Inf Retrieval J (2016) 19:378–415
DOI 10.1007/s10791-016-9280-8

Mining unstructured content for recommender systems: an ensemble approach

Marcelo G. Manzato[1] · Marcos A. Domingues[2] · Arthur C. Fortes[1] · Camila V. Sundermann[1] · Rafael M. D'Addio[1] · Merley S. Conrado[1] · Solange O. Rezende[1] · Maria G. C. Pimentel[1]

Received: 9 June 2015 / Accepted: 3 May 2016 / Published online: 24 May 2016

Abstract Recommendation of textual documents requires indexing mechanisms to extract structured metadata for attribute-aware recommender systems. Applying a variety of text mining algorithms has the advantage of capturing different aspects of unstructured content, resulting in richer descriptions. However, it is difficult to integrate them into a unique model so that these descriptions can efficiently improve recommendation accuracy. This article proposes a generic model based on ensemble learning that combines simple text mining methods in a post-processing approach. After executing each text mining technique, each set of metadata of a particular type is applied to the recommender module, which generates attribute-specific rankings. Then, the resulting recommendations are ensembled to generate a final personalized ranking to the user. We evaluated our ensemble technique with two attribute-aware collaborative recommenders (k-Nearest Neighbors and BPR-Mapping) and we demonstrate its generality by means of comparisons among different types of ensembles. We used two datasets from different domains, the first is from the *Brazilian Embrapa Agency of Technology Information* website, whose documents are written in Portuguese language, and the second is the HetRec MovieLens 2k, published by the *GroupLens Research Group*, whose movies' storylines are written in English. The experiments show that, particularly to the k-NN recommender, better accuracy can be obtained when multiple metadata types are combined. The proposed approach is extensible and flexible to new indexing and recommendation techniques.

Merley S. Conrado is currently Software Engineer, Intel Corp., Santa Clara, CA, USA.

✉ Marcos A. Domingues
madomingues@uem.br

Merley S. Conrado
merleyc@gmail.com

[1] Institute of Mathematics and Computer Science, University of São Paulo, São Carlos, Brazil

[2] Department of Informatics, State University of Maringá, Maringá, Brazil

Keywords Recommender systems · Ensemble learning · Personalized ranking · Metadata awareness · Unstructured content

1 Introduction

Recommender systems are an important technology to support users to deal with the increasing information overload. They provide suggestions of items and services, which are automatically selected to match the user's preferences and interests (Ricci et al. 2011). Research in this field aims to improve the quality of recommendations, occasionally adopting a multidisciplinary approach, including techniques from machine learning, natural language processing, text mining, multimedia, and so on.

Efforts to incorporate techniques from different areas usually share the goal of gathering and modeling the most important pieces of information needed in any personalized filtering technology, which are descriptions or metadata about the users' preferences and content prone to be recommended. Indeed, semantic and rich descriptions about items are very useful in content-based and attribute-aware collaborative recommenders, because users' profiles can be built based on such representations.

The use of metadata about the content depends on their structured viability to be automatically processed. The term "structured" data refers to a convenient format in which specific information can be easily found by the system, e.g., in relational databases (Manning et al. 2008). However, obtaining such organization of data is difficult to achieve not only because of the increasing availability of items on the Web but also due to the nature of the content to be recommended. Moreover, items can be multimedia (e.g. recommendation of videos, photos, songs, etc.), which require indexing tools to extract metadata for automatic retrieval. On the other hand, unstructured textual items do not have a clear and semantically overt structure to find specific information (Manning et al. 2008).

Particularly in the case of textual items (e.g. recommendation of Web pages, articles, books, etc.), a possible solution to extract metadata is to use unsupervised learning methods from the text mining and natural language processing areas. Examples include: i) topic hierarchies, which are efficient models to capture relevant information of textual data and to organize them into topics and subtopics (Marcacini et al. 2012); ii) named entities, which are specific information units usually classified in a predefined set of categories, such as person, location, time, organization, etc. (Sekine 2004); and iii) domain terms, a set of linguistic data which are representative of a particular scope (Conrado et al. 2013).

Using any textual indexing approach, such as those listed above, could result in a structured data representation that would facilitate their application as descriptors of items in content-based or attribute-aware collaborative recommenders (D'Addio et al. 2014; Domingues et al. 2014; Manzato et al. 2014). Nevertheless, each text mining technique has its own purposes, and thus, will explore the content in a particular way. For instance, in named entities recognition the techniques extract, from textual data, words or expressions that belong to some named entity category like name of people, organization, location, time, date, money, etc (Sekine 2004). In another approach, the textual data are clustered hierarchically and the most important words of each (sub)cluster are used as topics that represent the data (Marcacini et al. 2012). Alternatively, domain terms capture words that describe concepts in the domain of the textual data (Conrado et al. 2013).

