Exploiting a Determinant-based Metric to Evaluate a Word-embeddings Matrix of Items

Ludovico Boratto, Salvatore Carta, Gianni Fenu, Roberto Saia
Dipartimento di Matematica e Informatica
Università di Cagliari
Via Ospedale 72, 09124 - Cagliari, Italy
Email: {ludovico.boratto, salvatore,fenu,roberto.saia}@unica.it

***Abstract*—In order to generate effective results, it is essential for a recommender system to model the information about the user interests (*user profiles*). A profile usually contains preferences that reflect the recommendation technique, so collaborative systems represent a user with the ratings given to items, while content-based approaches assign a score to semantic/text-based features of the evaluated items. Even though semantic technologies are rapidly evolving and word embeddings (i.e., vector representations of the words in a corpus) are effective in numerous information filtering tasks, at the moment collaborative approaches (such as SVD) still generate more accurate recommendations. However, this might happen because, by employing classic profiles in form of vectors that collect all the preferences of a user, the power of word embeddings at modeling texts could be affected. In this paper we represent a profile as a matrix of word-embedding vectors of the items a user evaluated, and present a novel determinant-based metric that measures the similarity between an unevaluated item and those in the matrix-based user profile, in order to generate effective content-based recommendations. Experiments performed on three datasets show the capability of our approach to perform a better ranking of the items w.r.t. collaborative filtering, both when compared to a latent-factor-based approach (SVD) and to a classic neighborhood user-based system.**

***Index Terms*—data mining; semantic analysis; recommender systems; word embeddings; metrics;**

I. Introduction

The rapid growth of the number of companies that perform their activities in the so-called e-commerce environment generates an enormous amount of information, which must be correctly exploited in order to improve the quality and efficiency of the sales criteria [1]. This problem is effectively faced by *Recommender Systems* [2], which filter the information about their customers in order to get useful elements to produce effective suggestions to them. In order to perform this task, such systems need to define a set of profiles that model the preferences of their customers, and in this context the collaborative techniques, who usually represent a user with the ratings given to the items she evaluated, are in most of the cases more effective than the other techniques. The problem of the data sparsity, a side effect of the collaborative techniques, is effectively faced by the latent-factor-based techniques, such as SVD [3], which nowadays represent the state-of-the-art in this field.

Differently from the collaborative techniques, the content-based ones analyze each item (usually its textual description) by using a semantic tool able to convert it into a set of features. Even though semantic technologies are moving at a very rapid pace and state-of-the-art solutions, such as deep learning algorithms able to extract *word embeddings* [4] from a text corpus, have been successfully employed in numerous information filtering and retrieval tasks, at the moment collaborative filtering approaches continue to be more accurate at generating recommendations [5].

On the one hand, having a user profile represented by a unique vector of features guarantees quick comparisons (e.g., in a content-based system, a user profile can be easily compared to the vector that represents an item with a simple metric like the cosine similarity); moreover, the state-of-the-art tools allow us to extract the word embeddings from a corpus, in order to create a vector representation for each word (Google's *word2vec*[1]) or for each document (a *word2vec* extension usually known as *doc2vec*). In summary, such embeddings are able to reflect the semantic similarities between words based on their sentence-internal contexts in the document corpus. However, on the other hand, these vectors need to be summed to obtain a unique vector that represents the user profile (*additive compositionality* property) [6], and the power of this approach might be weakened by the fact that tens of vectors might have to be added to obtain a unique representation of a user, with the risk of losing the specific information of each user's evaluated item.

Therefore, the idea behind this paper is to *represent a user profile as a matrix of word-embeddings*, where each row is represented by an item a user positively evaluated, and to *develop a metric that evaluates the correlation between an item not evaluated by her and those in the matrix-based user profile*. The proposed metric is based on the concept of linear independence of the vector, by following the idea that *if an unevaluated item is linearly dependent to those in a matrix-based user profile, positively evaluated by the user, its features match with the feature of these, and we can recommend it to the user*.

Indeed, it should be observed how the vector representation of the items leads to some parallelisms between the concepts of *cosine similarity* and *linear independence*[2], since both of

[1] http://deeplearning4j.org/word2vec

[2] A set of vectors is linearly independent if no vector in the set can be defined as a linear combination of the other ones.

them allow us to evaluate the relation between two vectors on the basis of their position in the space. However, there is a substantial difference between these two approaches of evaluation, because the *cosine similarity* evaluates the similarity between two vector entities (items or user profiles expressed in terms of single vectors), while the evaluation of the *linear independence*, obtained by measuring the determinant of a matrix of vectors, composed by the vector representation of the items in a user profile and that of an item to evaluate added as last row, allow us to evaluate the similarity relation between all entities (all involved items).

On the basis of the above considerations, in order to generate effective content-based recommendations, this paper formalizes a novel criterion of evaluation based on the concept of matrix determinant. The proposed metric has been validated through a series of experiments, performed by using three real-world datasets, where we compared its effectiveness to rank the unevaluated items of a user, on the basis of the matrix composed by using the items positively evaluated by her, w.r.t. two approaches at the state of the art, such as the *user-based collaborative filtering* and *Singular Value Decomposition* (SVD).

