Application of the Tensor-Based Recommendation Engine to Semantic Service Matchmaking

Andrzej Szwabe, Michal Ciesielczyk, Pawel Misiorek, Michal Blinkiewicz
Institute of Control and Information Engineering
Poznan University of Technology
Poznan, Poland
Email: {firstname.lastname}@put.poznan.pl

***Abstract*—The paper presents a novel approach to semantic Web service matchmaking, which involves a use of multilinear data representation and processing. The proposed solution involves the use of a novel tensor data filtering method based on a set of covariance matrices derived from a hierarchical tensor structure. We provide results of experimental evaluation of the proposed solution conducted with the use of the Semantic Service Selection (S3) contest dataset. The evaluation has been done using the standard Information Retrieval methodology that assumes the methodologically correct partitioning of the dataset on mutually exclusive subsets: the training set and the testing set. The experimental evaluation results presented in the paper indicate superiority of the covariance-based tensor filtering method over other state-of-the-art tensor processing methods in terms of the matchmaking quality measured using mean average precision and Area Under the ROC curve (AUROC) measures.**

Keywords–Semantic Service Selection; tensor-based multirelational data modeling.

I. Introduction

Semantic Web Service (SWS) technologies are aimed at discovering and matching Web services using their functional and nonfunctional semantic representations. Due to practical importance of SWS solutions, in recent years the attention of scientific community has been paid on the development of methods which enable to embed the semantics into the discovery, matchmaking, and mediation processes [1]–[5]. In this paper, we investigate matchmaking of Web services described using the Semantic Annotations for Web Service Description Language (SAWSDL) standard which is based on enriching Web Services Description Language (WSDL) documents with semantic annotations in the form of references to ontologies. The presented research uses the widely-referenced [1]–[5] data collection – SAWSDL-TC [6] – developed for the purposes of the Semantic Service Selection (S3) contest [7].

A. Research Motivation

The most important part of each service matchmaker is its matching algorithm, which determines the means of the relevance measurement applied to a pair of Web services. The S3 contest editions have shown that the best results are achieved by the adaptive hybrid matchmakers, such as [2][4][7], which use a part of the test collection for optimization purposes. Hybrid matchmaker systems make use of several types of similarity algorithms for Web service descriptions (including logical and lexical similarity algorithms) and subsequently compute the overall similarity based on the importance weights of partial results optimized according to the cross-validation approach. One of the main goals of this paper is to investigate input data integration as an alternative to the widely-proposed integration at the level of final results provided by several hybridized subsystems (systems operating in accordance to different approaches to the Web service matchmaking task). In order to enable such a solution, we have used the tensor-based data representation which is suitable to integrate heterogeneous data. This approach results in no need for a further aggregation of all fragmentarily computed similarities.

The tensor-based data representation has been already recognized as a suitable tool for storing the multidimesional data in a compact way [8][9] that may be effectively used in many application areas related to machine learning [8][10]–[12]. We recognized the application of tensor data representation to the semantic service selection task as a promising approach, especially because the S3 task requires the need of retrieving the information from heterogeneous data sources. As a consequence, the experimental evaluation presented in this paper is focused on comparison of the proposed tensor-based data processing method with state-of-the-art methods.

B. Contribution

The main aim of this paper is to present the novel approach to the S3 task based on two-step processing consisting of the heterogeneous data integration step and the processing of the integrated data using the tensor model.

The important part of the paper contribution is related to evaluation methodology issues. In contrast to the methodology used in the S3 contest, the described experimental results are based on partitioning the dataset into a training and a testing set in such a way that the data used for testing the performance are not previously used to learn or tune the model. Alas, such an approach is not used in the S3 contest. According to the contest rules the participating matchmakers provide the recommendation results for the whole set of service requests described in the dataset. The S3 evaluation tool does not provide any additional set of reference matchings which may be used as a training set. For this reason, in this paper, we propose to consider the semantic service matchmaking task as a case of semi-supervised learning in which unlabeled data are used in conjunction with a small amount of labeled data [13]. Due to above-explained evaluation methodology differences, the provided performance evaluation does not contain a direct comparison of the proposed method's operation to the S3 contest results.

The rest of paper contribution includes: (i) the data integration framework, which enables the transformation of

SAWSDL and Web Ontology Language (OWL) documents into the set of n-tuples (or Resource Description Framework (RDF) statements) and then the aggregation of these data using the tensor-based data representation, (ii) the first tensor-based SWS matchmaking engine involving the use of the tensor-based data processing system based on a filtering method which applies the covariance data derived from a hierarchical structure of tensor flattenings, and (iii) the comprehensive comparison of several matchmaking algorithms including those based on the state-of-the-art tensor processing techniques (i.e., N-way Random Indexing [14] and Higher Order Singular Value Decomposition (SVD) [15]).

The paper is structured as follows. Section II provides a discussion on related work, which contains a brief presentation of state-of-the-art SWS matchmaking solutions, their limitations, as well as tensor-based data processing algorithms. The proposal of tensor-based semantic service recommendation system including the semantic data integration framework and tensor-based recommendation engine is given in Section III. Next, the tensor-based data representation and filtering method is provided in Section IV. Section V contains the description of the evaluation methodology and of the algorithms used for comparison. Section VI provides the experimental results and their analysis. Finally, the paper is concluded in Section VII.

II. Related Work

In this section, the advantages and limitations of leading state-of-the-art matchmaking solutions – in context of semantically annotated Web services – are discussed. Additionally, the tensor-based data processing assumptions and state-of-the-art algorithms are introduced.