In spite of the variety of ways to analyze the content which are specific to each technique and purposes, the combination of different views of the content may create a rich and meaningful set of descriptions that, when used together, will be able to improve the

efficiency of recommender systems. Indeed, the combination of different text mining techniques could explore the best of each one, capturing the semantics of the content in a richer and more detailed way.

However, developing a unique and integrated model composed of different features is difficult to achieve because of its complexity. In a previous work (Domingues et al. 2015), we proposed a model which used such variety of features, but the experience showed two main drawbacks: i) the improvement of results are prone to be marginal, mainly when the distinguished features are not combined efficiently; and ii) the integrated models lack extensibility in the sense that they will not take advantage of novel text mining algorithms developed in the future.

Towards tackling the two drawbacks identified above, we propose in this article a generic approach for integrating different text mining algorithms applied to the recommendation of textual documents. We propose a post-processing module based on ensemble learning, which combines the results of recommendations generated by specific metadata types (e.g. topic hierarchies, named entities and domain terms), generating a final personalized ranking with better accuracy. The main advantages of our approach are extensibility and flexibility, as different text mining and recommendation algorithms can be integrated into the model. Such integration, in turn, is accomplished automatically by the ensemble module, because it is able to learn the weights of contribution of each metadata type to the final recommendation.

We evaluated our proposal by comparison with two attribute-aware collaborative recommenders (k-Nearest Neighbors and BPR-Mapping), and we compare different types of ensembles to demonstrate the proposal's generality. The experiments were executed with two datasets from different domains: the first is from the *Brazilian Embrapa Agency of Technology Information* website, whose documents are written in Portuguese language, and the second is the HetRec MovieLens 2k, whose movies' storylines are written in English language. Our study shows that the proposed ensemble approach is able to provide better accuracy than individual rankings, as it automatically combines multiple metadata types which represent different aspects from the content.

This article is structured as follows: in Sect. 2 we overview research results related to text mining techniques and ensemble algorithms; in Sect. 3 we describe the algorithms we use in our work for the extraction of metadata from unstructured content. In Sect. 4 we present two attribute-aware collaborative recommenders which we used to evaluate our proposal. We then present the ensemble module we propose in Sect. 5, and detail results of its evaluation in Sect. 6. In Sect. 7 we present our conclusions, and discuss current limitations and future work.

2 Related work

Towards reviewing work related to our proposal, we first discuss approaches targeted at mining unstructured content; next, we provide an overview of ensemble-based recommender systems.

2.1 Text mining techniques

Some state-of-the-art approaches for extraction of metadata from Web content have been proposed in the literature. Some results focus on obtaining contextual information from

online reviews to improve item recommendation. For example, Li et al. (2010) compile a list of lexicons and use a string matching method to extract different types of contextual metadata from reviews. Hariri et al. (2011) propose a multi-labeled text classifier based on Labeled Latent Dirichlet Allocation. This classifier is trained with samples of text reviews labeled into their corresponding context (in this paper's case, trip types). The classifier is then used to probabilistically classify user reviews into those context categories. In our proposal we exploit unsupervised methods to learn topic hierarchies, entities and terms by analyzing the semantics of Web content and, then, use them as metadata to characterize the items. Thus, we do not need a lexicon or a set of labels, usually not available in Web content, to extract metadata.

Semeraro et al. (2009) apply a spreading activation algorithm to compute the correlation among terms from Web document and from a set of external knowledge sources related to linguistic knowledge, world knowledge, and social knowledge. They use the most correlated external terms as meaningful features/metadata in a content-based recommendation process. An important issue related to this approach is that it can only be used when external knowledge sources are available. In contrast, our proposal can be used with internal (found on the database) and external (i.e. information gathered in different Web sites or knowledge databases) data sources.

More recent works use reviews to extract feelings related to inherent characteristics of an item. For example, Qumsiyeh and Ng (2012) propose a system capable of generating recommendations for different multimedia items using information extracted from databases available in several trusted sites. Their method can compute the sentiment and degree of each considered aspect of an item, as genres, actors and reviews. Kim et al. (2012) propose MovieMine, a personalized search engine for movies based on previous reviews of a user and their ratings assigned to other items. Ganu et al. (2013) propose a restaurant recommender system that performs a soft clustering of users based on topics and sentiments extracted from reviews of visited restaurants.

