Random Indexing and Negative User Preferences for Enhancing Content-based Recommender Systems

Cataldo Musto, Giovanni Semeraro, Pasquale Lops, Marco de Gemmis

Department of Computer Science
University of Bari "Aldo Moro", Italy
{cataldomusto,semeraro,lops,degemmis}@di.uniba.it
http://www.di.uniba.it/

Abstract. The vector space model (VSM) emerged for almost three decades as one of the most effective approaches in the area of Information Retrieval (IR), thanks to its good compromise between expressivity, effectiveness and simplicity. Although Information Retrieval and Information Filtering (IF) undoubtedly represent two related research areas, the use of VSM in Information Filtering is much less analyzed, especially for content-based recommender systems.
The goal of this work is twofold: first, we investigate the impact of VSM in the area of content-based recommender systems; second, since VSM suffer from well-known problems, such as its high dimensionality and the inability to manage information coming from negative user preferences, we propose techniques able to effectively tackle these drawbacks. Specifically we exploited Random Indexing for dimensionality reduction and the negation operator implemented in the Semantic Vectors open source package to model negative user preferences. Results of an experimental evaluation performed on these enhanced vector space models (eVSM) and the potential applications of these approaches confirm the effectiveness of the model and lead us to further investigate these techniques.

Keywords: Content-based Recommender Systems, Dimensionality Reduction, Personalization, Vector Space Models

1 Introduction

The recent phenomenon of Web 2.0 and the consequent explosion of Social Web platforms contributed to the enormous growth of the available information and underlined the need for systems able to effectively manage this surplus of data. In this scenario, tools helping users in finding what they really need, such as Information Filtering systems [1], are rapidly emerging. These systems usually work in three steps:

1. *Training Step:* the system acquires information about a target user (what she knows, what she likes, the task to be accomplished, demographical or

contextual informations and so on). This step could be accomplished in an explicit or implicit way, that is to say, by asking users to explicitly express her preferences or by analyzing her behavior.

2. *User Modeling:* the information previously extracted are modeled and stored, according to the filtering model implemented in the system.
3. *Filtering:* finally, the system filters the information flow by exploiting the user profile. The goal of this step is to find the most relevant items for a target user, usually ranked according to a relevance criterion.

In recent years IR and IF followed two separated research paths, although the strong analogies between them have already been underlined by Belkin and Croft in 1992 [2]. Indeed, even if the goal is slightly different, both content-based recommendation and retrieval processes are carried out by processing a set of items represented as textual documents. In the first case the system performs a progressive filtering of the information not relevant for a target user in a space of items, while in the second one the system tries to retrieve the most relevant documents from the entire corpus w.r.t. the user informative need. Furthermore, the concept of 'query' (describing short-term user needs) is replaced in IF by the concept of 'user profile', that describes long-term user preferences and represents the input that triggers the whole recommendation process. Finally, typical IR-based weighting techniques (such as TF/IDF [3]) and measures (such as the Cosine Similarity [4]) can be easily applied in IF, for example to assign weights to the terms stored in a content-based user profile and to perform similarity calculations between items and a user profile. Anyway, despite these analogies, the impact of IR-based models in the area of IF has not yet been properly investigated.

In the area of Information Retrieval the Vector Space Model (VSM) emerged as one of the most effective state-of-the-art approaches, thanks to its good compromise between expressivity, effectiveness and simplicity. However, VSM suffers from at least three important problems: first, the approach is not incremental. This means that even the addition of a new item to the corpus requires that the whole vector space has to be generated again from scratch. This is a problem especially felt for real-world data, because the generation of high-dimensional vector spaces is a very complex and computationally expensive task. Furthermore, VSM cannot manage the information coming from negative user preferences. This is a well-known drawback, that can be ignored for IR systems (because the query usually contains only information about user informative needs), but not for the IF ones because as proved by many contributions in the area of text categorization (e.g., naive Bayes [5], SVM [6], etc.), both positive and negative preferences have to be modeled. Finally, VSM is not able to manage neither the latent semantics of each document nor the position of the terms that occur in it. For example, given a document and a permutation of its terms, their representation in the VSM is absolutely the same, although the conveyed information could be different.

The main contribution of this work is to exploit the overlapping between IR and IF research areas to evaluate the impact of IR-based models in the area of

IF by comparing their performance with respect to other content-based filtering models. Furthermore, we introduced two 'enhanced vectors space models' (eVSM) that exploit techniques able to overcome classical VSM problems by ensuring good efficiency, scalability and the ability of managing both latent semantics of documents and negative user preferences in a more effective way. Specifically, in this work we have used Random Indexing, an incremental technique for dimensionality reduction, and a negation operator based on quantum mechanics implemented in the Semantic Vectors [7] open source package to model negative user preferences.