The contributions of our work are the following:

- definition of a compositional criterion ($cdet$) able to extract a *determinant-based* value from a non-square matrix, by calculating the mean value of the determinant of a set of sub-matrices;
- formalization of the *Linear Independence Rate* (LIR) metric on the basis of the $cdet$ value, which allows us to evaluate the similarity between a user profile and an item in terms of linear independence of the vector representation of the involved items;
- evaluation of the capability of the LIR metric to measure the similarity between an unevaluated item and those in a user profile and to rank the items to recommend to a user, by comparing its performance with those of two state-of-the-art approaches, and by using three real-world datasets.

In the rest of this paper, we first introduce the literature related with the proposed strategy (Section II), continuing to define the adopted notation and the problem definition (Section III), and the approach used to define and implement the proposed new metric (Section IV). We complete the paper by presenting the results of the performed experiments (Section V) and some concluding remarks (Section VI).

II. Background and Related Work

In this section we briefly review some main concepts closely related with the present work.

User Profiling. In the e-commerce environment recommender systems play a determinant role. Their first implementations were based on the so-called *Collaborative Filtering* approach [7], [8], [9], [10], which assumes that users have similar preferences on a item, if they already have rated other similar items [11]. An alternative approach, known as *Content-based*, recommends items whose content is similar to that of the items previously evaluated by the user [12], [13].

The early content-based systems used relatively simple retrieval models, such as the Vector Space Model, with the basic TF-IDF weighting [14], [15], [16], [17], a spatial representation of the textual description of the items, where each of them is represented by a vector in a $n\text{-}dimensional$ space, and each dimension is related to a term from the overall vocabulary of a specific document collection. In other words, every document is represented as a vector of term weights, where the weight indicates the degree of association between the document and the term. In this way, it is possible to evaluate the similarity between items by comparing their vector representation, e.g., through the *cosine similarity* (CS) metric formalized in the Equation (1), a widespread measure based on the cosine of the angle between vectors.

$$C_S = cos(\vec{v_1}, \vec{v_2}) = \frac{\vec{v_1} \cdot \vec{v_2}}{\parallel \vec{v_1} \parallel \cdot \parallel \vec{v_2} \parallel} \tag{1}$$

Due to the fact that the approaches based on a simple bag of words are not able to perform a semantic disambiguation of the words in the item descriptions, the content-based recommender systems started to implement more sophisticate approaches of text analysis, such as those able to extract the semantic meaning from the item descriptions, in order to improve the system accuracy [12].

There are several approaches to create user profiles [18], some of them focus on *short-term* user profiles that capture features of the user's current search context, while others accommodate *long-term* profiles that capture the user preferences over a long period of time. As shown in [19], compared with the *short-term* user profiles, the use of a *long-term* user profile generally produces more reliable results, at least when the user preferences are fairly stable over a long time period. It should be noted that, regardless of the approach used to define the user profiles, almost all the state-of-the-art strategies take into account in a global manner the information related with the evaluated items, either by considering all their explicit features (e.g., content-based and collaborative user-based), or by exploiting latent characteristics (e.g., collaborative latent-factor-based). This means that a user profile is usually expressed as a single vector.

Neural Word Embeddings. In recent years, the interest for the neural language models has been increasing. This has happened because they have proved to be effective in order to move the representation of the words into a lower dimensional dense vector space via a hidden layer [6]. With respect to the canonical language models, they are able to provide a better representation of the words [20], by extracting the syntactic information instead of the simple bag-of-context, performing some non-linear transformations. The close words are considered semantically similar in this low dimensional vector space. The vectors used in the context of this work to define the user profile matrices are based on the so-called neural word embeddings, a numerical representation of the words performed by *word2vec*[3]. It represents a powerful tool for the developers, since it is able to train word vectors from a large document corpus, such that words with similar contexts end up having similar vectors.

In order to build its models, *word2vec* exploits a very simple neural net with a single hidden layer, and these models are used to produce the so-called word embeddings [21], and they are trained to obtain the linguistic contexts of all the words in a text corpus. Subsequently, they express each word as a numerical vector (usually composed by several hundred elements), which gives us information about the relation with the other words: this vector represents the *hidden layer* of the neural net. Word embeddings are largely employed nowadays in several NLP tasks, such as the representation of sentences and paragraphs [22], [23], relational entities [24], general text-based attributes, descriptive text of images [25], and nodes in graph structure [26]. Moreover, they have been recently adopted in recommender systems too, with promising results that, however, do not outperform the collaborative approaches [5].

Doc2vec (also known as *paragraph2vec* or *sentence embeddings*) is an extension of *word2vec* that represents a text of arbitrary length through a single numeric vector, instead of producing a set of *word2vec* vectors. In this paper, *doc2vec* is employed to generate the embedding of the items evaluated by the users.