A. State-of-the-art Web Service Matchmakers

The state-of-the-art Web service matchmakers make use of different knowledge representation formalisms and are usually referred to as *hybrid* solutions. They are known to achieve better results then logic-based only or non-logic-based only approaches in terms of the precision and recall measure [3]. Authors of the articles describing their hybrid matchmakers drew an attention to the problem of an aggregation of different matching results. Primarily weights of logical, text similarity and structural similarity matchings are set manually based on tests and analysis. It follows that any change of ontologies or services forces re-testing and re-analysis in order to select new appropriate weights [1]–[3][5]. Thus, a new *adaptive* approach has been proposed, which resolves this issue by letting the matchmaker learn what is the best adoption of weights. The main benefit of an adaptive approach is that a matchmaker settings are not dependent on a particular data collection. In order to adapt the system for a new dataset, it is sufficient to recalculate the weights in the off-line relearning process.

It should be stressed, however, that the evaluation procedure of the S3 contest [7] does not provide a separate set of reference matchings that may be used as a training set. Nevertheless, the contest participants' solutions based on the adaptive hybrid recommendations [1]–[5] use matchings from the test set in order to find the optimal set of weights in the procedure based on the k-fold cross-validation technique. In particular, the mentioned systems apply different machine learning techniques when determining the weights for particular strategies of the hybrid solution, including *logistic regression*, *simple linear regression* and *support vector regression* (SAWSDL-iMatcher [1]), *ordinary least squares* estimator (LOG4SWS.KOM [5]), and *support vector machine* (SAWSDL-MX2 [3]). Such an approach violates principles of recommendation systems evaluation [16]–[18] because it allows to learn from the information which is also the subject of testing. Another adaptive hybrid matchmaking system – URBE [2] – also assumes the system configuration phase in order to optimize the hybrid algorithm parameters, but, in this case, the authors have also conducted an evaluation assuming the partition of the set of reference matchings into mutually exclusive training and testing sets. However, this approach was applied only for the case of tests using dataset OWLS-TC of the S3 contest [7] track devoted to the OWL-S standard.

In contrast to the S3 contest evaluation methodology, the performance evaluation described in this paper assumes the explicit specification of the information about referential mappings used for training purposes and does not use these mappings in the testing phase. Such an approach is constitutional for the matchmaking system proposed in Section III, which assumes the application of input data integration instead of the integration done at the level of final results of using different strategies.

B. Tensor-Based Data Representation and Processing

The main goal of the paper is to propose a new approach to the S3 task involving the processing of the integrated data using the tensor model. Higher-order tensors are already used in many areas of research as a model for data representation [8]–[10][19], including the signal and image processing, higher-order statistic or scientific computing. At the same time, it may be observed that the tensor data model has been widely used for various information retrieval application, mainly by means of 3-rd order tensors used for multirelational data analysis, e.g., as presented in [11][12] for the case of processing RDF statements. It is well known that many problems in machine learning involve the processing of information with multiple aspects and high dimensionality. For such problems, the tensors are regarded as the most natural and compact representation for multidimensional data, however, they have to be accompanied by some low-rank approximation approach [8][9], e.g., based on tensor decomposition [19]–[21]. The semantic service matchmaking task, as an application scenario which involves the processing of multirelational and multidimensional data from heterogeneous sources (OWLs, SAWSDLs, textual data), seems to be another research area for which tensor data representation and processing methods may be efficiently applied.

It has to be admitted that the exponential grow of number of tensor elements observed with the increase of the number of tensor dimensions (usually referred to as tensor modes [10]) seems to be the main reason, why a significant part of the experimental research on tensor models is limited to the case of 3-rd order multidimensional structures [10][11][14][15][21].

In the context of multilinear data processing, the most widely known form of per-mode tensor filtering is the projection of tensor 'fibres' laying along the given tensor mode into a subspace spanned by the modes' principal components – the projection being the main tool of filtering based on Higher-order Singular Value Decomposition (HOSVD) [15] and Multilinear Principal Component Analysis [10]. However, the state-of-the-art solutions do not investigate the 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 [15][22]. Moreover, in order to be effective, the multilinear data modeling has to follow mathematical constrains derived from the area of statistics, algebra, or probability theory, [10][20][21][23]. The issues concerning the proper data centering necessary to provide the efficient multilinear principal component analysis are one of the examples of such a constrains [23]. In this paper, we present and experimentally evaluate the tensor filtering method involving the use of covariance matrices derived from different tensor structures, which addresses the above-mentioned issues.

III. Tensor-Based Semantic Service Recommendation System

The purpose of the proposed system is to provide accurate Web service recommendations – referred to as *offers* – for a given Web service description – referred to as *query*. Both *offers* and *queries* are assumed to be represented in the form of SAWSDL documents with references to objects described in OWL ontologies. The general architecture of the proposed system includes two main components:

1) The converter selecting essential information from SAWSDL descriptions, OWL documents, and reference matchings used to train the model and subsequently transforming them into a common representation.
2) The recommendation engine, described in Section IV, aimed at generating the high quality recommendations.

Thus, the quality of tuples, which are chosen as internal data representation of the system, generated by the converter is crucial for final recommendations accuracy. It should be also taken into account that the information from heterogeneous data is aggregated at the beginning of processing rather than, as in the case of the state-of-the-art solutions (discussed in Section II), as the last step.

As shown in Figure 1 an SAWSDL description is a WSDL document enhanced with semantic annotations linking various parts of the Web service description to corresponding OWL ontology classes.

