Schedae Informaticae Vol. 24 (2015): 103–112
doi: 10.4467/20838476SI.15.010.3032

Multilinear Filtering Based on a Hierarchical Structure of Covariance Matrices

Andrzej Szwabe, Paweł Misiorek, Michał Ciesielczyk
Institute of Control and Information Engineering
Poznań University of Technology
ul. M. Skłodowskiej-Curie 5, 60-965 Poznań, Poland
e-mail: *firstname.lastname@put.poznan.pl*

Abstract. We propose a novel model of multilinear filtering based on a hierarchical structure of covariance matrices – each matrix being extracted from the input tensor in accordance to a specific set-theoretic model of data generalization, such as derivation of expectation values. The experimental analysis results presented in this paper confirm that the investigated approaches to tensor-based data representation and processing outperform the standard collaborative filtering approach in the 'cold-start' personalized recommendation scenario (of very sparse input data). Furthermore, it has been shown that the proposed method is superior to standard tensor-based frameworks such as N-way Random Indexing (NRI) and Higher-Order Singular Value Decomposition (HOSVD) in terms of both the AUROC measure and computation time.

Keywords: tensor-based data modeling, multilinear PCA, random indexing, dimensionality reduction, multilinear data filtering, higher-order SVD.

1. Introduction

Tensor product, as enabling highly-expressive multi-modal data representations, has been investigated by many authors as the basic component of algebraic structures suitable for heterogeneous and multi-relational data representation [1–4]. Some of these authors have pointed out the fundamental relation between the order of a tensor and the 'structuredness' of data representation (originally having the form of a tuple set) – the property that explains why the tensor product is the core of 'the logic of multilinear data representation' [1].

On the other hand, the recent progress in tensor data processing [4] motivates work on solutions that enable estimation of probability assigned to any tuple that may be read from a multidimensional array (i.e., in a tensor) [3]. Obviously enough, to make a tensor prediction-providing, not just storage-providing, some form of filtering of the tensor's fibres must be performed [2, 5].

As we show in this paper, such a filtering should reflect variables' covariances visible in all the tensors that are flattened versions of the highest-order input tensor, not only these visible in the highest-order input tensor itself. To address this requirement, we propose a hierarchical approach to multilinear data modelling as means for robust, non-arbitral ('variational') fusion of heterogeneous data. It is worth stressing that the use of a hierarchy of differently flattened tensors is a direct consequence of the assumption of using multiple 'interpretations' of each tuple for construction of multiple alternative input tensors.

2. Related work

In the context of multilinear data processing, the most widely known form of per-mode tensor filtering is the projection of fibres laying along the given tensor mode into a subspace spanned by the modes' principal components – the projection being the 'workhorse' of filtering based on Higher-Order Singular Value Decomposition (HOSVD) [4,6] and Multilinear Principal Component Analysis [2]. However, so far no theoretical basis for optimality of the multilinear dimensionality reduction heuristics, as far as practical prediction quality, rather than some 'technical' criteria such as Frobenius norm preservation, is concerned [4, 6].

As empirically observed covariance is a necessary (although not sufficient) condition for causality, it forms the origin of knowledge that may be obtained as a result of pure observation [7]. Moreover, as one may realize, the core component of many widely-known data processing methods involves the use of covariance/coincidence matrix. The group of these method includes Reflective Random Indexing [8], Randomized Singular Value Decomposition in its RSVD-RRI variant [9] and any method based on the projection on the principal eigenvector such as PageRank [10]. Motivated by this observation, we aim at investigating the applicability of covariance matrices as a potential means for prediction-oriented tensor per-mode processing – obviously different to the analogical method based on the widely-known principle of the projection into a subspace spanned by the most significant eigenvectors.

Many authors have stressed that a multi-way data representation has the obvious advantage over any low-order representation of the same data: any low-order representation may be obtained as a result of 'lossy' (i.e., irreversible) summation/averaging of appropriate entries of the 'lossless' multi-way (i.e., matrix-based or tensor-based) representation of the same data (e.g., a matrix containing a set of vectors) [2, 11]. Therefore, representing input data in a multidimensional array (i.e., in a tensor) may be seen as enabling 'seeing' the modelled data from a higher number of 'perspectives', i.e., in a potentially more detailed way. On the basis of such a detailed, multi-way

data representation, many less detailed (i.e., 'generalized') representations of the same data may be obtained. Such an operation of 'neglecting' the variance of one or more tensor modes – obviously leading to obtaining a tensor of the order lower by one – is referred to as tensor 'flattening' [2]. Although hierarchical tensor decomposition methods have been already presented in some papers, the hierarchy that these works deal with is the hierarchy of tensor decompositions [12, 13], not the hierarchy of differently flattened tensors – the concept that is one of the key topics of the research reported in this paper.

