \, u^{(n)}_{q,j}} = \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon_{-n}} \frac{\partial \, \mathcal{H}_{m,[\mathbf{i},j|n]}}{\partial \, \hat{\theta}_{m,[\mathbf{i},j|n]}} \, \frac{\partial \, \hat{\theta}_{m,[\mathbf{i},j|n]}}{\partial \, u^{(n)}_{q,j}}, \tag{21}$$

where

$$\frac{\partial \, \mathcal{H}_{m,[\mathbf{i},j|n]}}{\partial \, \hat{\theta}_{m,[\mathbf{i},j|n]}} = \left(\mathcal{A}_{m,[\mathbf{i},j|n]} - \frac{1}{2} \right) - \frac{T_{m,[\mathbf{i},j|n]}}{2} \, \hat{\theta}_{m,[\mathbf{i},j|n]} \tag{22}$$

and from (17),

$$\frac{\partial \, \hat{\theta}_{m,[\mathbf{i},j|n]}}{\partial \, u^{(n)}_{q,j}} = \mathcal{B}^{(n)}_{m,\mathbf{i},q} = \sum_{\mathbf{r} \in \rho_{-n}} \mathcal{Q}_{m,[\mathbf{r},q|n]} \cdot \prod_{s=1, s \neq n}^{N} u^{(s)}_{r_s, i_s}. \tag{23}$$

Here, the index set ρ_{-n} is defined analogously to Υ_{-n}:

$$\rho_{-n} = \{1, 2, ..., R_1\} \times ... \times \{1, 2, ..., R_{n-1}\} \times \{1\} \times \{1, 2, ..., R_{n+1}\} \times ... \times \{1, 2, ..., R_N\}. \tag{24}$$

Setting the derivative (21) to zero results in

$$\sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon_{-n}} (2\mathcal{A}_{m,[\mathbf{i},j|n]} - 1)\ \mathcal{B}^{(n)}_{m,\mathbf{i},q} = \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon_{-n}} T_{m,[\mathbf{i},j|n]}\ \hat{\theta}_{m,[\mathbf{i},j|n]}\ \mathcal{B}^{(n)}_{m,\mathbf{i},q}. \tag{25}$$

Rewriting (17) as

$$\hat{\theta}_{m,[\mathbf{i},j|n]} = \sum_{t=1}^{R_n} \sum_{\mathbf{r} \in \rho_{-n}} \mathcal{Q}_{m,[\mathbf{r},t|n]}\ u^{(n)}_{t,j} \prod_{s=1, s\neq n}^{N} u^{(s)}_{r_s,i_s}\ +\ \Delta_{[\mathbf{i},j|n]}, \tag{26}$$

and applying to (25) we obtain

$$\sum_{t=1}^{R_n} u^{(n)}_{t,j}\ \mathcal{K}^{(n)}_{q,t,j} = \mathcal{S}^{(n)}_{q,j}, \tag{27}$$

where

$$\mathcal{S}^{(n)}_{q,j} = \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon_{-n}} (2\mathcal{A}_{m,[\mathbf{i},j|n]} - 1 - T_{m,[\mathbf{i},j|n]}\ \Delta_{[\mathbf{i},j|n]})\ \mathcal{B}^{(n)}_{m,\mathbf{i},q} \tag{28}$$

and

$$\mathcal{K}^{(n)}_{q,t,j} = \sum_{m=1}^{M} \sum_{\mathbf{r} \in \rho_{-n}} \mathcal{Q}_{m,[\mathbf{r},t|n]} \sum_{\mathbf{i} \in \Upsilon_{-n}} T_{m,[\mathbf{i},j|n]}\ \mathcal{B}^{(n)}_{m,\mathbf{i},q} \prod_{s=1, s\neq n}^{N} u^{(s)}_{r_s,i_s}. \tag{29}$$

For each n-mode coordinate $j \in \{1, 2, ..., I_n\}$, collect the j-th coordinate values of all n-mode basis vectors into a column vector $\mathbf{u}^{(n)}_{:,j} = (u^{(n)}_{1,j}, u^{(n)}_{2,j}, ..., u^{(n)}_{R_n,j})^T$. Analogously, stack all the $\mathcal{S}^{(n)}_{q,j}$ values in a column vector $\mathcal{S}^{(n)}_{:,j} = (\mathcal{S}^{(n)}_{1,j}, \mathcal{S}^{(n)}_{2,j}, ..., \mathcal{S}^{(n)}_{R_n,j})^T$. Finally, we construct an $R_n \times R_n$ matrix $\mathcal{K}^{(n)}_{:,:,j}$ whose q-th row is $(\mathcal{K}^{(n)}_{q,1,j}, \mathcal{K}^{(n)}_{q,2,j}, ..., \mathcal{K}^{(n)}_{q,R_n,j})$, $q = 1, 2, ..., R_n$. The n-mode basis vectors are updated by solving I_n linear systems of size $R_n \times R_n$:

$$\mathcal{K}^{(n)}_{:,:,j}\ \mathbf{u}^{(n)}_{:,j} = \mathcal{S}^{(n)}_{:,j}, \quad j = 1, 2, ..., I_n. \tag{30}$$

5.2 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.

For $\mathbf{r} \in \rho$ and $\mathbf{i} \in \Upsilon$ denote $\prod_{n=1}^{N} u^{(n)}_{r_n,i_n}$ by $C_{\mathbf{r},\mathbf{i}}$. For data index $\ell = 1, 2, ..., M$ and basis index $\mathbf{v} \in \rho$ we have

$$\frac{\partial\ \mathcal{H}}{\partial\ \mathcal{Q}_{\ell,\mathbf{v}}} = \sum_{m=1}^{M} \sum_{\mathbf{i} \in \Upsilon} \frac{\partial\ \mathcal{H}_{m,\mathbf{i}}}{\partial\ \hat{\theta}_{m,\mathbf{i}}}\ \frac{\partial\ \hat{\theta}_{m,\mathbf{i}}}{\partial\ \mathcal{Q}_{\ell,\mathbf{v}}}, \tag{31}$$

where

$$\frac{\partial\ \mathcal{H}_{m,\mathbf{i}}}{\partial\ \hat{\theta}_{m,\mathbf{i}}} = \left(\mathcal{A}_{m,\mathbf{i}} - \frac{1}{2}\right) - \frac{T_{m,\mathbf{i}}}{2}\ \hat{\theta}_{m,\mathbf{i}} \tag{32}$$

and $\frac{\partial\ \hat{\theta}_{m,\mathbf{i}}}{\partial\ \mathcal{Q}_{\ell,\mathbf{v}}} = C_{\mathbf{v},\mathbf{i}}$ if $m = \ell$ and $\frac{\partial\ \hat{\theta}_{m,\mathbf{i}}}{\partial\ \mathcal{Q}_{\ell,\mathbf{v}}} = 0$ otherwise.