Our work differs from the aforementioned since it is capable of using several text mining techniques to extract different features from unstructured content. Instead of using only one text mining technique, or developing a complex model which extracts different features at the same time, our proposal is based on a post-processing module that combines the results of recommendations from different algorithms and features. Thus, a number of recommendation strategies and text mining techniques can be integrated according to viability and the application domain.

2.2 Ensemble algorithms

Ensemble is a machine learning approach that uses a combination of similar models to improve the results obtained by a single model. In fact, several recent studies (e.g. (Jahrer et al. 2010)) demonstrate the effectiveness of an ensemble of several individual and simpler techniques, and show that ensemble-based methods outperform any single, more complex algorithm.

Bar et al. (2013) propose a systematic framework in which ensemble methods are applied to collaborative filtering (CF) models. They employ automatic methods for generating an ensemble of collaborative filtering models based on a single collaborative filtering algorithm (homogeneous ensemble). They demonstrate the effectiveness of their framework by applying several ensemble methods to various base collaborative filtering models.

In their recent work, Ristoski et al. (2014) propose a hybrid multi-strategy book recommender system using Linked Open Data. Their approach consists of training individual

recommenders and using global popularity scores as generic recommenders. The results of the individual recommenders are combined using an ensemble method and ranking aggregation. They show their approach delivers good results in different recommendation settings, and that it also allows to incorporate diversity of recommendations. However, their work is limited to structured item features available in DBpedia[1], which are extracted and organized in a different fashion than those found in user reviews.

Our proposal can be considered an ensemble-learning-based technique, as it automatically learns how to combine multiple rankings which were previously generated by single attribute-aware recommendation modules. The results of this procedure is the increase of recommendation accuracy, while maintaining extensibility and flexibility of the model.

3 Metadata extraction from unstructured content

In recommender systems, the extraction of metadata from unstructured content is a non-trivial task, because the data are prone to the occurrence of noise, irrelevant information, misspellings, etc. In addition, due to its unstructured nature, different information types can be mixed in a single document. In order to address these issues, we have applied three different text mining techniques, aiming to identify item features that can be useful to improve recommendation: topic hierarchies construction, named entity identification, and domain term extraction.

3.1 Topic hierarchies

Topic hierarchies strategy consists of organizing textual information of items into a hierarchical structure of topics using unsupervised learning methods. In this case, hierarchical text clustering algorithms are very useful to automatically organize textual collections into clusters and subclusters – based only on a measure of similarity between textual data. After obtaining the cluster structure, the most important words of each cluster are extracted and used to define topics from texts.

In this work, we use the Buckshot Consensus Clustering (BC^2) technique (Marcacini et al. 2012). This method is used for unsupervised learning of topic hierarchies from unstructured textual information describing the items (for example, Web page content). Consensus clustering combines different clustering solutions from a same dataset into a single clustering solution with better quality. For instance, if a textual data item is misplaced in some clustering solution, the same textual data item is not necessarily misplaced in other clustering solutions, thereby consensus clustering can yield better final solutions.

A brief description of BC^2 is as follows. Initially, several clustering structures are generated by running various clustering algorithms or, alternatively, repeated runs of the same algorithm with different parameter values. The clusters generated are aggregated by means of a co-association matrix $M(l,t) = \frac{f_{lt}}{c}$, where f_{lt} is the number of times that textual data items l and t are in the same cluster and c is the number of clustering solutions. In fact, the co-association matrix M represents a new (robust) concept of proximity among items, and the consensus clustering solution is obtained by applying the UPGMA hierarchical clustering algorithm on the matrix M (Marcacini et al. 2012). Finally, we identify the most important terms of each (sub)cluster, by using the F1 measure in the following strategy:

[1] http://dbpedia.org.

For each (sub)cluster composed of D documents and T candidate terms, we select the terms that would retrieve as many documents from D as possible while retrieving as few as possible documents outside of D. This is achieved by computing the F1 measure, where we retrieve documents from the whole text collection and consider as relevant only those documents in D.

This procedure is repeated for every (sub)cluster of the hierarchy, thereby extracting a topic hierarchy from textual information about the items. The topics are used as metadata by recommender systems.