3. Tensor representation and processing

We assume that the input data are given as n-tuples, where n is a number of attributes defining each event. In order to describe events in a format which enables comparing them in quantitative way the weighed n-tuples have been chosen, which may be described as follows:

$$\Gamma = (n, \mathcal{V}^{(1)}, \ldots, \mathcal{V}^{(n)}, \Lambda, \psi), \tag{1}$$

where $\mathcal{V}^{(i)}$, $(i = 1, \ldots, n)$, is a set of values which may be used as the i-th element of an n-tuple, Λ is a set of n-tuples of the form $(v^{(1)}, \ldots, v^{(n)})$ where $v^{(i)} \in \mathcal{V}^{(i)}$, and $\psi : \mathcal{V}^{(1)} \times \cdots \times \mathcal{V}^{(n)} \to \mathbb{R}$ is a function used to assign the weight. To model the set of n-tuples as a multidimensional array (referred to as a tensor) one has to define the tensor space $\mathcal{T} = \mathcal{I}^{(1)} \otimes \cdots \otimes \mathcal{I}^{(n)}$ where $\mathcal{I}^{(i)}$ is a basis of order $|\mathcal{V}^{(i)}| = n_i$ used to index elements of set $\mathcal{V}^{(i)}$. Finally, each set of n-tuples may be modelled as an element of $\mathcal{T}$. It is worth being stressed that a tensor space model that we consider (following [1]) does not involve modeling of dual spaces.

In the presented framework we assume that $\psi : \mathcal{V}^{(1)} \times \cdots \times \mathcal{V}^{(n)} \to \{0, 1\}$ and $\psi(v^{(1)}, \ldots, v^{(n)}) = 1$ if and only if $(v^{(1)}, \ldots, v^{(n)}) \in \Lambda$. Then, input data may be modelled as tensor $T = [t_{i_1,\ldots,i_n}]_{n_1 \times \cdots \times n_n}$ with binary entries.

Tensor-to-tensor transformation. Using the tensor-to-tensor transformation one may process the input tensor, usually of enormous size and sparsity, into a tensor of reduced cardinality of each of the modes. In general, tensor-to-tensor transformation is made according to the formula:

$$\widetilde{T} = T \times_1 U^{(1)} \times_2 \cdots \times_n U^{(n)}, \tag{2}$$

where $T \times_i U^{(i)}$ is a tensor by matrix multiplication transforming tensor fibres of i-th mode of tensor T into new fibres in the corresponding mode of output tensor $\widetilde{T}$ in such a way that the entries of a new fibre are just inner products of the old fibre and columns of matrix $U^{(i)}$. In order to clarify what the tensor fibre is, it is convenient to use the tensor unfolding concept, which is basically the Kronecker-product-based matrix representation of the tensor (see [6] for unfolding definition).

In such an interpretation, the tensor fibre is just a column of the unfolding, and the tensor-to-tensor transformation may be described in terms of matrix multiplications.

The entries of the result tensor of each tensor-to-tensor transformation may be calculated as follows:

$$\widetilde{t}_{j_1,\ldots,j_n} = \sum_{i_1 \in I^{(1)}} \cdots \sum_{i_n \in I^{(n)}} t_{i_1,\ldots,i_n} u^{(1)}_{j_1,i_1} \ldots u^{(n)}_{j_n,i_n}. \tag{3}$$

Transformation of input tensor into a state tensor of reduced size. Due to its multi-dimensional nature the input tensor suffers from its big size and high sparsity. In order to address these issues the proposed framework assumes the application of the preliminary dimensionality reduction similar to N-way Random Indexing (NRI) approach [3]. This step can be described as the tensor-to-tensor transformation using $n_i \times m_i$ matrices $U^{(i)}$ $(i = 1, \ldots n)$, where n_i and m_i are the cardinalities of i-th mode of the tensor before and after transformation, respectively. Each row of the transformation matrix (i.e, $(u^{(i)}_{k,1}, \cdots, u^{(i)}_{k,m_i})$) forms the random vector of specified length and specified seed [8] – each entry of the vector is set to be equal to 0 or 1, and then the vector is normalized using L^1-norm. We denote the result of transforming the input data using the matrices $U^{(i)}$ described above as state tensor $X = [x_{i_1,\ldots,i_n}]_{m_1 \times \cdots \times m_n}$.