By imposing $\frac{\partial\ \mathcal{H}}{\partial\ \mathcal{Q}_{\ell,\mathbf{v}}} = 0$, we get

$$\mathcal{T}_{\mathbf{v},\ell} = \sum_{\mathbf{r}\in\rho} \mathcal{P}_{\mathbf{v},\mathbf{r},\ell}\ \mathcal{Q}_{\ell,\mathbf{r}}, \tag{33}$$

where

$$\mathcal{T}_{\mathbf{v},\ell} = \sum_{\mathbf{i}\in\Upsilon} (2\mathcal{A}_{\ell,\mathbf{i}} - 1 - T_{\ell,\mathbf{i}}\ \Delta_{\mathbf{i}})\ C_{\mathbf{v},\mathbf{i}} \tag{34}$$

and

$$\mathcal{P}_{\mathbf{v},\mathbf{r},\ell} = \sum_{\mathbf{i}\in\Upsilon} T_{\ell,\mathbf{i}}\ C_{\mathbf{v},\mathbf{i}}\ C_{\mathbf{r},\mathbf{i}}. \tag{35}$$

To solve for expansion coefficients using tools of matrix algebra, we need to vectorize tensor indices. Consider any one-to-one function κ from ρ to $\{1, 2, ..., \prod_{n=1}^{N} R_n\}$. For each input tensor index $\ell = 1, 2, ..., M$,

- create a square $(\prod_{n=1}^{N} R_n) \times (\prod_{n=1}^{N} R_n)$ matrix $\mathcal{P}_{:,:,\ell}$ whose $(\kappa(\mathbf{v}), \kappa(\mathbf{r}))$-th element is equal to $\mathcal{P}_{\mathbf{v},\mathbf{r},\ell}$,
- stack the values of $\mathcal{T}_{\mathbf{v},\ell}$ into a column vector $\mathcal{T}_{:,\ell}$ whose $\kappa(\mathbf{v})$-th coordinate is $\mathcal{T}_{\mathbf{v},\ell}$,
- collect the expansion coefficients $\mathcal{Q}_{\ell,\mathbf{r}}$ in a column vector $\mathcal{Q}_{\ell,:}$ with $\kappa(\mathbf{r})$-th coordinate equal to $\mathcal{Q}_{\ell,\mathbf{r}}$.

The expansion coefficients for the ℓth input tensor $\mathcal{A}_\ell$ can be obtained by solving

$$\mathcal{P}_{:,:,\ell}\ \mathcal{Q}_{\ell,:} = \mathcal{T}_{:,\ell}, \quad \ell = 1, 2, ..., M. \tag{36}$$

5.3 Updates for the bias tensor

As before, when updating the bias tensor Δ, the expansion coefficients $\mathcal{Q}_{m,\mathbf{r}}$, $m = 1, 2, ...M$, $\mathbf{r} \in \rho$, and the basis sets $\{\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, ..., \mathbf{u}_{R_n}^{(n)}\}$ for all n modes $n = 1, 2, ..., N$ are kept fixed to their current values.

Fix $\mathbf{j} \in \Upsilon$. We evaluate

$$\frac{\partial\ \mathcal{H}}{\partial\ \Delta_{\mathbf{j}}} = \sum_{m=1}^{M} \sum_{\mathbf{i}\in\Upsilon} \frac{\partial\ \mathcal{H}_{m,\mathbf{i}}}{\partial\ \hat{\theta}_{m,\mathbf{i}}}\ \frac{\partial\ \hat{\theta}_{m,\mathbf{i}}}{\partial\ \Delta_{\mathbf{j}}}, \tag{37}$$

where $\frac{\partial\ \hat{\theta}_{m,\mathbf{i}}}{\partial\ \Delta_{\mathbf{j}}}$ is equal to 1 if $\mathbf{i} = \mathbf{j}$ and 0 otherwise.

Solving for $\frac{\partial\ \mathcal{H}}{\partial\ \Delta_{\mathbf{j}}} = 0$ leads to

$$\Delta_{\mathbf{j}} = \frac{\sum_{m=1}^{M} 2\mathcal{A}_{m,\mathbf{j}} - 1 - T_{m,\mathbf{j}} \cdot \sum_{\mathbf{r}\in\rho} \mathcal{Q}_{m,\mathbf{r}}\ C_{\mathbf{r},\mathbf{j}}}{\sum_{m=1}^{M} T_{m,\mathbf{j}}}. \tag{38}$$

6 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 $\{\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, ..., \mathbf{u}_{R_n}^{(n)}\}$, one for each n mode, $n = 1, 2, ..., N$. These are kept fixed.

The likelihood (12) to be maximized with respect to the expansion coefficients stored in tensor $\mathcal{Q}$ reads

$$\begin{aligned} \mathcal{L}(\mathcal{Q}; \mathcal{A}') = & \sum_{\mathbf{i},\ s.t.\ \mathcal{A}'_{\mathbf{i}}=1} \log \sigma \left(\sum_{\mathbf{r} \in \rho} \mathcal{Q}_{\mathbf{r}}\ C_{\mathbf{r},\mathbf{i}} + \Delta_{\mathbf{i}} \right) + \\ & \sum_{\mathbf{i},\ s.t.\ \mathcal{A}'_{\mathbf{i}}=0} \log \sigma \left(- \sum_{\mathbf{r} \in \rho} \mathcal{Q}_{\mathbf{r}}\ C_{\mathbf{r},\mathbf{i}} - \Delta_{\mathbf{i}} \right). \end{aligned} \tag{39}$$

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 data set $\mathcal{D}$ to $\mathcal{A}'$ in the Hamming distance sense,

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

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

$$\mathcal{Q}_{\mathbf{v}} \leftarrow \mathcal{Q}_{\mathbf{v}} + \eta \sum_{\mathbf{i} \in \Upsilon} C_{\mathbf{v},\mathbf{i}} \left[\mathcal{A}'_{\mathbf{i}} - \sigma \left(\sum_{\mathbf{r} \in \rho} \mathcal{Q}_{\mathbf{r}}\ C_{\mathbf{r},\mathbf{i}} + \Delta_{\mathbf{i}} \right) \right], \tag{40}$$

where $\eta > 0$.

7 Experiments

In this section, we compare the proposed generalized multilinear model for principal component analysis of binary tensors (GML-PCA) with multilinear principal component analysis of real-valued tensors (TensorLSI) [4] in two different scenarios. First, we evaluate the ability of the models to compress (and reconstruct) synthetic binary data tensors in a controlled set of experiments. Second, we illustrate our method on a real set of biological DNA sequences (represented as binary tensors) originating from different functional regions of genomic sequences.

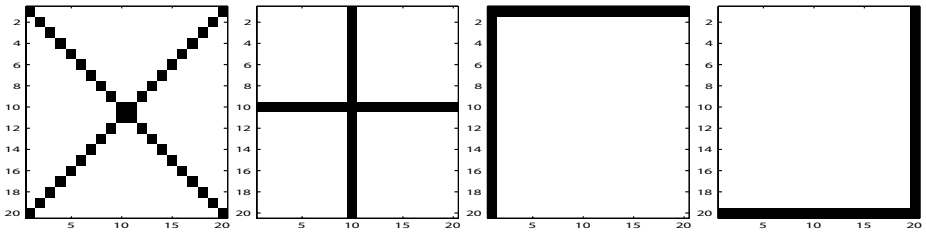

Figure 1: An example of basis tensors spanning a 4-dimensional Bernoulli natural parameter space.