Matrices, Linearity, and Vector Spaces. The concepts of matrix determinant, linearity, and vector spaces, cover a primary role in our context, since through them we can formalize and prove the correctness of a new metric able to measure the relations of similarity, in terms of linear dependence, between the vector representation of the items in a user profile and that of an item to evaluate. The matrix determinant is a mathematical function that assigns a number to every square matrix, so its domain is the set of square matrices, and its range is the set of numbers; more formally, we can write that $det : \Re^n \times \ldots \times \Re^n \rightarrow \Re$. Regardless of the method used to calculate the determinant (det) of a square matrix $N \times N$ (e.g., one of them is the *Leibniz* formula shown in Equation (2), where sgn is the sign function of permutations σ in the permutation group S_N, which returns +1 and -1, respectively for even and odd permutations), this value is related to the linear dependence relations between the vectors that compose the matrix.

$$det \begin{vmatrix} m_{1,1} & m_{1,2} & \ldots & m_{1,N} \\ m_{2,1} & m_{2,2} & \ldots & m_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ m_{N,1} & m_{N,2} & \ldots & m_{N,N} \end{vmatrix} = \sum_{\sigma \in S_N} sgn(\sigma) \prod_{i=1}^{N} m_{i,\sigma_i} \quad (2)$$

The independence of the N vectors can be verified by calculating the determinant (det) of the $N \times N$ matrix built by placing, one after the other, the n-tuples that express the vectors in a certain base. The vectors are independent when the determinant of the matrix is different from zero. A vector space (or linear space) is a mathematical structure composed by a collection of vectors that may be added together and multiplied (or, more correctly, scaled) by numbers called scalars. In other words, a vector space V is a set that is closed under finite vector addition and scalar multiplication. A vector sub-space (or linear sub-space) is a vector space that represents a subset of some other vector space of higher dimension.

[3] http://deeplearning4j.org/word2vec

III. Notation and Problem Definition

In this section we present the adopted notation and the problem definition.

Notation. We are given a set of users $U = \{u_1, \ldots, u_N\}$, a set of items $I = \{i_1, \ldots, i_M\}$, and a set R of ratings used to express the user preferences (e.g., $R = [1, 5]$ or $R = \{like, dislike\}$). The set of all possible preferences expressed by the users is a ternary relation $P \subseteq U \times I \times R$. We denote as $P_+ \subseteq P$ the subset of preferences with a positive value (i.e., $P_+ = \{(u, i, r) \in P | r \geq \overline{r} \vee r = like\}$) and as $P_- \subseteq P$ the subset of preferences with a negative value (i.e., $P_- = \{(u, i, r) \in P | r < \overline{r} \vee r = dislike\}$), where $\overline{r}$ indicates the mean value (in the previous example, $\overline{r} = 3$). We also denote as $I_u = \{i \in I | \exists(u, i, r) \in P_+ \wedge u \in U\}$ the set of items in the profile of a user u, for which there is a positive preference, and as $\hat{I}_u = \{i \in I | \exists(u, i, r) \in P_- \wedge u \in U\}$ the set of items in the profile of a user u, for which there is a negative preference.

Let $BoW = \{t_1, \ldots, t_M\}$ be the bag of words used to describe the items in I, we define as $V = \{v_1, \ldots, v_M\}$ the set of word embeddings vectors that represent the items in I, so $|V| = |I|$. We denote as Ξ_u the user profile matrix of size $|I_u| \times L$, were $|I_u|$ is the number of items, positively evaluated, in the profile of the user u, and L is the number of layers of the neural net (i.e., the cardinality of each vector).

We also define a profile age value $\alpha = [1, L\text{-}1]$ used to set the maximum number of items in user profile to take into account during the process of definition of the user models. In other words, through it we can limit the number of items to take into account in each user profile, i.e., those involved in the compositional determinant process. On the basis of this last definition, we introduce the notation $\Xi_{u,\alpha}$ to define the profile of the user $u \in U$, consisting of not more than α recent items $i \in I_u$, chronologically ordered, where we add an item $i \in I$ to evaluate as the last element.

Problem Definition. The aim is to get, for each item $i \in I$ not evaluated by a user, a *Linear Independence Rate* (LIR) able to measure the linear independence of its vector representation, in the space of the user profile (matrix $\hat{\Xi}_{u,i}$), i.e., a kind of global similarity with the other items positively evaluated by the user. Our goal[4] is to recommend an item i^* such that:

$$i^* = \underset{(i \in I, u \in U)}{\operatorname{argmin}} \; LIR(i, u) \quad (3)$$

[4] More specifically, in this paper we focus on the *top-n* recommendation problem, by selecting the n items with the lowest LIR value.

IV. Approach

In this section, we present the four steps performed to rank a set of items to recommend to a user.

- **Item Vectorization**: conversion of the set of items I in a set of vectors V, by using the *doc2vec* tool, a two-layer neural net approach at the state of the art;
- **Compositional Determinant**: definition of the compositional approach used to extract the determinant information ($cdet$) from the square and non-square user profile matrices;
- **Linear Independence Rate**: formalization of a new LIR metric, based on the $cdet$ information, able to evaluate the global similarity between a single item and all the items in a user profile;
- **Ranking Algorithm**: definition of the algorithm used to generate a ranked list of items, on the basis of the LIR metric.

In the following, we will describe in detail how each step works.