3.2 Named entities

According to Sekine (2004), the term Named Entity refers to the names of people, organizations and locations, as well as to expressions like time, date, money and percent numeric expressions. Named entity recognition is a task that involves identifying words or expressions that belong to categories of named entities (Mikheev et al. 1999). This process is divided into two subtasks (Nothman et al. 2013): i) identification of possible entities, and ii) categorization of entities. In the example illustrated by Mikheev et al. (1999), the sentence "Flavel Donne works as an analyst in the General Trends, which has been based in Little Spring since July 1998"; "Flavel Donne", "General Trends", "Little Spring" and "July 1998" are recognized as person, organization, location and time entities, respectively.

In this work, the named entity recognition is performed by using two tools: REMBRANDT[2] (Cardoso 2012) for the Portuguese dataset (Embrapa) and Stanford NER[3] (Finkel et al. 2005), also known as CRFClassifier, for the English dataset (HetRec MovieLens 2k).

REMBRANDT was designed to recognize named entities in texts written in Portuguese. It uses Wikipedia[4] as the knowledge base for the classification of entities and it has its own interface, called SASKIA, to interact with this base. The framework implements three main steps:

1. Recognition of numeric expressions and generation of candidates for named entities: the atomizer of Linguateca,[5] a resource center to the computational processing of Portuguese, is used to split the text into sentences and units, which makes it possible to recognize numeric expressions and to identify possible candidates for named entities;
2. Classification of named entities: candidates for named entities are first classified into the possible named entity classes (institution, place, person, among others) by SASKIA and then classified again using grammar rules. For example, in the sentence "Eu moro na Rua Brasil" (I live in the Brasil street), the named entity "Brasil" is first classified as "place" by SASKIA. Then, the street grammar rule is applied, and the named entity "Brasil" changes its classification to "street", a more specific class;
3. Reclassification of named entities without classes: task specific rules (heuristics) are used to detect relationships among named entities and, with these relationships, some named entities without classes may be associated to the same class of classified named entities related to them. The rules comprise the following heuristics: 1) named entities with the same text or named entities that are matched to the same Wikipedia page are labeled as being identical; 2) named entities that overlie other or that are separated by

[2] http://xldb.di.fc.ul.pt/Rembrandt.

[3] http://nlp.stanford.edu/software/CRF-NER.shtml.

[4] https://www.wikipedia.org.

[5] http://www.linguateca.pt.

a term like "de", "da", "do", "das", "dos" and/or "e" are analyzed to determine the type of relation between them. For example, when a category "happening" overlaps or is a neighbor of the category "local", the relation "occurs_in" is used; 3) named entities that are matched to the Wikipedia pages are analyzed in order to find relationships with neighboring named entities in the same sentence, through the links in the page; and 4) a series of grammar rules is applied to detect relations between named entities in the same sentence and that do not have links between them yet.

Stanford NER provides a general implementation of linear chain Conditional Random Field (CRF) sequence models (Lafferty et al. 2001). This named entity recognizer includes a four class model trained for CoNLL (Conference on Natural Language Learning) that classifies named entities into the following classes: Location, Person, Organization and Misc. Stanford NER also includes a seven class model trained for MUC (Message Understanding Conferences) that recognize the classes Time, Location, Organization, Person, Money, Percent and Date; and a three class model trained on both data sets (CoNLL and MUC) for the intersection of those class sets.

For this work, we used the seven class model to extract time and place entities from Web pages, and use them as metadata for recommender systems. Here, we assume that the entities related to location and organization are place entities.

3.3 Domain terms

As mentioned previously, we can use topic hierarchies or named entities to identify additional information available for items in a recommender system. Another way to recognize additional information is to analyze the lexical units that will describe concepts in the domain of these items. These lexical units are called *domain terms*, and the task of identifying these terms is called *term extraction* (Korkontzelos et al. 2008; Conrado et al. 2014).

The most common way to identify domain terms is to extract only the most frequent lexical units from texts, and is called *statistical term extraction*. Another way to identify the terms, called *linguistic term extraction*, is to consider only lexical units that follow some linguistic pattern – for example, extracting only the nouns because they are normally used to describe concepts (Aggarwal and Zhai 2012). Finally, a third way to extract terms consists of using, at the same time, some statistical and linguistic patterns – this approach is called *hybrid term extraction*.