7.1 Synthetic data

In order to evaluate the ability of the models to find a compact data representation and reconstruct the compressed data, we generated several datasets of high-dimensional binary tensors from underlying subspaces of the natural parameter space. Below, we describe generation of the synthetic binary tensors, give an overall outline of the experiments and summarize the results.

7.1.1 Generating Process

Our goal is to generate a set of M binary tensors $\mathcal{D} = \{\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_M\}$, where each element $\mathcal{A}_{m,\mathbf{i}}$ of $\mathcal{A}_m \in \{0,1\}^{I_1,I_2,...,I_N}$ is independently Bernoulli distributed with natural parameter $\theta_{m,\mathbf{i}}$ and the natural parameter tensor θ_m lies in a sub-space spanned by a set of linearly independent basis tensors $\{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_R\}$, $\mathcal{B}_r \in \{-1,1\}^{I_1 \times I_2 \times ... \times I_N}$, $r = 1, 2, ..., R$. Given such basis, each synthetic binary tensor $\mathcal{A}_m$ is generated as follows: First, a tensor θ_m containing Bernoulli natural parameters is constructed as a (random) linear combination of bases,

$$\theta_m = \sum_{r=1}^{R} \alpha_{mr} \mathcal{B}_r, \tag{41}$$

where the elements $\alpha_{m_r} \in \mathbb{R}$ of the mixing vector $\boldsymbol{\alpha}_m = (\alpha_{m_1}, \alpha_{m_2}, ..., \alpha_{m_R})$, are sampled from a uniform distribution over a given support. The elements $\mathcal{A}_{m,\mathbf{i}}$ of the binary data tensor $\mathcal{A}_m$ are then sampled from the Bernoulli distribution parametrized by $\theta_{m,\mathbf{i}}$ (see (4) and (5)):

$$\mathcal{A}_{m,\mathbf{i}} \sim P(\mathcal{A}_{m,\mathbf{i}} | \theta_{m,\mathbf{i}}) = \sigma(\theta_{m,\mathbf{i}})^{\mathcal{A}_{m,\mathbf{i}}} \cdot \sigma(-\theta_{m,\mathbf{i}})^{1-\mathcal{A}_{m,\mathbf{i}}}. \tag{42}$$

To illustrate that it is non-trivial to discern the underlying natural parameter subspace from the sample binary tensors, we randomly sampled and visualized 5 data tensors from Bernoulli natural parameter space spanned by bases shown in figure 1. The binary tensors are shown in figure 2.

7.1.2 Outline of the Experiments

In the next section we will use tensor decomposition to analyze a large-scale set of biological sequences represented through sparse second order binary tensors ($N = 2$) of sizes around $(I_1, I_2) = (250, 30)$. In this section we verify our method in a set of controlled experiments employing synthetically generated 2nd-order binary tensors of the same size $(250, 30)$. For the first experiment we generated 10 data sets, each containing $M = 10,000$ binary

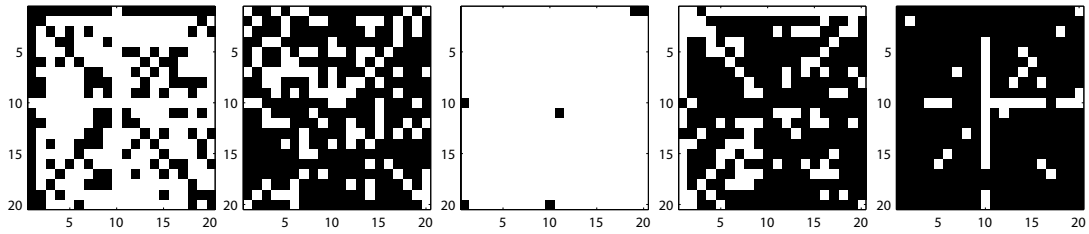

Figure 2: A sample of randomly generated binary tensors from the Bernoulli natural parameter space spanned by the bases shown in figure 1.

tensors, from a Bernoulli natural parameter spaces spanned by 10 linearly independent basis tensors. Each data set was sampled from a different natural parameter subspace. In the second experiment the setting remains the same, but the dimensionality of the underlying Bernoulli natural parameter space was increased from 10 to 30 (30 linearly independent basis tensors were employed). From each data set we hold out 20% $(2,000)$ binary tensors as a test set and let the models find the latent subspace on the remaining 80% $(8,000)$ tensors (training set).

After training the models, tensors from the hold-out set were "compressed" by projecting them onto the principal subspace thus obtaining their low dimensional representations in the natural parameter space. To evaluate the amount of preserved information, the compressed representations would need to be reconstructed back into the original binary tensor space. Note that since the models we consider represent binary tensors through continuous values in the natural parameter space (GML-PCA), or in $\mathbb{R}^{I_1 \times I_2}$ (TensorLSI), a straightforward deterministic reconstruction in the binary space is not appropriate. We use the area under the ROC curve (AUC) designed to compare different real-valued predictions of binary data. One way of determining the AUC value is to calculate the normalized Wilcoxon-Mann-Whitney statistic which is equal to AUC [5]. If we identify $\{x_1, x_2 \dots x_J\}$ as the model prediction outputs for all nonzero elements of tensors from the test set, and $\{y_1, y_2 \dots y_K\}$ as outputs for all zero elements, the AUC value for that particular prediction (reconstruction) of the test set of tensors is equal to

$$\texttt{AUC} = \frac{\sum_{j=1}^{J} \sum_{k=1}^{K} C(x_j, y_k)}{J \cdot K},$$

where J and K are the total number of nonzero and zero tensor elements in the test set, respectively, and C is a scoring function

$$C(x_j, y_k) = \begin{cases} 1 & \text{if } x_j > y_k \\ 0 & \text{otherwise.} \end{cases}$$

In our GML-PCA model, the basis tensors used for decomposition are constructed as outer products of fixed sets of column and row vectors, and so the desired number of principal tensors cannot be simply set, as in TensorLSI, to an arbitrary value. Instead, the number of principal tensors is determined by the number of basis column vectors times the number of basis row vectors. For example, to compare with TensorLSI using 4 principal tensors, we choose the portion of column and row vectors that gives 4 principal tensors and achieves the highest log-likelihood on the training set of 8,000 binary tensors. Of course, we could have used a separate validation set for model selection purposes, but