A. Item Vectorization

Given a set of documents (i.e., the descriptions of the items in the set I), in this first step we define and train a model by using the *doc2vec* neural net (its configuration parameters are listed and tested in a set of experiments reported in Section V-A). The result is a set V of vectors that represents the vector representation of the items in the set I, whose cardinality depends on the number L of the neural net layers (*layerSize* parameter). Such vectors are the semantic representation of each item in the context of the entire document corpus (i.e., the textual description of all the items).

Given a user $u \in U$ and the parameter α that indicates how many items we should consider in her profile, the output of this step is her user profile $\Xi_{u,\alpha}$, represented as a matrix that contains the word embeddings of the last α items she evaluated (sorted in chronological order), plus an empty row (that will be employed during the filtering process to evaluate the items not yet considered).

B. Compositional Determinant

Giving that there is not a mathematical definition of determinant of a non-square matrix, here we define a new operation based on the determinant concept, named *compositional determinant* ($cdet$). Through it we are able to calculate the average of the determinants of all square sub-matrices defined by decomposing the user profile matrix $\Xi_{u,\alpha}$, of size $(\alpha+1)\times L$, in $\left\lfloor \frac{L}{(\alpha+1)} \right\rfloor$ square sub-matrices of size $S \times S$, where S is defined according with the Equation 4.

$$S = \begin{cases} (\alpha+1), \; if\ (\alpha+1) \leq L \\ L, \; otherwise \end{cases} \tag{4}$$

On the basis of the S value, we calculate the determinant of each $S \times S$ sub-matrix, moving on $\Xi_{u,\alpha}$ by using a step S (i.e., without overlaps), calculating at the end, the average value of the obtained results. We can note that the maximum size of the square sub-matrices is the cardinality of the vectors, i.e., the value L of the *layerSize* parameter used to build the *doc2vec* model. It means that when we have $(\alpha+1) > L$, the process uses only the last L vectors of the user profile. It should be observed that the typical value of the L parameter (i.e., the size of the vectors) is in the order of several hundreds, for this reason it does not introduce significant limitations in the proposed approach, because it is reasonable to model the user preferences by using only the last evaluations, when they are some hundreds. This process has been exemplified in the Equation 5, where we hypothesize the values $\alpha = 1$ and $L = 6$, then $\left\lfloor \frac{6}{1+1} \right\rfloor = 3$ sub-matrices of size 2×2.

$$cdet \begin{pmatrix} a & b & c & d & e & f \\ g & h & i & l & m & n \end{pmatrix} = \frac{det\begin{pmatrix} a & b \\ g & h \end{pmatrix} + det\begin{pmatrix} c & d \\ i & l \end{pmatrix} + det\begin{pmatrix} e & f \\ m & n \end{pmatrix}}{3} \tag{5}$$

In spite of the fact that the mean value of the sub-matrices determinants does not have a canonical mathematical meaning, this value acquires one in our context, because it reports the linear independence between vector segments that characterize the same subset of features, as demonstrated in Theorem 1.

Theorem 1: Given the vector space of the features that characterize the vector representation of items in a domain, we can express it as sum of two or more sub-spaces that characterize subsets of features.

Proof 1: A vector space can be defined as a combination of sub-spaces by using a decomposition approach, e.g., given a space $\Re^3 = x\text{-}axis + y\text{-}axis + z\text{-}axis$, we can write any $\vec{w} \in \Re^3$ as a linear combination $c_1\vec{v}_1 + c_2\vec{v}_2 + c_3\vec{v}_3$ (where $\vec{v}$ is a member of the axis, and $c \in \Re$), as shown in Equation 6.

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = 1 \cdot \begin{pmatrix} w_1 \\ 0 \\ 0 \end{pmatrix} + 1 \cdot \begin{pmatrix} 0 \\ w_2 \\ 0 \end{pmatrix} + 1 \cdot \begin{pmatrix} 0 \\ 0 \\ w_3 \end{pmatrix} \tag{6}$$

On the basis of the consideration that $\Re^3 = x$-axis$+$ y-axis$+z$-axis, we can prove the consistency of the proposed compositional approach, since $cdet$ represents the mean value of the determinants calculated on a series of square sub-matrices composed by segments of vectors that belong to the same vector sub-space of the items features space.

It simply means that, by the $cdet$ information, we are able to evaluate subsets of features (in terms of linear dependence between their vector representations), and the calculation of the mean value of these results gives us a single value that reports the relations of similarity in the entire space of the features, as previously demonstrated.

C. Linear Independence Rate

On the basis of the $cdet$ operation, defined in Section IV-B, here we formalize the *Linear Independence Rate* (LIR), the metric that we will use to evaluate the similarity between an unevaluated item and the items in a user profile. In geometric terms, the vector of length L, which represents an item, is more similar to the others in the user profile when it is a linear combination of one or more of them. For instance, considering $v_3 = (0, 2, 0)$ the vector representation of an item to evaluate, and $v_1 = (1, 0, 0)$ and $v_2 = (0, 1, 0)$ the vector representation of the items (positively evaluated) in a user profile, we observe that v_3 is a linear combination of $v2$, i.e., v_3 represents an item with the same features of the item v_2, but with a different value. This is underlined by the zero value of the determinant calculated for the matrix defined by using these three vectors, i.e., $det\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 0 \end{pmatrix} = 0$, which becomes 2 in case of a vector v_3 that characterizes different features (e.g., $v_3 = (0, 0, 2)$), so we have $det\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{pmatrix} = 2$.