The proposed model assumes that before being used for the processing and querying procedures the state tensor needs to be preprocessed according to two following steps (i) *scaling in order to get the probability distribution* done as follows

$$x_{i_1,i_2,\ldots,i_n} := \frac{x_{i_1,i_2,\ldots,i_n}}{\omega}, \tag{4}$$

where ω is the number of n-tuples used to build state tensor X, and (ii) *preparing to be used in L^2-norm operations* done by taking each entry square root value, i.e:

$$x_{i_1,i_2,\ldots,i_n} := (x_{i_1,i_2,\ldots,i_n})^{1/2}. \tag{5}$$

State tensor querying. The state tensor querying procedure is aimed at reconstructing the entries of the input tensor. In general, this procedure may be seen as a tensor-to-tensor transformation (reverse to the state tensor creation step), but due to practical reasons it is defined as a procedure of reconstructing the single entry of the input data tensor. For a given n-tuple $\gamma = (k_1, \ldots, k_n)$ the query tensor $Q^\gamma = [q^\gamma_{i_1,\ldots,i_n}]_{m_1 \times \cdots \times m_n}$ is constructed as a tensor of the same size as the state tensor. Its entries are calculated according to the formula:

$$q^\gamma_{i_1,\ldots,i_n} = (u^{(1)}_{k_1,i_1})^{1/2} \cdot \ldots \cdot (u^{(n)}_{k_n,i_n})^{1/2}. \tag{6}$$

Then, the result of the state tensor querying procedure is calculated as an inner product of preprocessed state tensor X (according to (4) and (5)) and query tensor Q^γ, as follows:

$$\widetilde{t}_\gamma = \sum_{1 \le i_1 \le m_1} \cdots \sum_{1 \le i_n \le m_n} x_{i_1,\ldots,i_n} q^\gamma_{i_1,\ldots,i_n}. \tag{7}$$

The same querying procedure is applied to the filtered state tensor which is constructed according to the procedure described in the next section.

3.1. Covariance-based multilinear filtering

The proposed framework assumes the construction of filters for each tensor mode which are calculated as the linear combination of covariance matrices determined based on input state tensor X.

Extracting covariance data from the tensor data. It has to be stressed, that different relations in data may be seen depending on the choice of attributes used to model tensor modes. The construction of different tensors modelling the dependencies between given mode elements may be done by building the most detailed tensor, i.e., the tensor involving the use of a maximum possible number of modes corresponding to the set of all event attributes provided in the input data, and then consecutive procedure of so-called tensor flattening (i.e., aggregating the tensor entries across the mode being flattened/hidden). Specifically, operation of flattening the tensor $T = [t_{i_1,\ldots,i_n}]_{n_1\times\cdots\times n_n}$ over mode i leads to the new tensor T', such that:

$$T' = [t'_{i_1,i_2,\ldots,i_{i-1},i_{i+1},\ldots,i_n}]_{n_1\times n_2\times\ldots\times n_{i-1}\times n_{i+1}\times\ldots\times n_n},$$

where

$$t'_{i_1,i_2,\ldots,i_{i-1},i_{i+1},\ldots,i_n} = \sum_{1\leq j\leq n_i} t_{i_1,i_2,\ldots,i_{i-1},i_{i+1},\ldots,i_n}.$$

For a data set given as n-tuples, the number of different tensors that may be used in order to model the dependencies between the elements of a given mode is equal to the number of different subsets of the set of remaining attributes, i.e, is equal to $\sum_{i=0}^{n-1}\binom{n-1}{i} = 2^{n-1}$.

The set of all possible tensor flattening may be enumerated in the combinatorial way using labels based on binary numbers of length equal to the maximum number of modes. Let us assume that the most detailed state tensor, i.e., the tensor which involves modelling of each attribute as a mode is labelled by the binary number $111\ldots1$ (equal to 2^n-1). The operation of flattening the i-th mode leads to the flipping the bit at the i-th position of the binary code from 1 to 0. Finally, the totally flattened tensor (i.e., the tensor flattened to a scalar which is the sum of all tensor cells) is denoted by the binary number $000\ldots0$. We denote the flattenings of tensor X as X_j, where j corresponds to the flattening code ($0 \leq j \leq 2^n-1$). Each flattening except the totally flatten tensor (i.e., the tensor flatten to the scalar), and flattenings to one mode (i.e., to vectors), takes part in the procedure of filters' construction.