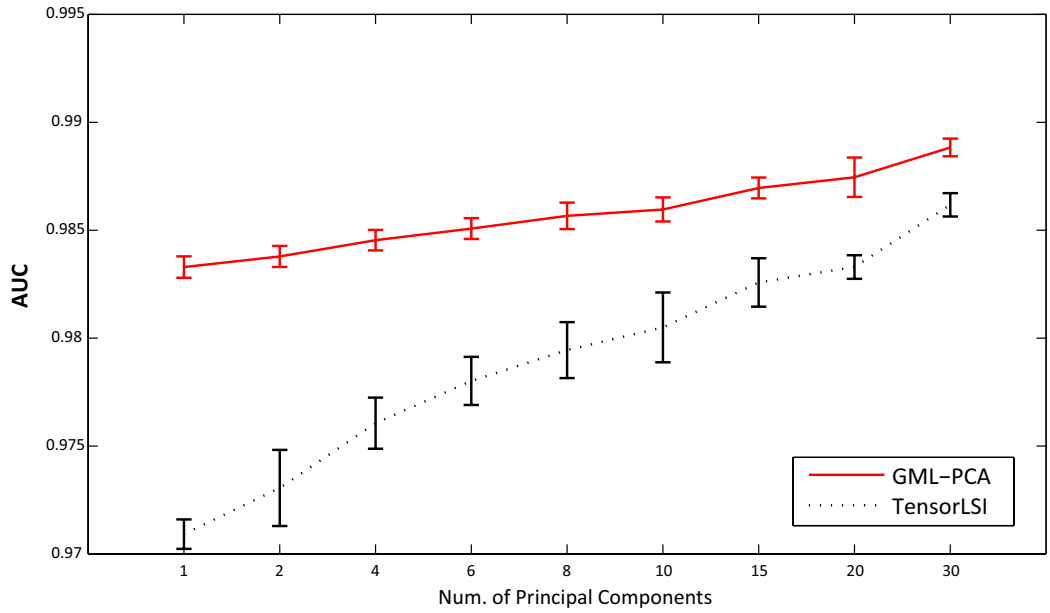

Figure 3: AUC analysis of hold-out binary tensor reconstructions obtained by the models using different number of principal components among 10 different sets of binary tensors. Each set was generated from different Bernoulli parameter space spanned by 10 linearly independent basis tensors.

with such highly constrained models (drastic dimensionality reduction), we found a close correspondence between training and hold-out set performances.

7.1.3 Results

To summarize the performance of TensorLSI and GML-PCA models to compress and subsequently reconstruct the sets of synthetic tensors, we calculate the mean and standard deviation of AUC values across the 10 test sets of binary tensors. Reconstruction results for binary tensors generated from Bernoulli natural parameter spaces spanned by 10 and 30 bases tensors are summarized in figures 3 and 4, respectively. The GML-PCA model outperforms the real-value tensor decomposition technique (TensorLSI) in both experiments. The differences between AUC values of GML-PCA and TensorLSI for 30 bases are smaller than those for 10 bases, although the differences are almost always significant.

7.2 Visualization of DNA Sequences Based on Subsequence Structure

Bioinformatics involves the development of computational tools for systematic analysis and visualization of DNA, RNA and protein sequence data. To uncover specific regulatory circuits controlling gene expression, biologists need to first confidently map broader functional regions implanted in genomic sequence, such as *promoters*. Promoters are regions upstream of a gene, used by the transcriptional machinery. The process of expressing a gene is carefully regulated by the timely binding of both general and gene-specific regulatory proteins and RNA (or complexes thereof) to its promoter. We thus expect that promoters contain regulatory binding sites (typically 5-15 nucleotide degenerate sequence patterns). This section investigates the ability of GML-PCA to recognize in an unsupervised manner promoter sequences (forming our "positive" class).

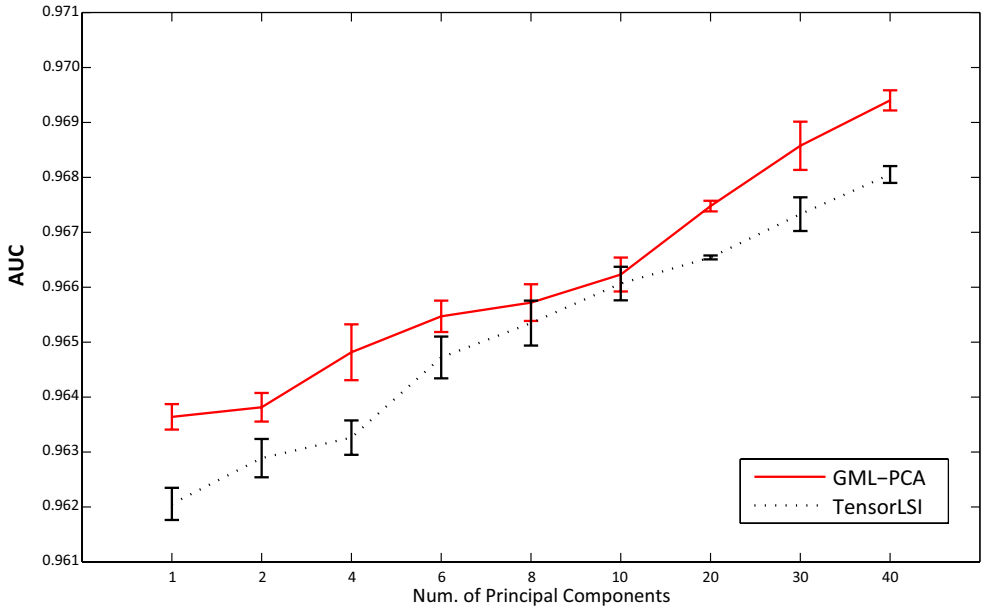

Figure 4: AUC analysis of hold-out binary tensor reconstructions obtained by the models using different number of principal components among 10 different sets of binary tensors. Each set was generated from different Bernoulli parameter space spanned by 30 linearly independent basis tensors.

A gene contains *exons* and *introns*, where exons primarily code for protein-forming amino acids. Not directly coding for amino acids, introns may contain important control signals for splicing the gene product. Intronic sequences may also contain sites to which a different category of regulatory proteins (and RNA) bind to modulate the efficiency of transcription. Many current state-of-the-art systems for analyzing and predicting promoter regions (e.g. [12, 18]) are based on the underlying principle that sequences from different functional regions differ in a local term[2] composition. Following [12, 18], we use intronic sequences as a negative set.

Based on this principle we use a suffix tree based extraction of statistically significant terms, preserving the within-class (functional regions) frequencies. Given such terms, the local term composition of a DNA sequence is obtained in the form of a binary second-order tensor (matrix), where rows represent terms, columns positions within the sequence and the binary tensor elements indicate the presence/absence of a term in the sequence at a given position.

To reveal the 'dominant trends' in a real world large-scale dataset of annotated DNA sequences, we compress the binary tensors representing the sequences into their low dimensional representations and visualize their distributions. Note that based on the underlying assumption about the differences in local term composition, we expect some separation between sequences from different functional regions of DNA, even though the decomposition/compression models themselves are fitted in a completely unsupervised manner.

[2]As a term, we denote a short and widespread sequence of nucleotides that has or may have a biological significance.