To summarize, this approach allows us to evaluate the similarity on the basis of the shared elements between vectors of features, as well as on the basis of their values, e.g., two vectors $v_1 = [1, 0, 0]$ and $v_2 = [0, 1, 0]$ do not have any common feature, while the vectors $v_3 = [1, 0, 0]$ and $v_4 = [2, 1, 0]$ have a common feature, although with a different value. By adopting the compositional process explained in Section IV-B, we evaluate the *Linear Independence Rate* (LIR) of an item i by placing its vector representation as the last element of the user profile, calculating the compositional determinant of the matrix $\Xi_{u,\alpha}$ of size $(\alpha+1) \times L$ as shown in the Equation 7. It should be noted that, on the basis of the meaning of the determinant measure, we consider closer to the preferences of a user the items with a $cdet$ value as close as possible to zero, thus we take into account the absolute value of $cdet$.

$$LIR(i, u) = |cdet(\Xi_{u,\alpha})| \tag{7}$$

D. Ranking Algorithm

On the basis of the information extracted from the user profiles through the proposed LIR metric, here we present the Algorithm 1, used to generate a ranked list of items on the basis of this information. It takes as input all the items i in the training set I, a user u, the number of layers L to use in the process of vectorization performed by *doc2vec*, and the age parameter α that determines how many recent items we have to take into account during the evaluation of the items (i.e., the last α items in the user profile, positively evaluated by u). It returns as output a list Rec of the items not evaluated by the user u, ranked on the basis of our LIR metric.

In *steps* from *2* to *4*, we verify if the value assigned to the $(\alpha+1)$ parameter is greater than L (i.e., the maximum size allowed for the square sub-matrices used in the compositional process, as described in Section IV-B), limiting it to $(L-1)$ when it happens. In *step 5* we create the set V of vectors of each item $i \in I$, by using the *doc2vec* tool. In *step 6* we define the matrix M, composed by the vector representation of the last α items of size L in the profile of the user u, and complete this process by adding, as the last row of the matrix, an empty vector of the same L size (*step 7*). In *steps* from *8* to *15*, we process all items $i \in I$ that the user u has not evaluated yet (*step 9*). For each of them, we get its vector representation (*step 10*), and in the *step 11* we use it to fill the last row of

Algorithm 1 *Items evaluation and ranking*

Input: I=Set of items, u=User, L=Layers, α=Age
Output: Rec = List of ranked items
1: **procedure** GETRANKEDITEMS(I,u,α)
2: **if** $(\alpha+1) > L$ **then**
3: $\alpha = (L-1)$
4: **end if**
5: V=Doc2VecVectorization(I)
6: M=DefineUserProfileMatrix(V,I_u,α)
7: M=AddEmptyVectorAsLastRow(M);
8: **for** each i in I **do**
9: **if** i NOT IN I_u AND i NOT IN $\hat{I}_u$ **then**
10: v=GetItemVector(V, i)
11: M=FillLastMatrixRow(M, v)
12: LIR=CalculateLIR(M)
13: $Rec \leftarrow (i, LIR)$
14: **end if**
15: **end for**
16: Return $SortItemsByDescLIR(|Rec|)$
17: **end procedure**

the matrix M (i.e., the empty one). The next operation is the calculation of the LIR value (*step 12*) of the matrix M, as explained in the Section IV-C, by adopting the compositional process of Section IV-B. The LIR evaluation of each item is inserted in the set Rec (*step 13*), where the absolute value of the elements is sorted in descending order, on the basis of the LIR value, and returned as output at the end of the process (*step 16*).

V. EVALUATION

This section describes the experimental environment, the used datasets, the adopted strategy and the involved metric, concluding with the obtained results and their discussion.

A. Environment

The environment for this work is based on the Java language, with the support of DL4J[5], the scientific computing engine used to perform the *doc2vec* process. The experimental framework was developed by using a machine with an Intel i7-4510U, quad core (2 GHz × 4) and a Linux 64-bit Operating System (Debian Jessie) with 4 GBytes of RAM. The item evaluation approaches at the state of the art, to which we compare to test the proposed metric, are SVD and a classic *User-Based Nearest Neighbors Collaborative Filtering* approach (CF). They allow us to compare our strategy with the two different strategies widely used in the context of recommender systems, one in which the dimensionality of the features space is reduced (SVD), and one in which it is processed as it is (CF). The Mahout framework[6] was used to implement the two aforementioned state-of-the-art approaches. We choose to compare our approach with two collaborative filtering approaches instead than a content-based one, because the first of them (i.e., CF), which is based on a classic neighborhood model, is one of the most common approaches, while the second one (i.e., SVD), which is based on the latent factor model, is nowadays considered the top performing recommendation approach.

