tensorBF: an R package for Bayesian tensor factorization

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Results We present the R package *tensorBF*, which is the first R package providing Bayesian factorization of a tensor. Our package implements a generative model that automatically identifies the number of factors needed to explain the tensor, overcoming a key limitation of traditional tensor factorizations. We also recommend best practices when using tensor factorizations for both, explorative and predictive analysis with an example application on drug response dataset. The package also implements tools related to the normalization of data, informative noise priors and visualization. **Conclusions** The *tensorBF* package allows Bayesian factorization of tensor datasets in the R statistical environment and is made freely available at `https://cran.r-project.org/package=tensorBF`.

1 Introduction

A key question that tensor factorization can answer is, which parts of the drug-responses are specific to a particular cancer and which are common across various cancers. Elucidating such effects can generate hypothesis on personalised therapies, as well as increase understanding on drug action mechanisms.

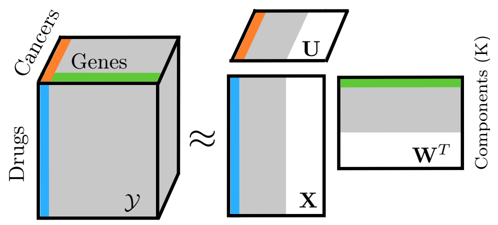

Figure 1: Illustration of tensor factorisation. The tensor $\mathcal{Y}$ can be factorized into a low-dimensional component space $\mathbf{X}$,$\mathbf{W}$ and $\mathbf{U}$ which represents relationships across the drugs, genes and cancers. tensorBF automatically prunes out excessive components (shaded white in component matrices).

Fig 1 presents the well-known trilinear CP factorization of a tensor. The CP (Canonical Decomposition / Parafac) factorizes a tensor into a sum of rank-one tensors, each of which can be represented as latent variables (factors or components) in all modes [Carroll and Chang, 1970, Harshman, 1970]. For the tensor $\mathcal{Y} \in \mathbb{R}^{N \times D \times L}$, CP identifies the latent variables $\mathbf{X} \in \mathbb{R}^{N \times K}$, $\mathbf{W} \in \mathbb{R}^{D \times K}$, and $\mathbf{U} \in \mathbb{R}^{L \times K}$ as

$$\mathcal{Y} \approx \sum_{k=1}^{K} \mathbf{X}_k \circ \mathbf{W}_k \circ \mathbf{U}_k. \tag{1}$$

While several factorization methods exist for tensors, like the Tucker model [Tucker, 1966], CP factorization is easier to interpret making it a promising choice for many biological applications. Recently, Bayesian tensor factorizations have been demonstrated to overcome some of the limitations including automatic determination of the number of components [Khan and Kaski, 2014, Hore et al., 2016], however, R package for Bayesian factorization of a tensor do not exist.

We present tensorBF, an R package to analyze natural tensor structures in the data. The package implements the Bayesian CP factorization of a tensor to infer latent factors (components) that are not obvious from the data itself. Additionally, it provides tools for analyzing the components and relationships between the variables.

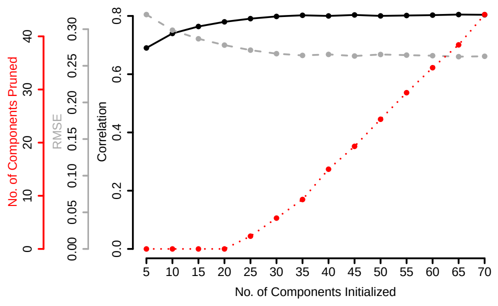

Figure 2: Illustrating component selection with tensorBF on CMAP data set. The plot shows on the y-axis, the Pearson Correlation, Root Mean Squared Error (RMSE), and the no. of components pruned (in red) on the missing values prediction task, as a function of the number of components K used to initialize the model (the x-axis).

Implementation

Our package tensorBF implements the Bayesian formulation of the tensor factorization problem of Eqn (1), by assuming normal distribution with conjugate priors. A sparsity parameter is introduced that shuts down excessive components by setting them to zero (white in Fig 1), making it possible for the model to learn the true number of components automatically. Besides, the package implements feature-level sparsity for the latent variable matrices. Supplementary File 1 provides the details of the modeling assumptions and inference using Gibbs sampling.