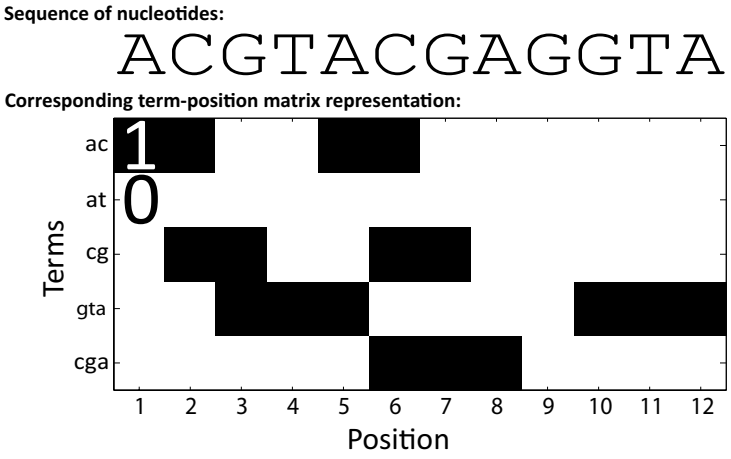

Figure 5: An illustrative example of representing a short sequence of nucleotides by a binary second-order tensor (matrix) where rows represent terms and columns positions.

7.2.1 Biological Sequence Data and its Representation

We use the dataset of promoter and intron (non-promoter) sequences employed in [12]. From the Database of Transcription Start Sites (DBTSS), version 5.2 [21], which includes 30,964 human promoter sequences, we extracted from each sequence a subsequence from 200bp upstream to 50bp downstream relative to the position of a transcription start site (TSS). Regulatory components primarily bind to the DNA relatively close to the TSS. The same number of intron sequences with length of 250bp were randomly selected from the Exon-Intron Database [16], release Sept.2005. To represent the sequences, we identify terms over the alphabet of nucleotides $\mathcal{N} \in \{a, c, g, t\}$ that are statistically significant longest words preserving the within-class frequencies. For this purpose we use a suffix tree construction. Typically, such a construction is guided by two main characteristics: **(1)** a 'significance criterion' used to decide whether to continue with expanding a particular suffix and **(2)** construction parameters guiding the suffix extension process. As the significance criterion we employ the Kullback-Leibler divergence between the promoter and intron class distributions, given by the candidate term w and its possible extension ws, $s \in \mathcal{N}$, weighted by the prior distribution of the extended term ws. The suffix tree is built in a bottom-up fashion, starting with four leaf nodes labeled with the symbols from $\mathcal{N}$. A term w is extended with a symbol s if

$$P(ws) \sum_{c=\{0,1\}} P(c|ws) \log_2 \frac{P(c|ws)}{P(c|w)} \geq \epsilon_{KL},$$

where the classes of sequences from promoter regions and intron regions are denoted by $c = 1$ and $c = 0$, respectively. Size of the suffix tree depends on values of the construction parameters $\epsilon_{KL}, \epsilon_{grow} > 0$. The parameter ϵ_{grow} represents the minimal frequency of a word in the training sequences to be considered a candidate for the tree construction. More details on general principles behind suffix tree construction can be found e.g. in [15, 20].

Besides the composition of specific terms, their position within the sequences may be an important factor (especially for promoter sequences aligned with respect to the TSS site). To capture both the term composition and position, we represent the DNA sequences as

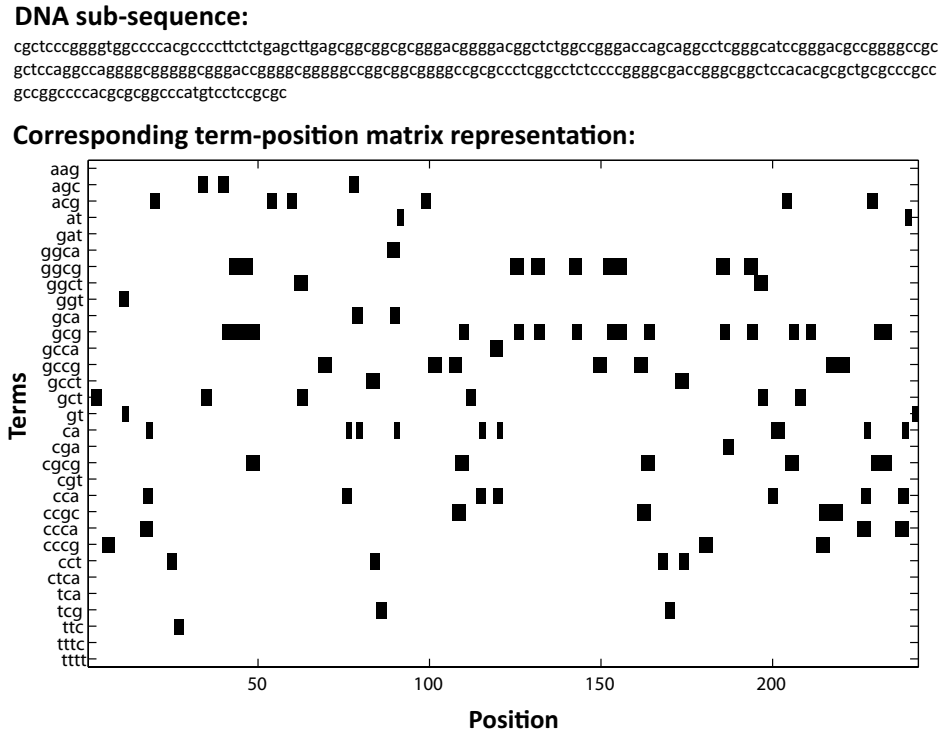

Figure 6: An example of representing a promoter DNA sub-sequence from real biological dataset by a binary second order tensor where rows represent terms and columns positions.

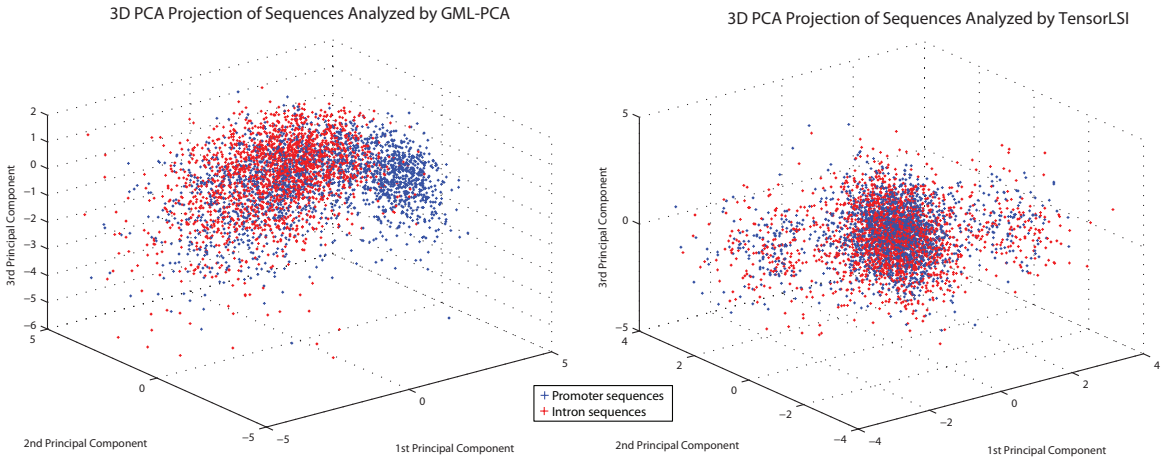

Figure 7: Thee-dimensional PCA projections of 10% randomly sampled promoter and intron sequences from the tensor space spanned by 10 basis tensors obtained by the GML-PCA (left plot) and TensorLSI (right plot) models.

binary second-order tensors $\mathcal{A}$ where rows i_1 represent terms, columns i_2 positions within the sequence, and the binary tensor element $\mathcal{A}_{i_1,i_2}$ is an indicator whether the sequence represented by $\mathcal{A}$ has a term i_1 at position i_2.