Overall centring. In order to provide the covariance data about elements of a given mode, each state tensor flattening has to be centred. The simplest way to provide the covariance matrix is to centre across the tensor slices corresponding to the elements of this mode. The centring operation is provided by the subtraction of the mean of values in cells of a given tensor slice. However, this operation is not regarded as a most effective data centring [11]. Instead, so-called overall centring [11] should be used as the operation which leads to the minimum Frobenius norm of the covariance matrix. The overall centring may be done by consecutive centring of tensor fibres in each mode, i.e., for a given mode all fibres are centred and then this procedure is repeated for the next mode and so on. Equivalently, the overall centring

operation for a given h-mode tensor X_j ($h \leq n$) may be formulated in terms of the inclusion-exclusion principle (see [11] for details). We denote the result of centring procedure applied for flattening X_j as X_j^c.

Generation of covariance matrices. Using the data collected in each centred tensor X_j^c we construct the matrices describing the relation among elements of the given mode, as follows:

- the unfolding matrix $X_j^{c,(i)} \in \mathbb{R}^{J_i \times (J_1 \times \cdots \times J_{i-1} \times J_{i+1} \times \ldots J_n)}$ is constructed, which collects i-th mode fibres of centred state tensor X_j^c as columns,
- then, the symmetric matrix $A_j^{(i)} = [a_j^{(i)}]_{m_i \times m_i}$ such that:

$$A_j^{(i)} = X_j^{c,(i)} \left(X_j^{c,(i)} \right)^T \tag{8}$$

is obtained as a matrix representing the covariance between random dimensions used to enumerate the i-th mode. Finally, $A_j^{(i)}$ is the covariance matrix for elements of i-th mode constructed from the j-th flattening of state tensor X.

The example illustrating the process of flattening and covariance matrix creation has been presented in Table 1.

Table 1. Flattenings and covariance matrices for the 4-mode tensor describing events concerning users (1st mode), items (2nd mode), time (3rd mode), and location (4th mode).

Tensor modes	Flattening code and symbol	Covariance matrices
user,item,time,location	1111, X_{15}	$A_{15}^{(1)}$,$A_{15}^{(2)}$,$A_{15}^{(3)}$,$A_{15}^{(4)}$
user,item,time	1110, X_{14}	$A_{14}^{(1)}$,$A_{14}^{(2)}$,$A_{14}^{(3)}$
user,item,location	1101, X_{13}	$A_{13}^{(1)}$,$A_{13}^{(2)}$,$A_{13}^{(4)}$
user,time,location	1011, X_{11}	$A_{11}^{(1)}$,$A_{11}^{(3)}$,$A_{11}^{(4)}$
item,time,location	0111, X_7	$A_7^{(2)}$,$A_7^{(3)}$,$A_7^{(4)}$
user,item	1100, X_{12}	$A_{12}^{(1)}$,$A_{12}^{(2)}$
user,time	1010, X_{10}	$A_{10}^{(1)}$,$A_{10}^{(3)}$
user,location	1001, X_9	$A_9^{(1)}$,$A_9^{(4)}$
item,time	0110, X_6	$A_6^{(2)}$,$A_6^{(3)}$
item,location	0101, X_5	$A_5^{(2)}$,$A_5^{(4)}$
time,location	0011, X_3	$A_3^{(3)}$,$A_3^{(4)}$

Constructing the filter based on covariance matrices. For mode i the optimal filter $F^{(i)}$ is constructed as a sum of an identity transformation and the average of matrices $A_j^{(i)}$. In particular, we have:

$$F^{(i)} = I_i + \frac{1}{k} \sum_j A_j^{(i)}, \tag{9}$$

where I_i is the identity matrix of size m_i, and k is a number of covariance matrices built for the i-th mode. We assume that before applying the filters the tensor X is centred according to overall centring [11] approach. The filters $F^{(i)}$ are used in order to transform centred tensor X^c into its filtered version $\widetilde{X}^c$ according to the formula: $\widetilde{X}^c = X^c \times_1 F^{(1)} \times_2 \cdots \times_n F^{(n)}$. At the next step the prediction tensor $\widetilde{X}$ is calculated as $\widetilde{X} = X - X^c + \widetilde{X}^c$. Finally, the tensor $\widetilde{X}$ is used for calculating the prediction results according to the querying procedure described by equations (6) and (7). It has to be stressed that the prediction tensor may be additionally transformed using the HOSVD approach [6] what leads to reduction of tensor size and, as consequence, shortens the time needed for querying. The experimentation part of the paper involves the evaluation of the presented method both with and without the final HOSVD-based dimensionality reduction step.