The paper is organized as follows: related work are described in Section 2. Section 3 introduces the techniques we exploited in this work, such as Random Indexing, while in section 4 we focus the attention on the description of both filtering models. Results emerged from the experimental evaluation are described in Section 5. Finally, future directions are sketched in Section 6.

2 Related Work

Vector Space Model, introduced by Salton et al. [8] in 1975, is considered as one of the most effective retrieval models in the IR research community. Raghavan [9] gives a good overview of Vector Space model issues in the area of IR. The use of VSM as content-based filtering model [10] has been previously investigated by Cohen and Hirsh [11]and Nouali and Blache [12]. Berry et al. [13] pointed out the need for dimensionality reduction techniques as a mean to improve the effectiveness and the scalability of VSMs. LSA [14] and PLSI [15] are two of the most well-known techniques that perform this step, but their computational complexity is of hindrance to implement these approaches in real-world applications. In these scenarios effective techniques for dimensionality reduction such as Random Indexing [16], emerged. The effectiveness of this approach has already been demonstrated in [17] with an application for image and text data. Recently the research about semantic vector space models gained more and more attention: the survey by Turney and Pantel [18] about the use of VSM for semantic processing of text analyzed the main issues and the first packages developed in this area, such as S-Space[1] and Semantic Vectors (SV)[2]. The SV package was implemented by Widdows [7]: it implements a Random Indexing algorithm and defines a negation operator based on quantum mechanics [19]. Some initial investigations about the effectiveness of the Semantic Vectors for retrieval and filtering tasks are reported in [20] and [21].

3 eVSM for Content-based Recommender Systems

In this section we will describe the techniques exploited for building enhanced vector space models. In our opinion, a VSM can be defined *enhanced* if:

[1] http://code.google.com/p/airhead-research/
[2] http://code.google.com/p/semanticvectors/

1. The whole vector space is built in an *incremental way*;
2. The model is able to catch the *semantics* of documents;
3. The model is able to manage the information coming from *negative evidences*.

In our approach we tackled the first two issues through the introduction of Random Indexing, while the last one is managed by exploiting Semantic Vectors. In this section we will give a complete overview of both the theoretical basis of the Random Indexing approach and the main features implemented in the Semantic Vectors open source package.

Hereafter we could refer to the *items to be filtered* and to the *user profiles* as *documents*. Indeed, in a content-based filtering model the terms are considered synonyms because we assume that items to be filtered are described by means of some textual content. For example, in a movie recommendation scenario we can assume that an item (movie) will be represented by its title, cast, plot and so on.

3.1 Random Indexing

Random Indexing is an efficient, scalable and incremental technique for dimensionality reduction. Following this approach, we can represent terms and documents as points in a vector space with a considerable reduction of the features that describe them. To sum up, through this model we can obtain results comparable to other well-known methods (such as Singular Value Decomposition), but with a tremendous savings of computational resources.

This approach belongs to the class of the so-called *distributional models*. These models state that the meaning of a word can be inferred by analyzing its use (that is to say, its *distribution*) within a corpus of textual data. According to the *distributional hypothesis* "words that occur in the same contexts tend to have similar meanings". For example, we can state that the terms *wine* and *beer* have similar meanings because they often co-occurs with the same words (e.g. *drink*) The goal of Random Indexing is to shift the classical VSM representation based on a n-dimensional term-document matrix towards a more compact and flexible k-dimensional term-context matrix (see figure 1).

Fig. 1. Dimensionality reduction process through Random Indexing

This dimensionality reduction is obtained by multiplying the original term-document matrix with a matrix R built in a random way. Formally, given a

corpus of n terms and m documents represented in the original matrix A, the reduced k-dimensional matrix B is obtained as follows:

$$A^{n,m} * R^{m,k} = B^{n,k} \tag{1}$$

So, the key concept behind the building of the *random* matrix is the definition of the concept of "context". Given a word, we could think at the *context* as a piece of text, variable in size, which surrounds that word. Following the famous Wittgenstein sentence *"meaning is its use"*, in Random Indexing the context is exploited to infer the meaning of the word by analyzing the meaning of the other words that more often co-occur within its own context.