An example of a short illustrative sequence representation by a binary tensor is shown in figure 5. The tensor representation of a real promoter sequence randomly selected from the dataset is presented in figure 6.

We used the following values of the construction parameters: $\epsilon_{grow} = 5 \times 10^{-3}$ and $\epsilon_{KL} = 7 \times 10^{-5}$. This setting yields a set of 31 terms. Larger term sets (obtained using lower values of construction parameters) did not improve the separation of sequences from different function regions in the final visualizations and smaller term sets (resulting from more stringent parameter settings) made the separation weaker. Each DNA sequence was represented by a binary matrix with 250 columns and 31 rows. We compressed the binary matrices via TensorLSI decomposition using 10 principal tensors. To get a corresponding setting for GML-PCA, we decomposed the data with 10 principal tensors obtained as outer products of 5 column and 2 row basis vectors. In general, for both models, employing a higher number of basis tensors did not improve the separation of promoter sequences from the introns.

7.2.2 Visualizations

Both decomposition methods represent the sequential data as 10-dimensional vectors of expansion coefficients. To visualize the distribution of such representations, we used principal component analysis (PCA) and projected the real-valued 10-dimensional expansion vectors onto the three-dimensional space defined by the three leading principal vectors. Visualizations of promoter and intron sequences decomposed via GML-PCA and TensorLSI are shown in figure 7 left and right, respectively. Promoter sequences are illustrated in the plots by blue and intron sequences by red dots. Based on the TensorLSI plot, the compressed expansion vectors completely fail to discriminate between the promoter and intron classes. However, with the same term-position representations, GML-PCA clearly separates a large subset of promoters from introns.

For a more involved analysis, we project the 10-dimensional expansion coefficients of the GML-PCA model onto the leading two-dimensional principal subspace (see figure 8). Detailed analysis of the individual sequences (not reported here) reveals that the rightmost region, populated primarily by promoter sequences, have frequent occurrences of terms GGCG, GCG, CGCG and CCGC. This indicates a high concentration of di-nucleotides CG around the TSS. These so-called CpG islands are known to be associated with functional promoter regions–approximately 60% of mammalian genes [6]. More specific signals are found directly at the TSS (the band at "200"), which is known to be a key site for the RNA polymerase transcriptional machinery. GT and AG are known signals for splicing and are thus expected to occur in introns. Indeed, these words tend to occur predominately in the intron-rich regions in figure 8.

For a deeper analysis of the composition difference between promoter and intron sequences, a user interaction can be integrated into the visualization to select and visualize the term composition of interesting individual sequences. An illustrative visualization of manually selected sequences is shown in figure 8 where matrices for promoters and introns are denoted by letters P and I, respectively. Based on our previous analysis, we highlighted important terms that have a strong influence on the sequence position in the visualization space (marked with black dots). For illustration purposes, we selected two pairs of 'close' promoter sequences: the pair (P-3,P-4) is more separated from the introns than the pair (P-1,P-2). Based on the term compositions, promoter sequences P-3 and P-4 have higher occurrences of terms GGCG and GCG (regarded as a strong signal of promoter sequence by the model). A general topographic organization of the visualization plot is clearly visible, with 'close' sequences representations on the plot having 'similar' term composition structure (e.g three intron sequences I-1, I-2 and I-3, and related promoter structure in P-5).

7.2.3 Functional enrichment analysis of promoter sequences

The DNA-binding sites of transcription factors, are often characterized as relatively short (5-15 nucleotides) and degenerate sequence patterns. They may occur multiple times in promoters of the genes the expression of which they modulate. To further validate that GML-PCA indeed picks up biologically meaningful patterns, we searched the compressed feature space of promoters for biologically relevant structure (including that left by transcription factors). Genes that are transcribed by the same factors are often functionally similar [3]. Carrying specific biologically relevant features, suitable representations of promoters should correlate with the roles assigned to their genes. If the projection to a compressed space highlights such features, it is testament to a method's utility for processing biological sequences.

The Gene Ontology (GO; [1]) provides a controlled vocabulary for the annotation of genes, broadly categorized into terms for cellular component, biological process and molecular function. In an attempt to assign biologically meaningful labels to promoters, all sequences were mapped to gene identifiers. In cases of multiple promoters for the same identifier, we picked one sequence randomly. In cases of multiple gene identifiers for the same promoter sequence, we picked the identifier with the greatest number of annotations. Using the Gene Ontology (June 2009), we could thus assign zero or more GO terms to each promoter sequence. In total there are 8051 unique GO terms annotating 14619 promoters.

Recall that in this experiment GML-PCA decomposes binary tensors into a (linear) combination of 10 basis tensors in the Bernoulli natural parameter space. Each pro-