4. Experiments

In this paper, we present an experimental verification of the proposed method – Covariance-based Multilinear Filterning, referred to as '*CMF*' and '*R-CMF*' for CMF enhanced by post-processsing based on HOSVD. The scope of the experiments has been limited to a recommendation scenario that involves extreme data sparsity. Although such a scenario is considered as challenging, it is also very common in the area of e-commerce, as an online merchandising recommender system is frequently provided with a few ratings per user [14].

In our experiments we followed the approach to the evaluation of a recommender system proposed in [14]. We used one of the most widely referenced data sets – the MovieLens ML100k set, which contains 100.000 ratings (in a 1 to 5 scale) for 1682 movies given by 943 unique users. Each above-average rating (i.e., equal to 4 or 5) has been treated as an indication that a given user likes a given movie. Analogically, below-average ratings have been treated as an indication that a given user dislikes a given movie. In companion to the rating data, we used the information about each movie's genre (19 distinct values). As a result, the input data was modelled as a tensor T with size equal to $2 \times 943 \times 1682 \times 19$ containing 100.000 non-zero entries. Finally, we randomly divided the data set into a training set and a testing set. In order to address the high sparsity scenario the rating data was divided according to a training ratio equal to 0.1. The goal of the recommendation algorithm was to predict whether each rating in the test set is positive (given user likes a given movie) or negative (given user dislikes a given movie). Each recommendation quality measurement result presented in this paper represents the averaged result of 50 individual experiments.

We have compared our method, both in terms of recommendation accuracy and computational efficiency, with other methods presented in the relevant literature, such as NRI [3], HOSVD [6], as well as with a baseline method based on typical SVD-based collaborative filtering [15] (referred to as '*2-mode CF*'). Additionally, we have tested the method that is established on a concept of preliminary NRI-based dimensionality reduction and vector space optimization based on HOSVD (herein

referred to as '*NRI+HOSVD*'). For each of these methods we set all the necessary parameters optimally in order to provide the best possible accuracy. In explicit, we set the k-cut in *2-mode CF* to 7, and the size of the final 'core tensor' in HOSVD to $(2 \times 18 \times 24 \times 8)$. Additionally, we limited the size of the state tensor X to $(2 \times 64 \times 64 \times 19)$, and we used the random vector's *seed* value to 4 (herein only the dimensionality of modes representing the users and movies is reduced, as the size of genres and likes/dislikes mode is comparatively small).

To obtain quantitative recommendation accuracy results, we have evaluated an ordered list of user-item recommendations, generated by each of the algorithms, by means of the AUROC measure [14]. The results presented in Table 2 clearly indicate that, in a high data sparsity scenario, all of the tensor-based methods enable to provide higher quality recommendations than a typical matrix factorization method (probably due to the limitations of data representation model). As we have confirmed experimentally, the HOSVD algorithm is more accurate (in terms of AUROC) than the NRI based on simple random indexing. On the other hand, while it is possible to provide high quality results using HOSVD on a full size input tensor, applying such a procedure on a state tensor of reduced size does not lead to increased recommendation quality. Finally, what is most important, we have shown that the proposed algorithms based on CMF allow to obtain better results, as far as a small training ratio is concerned, than all other methods used for comparison.

Table 2. The average AUROC results for all tested algorithms; the best and second best result is correspondingly highlighted by bold and slanted font setting.

2-mode CF	HOSVD	NRI	NRI+HOSVD	CMF	R-CMF
0.527 (±0.004)	0.562 (±0.004)	0.550 (±0.004)	0.549 (±0.004)	**0.585** (±0.004)	*0.582* (±0.005)

The average execution times (single-threaded) for the compared tensor-based methods have been presented in Table 3. In methods based on NRI, in case of sparse data, the complexity of such algorithms is dependent on the number of input tuples and not on the size of the input tensor (as in the case of HOSVD) and may be flexibly adapted to the amount of available computational resources. Thus, as it has been shown, the NRI-based methods allow to significantly reduce the time needed to build the model. On the other hand, the model query time depends on the final size of the processed tensor. Due to the fact that, in general, it is not possible to achieve the extent of dimensionality reduction as in the case of SVD using RI while providing high quality results, the query execution times for methods based solely on NRI are higher. Contrarily, combining NRI-based preprocessing with HOSVD-based vector space optimization enables to reduce both the model construction time and query time. Consequently, as shown in our experiments, the *NRI+HOSVD* and *R-CMF* algorithms require, in total, the lowest execution times.