In general, the "meaning" of a term (its position in the Vector Space) is obtained by following these steps:

1. *A context vector* is assigned to each term. This vector has a fixed dimension (k) and it can contain only values in $\{-1, 0, 1\}$. Values are distributed in a random way, but the number of non-zero elements is much smaller.
2. The Vector Space representation of a *term* (denoted by $\boldsymbol{t}$) is obtained by summing the context vectors of all the terms it co-occurs with.
3. The Vector Space representation of a *document* (denoted by $\boldsymbol{d}$) is obtained by summing the context vectors of all its terms.
4. The Vector Space representation of a *user profile* for a user u (denoted by $\boldsymbol{p}_u$) is obtained by combining the context vectors of all the *terms* that occur in the documents liked in the past by the user u. The unique difference between the filtering models proposed in this work is the way previously liked documents are combined.

Given a set of documents, by following this approach we can build a low-dimensional Vector Space that guarantees scalability, effectiveness and a better semantic modeling of the documents since each term is no longer represented in an atomic way, as in the classical keyword-based methods, but its position in the space depends on the terms it co-occurs with. The main advantage behind Random Indexing (whose theoretical reliability has been proved by the studies about near-orthogonality by Hecht-Nilsen [22]) is that in this low-dimensional space, as stated by Johnson and Lindestrauss in their lemma [23], the distance between points is preserved (Figure 2) so it is possible to perform calculations and compute similarity between items represented in the vector space with a minimum loss of accuracy balanced by the enormous gain in efficiency.

3.2 Semantic Vectors

Through Random Indexing we can build low-dimensional vector spaces that maintain the original expressivity of the model. However, they still inherit a classic problem of VSM: the information coming from negative evidences is not managed in any way and does not contribute to the position that the item assumes in the vector space. In content-based recommender systems, especially for

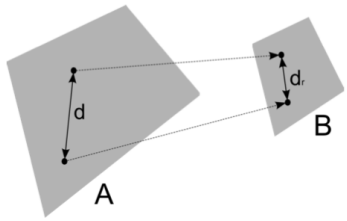

Fig. 2. A visual explanation of the Johnson-Linderstrauss lemma

building user profiles, this is an important aspect because negative user preferences have to be modeled, too. In order to tackle this problem we exploited the Semantic Vectors (SV) open-source package, a set of libraries that implements a Random Indexing approach and extends it by introducing a negation operator based on quantum mechanics. In SV the negation operator is used mainly to define queries that contain negative terms, such as *A not B*, for retrieval tasks. From a theoretical point of view, this kind of query represents the projection of the vector A on the subspace orthogonal to those generated by the vector B. Intuitively, in our recommendation models we will define two vectors: the first for modeling positive preferences and the second modeling negative ones. The negation operator will be used to identify the subspace that will contain the items as close as possible to the positive preference vector and as far as possible to negative one. In the next section we will analyze thoroughly this aspect.

4 Recommendation models

The recommendation approaches proposed in this work try to prove that the exploitation of the classical IR-based measures can be useful for filtering items represented as points in an *enhanced vector space*. The main idea behind our models is to build a vector space for each user, where both user profile and items to be filtered are represented through the techniques described in the previous section. Next, by exploiting the classical similarity measures between vectors (such as the classical *cosine similarity*) it is possible to efficiently obtain the set of the most relevant items for the target user, that is to say, the points in the space that are nearest to her profile.

In this work we proposed four different recommendation models, all based on Random Indexing and Semantic Vectors. The main difference between the approaches lies in the way the evidences about both positive and negative user preferences are combined in order to model the user profile in the vector space.

4.1 Random Indexing-based model

This approach is based on the assumption that the information coming from the items a user liked in the past can be a reliable source of information to build

accurate user profiles. Therefore, let $d_1..d_n \in D$ be a set of already rated items, and $r(u, d_i)$ $(i = 1..n)$ the rating given by the user u to the item d_i. We can describe the set of positive items for user u, denoted by I_u, as follows:

$$I_u = \{d \in D | r(u, d) \geq \beta\} \tag{2}$$

Thus, given a threshold β, the profile of a user consists of the set of the terms occurring in the documents she liked in the past. As stated above, the *Random Indexing* is exploited to build the user profile in an incremental way, that is to say by simply summing all the *document vectors* for each document in I_u. Let $|I_u|$ be the cardinality of the set I_u and let $\boldsymbol{d}_i$ be the vector space representation of the document d_i, we can define the user profile $\boldsymbol{p}_u$ as follows:

$$\boldsymbol{p}_u = \sum_{i=1}^{|I_u|} \boldsymbol{d}_i \tag{3}$$

That is undoubtedly the simplest Random Indexing-based filtering model that could be defined. In the experimental evaluation we will refer to this model as **RI**.