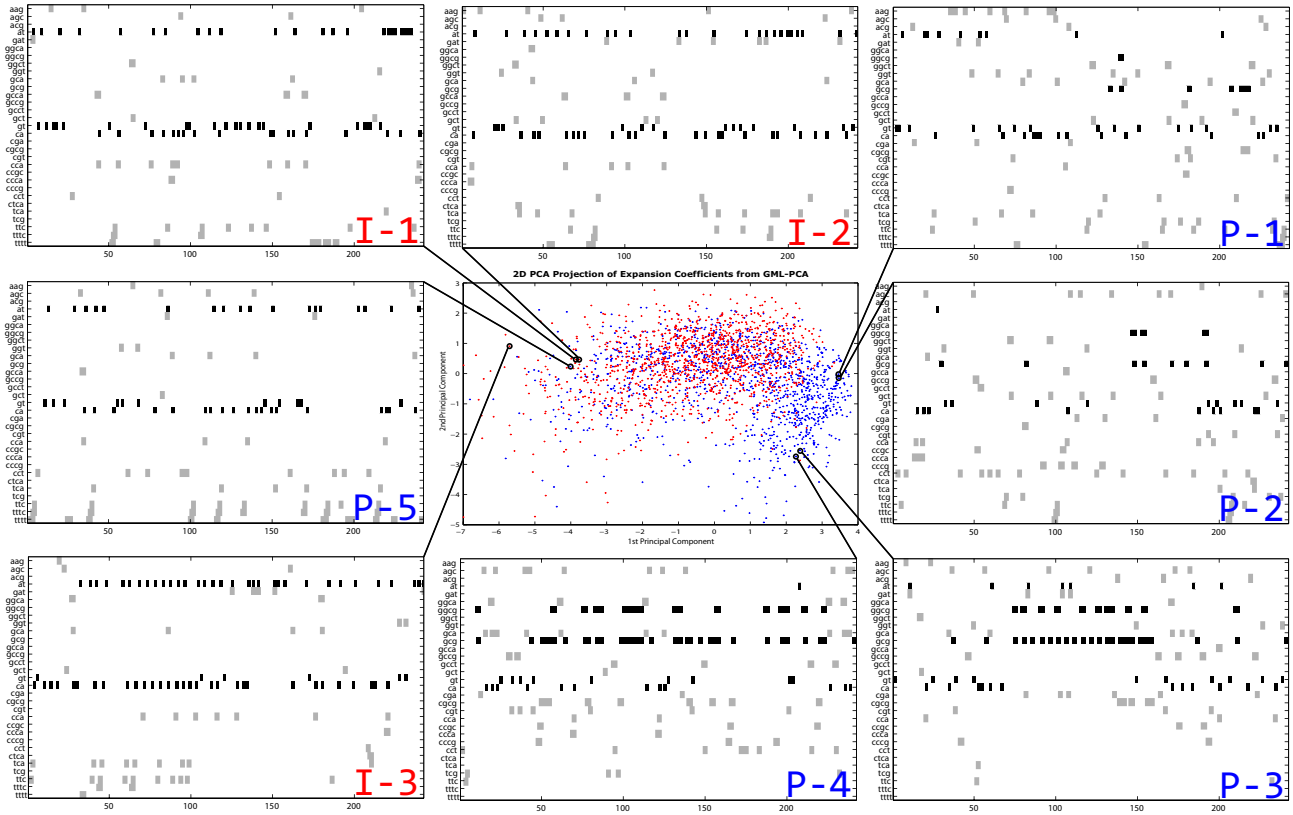

Figure 8: Detailed visualization of second-order binary tensors for manually selected promoter and intron sequences. Important terms that have a strong influence on the sequence coordinates in the central 2D plot are marked with black dots.

moter sequence from the dataset can thus be represented by a 10-dimensional expansion coefficient vector. For visualization purposes, standard PCA is then used to project the expansion vectors into a 2- and 3- dimensional space (selected to have the highest principal values). To evaluate whether promoters deemed similar by GML-PCA are also functionally similar, we need first to design a methodology for calculating the 'distance' between each pair of promoters. Naively, one may be tempted to use the usual Euclidean distance in the 10-dimensional coordinate space of natural parameters. However, this is not correct, since **(1)** the basis tensors are not orthogonal; **(2)** they span a space of Bernoulli natural parameters that have a nonlinear relationship with the data values. To determinate the model-based 'distance' between two promoter sequences m and l in a principled manner, we propose to calculate a sum of average symmetrized Kullback-Leibler divergences between noise distributions for all corresponding tensor elements $\mathbf{i} \in \Upsilon$:

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

where KL divergence between two Bernoulli distributions defined by their means (see (4)) is equal to

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

The following test suite aims to quantify if the compressed promoter representations are biologically meaningful. In each test, we select one promoter as a reference. The test is repeated until all promoters have been selected. Given a reference promoter m, we label the group of all promoters l within a pre-specified distance $\mathrm{D}(m,l) < \mathrm{D}_0$ as "positives" and all others as "negatives". Hence, the positive set of the reference promoter m reads:

$$S_m = \{l \mid \mathrm{D}(m,l) < \mathrm{D}_0\}$$

In the tests we consistently use a distance of $\mathrm{D}_0 = 25$, usually rendering over one hundred "positives". For each GO term (ultimately, in the full Gene Ontology, but in practice, we look only at those assigned to the reference promoter), Fisher's exact test resolves if it occurs more often amongst "positives" than would be expected by chance. (The null hypothesis is that the GO term is not attributed more often than by chance to the "positives".) A small p-value indicates that the term is "enriched" at the position of the reference promoter m. We adjust for multiple hypothesis testing and set the threshold at which to report a term as significant accordingly ($p < 5{\cdot}10^{-7}$). To understand the tendency of false discovery, we also repeated the tests (with the same significance threshold) after shuffling the points assigned to promoters. Re-assuringly, in no case did this permutation test identify a single GO term as significant.

In total, at the aforementioned level of significance, we found 75 GO terms that were enriched around one or more reference promoters. The observation that a subset of promoter sequences are functionally organized after decomposition into 10 basis tensors adds support to the methods' ability to detect variation at an information-rich level. More specifically, we find a number of terms that are specifically concerned with chromatin structure (that packages the DNA), e.g. GO:000786 "Nucleosome", GO:0006333 "Chromatin assembly or disassembly" and GO:0065004 "Protein-DNA complex assembly". Interestingly, we found several enriched terms related to development, e.g. GO:0022414

"Reproductive process" and GO:0007565 "Female pregnancy". Anecdotally, we note that CpG islands (that are clearly distinct in the promoter sequence data) are associated with open DNA, leading to constitutive gene expression. Speculatively, genes associated with CpG-rich promoters need to control local chromatin. Moreover, it was recently observed that under-methylation of such otherwise methylation-prone regions is established by developmental cues [19], suggesting a link between CpG islands and development.

We use the following method to visualize the grouping of promoters assigned the same GO term t. Given a promoter m, we consider two events: e_t - a promoter assigned the GO term t; $e_{\neq t}$ - the complement of e_t. The probability $P_t(e_t|S_m)$ is determined by calculating the proportion of promoters with the GO term t in S_m. We do this by first normalizing the counts so that *a priori* (across all promoters) the probability of drawing a positive (a promoter assigned the GO term t) and drawing a negative (a promoter not assigned that GO term) is equal.

By performing the procedure above, we have for each promoter m and for each GO term t assigned to it a value $P_t(e_t|S_m)$ that expresses how organized the space around m is with respect to the GO term t. Figure 9 shows a contour plot of $P_t(e_t|S)$ that interpolates the values $P_t(e_t|S_m)$ calculated for all promoters m with a GO term t=GO:0000003 "Reproduction" when mapped to the PCA-derived 2-dimensional space. Dark regions indicate high level of enrichment. The star marker highlights the promoter NM-002784 with the highest level of enrichment for the GO term GO:0000003. The plot reveals islands of highly enriched promoter regions related to a particular biological function.

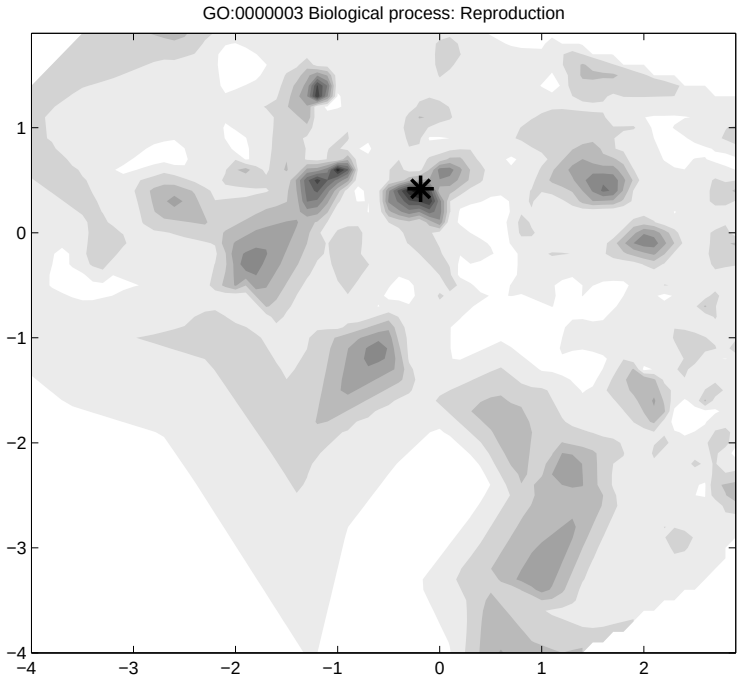

Figure 9: Promoter regions assigned to GO:0000003 biological process: Reproduction.