Table 3. The average execution times (in seconds); in each row, the best and second best results are correspondingly highlighted by bold and slanted font setting.

	HOSVD	NRI	NRI+HOSVD	CMF	R-CMF
Model build	48.6	**9.0**	*9.1*	*9.1*	9.2
Model query	*28.7*	77.7	**20.9**	77.7	**20.9**
Total	77.3	86.7	**30.0**	86.8	*30.1*

5. Conclusions

The paper contributes with a formal description of a novel tensor-based data representation and processing framework. The core part of the framework is a novel multilinear filtering method which involves the use of a hierarchical structure of covariance matrices extracted from the so-called state tensor. Based on the experiments involving the use of widely-referenced data sets, we have analysed the performance of the proposed method in the 'cold-start' personalized recommendation scenario. Firstly, we have confirmed that each of the methods based on multilinear modelling outperforms the standard collaborative filtering based on matrix factorization what leads us to the conclusion that tensor-based data representation enables more effective data representation. Secondly, we have shown that the use of filters based on the hierarchical structure of covariance matrices built using various tensor flattenings (i.e., various data generalization), enables to extract additional dependencies in input data what, in consequence, leads to a further performance improvement. Additionally, we have confirmed that, as a result of application of dimensionality reduction techniques (both based on RI and HOSVD), the proposed method may be effectively used for large data sets and for multi-mode processing – without the common limitation of 3-mode structures processing [1].

Acknowledgement

This work is supported by the Polish National Science Centre, grant DEC-2011/01 /D/ST6/06788.

6. References

[1] Nickel M., Tresp V., *An Analysis of Tensor Models for Learning on Structured Data.* In: Machine Learning and Knowledge Discovery in Databases. 8189 of LNCS. Springer Berlin Heidelberg 2013, pp. 272–287.

[2] Lu H., Plataniotis K.N., Venetsanopoulos A.N., *Multilinear principal component analysis of tensor objects for recognition.* In: 18th International Conference on Pattern Recognition, ICPR 2006. vol. 2., 2006, pp. 776–779.

[3] Sandin F., Emruli B., Sahlgren M., *Incremental dimension reduction of tensors with random index.* March 2011, pp. 240–56.

[4] Grasedyck L., Kressner, D., Tobler C., *A literature survey of low-rank tensor approximation techniques.* GAMM–Mitteilungen, 2013, 36.1, pp. 53–78.

[5] Baldassarre L., Rosasco L., Barla A., Verri A., *Multi-output learning via spectral filtering.* Machine learning, 2012, 87(3), pp. 259–301.

[6] De Lathauwer L., De Moor B., Vandewalle, J., *A multilinear singular value decomposition.* SIAM J. Matrix Anal. Appl, 2000, 21, pp. 1253–1278.

[7] Pearl J., *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, 1988.

[8] Cohen T., Schvaneveldt R., Widdows D., *Reflective Random Indexing and indirect inference: a scalable method for discovery of implicit connections.* Journal of Biomedical Informatics, 2010, 43(2), pp. 240–56.

[9] Ciesielczyk M., Szwabe A., *RSVD-based Dimensionality Reduction for Recommender Systems.* International Journal of Machine Learning and Computing, 2011, 1(2), pp. 170–175.

[10] Brin S., Page L., *The anatomy of a large-scale hypertextual web search engine.* Proceeding WWW7 Proceedings of the seventh international conference on World Wide Web 7, 1998, 30(1-7), pp. 107–117.

[11] Kroonenberg P. M., *Three-mode principal component analysis: Theory and applications.* vol. 2. DSWO press; three-mode.leidenuniv.nl, 1983.

[12] Grasedyck L., *Hierarchical singular value decomposition of tensors.* SIAM Journal on Matrix Analysis and Applications, 2010, 31(4), pp. 2029–2054.

[13] Kolda T.G., Bader B.W., *Tensor decompositions and applications.* SIAM review, 2009, 51(3), pp. 455–500.

[14] Herlocker J.L., Konstan, J., Terveen L.G., Riedl, J., *Evaluating collaborative filtering recommender systems.* ACM Transactions on Information Systems, 2004, 22(1), pp. 5–53.

[15] Koren Y., Bell R., Volinsky C., *Matrix factorization techniques for recommender systems.* Computer, 2009, 8, pp. 42–49.