Given tensor $\mathcal{Y} \in \mathbb{R}^{N\times D\times L}$, the package tensorBF implements a Bayesian formulation of the tensor factorization problem. Our package implements the method assuming CP factorisation (CANDECOMP/ PARAFAC, by Carroll and Chang [1970], and Harshman [1970]) for a three-mode tensor into corresponding latent variables $\mathbf{X} \in \mathbb{R}^{N\times K}$, $\mathbf{W} \in \mathbb{R}^{D\times K}$ and $\mathbf{U} \in \mathbb{R}^{L\times K}$. The CP factorisation is represented as:

$$\mathcal{Y} = \sum_{k=1}^{K} \mathbf{X}_k \circ \mathbf{W}_k \circ \mathbf{U}_k + \boldsymbol{\epsilon}.$$

where $\boldsymbol{\epsilon} \in \mathbb{R}^{N\times D\times L}$ is a noise tensor.

The model tensorBF assumes the following distributional assumptions:

$$\begin{aligned}
y_{n,d,l} &\sim \mathcal{N}(z_k.\mathbf{x}_n^T.(\mathbf{w}_d * \mathbf{u}_l), \tau^{-1}) \\
x_{n,k} &\sim \mathcal{N}(0, (\lambda^x_{n,k})^{-1}) \\
u_{l,k} &\sim \mathcal{N}(0, (\lambda^u_{l,k})^{-1}) \\
w_{d,k} &\sim \mathcal{N}(0, (\lambda^w_{d,k})^{-1}) \\
z_k &\sim Bernoulli(\pi_k), \\
\pi_k &\sim Beta(a^\pi, b^\pi) \\
\tau &\sim Gamma(a^\tau, b^\tau)
\end{aligned}$$

$$\lambda^{x}_{n,k} \sim \begin{cases} 1, & \text{dense,} \\ Gamma(a^{\alpha}, b^{\alpha}), & \text{sparse.} \end{cases}$$

$$\lambda^{w}_{d,k} \sim \begin{cases} 1, & \text{dense,} \\ Gamma(a^{\alpha}, b^{\alpha}), & \text{sparse.} \end{cases}$$

$$\lambda^{u}_{l,k} \sim \begin{cases} 1, & \text{dense,} \\ Gamma(a^{\alpha}, b^{\alpha}), & \text{sparse.} \end{cases}$$

where $*$ is an element-wise vector product, τ is the noise precision, and $Gamma(a, b)$ is the Gamma distribution with a shape a and a rate b.

The z_k variables encode the automatic component selection and control the total number of non-zero components in the model. The binary values in z_k switch the component k on or off. If $z_k = 0$, all values in $\mathbf{w}_k$ become zero effectively pruning the component; when $z_k = 1$, values in $\mathbf{w}_k$ are sampled from a normal distribution yielding non-zero values that capture meaningful variation in the data. This is achieved through the Beta-Bernoulli construct.

The package provides several practically useful choices for the modeling assumptions, especially when the data modes are imbalanced, i.e. "small n and large p", or data contains heavily noised measurements, both of which occur commonly in many bioinformatics datasets.

As one key characteristic, the package makes it possible to choose dense or sparse priors for each of the loading matrices, based on application scenario. It is recommended to use sparse settings on the mode with large dimensions or when there is a prior belief in the sparseness of the structure. These parameters can be selected using `ARDX` `ARDW`, and `ARDU` logical parameters in the `getDefaultOpts()` function.

The inference of the model is performed via Gibbs sampling. The package provides options for varying the burnin, sampling and thinning iterations with default recommended values based on application on real data sets. The computational complexity of the model is linear in the number of dimensions and cubic only in the number of components K, where K is generally much smaller than the data dimensionality, making it feasible for K to the tune of a few hundreds.

Results and Discussion

Model Inference and Initialization

The factorization of a 3-mode tensor $\mathcal{Y}$ can be inferred using `model <- tensorBF(Y)`, with the default options. Depending on the modeling assumptions and application setting, the function can take a variety of parameter choices as inputs. For instance, the number of components to initialize the model, how to normalize the data and an informative noise prior, that is, a user's belief on how much of the data variance should be explained with the components. A full description of the possible options is given in the functions `getDefaultOpts()` and `tensorBF()` documentation. The tensor can be normalized over different modes and ways, using `norm.fibercentering()` and `norm.slabscaling()`. If the features in a particular mode are deemed equally important, they should be scaled. However, if the variance is a proxy for the feature's importance, scaling should not be done. The package manual contains simplified examples and `demo()`, demonstrating the usage of the functions on simulated data. The methods computational complexity is linear in the data dimensions and cubic only in K. The package took $\sim$1 hour for a single chain on the CMap data.

Missing Values Prediction

The package can handle missing values by simply including them as `NA`s. The model parameters are sampled based on the observed data only, and `predictTensorBF()` predicts the missing values.