The introduced framework processes every SAWSDL document along with other linked XML or OWL files, and transforms the acquired information into a common representation. Specifically, for each SAWSDL document the *portType* element, constituting the interface of the Web service, is parsed. The *portType* consists of a set of *operations* having exactly specified *input* and *output*, which in turn reference corresponding *messages*. Every *message* has a list of elements associated with a specific *types* expressed in the XML Schema (XSD) language, which in turn may reference to corresponding OWL ontology classes. As shown in Figure 1, the OWL classes are subsequently linked with the related instances from the ontology (super- and sub-classes). All of the human-readable names – appearing in SAWSDL documents as values of the

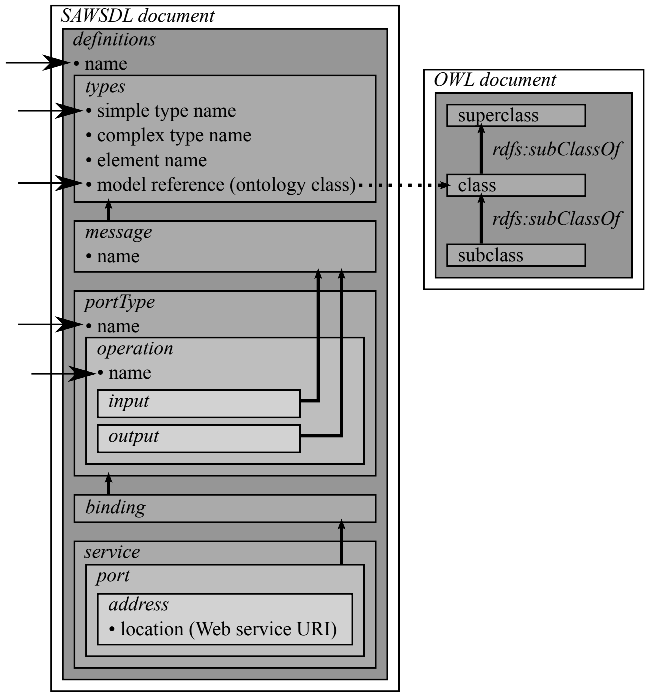

Figure 1. Data retrieved from SAWSDL and OWL documents in order to build the augmented representations of the matchings.

schema fields pointed out in Figure 1 – are being tokenized and included into the common data representation.

Finally, every matching used to train the model is augmented with the acquired semantic descriptions. Specifically, each matching consists of an information whether two corresponding Web services, depicted by an Uniform Resource Identifier (URI), are relevant or not. Subsequently, the augmented descriptions of every Web service used by our system have been built from the following attributes:

- tokenized Web service name or URI,
- tokenized *portType* name,
- tokenized *operation* names,
- XSD simple data type names,
- URIs pointing to corresponding OWL ontology classes.

The tensor-based recommendation engine operates on data provided in the form of n-tuples or its RDF equivalent.

A. Tuple-Based Representation of the Input Data

For the purposes of representing the tuple-based SAWSDL Web services descriptions we use a 3-mode tensor. The first tensor mode is used to model the information on service relevance (i.e., relevant/nonrelevant indicator). The second and the third modes represent the augmented semantic descriptions of a *request* and an *offer*, respectively, related to the matching specified by the first mode. Such a description is represented by a vector built as an L^1-normalized sum of index vectors (uniquely assigned random vectors) of related terms (i.e., the terms from the specific augmented Web service description).

An index vector is a uniquely assigned random vector for the object that it represents, as defined in [24].

Listing 1 shows a reduced (for the purpose of presentation) 3-tuple example. Line 1 indicates that the two considered Web services are relevant. Line 2 and 3 represent the terms from tokenized Web service request and offer descriptions, respectively.

```
('relevant',
 ['shopping mall', 'camera', 'price', 'Mid-level-
     ontology.owl#ShoppingMall', extendedCamera.
     owl#Camera'],
 ['shopping mall', 'purchasable', 'item', 'Mid-
     level-ontology.owl#ShoppingMall', '
     extendedCamera.owl#PurchaseableItem'])
```

Listing 1. The 3-tuple example.

As specified formally in the next section, the tensor that represents the whole input data set is simply the sum of the individual rank-one tensors, while each of these tensors represents a single tuple form the data set. Thus, the elementary procedure of the input tensor construction process is the computation of the rank-one tensor representing a given tuple. Such a rank-one tensor is obtained as an outer product of the vectors representing all the consecutive elements of the given tuple.

Let us refer to the example presented in Listing 1 once again. As the first element of the tuple corresponds to the single token of 'relevant', the first argument of the tensor product is simply the vector representing this element in the vector space corresponding to all the first values of the tuples. In contrast to the simplest case, when the given tuple element is a set of elements, rather than a single element, the vector representing such a set of elements is built as a superposition of the vectors representing the elements. For example, when the second element of the tuple is a set of the following five elements the second argument of the tensor product is the normalized sum of the below-enlisted five vectors:

```
('shopping mall', 'purchasable', 'item', 'Mid-
     level-ontology.owl#ShoppingMall', '
     extendedCamera.owl#PurchaseableItem')
```

Listing 2. Elements of the second argument in the example 3-tuple.

It is worth noticing that the proposed approach does not require the model reference instances of any web service to be linked to classes from the same ontology. Any element of any given set (constituting the given tuple element) is treated simply as a regular token, i.e., exactly the same way as an 'ordinary' token (representing a word found in some text) is treated. Thus, there is no obstacles limiting the use of different ontologies in order to describe the inputs and the outputs of the same web service, not to mention the inputs and the outputs of different web services.

B. RDF-Based Representation of the Input Data

The RDF-based representation of the SAWSDL and OWL documents is obtained in a similar manner as the tuple-based representation except that the result is saved into RDF statements rather than tuples. First of all, the information whether two Web services are relevant is formed by the triple which subject is a *request* URI, object is an *offer* URI and predicate is one of the *isRelevant* or *isNonRelevant* properties. The augmented semantic description, derived from SAWSDL and OWL documents, is stored as triples with the subject being *request* or *offer* URI, predicate indicating the corresponding attribute property, and object containing associated information – such as a term or type (both in form of a literal) or an OWL class reference (in form of an URI). Thus, it should be also taken into account that, typically, one tuple is represented by more than one RDF statement. As an example, Listing 3 shows the same tuple as in Listing 1 in an RDF format (Turtle notation).

```
<shoppingmall_cameraprice.wsdl>
   :isRelevant <shoppingmall_purchaseableitemprice
      .wsdl> ;
   :terms "shopping mall", "camera", "price" ;
   :input_owl_uri_ref <Mid-level-ontology.owl#
      ShoppingMall> ;
   :output_owl_uri_ref <extendedCamera.owl#Camera>
      .
<shoppingmall_purchasableitemprice.wsdl>
   :terms "shopping mall", "purchasable", "item";
   :input_owl_uri_ref <Mid-level-ontology.owl#
      ShoppingMall> ;
   :output_owl_uri_ref <extendedCamera.owl#
      PurchaseableItem> .
```

Listing 3. The example tuple in an RDF format (Turtle notation, with URI prefixes removed).