4.2 Weighted Random Indexing-based model

The main drawback of the RI method is that the user profile $\boldsymbol{p}_u$ is built without taking into account the ratings provided by the target user for the items she liked. In other terms, it is *independent* from the rates provided by the target user (provided that they are above or below the threshold β). The second model, called *Weighted Random Indexing-based (**W-RI**)*, enriches the previous one by simply associating to each *document vector*, before combining it, a weight equal to the rating provided by the user for it. More formally:

$$\boldsymbol{p}_u = \sum_{i=1}^{|I_u|} \boldsymbol{d}_i * r(u, d_i) \tag{4}$$

In this way the model will increase the weight of the items liked by the user.

4.3 Semantic Vectors-based model

The main idea behind Semantic Vectors-based model (**SV**) is to exploit the negation operator to represent in the user profile both positive and negative preferences, as in the classical text classification approaches (e.g. Naïve Bayes, Support Vector Machines and so on). We can think at this model as an extension of the previously described RI model. Unlike RI, in which a single user profile $\boldsymbol{p}_u$ is built, in SV filtering model two user profile vectors, one for positive preferences

and one for negative ones, are inferred. The set of positive items I_u^+ and the positive user profile vector $\boldsymbol{p}_{+u}$ are identical to the set of positive items I_u and the user profile $\boldsymbol{p}_u$ in RI, while the set of negative items, denoted by I_u^-, is defined as follows:

$$I_u^- = \{d \in D | r(u, d_i) < \beta\} \tag{5}$$

The negative user profile vector, denoted by $\boldsymbol{p}_{-u}$, is built by summing the vector space representations of the items in I_u^-. Formally:

$$\boldsymbol{p}_{-u} = \sum_{i=1}^{|I_u^-|} \boldsymbol{d}_i \tag{6}$$

Thus, given the profile vectors $\boldsymbol{p}_{+u}$ and $\boldsymbol{p}_{-u}$ we can use Semantic Vectors to instantiate the vector $\boldsymbol{p}_{+u}$ NOT $\boldsymbol{p}_{-u}$, that is exploited to find the items represented in the vector space that contain as much as possible features that occur in the documents in I_u^+ and as less as possible features from I_u^-.

4.4 Weighted Semantic Vectors-based model

As RI, the SV model has its weighted counterpart, called **W-SV**. This model shares the same idea of the W-RI model and the same weighting schema described in 4.2, with the unique difference that in the negative profile I_u^- the items with a lower rate are given higher weights in order to exclude as much as possible the features disliked by the target user. More formally, the set I_u^+ and I_u^- are built by following the same formula introduced in the previous section, while the vectors $\boldsymbol{p}_{+u}$ and $\boldsymbol{p}_{-u}$ are inferred in this way:

$$\boldsymbol{p}_{+u} = \sum_{i=1}^{|I_u^+|} \boldsymbol{d}_i * r(u, d_i) \tag{7}$$

$$\boldsymbol{p}_{-u} = \sum_{i=1}^{|I_u^-|} \boldsymbol{d}_i * (MAX - r(u, d_i)) \tag{8}$$

where MAX is the highest rating that can be assigned to a document.

5 Experimental Evaluation

The goal of the experimental evaluation was to measure the effectiveness of RI and SV models, as well as of their weighted variants W-RI and W-SV, in term

of predictive accuracy and goodness of the proposed ranking. The experimental session has been carried out on a subset of the 100k MovieLens dataset[3], containing 40,717 ratings provided by 613 different users on 520 movies. Since content-based information were crawled from the English version of Wikipedia, we excluded from the original MovieLens dataset the movies without a Wikipedia entry. In Table 1 contains some statistics about the dataset: the original term-document matrix contained 7.351 rows (*features*) and 520 columns (*items*) on average. Since the dimension of each *context vectors* was set to 200, after Random Indexing the size of the matrix was reduced by 62% (from 520 to 200 columns).