In this article, we use the DF_POS method that performs a hybrid term extraction, by selecting terms that are noun and appear in more than one document. We applied the DF_POS method to two sets of candidate words: one generated from the stemming pre-processing and the other generated from the lemmatization pre-processing. We applied the DF_POS method as follows:

Step 1: We pre-process the texts, aiming to standardize the data, and remove words that cannot be classified as terms. First, we convert all letters to lower case and remove stop-words, special characters, punctuation, numbers, accents, and words composed by only one character. Second, we annotate the remaining texts using the Stanford parser (Socher et al. 2013). Finally, we either stem the words from the texts or lemmatize them, producing two sets of terms. For our Portuguese dataset, we used the PTStemmer,[6] a stemming

[6] http://code.google.com/p/ptstemmer.

toolkit for the Portuguese language; and the lemmatizer annotator available on the LX-Center,[7] a NLP toolkit for the Portuguese language. For our English dataset, we used the well-known Porter stemming algorithm (Porter 1997), and used the lemmatizer available in the Stanford CoreNLP[8] (Manning et al. 2014).

Step 2: We perform the statistical term extraction for the stemmed and lemmatized candidate sets. The statistical part of this term extraction method removes the words that occur only in one document in the dataset, because these words are not representative, since they may be misspelling, noise or terms specific for only one document;

Step 3: The linguistic term extraction is carried out. Considering the words extracted in step 2, the linguistic part of our method removes the words that are not nouns since nouns normally represent concepts for a document.

Following these three steps, we generate two lists of words (i.e. terms) that represent the dataset. Each set of terms is used separately as additional information for the items in a recommender system with the aim of providing better recommendations.

4 Attribute-aware collaborative recommenders

We use the text mining techniques summarized in previous section to extract features from unstructured content: our aim is to use them as metadata with attribute-aware collaborative filtering algorithms.

In this work, we have used two attribute-aware collaborative filtering algorithms available in the literature, *k*-Nearest Neighbors (*k*-NN) (Desrosiers and Karypis 2011) and BPR-Mapping (Gantner et al. 2010), which we review in remainder of this section. Both recommenders were chosen because of good results we obtained when testing them with a set of metadata obtained with a single text mining technique, and because we wanted to evaluate the ensemble module using the two most well-known subclasses of collaborative filtering: *k*-NN and latent factors (Bobadilla et al. 2013). It is worth mentioning, however, that other recommender algorithms can be used in the system, as the ensemble only uses the attribute-specific rankings they generate.

4.1 *k*-Nearest Neighbors

The Item-based *k*-Nearest Neighbors (*k*-NN) is a well-known CF algorithm that generates recommendations by computing the similarity among items and taking into account the most similar items (called neighbors) (Desrosiers and Karypis 2011). We have extended this model so that the similarities among items are dictated by the item descriptor vectors instead of item rating vectors. We briefly describe the model next; a more detailed description can be found elsewhere (Koren 2010).

A common approach of CF algorithms is to adjust the data for accounting item and user bias. These effects are tendencies of users to prefer items in different manners (higher or lower ratings), or items that tend to be rated differently than the others. We encapsulate

[7] http://lxcenter.di.fc.ul.pt/index.html.

[8] http://stanfordnlp.github.io/CoreNLP.

these effects within the baseline estimates. A baseline estimate for an unknown rating $\hat{r}_{ui}$ is denoted by:

$$b_{ui} = \mu + b_u + b_i, \tag{1}$$

where μ is the global average rating, i.e., the average rating of all user-item pairs available; i and u correspond to an item and user, respectively; and b_i and b_u are the item's and user's deviations from the average. To estimate b_u and b_i one can solve a least squares problem. We adopted a simple approach which will iterate a number of times the following equations:

$$b_i = \frac{\sum_{u:(u,i)\in K}(r_{ui} - \mu - b_u)}{\lambda_1 + |\{u|(u,i) \in K\}|}, \tag{2}$$

$$b_u = \frac{\sum_{i:(u,i)\in K}(r_{ui} - \mu - b_i)}{\lambda_2 + |\{i|(u,i) \in K\}|}, \tag{3}$$

where k is the set of known items and r_{ui} is a score indicating how much a user u likes an item i. In our experiments, we iterate 10 times these equations, and set the constants λ_1 e λ_2 to 10 and 15, respectively, since these are values suggested by the literature (Koren 2010).