Note that the statement form of a subject-predicate-object expression is also known as a triple – or equivalently as a 3-tuple – in the RDF terminology. Therefore, in this paper, as not to introduce confusion, RDF data is referred to only as statements, while the tuple-based representation refers solely to the representation described in Section III-A.

IV. Tensor-Based Recommendation Engine

The semantic Web service matchmaking algorithm presented in this paper is based on multilinear filtering framework proposed in [25]. In this section, the main features of this framework are presented. Moreover, all the settings and assumptions made in order to use this framework for the semantic service selection task have been provided.

A. Tensor-Based Representation of a Tuple Set

We assume that the heterogeneous data on Web services is transformed to the integrated set of n-tuples, where n is a number of attributes defining each event of relevance (or irrelevance) for a given pair of services. 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 modeled as an element of $\mathcal{T}$.

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 modeled as tensor $T = [t_{i_1,\ldots,i_n}]_{n_1 \times \cdots \times n_n}$ with binary entries. For the service matchmaking task based on S3 dataset [6], the set of used tuples contains events describing that a given service offer is relevant or irrelevant to a given service request.

Finally, it should be noted, that though the introduced function ψ herein returns binary values only, the model may be easily extended to use a weighted relevance information. In particular, $\psi'(v^{(1)}, \ldots, v^{(n)}) = \beta$, where β indicates the weight assigned to an n-tuple. In such a case, $\beta = 0$ if $(v^{(1)}, \ldots, v^{(n)}) \notin \Lambda$, and $\beta \in \mathbb{R}^+$ otherwise.

B. Tensor-Based Processing

The proposed multilinear filtering framework is based on tensor data modeling involving the use of so-called tensor-to-tensor transformations [25]. In general, tensor-to-tensor transformation is made according to the formula [25]:

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

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 $U^{(i)}$. The entries of the result tensor of each tensor-to-tensor transformation may be calculated as follows:

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

1) Transformation of input tensor into a state tensor of reduced size: Due to its multidimensional 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 [14]. 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 [24] – 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.:

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

2) Tensor querying: The 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.

C. Covariance-Based Multilinear Filtering

The proposed filtering algorithm is based on the application of the covariance data derived from a hierarchical structure of tensor flattenings. 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. The algorithm consists of steps described below. More details on the method may be found in [25].

1) 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 modeling 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). Such a collection of different tensor structures is referred to as tensor network [8][9]. We denote the flattenings of tensor X as X_j, where j corresponds to the flattening code $(0 \le j \le 2^n - 1)$ defined in a way described in [25]. 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.

2) Overall centering: In order to provide the covariance data about elements of a given mode, each state tensor flattening has to be centered. The simplest way to provide the covariance matrix is to center across the tensor slices corresponding to the elements of this mode. The centering 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 centering [21][23]. Instead, so-called overall centering [21] should be used as the operation which leads to the minimum Frobenius norm of the covariance matrix. The overall centering may be done by consecutive centering of tensor fibres in each mode, i.e., for a given mode all fibres are centered and then this procedure is repeated for the next mode and so on. We denote the result of centering procedure applied for flattening X_j as X_j^c.

3) Generation of covariance matrices: Using the data collected in each centered 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 centered state tensor X_j^c as columns, and 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.

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 centered according to overall centering [21] approach. The filters $F^{(i)}$ are used in order to transform centered 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).

D. Complexity of the proposed method

Reducing the space and time consumption is a key issue for the tensor-based approach in which the complexity may grow exponentially with the number of tensor modes used in the model. Therefore, it is crucial to provide the dimensionality reduction step in the earliest phase of computing as possible, ideally, in the phase of data storing in the tensor structure. First of all, it has to be stressed that the tensor $\widetilde{X}$ used for prediction may be additionally transformed using the HOSVD approach [15] that leads to reduction of tensor size and, as consequence, shortens the time needed for querying. Furthermore, according to the research on existing state-of-the-art tensor-based data processing frameworks, including the incremental tensor analysis approach [26] and ALS-based tensor solutions [27], the space and time consumption for this kind of solutions may be efficiently reduced by using the approximation approach avoiding the diagonalization step, the fast approximation methods for finding principal components as well as random sampling techniques.

In particular, in the case of our approach, the space complexity of the proposed method is directly related to the size of a state tensor used to accumulate the data. In the case of the application scenario presented in this paper, we limit to 3-rd order tensors, so the space complexity is bounded by $O(m_1 m_2 m_3)$, where m_i is a cardinality of the i-th mode of the state tensor (as defined in Section IV-B). The computational cost of the method depends on the cost of state tensor construction based on accumulation of ω tuples ($O(\omega m_1 m_2 m_3)$), and the cost of the construction of covariance matrices from tensor unfoldings ($O(m_1^2 m_2 m_3 + m_1 m_2^2 m_3 + m_1 m_2 m_3^3) = O((m_1 + m_2 + m_3) m_1 m_2 m_3)$). In the case of the applying the additional dimensionality reduction step based on HOSVD, the additional cost of $O(h m_1 m_2 m_3)$ have to be taken into account, where h is the reduced number of dimensions. Since, in the application scenario presented in this paper, ω is greater than m_i for each i as well as than h, we have observed the biggest computational cost in the phase of state tensor construction. However, it has to be stressed, that the time of the state tensor accumulation may be efficiently shortened by taking into account that tensor structures corresponding to tuples are sparse. Nevertheless, due to the relatively small size of the state tensor used in the evaluation (see Section V-D), we have not applied such an optimization step.