Table 1. Content-based MovieLens dataset statistics. The average number of features was calculated by counting the features occurring on average in the documents rated by each user

Items	520	Ratings	40,717
Ratings (avg. per user)	66.44	Positive ratings	83.8%
Features	24,975	Features (avg. per user)	7,351

User profiles were learned by analyzing the ratings stored in the MovieLens dataset. Each rating was expressed as a numerical vote on a 5-point Likert scale, ranging from 1=strongly dislike to 5=strongly like. All the ratings above 2 were considered as positive, while the ratings under this threshold were considered as negative. The session was organized through a 5-fold cross validation: for each fold and for each user we built a vector space for the user profile and the items to be filtered. By exploiting a simple cosine similarity measure we ranked the items, assuming the nearest ones as the most relevant. The metric used to evaluate the effectiveness of the approaches was the *Average Precision@n*, where n was set to 1, 3, 5 and 10. We preferred the Average Precision@n instead of the simple Precision@n because it takes into account also the position of the correctly classified items.

Specifically, in our experimental evaluation we tried to give an answer to three questions:

1. Does the weighting scheme improve the predictive accuracy of the recommendation models?
2. Does the negation operator improve the predictive accuracy of the recommendation models?
3. How do the recommendation models perform w.r.t. other content-based filtering approaches?

As shown in Figure 3 the weighting scheme, even in this naive form, improves the predictive accuracy of the system for all the metrics. The improvement is greater for the AV-P@1 and AV-P@3. This is a good outcome because in this kind of task it is crucial to put *good* items at the top of the recommendation list.

[3] http://www.grouplens.org/node/73

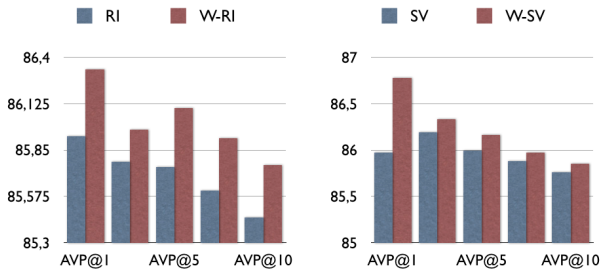

Fig. 3. Analysis of the impact of the weighting schema by comparing RI vs. W-RI and SV vs. W-SV

Figure 4 shows that also the introduction of the negation operator is able to improve the predictive accuracy of the system in all metrics. In this case we can note a lower improvement for the W-SV model. This could suggest to introduce different weighting techniques for the negative component of the user profile.

Finally, in Table 2 the results obtained by our recommendation models are compared w.r.t. a naive Bayes filtering algorithm (described in [24]) and a classical VSM based on the complete term/document matrix without any dimensionality reduction. As shown in Table 2, the *W-SV* model gained the best results, with an increase of the Average Precision between 0.1% and 0.4% w.r.t. the bayesian classifier and around 0.5% w.r.t. the original VSM. Finally, none of the experiments obtained a statistically significant difference between the values of Average Precision. This outcome has been certainly influenced by the extreme imbalance of the dataset (over 80% of positive ratings) and should be verified again through deeper experimental evaluations.

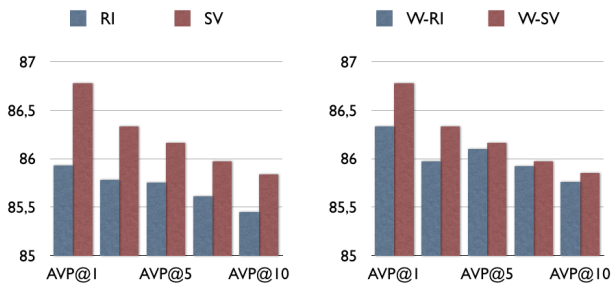

Fig. 4. Analysis of the impact of the negation operator by comparing RI vs. SV and W-RI vs. W-SV

Table 2. Average Precision

Metric	RI	W-RI	SV	**W-SV**	*TF/IDF*	*Bayes*
AV-P @1	85,93	86,33	85,97	**86,78**	*86,27*	*86,39*
AV-P @3	85,78	85,97	86,19	**86,33**	*85,85*	*85,97*
AV-P @5	85,75	86,10	85,99	**86,16**	*86,70*	*85,83*
AV-P @10	85,45	85,76	85,76	**85,85**	*85,58*	*85,75*