[5]http://deeplearning4j.org/
[6]http://mahout.apache.org/

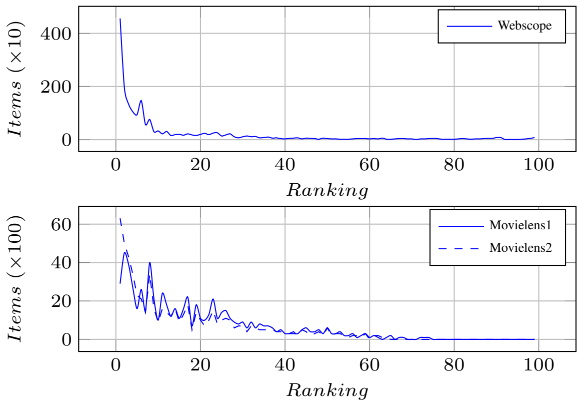

Fig. 1. $Linear\ similarity\ evaluation$

1) Parameters Setup: The optimal value of the *doc2vec* parameters during the process of vectorization has been chosen by a preliminary training, i.e., 10 *epochs*, 100 *layers* (which represent the size of the vector, i.e., the value of L), 0.025 *learning rate*, 0.001 *minimum learning rate*, and 100 *batch size*. Also for the Mahout implementation of SVD, the number of the *target features* and that of the *training steps* (respectively, 10 and 0.05) has been chosen through a preliminary training. With regard to the CF approach, the distance function chosen is the *Pearson correlation*, since it represents one of the most common measures of correlation used in this context [27]. The threshold value used to decide the number of neighbors to consider when predicting the ratings (*similarity thresholds*) has instead been chosen through a specific set of experiments (i.e., $threshold = 0.1$).

We performed all experiments by applying our approach to a maximum of 50 positively evaluated items, for each user profile (i.e., a value of $\alpha = 50$). The choice to perform the experiments by using only 50 items (i.e., by using less information about a user, since with $L = 100$ we could have employed up to 99 items) was made since with this value the results did not worsen but we could have a significant speed-up on the computational side, since the amount of information to process was much more reduced (with $\alpha < 50$ the performance had a statistically significant worsening).

B. Datasets

The experiments have been performed by using three different real-world datasets, extracted by two standard benchmarks for recommender systems: Yahoo! Webscope R4[7] and Movielens 10M[8].

Yahoo! Webscope (R4). The first dataset contains a large amount of data related to user preferences expressed on the Yahoo! Movies community that are rated on the base of two different scales, from 1 to 13 and from 1 to 5 (we use the latter). The entire dataset (i.e., training and test data) is composed by *7,642* users ($|U|$), *11,916* items ($|I|$), and *221,367* ratings ($P \subseteq U \times I \times R$). In the following of this paper we refer to this dataset as *Webscope*.

Movielens 10M. The other two datasets used in this work were extracted from MovieLens 10M dataset (a movie recommendation website), where the items are rated by the users on a scale from 1 to 5. Since Movielens 10M dataset does not contain any textual description of the items, to obtain this information to use in the vectorization process made by *doc2vec*, we used a file provided by the previously presented Webscope (R4) dataset, which maps the Movielens ID with those in the Webscope dataset. The two entire datasets (i.e., training and test data) are composed by *9,683* users ($|U|$), *3,516* items ($|I|$), and *1,051,680* ratings ($P \subseteq U \times I \times R$). In the following of this paper we refer to these datasets as *Movielens1* and *Movielens2*.

C. Strategy

The criterion adopted for obtaining the training and the test sets to use in the experiments was the *K-fold cross validation* [28], with $K = 3$. It means that each dataset (i.e., *Webscope*, *Movielens1*, and *Movielens2*) was divided into three disjoint partitions, and the training and testing process has been performed in three steps. At each step, two partitions (i.e., K-1) were used as training set, whereas the remaining partition was used as test set. The steps were repeated until each of the three disjoint partitions was used as test set, and the results were averaged over the three runs.

We evaluate our proposal through a comparative analysis, by considering the ranking generated by all the approaches (i.e., SVD, CF, and our LIR approach) to each item not evaluated by a user, on the basis of the analysis of the user profiles information. The comparisons have been made by measuring the *Mean Reciprocal Rank* (MRR), a statistical measure [29] widely adopted to evaluate the processes that produce a list of ranked results, giving the measure of how early we get the relevant results, as explained in Section V-D.

According with the Mahout documentation, we use the instruction *RandomUtils.useTestSeed()* in the Java code to ensure the repeatability of the performed experiments.

We evaluated our LIR metric from two different perspectives:

1) **Linear similarity evaluation:** we verify the ability of the proposed LIR metric to give us information about the similarity between a user profile and an item, in terms of linear dependence. Considering the set of items I, for each user $u \in U$ we rank each item $i \in I$ that belongs to the most positively evaluated item genre in her/his profile I_u (i.e., the genre with the highest number of preferences). We perform this operation on the basis of the value of the compositional determinant $cdet$ calculated in each user profile matrix $\Xi_{u,\alpha}$ (i.e., the matrix where the last row is the item to evaluate). The aim is to evaluate the rank given to the items that belong to the most preferred genre of each user, in order to verify the effectiveness of the compositional approach

[7]http://webscope.sandbox.yahoo.com
[8]http://grouplens.org/datasets/movielens/

explained in Section IV-B, in terms of modeling of the user preferences.