V. Evaluation

It should be noted that a typical comparison of different matchmaking systems is not the main goal of the experimental research presented in this paper, as we focus on evaluating of several tensor processing methods in the experimental scenario of SWS matchmaking. We assume that each of the compared tensor processing methods is applied to process the same data obtained by means of data integration framework being a part of the system presented in Section III, given using the tuple-based internal data representation (n-tuples or RDF statements). The application of tensor-based data representation and processing methods have been already investigated in several domains for which – similarly as for the S3 task – the input data is multirelational or multidimensional.

A. SAWSDL-TC3 Dataset Use

The experimental evaluation presented in the paper is based on the use of the publicly available SAWSDL test collection – SAWSDL-TC3 [6]. The dataset provides 1080 semantic Web services written in SAWSDL (for WSDL 1.1) from 9 domains (education, medical care, food, travel, communication, economy, weapon, geography, simulation) and consists of both SAWSDL and OWL documents. The S3 SAWSDL-TC is divided into three main sets. The first and second set contain SAWSDL documents representing *queries* and potential *query* matches – *offers*, respectively. The third set consists of related OWL ontologies.

Additionally, the SAWSDL-TC3 contains the XML file `sawsdl-tc3.xml` describing the information on relevance between 4178 Web service pairs (i.e., *query* and *offer* pairs). The relevance information is provided using two independent relevance grades — *binary* and *4-graded*. For the purposes of the experiments presented herein, the *binary* relevance has been chosen. Nonetheless, it should be noted, that experiments could be easily extended – as briefly discussed in Section IV-A – to use the *4-graded* relevance information. For instance, one could set $\beta = 1.0$ if the grade was '*relevant*', $\beta = 0.5$ if '*potentially relevant*', and $\beta = 2.0$ if '*highly relevant*'. It should be kept in mind, however, that in order to adjust properly these weights for a specific dataset, a parameter optimization technique (such as cross-validation on the available training data) should be used.

B. Evaluation Scenarios

In order to experimentally evaluate the compared solutions we have used a part of the test collection, containing information about relevant and nonrelevant Web service matches, in the learning process. Moreover, as in other approaches the learning process itself is performed off-line (i.e., before the matchmaking). However, contrarily to the research presented in the literature, in our experiments we have tested how the training ratio tr – indicating the percentage of the entire test collection that is used to train the model – affects the quality of matchmaking. What is more, following the Information Retrieval experiment design practices [28][29], we did not use the full test collection during the evaluation (what is allowed in the case in S3 contest) but only the remaining part of the data (that was not used in the learning process) instead. For instance, for $tr = 0.2$ the remaining 80% of the test collection has been used to evaluate the predicted Web service matchings. Therefore, due to differences in the methodology, the final results are not directly comparable with the S3 contest results.

Such an evaluation methodology (i.e., based on both data sources) has been chosen as it does not assume that the textual and structural similarities between the items (here represented by semantic Web services) is directly correlated with the matching relation, and thus it may be considered as more comprehensive. In other words, the algorithm is expected to adapt to the specified task – as in a typical semi-supervised learning task – by inferring the meanings of the relations contained in the SAWSDL and OWL documents.

Apart from the hybrid scenario involving using both SWS descriptions and partial information about the relevant or nonrelevant matches, we additionally investigated a simplified scenario involving only the information about the Web service matches. Our motivation for such an approach is an attempt to show how much an algorithm is able to learn using sample mappings only, and how much the matching quality may increase by adding supplementary semantic information.

Finally, the quality of the generated Web service matches has been evaluated using typical Information Retrieval measures described in Section V-C. To compensate for the impact that the randomness of the dataset partitioning has on the results of the presented methods, all figures in this paper show series of values that represent the averaged results of 100 individual experiments. As a result, the standard error of each presented mean is less than 0.005.

C. Recommendation Accuracy Measures

Following other articles in the literature relevant to the Web service matchmaking, we have used the Mean Average Precision (MAP) measure:

$$MAP = \sum_{i=1}^{n} ap_i/n \tag{10}$$

where n is the number of requests tested and ap_i is the average precision for the i-th request. Particularly, the ap is defined as:

$$ap = \sum_{k=1}^{m} P(k)/min(m, r) \tag{11}$$

where r is the number of relevant matchings, m is the recommendation list length, and $P(k)$ denotes the precision at k-th prediction in the recommendation list. Specifically, the precision $P(k)$ is the ratio of correct matchings up to the position k over the k, and is equal to 0 when the k-th prediction is invalid.

Additionally, we have measured the Area Under the ROC curve (AUROC) as it directly allows to establish the probability of making correct or incorrect decisions by a system about whether a matching is relevant. According to [30], AUROC is equivalent to the probability of the system being able to choose properly between two items, one randomly selected from the set of relevant items, and one randomly selected from the set of non-relevant items. Hence, it allows one to abstract from any particular precision-recall proportion. Specifically, for an ordered list of predicted matchings R, AUROC is defined as:

$$AUROC = \frac{1}{|R|} \sum_{i=1}^{F} (s_i - i) \; , \tag{12}$$

where the probability score s_i is indicated by rank of the i-th true positive in R, and F is the number of false positives in R. In particular, if all relevant matchings appear before all nonrelevant matchings in the list, one will have a perfect ROC curve and $AUROC = 1$.

D. Recommendation Methods under Evaluation

We have compared the accuracy of our method to the accuracy of state-of-the-art tensor-based processing methods presented in the relevant literature. In order to perform such a comparison, we have developed our implementations of N-way random indexing (NRI) [14], HOSVD [15], joint feature mapping via tensor product [31], and a typical SVD-based matrix factorization [32]. The matrix factorization, herein referred to as '*MF (matchings)*', was performed on a matrix containing information about known Web service matchings, as it is not possible to unambiguously encode more relations in such a structure. As a consequence, we used *MF (matchings)* as a baseline method allowing to distinguish whether the use of tensor-based algorithms provides any significant benefit compared with classical matrix factorization.