The goal of the recommender algorithm is to find similar items preferred by a user and to predict a score based on the scores of those similar items. In this way, a score is predicted for an unobserved (u, i) pair by considering the similar items he/she provided a preference. In order to find similar items, a similarity measure is employed between items. It can be based on several correlation or distance metrics, such as the Pearson correlation coefficient, p_{ij}, which measures the tendency of users to like items i and j similarly (Koren 2010). Another similarity measure is the cosine correlation, used mainly in the information retrieval area and in content-based algorithms (Lops et al. 2011). Using the datasets, we performed preliminary tests with both correlation metrics by considering a validation subset, and found out that the Pearson correlation coefficient presented better results. The final similarity measure is a shrunk correlation coefficient, s_{ij}:

$$s_{ij} = \frac{n_{ij}}{n_{ij} + \lambda_3} p_{ij}, \tag{4}$$

where n_{ij} is the number of features that both items i and j have in common, and λ_3 is a regularization constant, set as 100, according to the literature (Koren 2010).

Finally, we identify the k items preferred by u that are most similar to i, the k-nearest neighbors. We denote this set as $S^k(i;u)$. Using this set, the final score is an average of the k most similar items' scores, adjusted to their baseline estimate:

$$r_{ui} = b_{ui} + \frac{\sum_{j\in S^k(i;u)} s_{ij}(r_{uj} - b_{uj})}{\sum_{j\in S^k(i;u)} s_{ij}}. \tag{5}$$

4.2 Latent factors

The BPR-Mapping was proposed by Gantner et al. (2010) to use a linear mapping to enhance the item factors which will later be used in an extended matrix factorization prediction rule. Such extension of matrix factorization is optimized for Bayesian Personalized Ranking (BPR-MF) (Rendle et al. 2009) that can deal with the cold-start problem,

yielding accurate and fast attribute-aware item recommendation. Gantner et al. (2010) address the case where new users and items are added by first computing the latent feature vectors from attributes like the user's age or movie's genres, and then using those estimated latent feature vectors to compute the score from the underlying matrix factorization (MF) model.

The model considers the matrix factorization prediction rule:

$$r_{ui} = b_{ui} + p_u^T q_i = b_{ui} + \sum_{f=1}^{k} p_{uf} q_{if}, \tag{6}$$

where b_{ui} is defined equally to the one used in the k-NN algorithm, each user u is associated with a user-factors vector $p_u \in \mathbb{R}^f$, and each item i with an item-factors vector $q_i \in \mathbb{R}^f$.

From this model, the item factors are mapped according to their attributes:

$$r_{ui} = b_{ui} + \sum_{f=1}^{k} p_{uf} \phi_f(\mathbf{a}_i), \tag{7}$$

where $\phi_f : \mathbb{R}^n \rightarrow \mathbb{R}$ is defined as:

$$\phi_f(\mathbf{a}_i) = \sum_{g=1}^{n} \upsilon_{ug} a_{ig}, \tag{8}$$

which is a function that maps the item's attributes to the general preferences r_{ui} and $\mathbf{a}_i$ is a boolean vector of size n whose elements a_{ig} represent the presence of attributes. The parameter υ_{ug} is a weight indicating how much user u likes attribute g and is learned through the LearnBPR algorithm, illustrated in Algorithm 1 (Rendle et al. 2009). In this algorithm, D_K is a set triples composed of a user and a pair of items, where the first item is known by the user and the second is unknown. The symbol Θ represents the parameters of the model to be learned, Λ_Θ is a set of regularization constants, and α is the learning rate.

Input: D_K
Output: Learned parameters Θ
Initialize Θ with random values
for *count = 1,...,#Iter* **do**
  draw (u, i, j) from D_K
  $s_{uij} \leftarrow r_{ui} - r_{uj}$
  $\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-s_{uij}}}{1+e^{-s_{uij}}} \cdot \frac{\partial}{\partial \Theta} s_{uij} - \Lambda_\Theta \Theta \right)$
end

Algorithm 1: Learning through LearnBPR (Rendle et al. 2009)

In this way, the BPR-Mapping first computes the relative importance between two items (i.e. without taking into account their positions in the ranking):

$$\begin{aligned} s_{uij} &= r_{ui} - r_{uj} \\ &= \sum_{g=1}^{n} \upsilon_{ug} a_{ig} - \sum_{g=1}^{n} \upsilon_{ug} a_{jg} \\ &= \sum_{g=1}^{n} \upsilon_{ug} (a_{ig} - a_{jg}). \end{aligned} \tag{9}$$

And then, the partial derivative with respect to υ_{ug} is taken:

$$\frac{\partial}{\partial \upsilon_{ug}} s_{uij} = (a_{ig} - a_{jg}), \tag{10}$$

which is applied to Algorithm 1 considering that $\Theta = (\upsilon_*)$ for all set of users and attributes.