6 Conclusions and Future Directions

In this work we introduced the first results emerged from an initial investigation on the impact of enhanced VSM, such as Random Indexing-based and Semantic Vectors-based ones, on Content-based Recommender Systems. The main outcome of the experimental evaluation is that, even in this first prototype and even with a naive weighting scheme, the filtering model shows an accuracy comparable to that obtained by other content-based filtering techniques such as the Bayesian classifier. Furthermore, the introduction of a negation operator, a totally novel aspect for VSM, lets us manage also the information about the disliked items and their features. The results obtained with the W-SV model represents a promising starting point for further investigations in this area. In the future we will introduce other weighting schemas and we will compare the results with those obtained by other algorithms capable of managing negative user feedbacks (e.g. Rocchio). Another important aspect to be investigated is the impact of Natural Language Processing techniques on the model. We will try to analyze the impact of single lexical categories on the accuracy of filtering tools. This task will be accomplished by comparing the effectiveness of the system with user profiles built by exploiting only a single category (for example, only names, only verbs or only entities). Finally, a promising future direction could be represented by the exploitation of Linked Data in order to shift the classical keyword-based profiles towards a more complex structure in which relationships are explicitly coded and can be used for recommendation tasks.

References

1. U. Hanani, B. Shapira, and P. Shoval, "Information filtering: Overview of issues, research and systems," *User Model. User-Adapt. Interact.*, vol. 11, no. 3, pp. 203–259, 2001.
2. N. Belkin and B. Croft, "Information filtering and information retrieval," *Comm. ACM*, vol. 35, no. 12, pp. 29–37, 1992.
3. R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval.* Addison-Wesley, 1999.
4. P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining.* Pearson Education, 2006.
5. S.-B. Kim, K.-S. Han, H.-C. Rim, and S.-H. Myaeng, "Some effective techniques for naive bayes text classification." *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 11, pp. 1457–1466, 2006.

6. T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *European Conference on Machine Learning (ECML)*, 1998.
7. D. Widdows, "Orthogonal negation in vector spaces for modelling word-meanings and document retrieval," in *ACL*, 2003, pp. 136–143.
8. G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, no. 11, pp. 613–620, 1975.
9. V. V. Raghavan and S. K. M. Wong, "A critical analysis of vector space model for information retrieval," *Journal of the American Society for Information Science*, vol. 37, no. 5, pp. 279–287, 1986.
10. P. Lops, M. de Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recomender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp. 73–105.
11. W. W. Cohen and H. Hirsh, "Joins that generalize: Text classification using WHIRL." in *KDD*, 1998, pp. 169–173.
12. O. Nouali and P. Blache, "A semantic vector space and features-based approach for automatic information filtering." *Expert Syst. Appl.*, vol. 26, no. 2, pp. 171–179, 2004.
13. M. W. Berry, Z. Drmac, and E. R. Jessup, "Matrices, Vector Spaces and Information Retrieval," *SIAM Review*, vol. 41, no. 2, pp. 335–362, 1999.
14. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *JASIS*, vol. 41, no. 6, pp. 391–407, 1990.
15. T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd Annual International SIGIR Conference*, 1999.
16. M. Sahlgren, "An introduction to random indexing," in *Methods and Applications of Semantic Indexing Workshop, TKE 2005*, 2005.
17. E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in *KDD '01*. ACM, 2001, pp. 245–250.
18. P. D. Turney and P. Pantel, "From frequency to meaning: Vector space models of semantics." *J. Artif. Intell. Res. (JAIR)*, vol. 37, pp. 141–188, 2010.
19. C. J. van Rijsbergen, *The Geometry of Information Retrieval*. Cambridge, UK: Cambridge University Press, 2004.
20. P. Basile, A. Caputo, and G. Semeraro, "Semantic vectors: an information retrieval scenario," in *IIR 2010 - Proceedings of the First Italian Information Retrieval Workshop, Padua, Italy, January 27-28, 2010*, M. Melucci, S. Mizzaro, and G. Pasi, Eds., pp. 1–5.
21. C. Musto, "Enhanced vector space models for content-based recommender systems," in *Proceedings of the fourth ACM conference on Recommender systems*, ser. RecSys '10. New York, NY, USA: ACM, 2010, pp. 361–364. [Online]. Available: http://doi.acm.org/10.1145/1864708.1864791
22. R. Hecht-Nielsen, "Context vectors: general purpose approximate meaning representations self-organized from raw data," *Computational Intelligence: Imitating Life, IEEE Press*, pp. 43–56, 1994.
23. W. Johnson and J. Lindenstauss, "Extensions of lipschitz maps into a hilbert space," *Contemporary Mathematics*, 1984.
24. P. Lops, M. de Gemmis, G. Semeraro, C. Musto, F. Narducci, and M. Bux, "A semantic content-based recommender system integrating folksonomies for personalized access," in *Web Personalization in Intelligent Environment*, G. Castellano, L. C. Jain, and A. M. Fanelli, Eds. Springer (Berlin), 2009, pp. 27–47.