We have also evaluated feature mapping via tensor product, herein referred to as '*Feature Mapping*', as it has been reported in [31] and followed in [33] that such a model allows to exploit not only the direct relations between individual objects but also the associated textual features. In this paper, in order to adapt the algorithm to the Web service matching scenario, in accordance with [33] we represent each pair of Web services (i.e., a request r and an offer o) as an outer product $r \otimes o$ of two corresponding vectors represented in a feature space of

Web service descriptions. Subsequently, the resultant tensor is defined as $\sum_i \sum_j (r_i \otimes o_j) m_{i,j}$, where $m_{i,j}$ indicates whether the matching between r_i and o_j is relevant ($m_{i,j} = 1$), nonrelevant ($m_{i,j} = -1$) or unknown ($m_{i,j} = 0$). Finally, Principal Component Analysis (PCA) is applied in order to reduce the noise and extract the most salient features.

The HOSVD algorithm has been performed on RDF data expressed in a form of a 3-rd order tensor, in which every predicate is represented as an adjacency matrix – forming the slice of the tensor – between subjects and objects. This model has been already used in the relevant literature [11][12] for the task of multirelational statistical learning. Contrarily, we were not able to obtain any meaningful results by applying solely HOSVD on a tensor model build from tuples (as presented in Section III-A), thus we omitted these results in the presented evaluation.

Finally, we have evaluated the effectiveness of the proposed covariance-based multilinear filtering (CMF) regardless of the underlying data representation model. For that reason we additionally present MAP and AUROC results of the experiments performed solely on a NRI-reduced vector space and the probabilistic state tensor introduced in Section IV-B, herein referred to as '*Probabilistic ST*'. By that means, the ability of CMF to process tensor spaces of reduced dimensionality is verified.

All the above described methods have been evaluated using the same data (correspondingly in the form of a n-tuple or RDF) as their input. The combinations of parameters (such as the k-cut or the core tensor size) that lead to the best recommendation quality (i.e., the highest AUROC value) were considered optimal, and used in experiments illustrated in this paper.

For purposes of evaluated methods we have used the framework introduced in Section III in order to construct both tuple-based and RDF-based internal representations of the input data. In particular, the n-tuple representation of the dataset is processed into the 3-rd order tensor structure of size $(2 \times 110 \times 615)$ in order to store the data concerning 4178 n-tuples. As described in Section III-A, the first tensor mode contains information about the relevance (i.e., relevant, non-relevant). Subsequently, the second tensor mode – concerning the *queries* – is constructed using vectors of length 110, while the third mode – concerning the *offers* – is built using vectors of length 615.

In the case of data given as RDF statements, the tensor of the size $(1084, 7, 1909)$ is constructed – according to Section III-B – to store the data on 15537 triples. In the presented experiments, we also investigate the standard collaborative filtering approach using the request-offer matrix of size $(42, 1043)$ containing only the data on service relevance modeled using 4178 non-zero values from the set $\{-1, 1\}$.

VI. Experimental Results

The results of our evaluation performed, using MAP and AUROC measures, are presented in Figures 2 and 3, respectively. The comparison has been performed with the use of different training ratios tr, ranging from 0.05 to 0.9. As it has been confirmed experimentally, the introduced algorithm – CMF – allows to achieve higher quality matchings, both in terms of MAP and AUROC and for all training ratios, as compared to other evaluated methods.

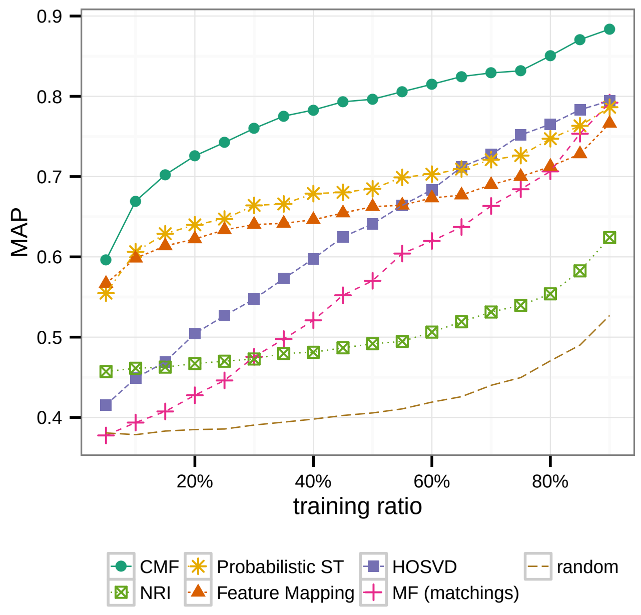

Figure 2. The MAP results.

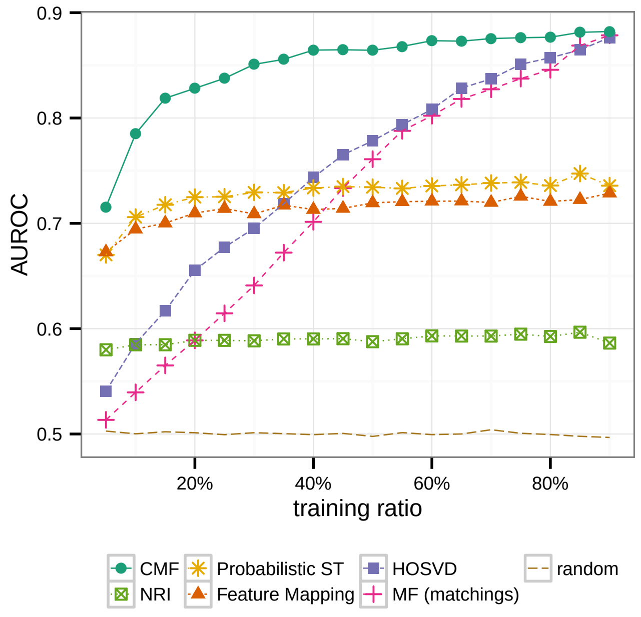

Figure 3. The AUROC results.