2) **Ranking accuracy evaluation:** we test the ability of our approach to infer the future choices of the users, by comparing the rank assigned to the items in the test set by using the LIR metric, with those assigned to the same items by the two approaches at the state of the art taken into account (i.e., the user-based collaborative filtering and SVD). In this experiment we take into account the items positively evaluated by the users (i.e., those in their user profiles with a rating ≥ 3).

D. Metric

The *Mean Reciprocal Rank* (MRR) is a statistical measure largely adopted in order to evaluate the ranking generated for a set of elements that belong to a certain domain. The reciprocal rank is the multiplicative inverse of the rank of the first correct element. Its mathematical formalization is shown in Equation (8), where E is the number of evaluations (i.e., in our context, the number of evaluated items).

$$MRR = \frac{1}{|E|}\sum_{i=1}^{|E|}\frac{1}{\text{rank}_i} \quad (8)$$

E. Experimental Results

In this section we present the results of the experiments introduced in Section V-C.

1) Linear Similarity Evaluation: The idea behind this process of validation is the same that gives a meaning to the proposed LIR metric, an important *determinant* property: *a square matrix with duplicate rows has zero determinant*. On the basis of this property, we can assume that the more similar is the vector representation of a unevaluated item to one of those in the user profile, the more the value of the determinant will be close to zero. As shown in Figure 1, where the rank was normalized in a range from 1 to 100, the effectiveness of the proposed metric is validated with all three datasets, i.e., almost all items have been ranked in the first part of the interval (within the top-10 positions), and this means that the items that belong to the genre preferred by the users have been recognized as similar with those in their user profiles.

2) Ranking Accuracy Evaluation: Now that we verified the validity of the proposed metric in terms of item linear similarity evaluation, Figure 2 reports that our approach is able to model, in the best way (w.r.t. other approaches), the user preferences, by assigning an high rank to those most similar to the real user preferences. In fact, regardless of the number of ranked items (*x-axis*), the MRR values of our approach (*y-axis*) overcome those of the other ones at the state of the art (i.e., *content-based* and *SVD*).

The independent-samples *two-tailed Student's t-tests* highlighted that there is statistical difference between the results ($p < 0.05$).

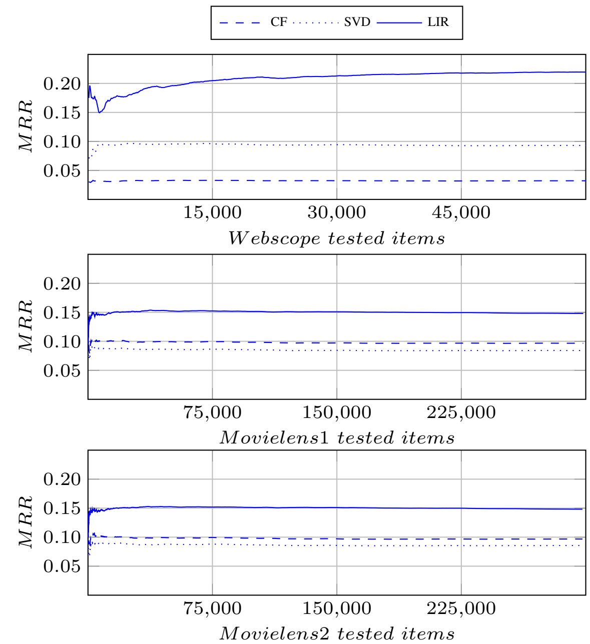

Fig. 2. *Ranking accuracy evaluation*

3) Discussion: Through the first experiment of Section V-E1 we validate the proposed LIR model in terms of its capability to assign the highest rank (i.e., a lowest value of LIR) to the items that reflect the real user tastes, because in all the three real-world datasets taken into account the items that belong to the most positively evaluated genre of the users are ranked in the first positions. As we can observe in Figure 1, in the $Webscope$ dataset, almost all of them are in the first 10%, while in the $Movielens1$ and $Movielens2$ datasets, almost all of them are in the first 20%. The other results, reported in Section V-E2 and summarized in Table I, show that the proposed LIR metric outperforms the canonical state-of-the-art ones. In fact, in all the three real-world datasets, our approach based on the LIR metric obtains the highest value of MMR: as summarized in Table I, in the $Webscope$ dataset we obtain a value of 0.22 against the values of 0.03 and 0.09 of the other approaches, while in the $Movielens1$ and $Movielens2$ datasets, we obtain a value of 0.147 and 0.148, against the values of 0.096 and 0.085 of the other approaches at the state of the art.

The formalization of a mean value of determinant ($cdet$) allowed us to evaluate the similarity between an item and a user profile by exploiting the concept of determinant of a matrix, also when it is a non-square matrix. Its ability to investigate about the similarity relations between vector representations of the items in a domain, performed in terms of linear independence between them, represents a new powerful instrument of evaluation. We obtained a twofold result: we were able to predict the punctual preferences of the users in terms of similar items, but we were also able to predict the future choices of them, when the involved items are not similar to those previously evaluated, as can happen with the items in

$Dataset$	$Evaluations$	MRR		
		CF	SVD	LIR
$Webscope$	$62,600$	0.032	0.092	$\mathbf{0.220}$
$Movielens1$	$298,500$	0.096	0.084	$\mathbf{0.147}$
$Movielens2$	$298,000$	0.096	0.085	$\mathbf{0.148}$

TABLE I
$MRR\ best\ performance$

the test set (as indicated by the low value of MRR).