Component Selection

The tensorBF package infers the number of components automatically. In practice, this is achieved by initializing the model with a high number of components K (default choice: 20% of the sum of lower two modes) and the method prunes any excessive components. The `noiseProp` in `tensorBF()` defines the proportion of variance that is expected to be explained with the components. In case, the data is expected to be heavily noisy, as with many real datasets, experimenting different choices of `noiseProp` will aid in component selection.

We explain component selection practice with a real tensor dataset of Fig 1. Fig 2 plots the methods behaviour as a function of an increasing number of initial K. The key observation here is that the performance improves until $K \leq 30$, after which it stabilizes to the best result. Around the same mark, the model starts to prune all the excessive components indicating that it has already explained the data sufficiently. Therefore, in practice, we suggest to initialize K to a higher enough value and let the model choose the component number automatically. An appropriate K can be identified as one that prunes at least several excessive components.

Analysis and Visualizations

1.1 Connectivity Map dataset

The Connectivity Map (a.k.a CMap) dataset [Lamb et al., 2006] contains post-treatment gene expression responses of a large set of drugs on three cancer cell lines, namely HL60 (Blood), MCF7 (Breast) and PC3 (Prostate). We used post-treatment differential gene expression responses of $N = 78$ drugs over $D = 1106$ genes as measured over the $L = 3$ cancer lines as a data tensor. We chose only a subset of drugs and genes from the Connectivity Map dataset for demonstration purposes. We processed the data such that the gene expression values represent up (positive) or down (negative) regulation from the untreated (base) level.

1.2 Experimental Setup

We adopted a robust setting for the demonstration of the component selection procedure. Specifically, we repeated each setting ten times, computing the average prediction performance. In each iteration, 5% random missing values were introduced in the tensor and prediction performance was computed over them.

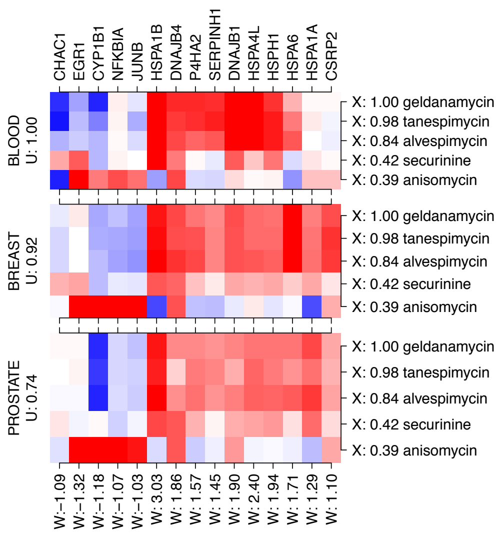

Figure 3: A component showing the relationship between the latent variables $\mathbf{X}$, $\mathbf{W}$ and $\mathbf{U}$ plotted using the function *plotTensorBF()*.

The factorization explains relationships between the variables through K components. The components can be visualized using `plotTensorBF()`. An example of such visualization is shown in Fig 3. The values of the latent variable $\mathbf{X}$ indicate that the response is primarily driven by the top 3 drugs in several HSP genes $\mathbf{W}$. High latent scores in $\mathbf{U}$ show that this response is common across all three cancers, and can, therefore, be interpreted as a Heat Shock Protein response of HSP90 inhibitors in all three cancers.

Conclusions

The tensorBF package factorizes a tensor into low-dimensional latent factors, inferring meaningful relationships. The package provides essential tools ranging from normalization to automatic component selection, and from setting informative noise prior to interpreting the factorization. The package is a new contribution in the data analysis domain focusing on tensors with a fully Bayesian treatment of the latent factors.

Availability and Requirements

The tensorBF package is available at CRAN - a global repository of R packages `https://cran.r-project.org/package=tensorBF`. The R package tensor is required for installation of tensorBF.

References

J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition. *Psychometrika*, 35(3):283–319, 1970.

Richard A Harshman. Foundations of the parafac procedure: models and conditions for an explanatory multimodal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.

Victoria Hore, Ana Viñuela, Alfonso Buil, Julian Knight, Mark I McCarthy, Kerrin Small, and Jonathan Marchini. Tensor decomposition for multiple-tissue gene expression experiments. *Nature Genetics*, 48(9):1094–1100, 2016.

Suleiman A Khan and Samuel Kaski. Bayesian multi-view tensor factorization. In Toon Calders et al., editors, *Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2014*, pages 656–671. Springer Berlin Heidelberg, 2014.

Justin Lamb et al. The connectivity map: Using gene-expression signatures to connect small molecules, genes, and disease. *Science*, 313(5795):1929–1935, 2006. doi: 10.1126/science.1132939.

Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.