As it has been shown, the accuracy of *MF (matchings)* is almost linearly dependent on the training ratio. Particularly, for the smallest $tr = 0.05$ matrix factorization achieves the lowest score – similar to a random one, while for the highest $tr = 0.9$ the obtained results are comparable to other best performing methods (i.e., in terms of AUROC).

An algorithm generating random recommendations obtained $AUROC = 0.5$ for all tr – as expected due to its probabilistic interpretation, what additionally confirms the reliability of the AUROC measure. On the other hand, due to correspondingly smaller test set – more precisely smaller number of relevant matchings for every offer – for higher tr the MAP values for a random algorithm are also respectively higher. Therefore, for comparison, we included the random method in the presented evaluation.

It may be also observed that the HOSVD algorithm, performed on a 3-rd order tensor build from RDF statements, enabled us to obtain slightly higher results – although still statistically significantly higher – than *MF (matchings)*. Nevertheless, the results of HOSVD, even with optimally adjusted size of the core tensor, are still heavily dependent on tr. Particularly, for $tr < 0.1$ HOSVD performed only slightly better than the random method despite processing all of the semantic information extracted from SAWSDL files.

The algorithm based on feature mapping via tensor product, compared to the baseline *MF matchings*, clearly enabled to obtain higher AUROC and MAP values for smaller training ratios (i.e., $tr < 0.4$). In opposition, for denser train set the results are not so apparently conclusive. Specifically, *Feature Mapping* achieves higher quality results in terms of MAP than the baseline method for almost all tested tr. At the same time, in the case of the AUROC measure its performance is almost constant (with only relatively small gains for higher tr) and significantly inferior to a simple matrix factorization. Such a finding may be caused by the fact that AUROC probabilistically reflects the system's performance (see Section V-C) – which is rather independent from the amount of behavioral data (herein – known matchings) in case of content-based methods. On the other hand, the MAP measure takes into account the number of relevant matchings in the test set (as it has been shown on the case of random matchings).

Although the main purpose of NRI is to reduce the dimensionality of the input tensor, and not multiple factor analysis, we included this algorithm in the evaluation as the introduced CMF method is partially based on the NRI concept. As shown in Figure 3, the ability of NRI to provide high quality recommendations is independent of the number of input training matches – probably due to the fact that it merely reflects the co-occurrences of the terms in the requests and in the offers. It should be also noted that although the addition of scaling and L^2-normalization – in *Probabilistic ST* method – enabled to significantly improve the performance of tensor-based processing, such an algorithm still does not allow one to provide higher quality results than HOSVD or even *MF (matchings)* in case of higher tr. Additionally, we have performed experiments using a 3-rd order tensor build from RDF statements and processing methods such as NRI, *Probabilistic ST* and CMF. However, due to definitively lower quality of the provided recommendations we have omitted these results from the final evaluation so as not to obscure the presented results.

Therefore, it may be stated that in the application scenario investigated in this paper, the tuple-based probabilistic tensor modeling combined with covariance-based multilinear filtering enables to outperform other tensor-based methods, regardless of the amount of known matchings (herein depicted by tr).

VII. Conclusion and Future Work

The experimental evaluation results presented in the paper are expressed in terms of AUROC and MAP results. It is worth stressing that the evaluation has been done using the standard Information Retrieval methodology that assumes partitioning of the dataset on the training and testing sets in such a way that the data used for testing the performance cannot be used to learn or tune the model. Quite surprisingly, such a methodologically correct approach differs from the evaluation methodology used by the authors that have taken part in the S3 contest, as they frequently use the same data on matchings between services for both for the matchmaker system parameters tuning (e.g., by means of the cross-validation approach) and for the final performance evaluation. The results presented in the paper indicate the superiority of the proposed combination of the tuple-based probabilistic tensor modeling and the covariance-based multilinear filtering over other tensor-based methods, including NRI and HOSVD-based RDF processing – the superiority that is clearly visible regardless of the amount of matchings included in the training set.

Finally, it has to be stressed that contrarily to the state-of-the-art algorithms such as [3]–[5] the proposed Semantic Service Recommendation System does not rely on any kind of predefined rules customized for SAWSDL matchmaking. As introduced in Section IV, the recommendation engine is virtually unconstrained regarding any data structure, and thus it may be easily applied in other domains, as already shown in [25]. For that reason, for future work we plan to extend our research to address other semantic matchmaking tasks. Another potential directions of the further research would be an extended use of the referenced ontologies, and conducting the experiments involving the 4-graded relevance information, in addition to the presented herein binary relevance.

Acknowledgement

This work is supported by the Polish National Science Centre, grant DEC-2011/01/D/ST6/06788, and by the Polish National Centre for Research and Development, grant PBS1/B3/14/2012.

References

[1] T. Wang, D. Wei, J. Wang, and A. Bernstein, "SAWSDL-iMatcher: A Customizable and Effective Semantic Web Service Matchmaker," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 9, no. 4, 2012, pp. 402–417. [Online]. Available: http://www.websemanticsjournal.org/index.php/ps/article/view/239

[2] P. Plebani and B. Pernici, "URBE: Web Service Retrieval Based on Similarity Evaluation," Knowledge and Data Engineering, IEEE Transactions on, vol. 21, no. 11, nov. 2009, pp. 1629 –1642.

[3] M. Klusch, P. Kapahnke, and I. Zinnikus, "Adaptive Hybrid Semantic Selection of SAWSDL Services with SAWSDL-MX2." Int. J. Semantic Web Inf. Syst., 2010, pp. 1–26.

[4] M. Klusch and P. Kapahnke, "The iSeM matchmaker: A flexible approach for adaptive hybrid semantic service selection," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 15, 2012, pp. 1–14.