VI. Conclusions and Future Work

This paper aimed at introducing a novel metric able to improve the item evaluation process in the context of a recommendation task. This new determinant-based metric allows us to measure the similarity between a unevaluated item and those in the user profile, expressed as a matrix of word-embedding vectors of the items positively evaluated by a user, in terms of *linear independence* between their vector representations.

In the first experiment, we verified the correctness of the compositional approach ($cdet$), which represents the core of the proposed metric, since through it we can extract the determinant information from a matrix composed by the items positively evaluated by a user and a unevaluated one. In the second experiment, we instead show that the proposed LIR metric overcomes the canonical state-of-the-art metrics, in terms of capability to model the user preferences, i.e., all results report a strong improvement in the process of rating of the unevaluated items, where our metric assigns an higher score (w.r.t. the approaches at the state of the art, to which we compared) to those in the test sets, positively evaluated by the users.

Future work will consider the LIR evaluation of the matrices composed by the items in the user profiles negatively evaluated by the users, in order to better define the users' tastes, also in terms of unwanted items, exploiting this new information to improve the performance of a recommender system, for instance, by verifying the preferences collision (i.e., when very similar items are rated both positively and negatively by a user).

Acknowledgment

This work is partially funded by Regione Sardegna under project NOMAD (Next generation Open Mobile Apps Development), through PIA - Pacchetti Integrati di Agevolazione "Industria Artigianato e Servizi" (annualità 2013), and by MIUR PRIN 2010-11 under project "Security Horizons".

References

[1] S. Sivapalan, A. Sadeghian, H. Rahnama, and A. M. Madni, "Recommender systems in e-commerce," in *World Automation Congress (WAC), 2014*. IEEE, 2014, pp. 179–184.

[2] F. Ricci, L. Rokach, and B. Shapira, Eds., *Recommender Systems Handbook*. Springer, 2015.

[3] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 426–434.

[4] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu, "Learning semantic hierarchies via word embeddings." in *ACL (1)*, 2014, pp. 1199–1209.

[5] C. Musto, G. Semeraro, M. de Gemmis, and P. Lops, "Learning word embeddings from wikipedia for content-based recommender systems," in *Advances in Information Retrieval*. Springer, 2016, pp. 729–734.

[6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[7] W. Hill, L. Stead, M. Rosenstein, and G. Furnas, "Recommending and evaluating choices in a virtual community of use," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., 1995, pp. 194–201.

[8] G. Karypis, "Evaluation of item-based top-n recommendation algorithms," in *Proceedings of the tenth international conference on Information and knowledge management*. ACM, 2001, pp. 247–254.

[9] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *Internet Computing, IEEE*, vol. 7, no. 1, pp. 76–80, 2003.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001, pp. 285–295.

[11] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, p. 4, 2009.

[12] P. Lops, M. De Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender systems handbook*. Springer, 2011, pp. 73–105.

[13] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*. Springer, 2007, pp. 325–341.

[14] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[15] D. Billsus and M. J. Pazzani, *A hybrid user model for news story classification*. Springer, 1999.

[16] H. Lieberman *et al.*, "Letizia: An agent that assists web browsing," *IJCAI (1)*, vol. 1995, pp. 924–929, 1995.

[17] M. J. Pazzani, J. Muramatsu, D. Billsus *et al.*, "Syskill & webert: Identifying interesting web sites," in *AAAI/IAAI, Vol. 1*, 1996, pp. 54–61.

[18] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli, "User profiles for personalized information access," in *The adaptive web*. Springer, 2007, pp. 54–89.

[19] D. H. Widyantoro, T. R. Ioerger, and J. Yen, "Learning user interest dynamics with a three-descriptor representation," *Journal of the American Society for Information Science and Technology*, vol. 52, no. 3, pp. 212–225, 2001.

[20] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations." in *HLT-NAACL*, 2013, pp. 746–751.

[21] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 384–394.

[22] N. Djuric, H. Wu, V. Radosavljevic, M. Grbovic, and N. Bhamidipati, "Hierarchical neural language models for joint representation of streaming documents and their content," in *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2015, pp. 248–255.

[23] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *arXiv preprint arXiv:1405.4053*, 2014.

[24] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in Neural Information Processing Systems*, 2013, pp. 2787–2795.

[25] R. Kiros, R. Zemel, and R. R. Salakhutdinov, "A multiplicative model for learning distributed text-based attribute representations," in *Advances in Neural Information Processing Systems*, 2014, pp. 2348–2356.

[26] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[27] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *SIGIR*. ACM, 1999, pp. 230–237.

[28] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, no. 2, 1995, pp. 1137–1145.

[29] D. Binkley and D. Lawrie, “Learning to rank improves ir in se,” in *2014 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2014, pp. 441–445.