In this section we described two attribute-aware collaborative filtering algorithms which use metadata extracted from unstructured textual content using a set of text mining algorithms. As previously exposed, the application of different analysis on the content has the potential to create richer and more detailed descriptions, but the combination of a variety of text mining techniques into a single model is difficult to achieve in an efficient way. Our proposal addresses this issue by means of an ensemble approach, which is detailed in the next section.

5 Proposal of an ensemble approach

In this article we propose a generic approach to combine multiple metadata types extracted from unstructured textual content using a variety of text mining algorithms. The combination is accomplished in a post-processing module based on ensemble learning, which will aggregate attribute-specific rankings according to the users' preferences. Figure 1 shows a representation of the proposed approach.

The set of items in the database is analyzed by different text mining algorithms, generating a variety of metadata types for the items. As illustrated in Fig. 1a, in the work we present in this paper we use topic hierarchies, named entities and domain terms; however, additional techniques can be used to extract structured descriptions from the items. Users' interactions (Fig. 1b) are also analyzed, in order to construct a profile representation of each user containing his/her main interests. In this work, we use only implicit feedback, which is dictated by the users' viewing history of items. Both pieces of information (items' descriptions and users' interactions) are the two fundamental elements for recommender algorithms. Therefore, along with information about users, we apply each of the metadata types (domain terms, topics and named entities) in a particular recommender (Fig. 1c) which, in the context of the work reported in this article, is either Item-based k-NN or BPR-Mapping. Other content-based or attribute-aware collaborative filtering algorithms could be used, as long as they generate an attribute-specific ranking of items for each user according to his/her preferences (Fig. 1d).

The set of users' interactions and items' metadata is also used in the training phase of our model (Fig. 1e). In this module, the system will iterate over the training samples in order to adjust a set of weights that controls how much each metadata type influences the final recommendations, based on the users' interests. This is an important step because the

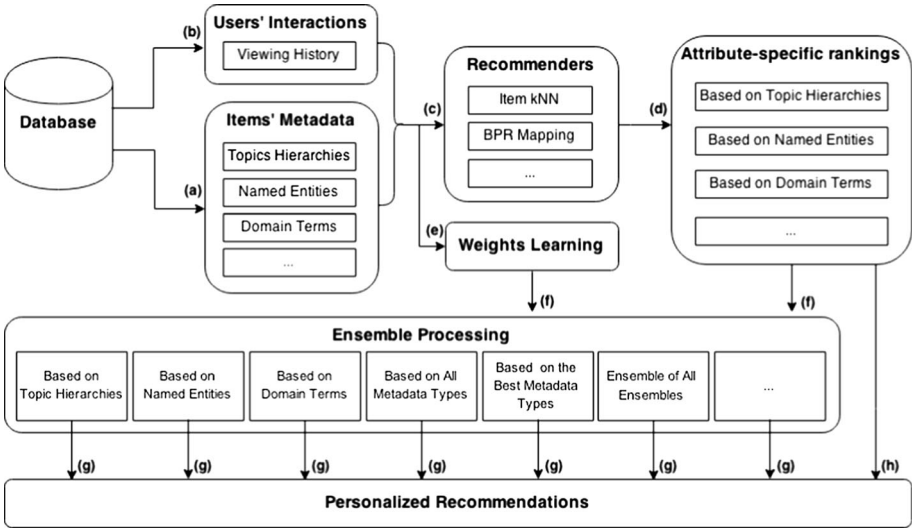

Fig. 1 Schematic view of the proposed approach

model can adapt itself to a variety of metadata types and combinations that are used at a particular time in the system.

After learning the set of weights, they are used to combine the involved attribute-specific rankings in the ensemble processing module (Fig. 1f). The goal of this module is to aggregate two or more rankings into a unique and final list of items (Fig. 1g), which will be recommended to the user. Our hypothesis is that this aggregated ranking will have a better accuracy when compared to any of the attribute-specific rankings, as it will consider and combine additional metadata types about the items generated by different text mining algorithms. Such attribute-specific rankings are used in our work as baselines, so we also use these rankings as final recommendations for the sake of evaluating our proposal (Fig. 1h).