[5] S. Schulte, U. Lampe, J. Eckert, and R. Steinmetz, "LOG4SWS.KOM: Self-Adapting Semantic Web Service Discovery for SAWSDL," in Proceedings of the 2010 6th World Congress on Services, ser. SERVICES '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 511–518. [Online]. Available: http://dx.doi.org/10.1109/SERVICES.2010.40

[6] Semantic Service Selection Contest SAWSDL track dataset (SAWSDL-TC). http://projects.semwebcentral.org/projects/sawsdl-tc/. [retrieved: February, 2015]

[7] Semantic Service Selection Contest webpage. http://www-ags.dfki.uni-sb.de/~klusch/s3/index.html. [retrieved: February, 2015]

[8] N. Vervliet, O. Debals, L. Sorber, and L. De Lathauwer, "Breaking the curse of dimensionality using decompositions of incomplete tensors: Tensor-based scientific computing in big data analysis," Signal Processing Magazine, IEEE, vol. 31, no. 5, Sept 2014, pp. 71–79.

[9] A. Cichocki, "Era of big data processing: A new approach via tensor networks and tensor decompositions," CoRR, vol. abs/1403.2048, 2014. [Online]. Available: http://arxiv.org/abs/1403.2048

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

[11] M. Nickel and V. Tresp, "An Analysis of Tensor Models for Learning on Structured Data," in Machine Learning and Knowledge Discovery in Databases, ser. Lecture Notes in Computer Science, H. e. Blockeel, Ed. Springer Berlin Heidelberg, 2013, vol. 8189, pp. 272–287. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40991-2_18

[12] T. Franz, A. Schultz, S. Sizov, and S. Staab, "Triplerank: Ranking semantic web data by tensor decomposition," in The Semantic Web - ISWC 2009, ser. Lecture Notes in Computer Science, A. Bernstein, Ed. Springer Berlin Heidelberg, 2009, vol. 5823, pp. 213–228.

[13] O. Chapelle, B. Scholkopf, and A. Zien, "Introduction to semi-supervised learning," in Semi-Supervised Learning, O. Chapelle, B. Scholkopf, and A. Zien, Eds. The MIT Press, 2006, pp. 1–8. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/21243728

[14] F. Sandin, B. Emruli, and M. Sahlgren, "Incremental dimension reduction of tensors with random index," CoRR, Mar. 2011, pp. 240–56. [Online]. Available: http://arxiv.org/abs/1103.3585

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

[16] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," ACM Trans. Inf. Syst., vol. 22, no. 1, Jan. 2004, pp. 5–53. [Online]. Available: http://doi.acm.org/10.1145/963770.963772

[17] C. D. Manning, P. Raghavan, and H. Schtze, Introduction to information retrieval. Cambridge University Press, NY, USA, 2008.

[18] T. Mitchell, Machine Learning. McGraw-Hill, New York, NY, USA, 1997.

[19] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," SIAM Review, vol. 51, no. 3, 2009, pp. 455–500. [Online]. Available: http://dx.doi.org/10.1137/07070111X

[20] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," Psychometrika, vol. 31, no. 3, 1966, pp. 279–311. [Online]. Available: http://dx.doi.org/10.1007/BF02289464

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

[22] L. Grasedyck, D. Kressner, and C. Tobler, "A literature survey of low-rank tensor approximation techniques," ArXiv e-prints, Feb. 2013, pp. 53–78.

[23] R. Bro and A. K. Smilde, "Centering and scaling in component analysis," Journal of Chemometrics, vol. 17, no. 1, 2003, pp. 16–33. [Online]. Available: http://dx.doi.org/10.1002/cem.773

[24] T. Cohen, R. Schvaneveldt, and D. Widdows, "Reflective Random Indexing and indirect inference: a scalable method for discovery of implicit connections." Journal of biomedical informatics, vol. 43, no. 2, Apr. 2010, pp. 240–56. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/19761870

[25] A. Szwabe, P. Misiorek, and M. Ciesielczyk, "Multilinear Filtering Based on a Hierarchical Structure of Covariance Matrices," Schedae Informaticae, vol. 24, 2015, in press. [Online]. Available: http://ncn6788.cie.put.poznan.pl/images/ncn6788-tfml2015.pdf

[26] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos, "Incremental tensor analysis: Theory and applications," ACM Trans. Knowl. Discov. Data, vol. 2, no. 3, Oct. 2008, pp. 11:1–11:37. [Online]. Available: http://doi.acm.org/10.1145/1409620.1409621

[27] I. Pilászy, D. Zibriczky, and D. Tikk, "Fast als-based matrix factorization for explicit and implicit feedback datasets," in Proceedings of the Fourth ACM Conference on Recommender Systems, ser. RecSys '10. New York, NY, USA: ACM, 2010, pp. 71–78. [Online]. Available: http://doi.acm.org/10.1145/1864708.1864726

[28] R. A. Bailey, Design of Comparative Experiments, ser. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2008.

[29] C. D. Manning, P. Raghavan, and H. Schutze, Introduction to information retrieval. New York, NY, USA: Cambridge University Press, 2008, no. c. [Online]. Available: http://www.langtoninfo.com/web_content/9780521865715_frontmatter.pdf

[30] T. Fawcett, "An introduction to roc analysis," Pattern Recogn. Lett., vol. 27, no. 8, Jun. 2006, pp. 861–874. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2005.10.010

[31] J. Basilico and T. Hofmann, "Unifying collaborative and content-based filtering," in Proceedings of the Twenty-first International Conference on Machine Learning, ser. ICML '04. New York, NY, USA: ACM, 2004, pp. 9–. [Online]. Available: http://doi.acm.org/10.1145/1015330.1015394

[32] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," Computer, 2009, pp. 42–49. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5197422

[33] S.-T. Park and W. Chu, "Pairwise Preference Regression for Cold-start Recommendation," in Proceedings of the Third ACM Conference on Recommender Systems, ser. RecSys '09. New York, NY, USA: ACM, 2009, pp. 21–28. [Online]. Available: http://doi.acm.org/10.1145/1639714.1639720