To further investigate the enrichment of biologically meaningful information we performed a motif discovery on the sets S_m. As an example, consider a reference promoter

NM-002784 (highly enriched for GO:0000003). S_m contains 38 promoter sequences that were passed to MEME [2]. Several statistically significant motifs were found. The most significant 12-nucleotide motif was then passed to Tomtom [8] that (reassuringly) recognized the motif as the binding site of transcription factor Sfpi1 ($p = 0.0003$, $q = 0.23$, UniPROBE database). We stress that the main purpose of this experiment was to illustrate the potential of GML-PCA to take stock of more complicated patterns than zero-order compositional biases. A more involved application of our binary tensor decomposition in specific analysis of biological sequences (genomic or aminoacid) is a matter for our future research.

8 Conclusion

Current data processing tasks often involve manipulation of binary tensors. However, a principled systematic framework for decomposition of binary tensors has been missing. We closed this gap by introducing a generalized multilinear model for principal component analysis of binary tensors - GML-PCA. In the model formulation, to account for a binary nature of the data, each tensor element is modeled by a Bernoulli noise distribution and constrained to lie in a subspace spanned by a reduced set of basis (principal) tensors. We derived a simple closed form iterative scheme for parameter estimation.

Experiments with synthetic data generated from linear Bernoulli natural parameter sub-spaces showed that GML-PCA is better suited for compression and reconstruction of binary tensors than multilinear principal component analysis (TensorLSI) designed for real valued tensors. We have also investigated the ability of GML-PCA and TensorLSI models for unsupervised analysis of DNA sub-sequences (represented as binary tensors) from different functional regions based on the local term composition. By visualizing sub-sequence distributions in the principal sub-spaces spanned by the basis tensors, it transpires that the TensorLSI model could not reveal any discriminatory trends, while the GML-PCA could. After detailed analysis, the discriminatory trends were identified as one of the most known signals in the promoter analysis domain verified by in-vivo biological experiments. To further investigate the method's utility for processing biological sequences, we searched the compressed feature space of promoters for biologically relevant structure. After assigning biologically meaningful labels to analyzed promoters, we found 75 GO terms that were enriched around one or more promoters. The observation that a subset of promoter sequences are functionally organized adds support to the method's ability to detect variation at an information-rich level.

Acknowledgment

Jakub Mažgut was supported by the Scientific Grant Agency of Republic of Slovakia, grant No. VEGA 1/0508/09. Peter Tiňo was supported by the DfES UK/Hong Kong Fellowship for Excellence. Mikael Bodén was supported by the ARC Centre of Excellence in Bioinformatics and the 2009 University of Birmingham Ramsay Research Scholarship Award. Hong Yan was supported bya grant from the Hong Kong Research Grant Council (Project CITYU123809).

References

[1] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet*, 25(1):25–29, May 2000.

[2] T. L. Bailey, M. Boden, F. A. Buske, M. Frith, C. E. Grant, L. Clementi, J. Ren, W. W. Li, and W. S. Noble. Meme suite: tools for motif discovery and searching. *Nucleic acids research*, 37(Web Server issue):W202–208, July 2009.

[3] M. Bodén and T. L. Bailey. Associating transcription factor-binding site motifs with target GO terms and target genes. *Nucleic Acids Res*, 36(12):4108–4117, Jul 2008.

[4] D. Cai, X. He, and J. Han. Tensor space model for document analysis. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 625–626, New York, NY, USA, 2006. ACM.

[5] C. Cortes and M. Mohri. Auc optimization vs. error rate minimization. In *Advances in Neural Information Processing Systems*. MIT Press, 2003.

[6] S.H. Cross, V.H. Clark, and A.P. Bird. Isolation of CpG islands from large genomic clones. *Nucl. Acids Res.*, 27(10):2099–2107, 1999.

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

[8] S. Gupta, J. Stamatoyannopoulos, T. Bailey, and W. Noble. Quantifying similarity between motifs. *Genome Biology*, 8(2):R24, 2007.

[9] L. Haiping, K.N. Plataniotis, and A.N. Venetsanopoulos. Multilinear principal component analysis of tensor objects for recognition. *Pattern Recognition, International Conference on*, 2:776–779, 2006.

[10] K. Jia and S. Gong. Multi-modal tensor face for simultaneous super-resolution and recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1683–1690 Vol. 2, Oct. 2005.

[11] E. Kofidis and P. A. Regalia. On the best rank-1 approximation of higher-order supersymmetric tensors. *SIAM J. Matrix Anal. Appl.*, 23(3):863–884, 2001.

[12] X. Li, J. Zeng, and H. Yan. Pca-hpr: A principle component analysis model for human promoter recognition. *Bioinformation*, 2(9):373–378, 2008.

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

[14] N. Renard and S. Bourennane. An ica-based multilinear algebra tools for dimensionality reduction in hyperspectral imagery. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal, ICASSP 2008*, volume 4, pages 1345–1348, April 2008.

[15] D. Ron, Y. Singer, and N. Tishby. The power of amnesia: learning probabilistic automata with variable memory length. *Mach. Learn.*, 25(2-3):117–149, 1996.

[16] S. Saxonov, I. Daizadeh, A. Fedorov, and W. Gilbert. Eid: The exonintron databasean exhaustive database of protein-coding intron-containing genes. *Nucleic Acids Research*, 28(1):185–190, 2000.

[17] A. Schein, L. Saul, and L. Ungar. A generalized linear model for principal component analysis of binary data. In *Proceedings of the 9 th International Workshop on Artificial Intelligence and Statistics*, 2003.

[18] S. Sonnenburg, A. Zien, P. Philips, and G. Ratsch. Poims: positional oligomer importance matrices–understanding support vector machine-based signal detectors. *Bioinformatics*, 24(13):i6–14, July 2008.

[19] R. Straussman, D. Nejman, D. Roberts, I. Steinfeld, B. Blum, N. Benvenisty, I. Simon, Z. Yakhini, and H. Cedar. Developmental programming of CpG island methylation profiles in the human genome. *Nat Struct Mol Biol*, 16(5):564–571, 2009.

[20] P. Tino and G. Dorffner. Predicting the future of discrete sequences from fractal representations of the past. *Machine Learning*, 45(2):187–217, 2001.

[21] H. Wakaguri, R. Yamashita, Suzuki Y., S. Sugano, and K. Nakai. Dbtss: database of transcription start sites. *Nucleic Acids Research*, 36(Database-Issue):97–101, 2008.

[22] H. Wang and N. Ahuja. Compact representation of multidimensional data using tensor rank-one decomposition. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 1*, pages 44–47, Washington, DC, USA, 2004. IEEE Computer Society.