As aforementioned, the proposed model is flexible and extensible to different combinations of metadata types. In this way, the ensemble processing module illustrated in Fig. 1 has a set of predefined use cases which are evaluated in Sect. 6. They are:

- *Based on Topic Hierarchies (Ensemble_Topics):* ensemble using metadata generated by different configurations of the topic hierarchies construction technique. These configurations are related to the level of granularity of topics used to describe items;
- *Based on Named Entities (Ensemble_Entities):* ensemble using metadata composed of different classes of named entities (place, time and the combination of both);
- *Based on Domain Terms (Ensemble_Terms):* ensemble using metadata composed of terms which were extracted using two different methods (lemmatization and stemming);
- *Based on All Metadata Types (Ensemble_All_Metadata):* ensemble using metadata generated by all text mining algorithms and their variations;
- *Based on the Best Metadata Types (Ensemble_Best_Metadata):* ensemble using metadata generated by the best variation of each text mining algorithm. Such best variation is chosen after executing the algorithms with isolated metadata types;

– *Ensemble of All Ensembles (Ensemble_All_Ensembles):* the results of the ensembles of topic hierarchies, named entities and domain terms are combined in a second ensemble.

The modules illustrated in Fig. 1 are executed according to a number of steps, which are detailed next.

5.1 Step 1: Data division and execution of recommender algorithms

The first step consists of generating the attribute-specific rankings. To do that, the database is firstly divided into training and test sets, where the former is also divided into two subsets: the first part is used in the evaluation of the proposal, and the second is used for learning the ensemble weights as detailed in Step 2.

The training set, which is composed of user's interactions and textual items, is processed as follows: initially, we apply n text mining techniques in order to generate the structured items' metadata of n types. Then, the set of items, metadata types and users' interactions are used to train n recommender algorithms, one for each type of metadata. This training procedure is dependent on the recommender technique used; in case of Item-based k-NN or BPR-Mapping, Sect. 4 provides a description of this task. In summary, each instance of a recommender receives as input a set of metadata of a particular type, and outputs an attribute-specific personalized ranking for each user.

5.2 Step 2: Learning of weights for ensembling

The second step consists of learning a set of weights which will be used to combine all considered rankings in the ensemble module. Each weight is adjusted according to the relevance of each type/set of metadata, and they are used in the ensemble by means of a linear function, represented by r_{ui}^{final}:

$$r_{ui}^{final} = \beta_a r_{ui}^{a} + \beta_b r_{ui}^{b} + ... + \beta_n r_{ui}^{n}. \tag{11}$$

In Equation 11, the parameters $r_{ui}^{a}, r_{ui}^{b}, ..., r_{ui}^{n}$ indicate the scores computed previously by each recommender algorithm instance for a (u, i) pair, and $\beta_a, \beta_b, ..., \beta_n$ are the weights of each individual score for the final prediction, which have to be learned. In other words, $\beta_a, \beta_b, ..., \beta_n$ dictate how much the different types of metadata influence the aggregated ranking to be recommended to the user.

To learn the weights $\beta_a, \beta_b, ..., \beta_n$, we use the LearnBPR algorithm, as shown in Algorithm 1. We chose to use this algorithm because it is able to efficiently deal with implicit feedback, as it considers the relative order between a pair of items to optimize the final ranking.

Thus, considering Algorithm 1, we first have to compute s_{uij} (Algorithm 1, line 4), where in this case, we substitute r_{ui} and r_{uj} for the final prediction rule specified in Equation 11, yielding:

$$s_{uij} := \beta_a(r_{ui}^{a} - r_{uj}^{a}) + \beta_b(r_{ui}^{b} - r_{uj}^{b}) + ... + \beta_n(r_{ui}^{n} - r_{uj}^{n}). \tag{12}$$

According to Equation 11, our model is composed of the parameters $\Theta = \{\beta_a, \beta_b, ..., \beta_n\}$, which have to be learned. The next instruction in Algorithm 1, line 5, is the adjustments of the involved parameters:

$$\Theta \leftarrow \Theta + \alpha\left(\frac{e^{-s_{uij}}}{1+e^{-s_{uij}}} \cdot \frac{\partial}{\partial\Theta} s_{uij} - \Lambda_\Theta \Theta\right), \tag{13}$$

where α is the learning rate, Λ_Θ is the set of regularization constants and $s_{uij} = r_{ui}^{final} - r_{uj}^{final}$ (Equation 12).

In order to adjust each parameter as shown, we have to calculate the partial derivative of s_{uij} in relation to Θ, as follows:

$$\frac{\partial}{\partial\Theta} s_{uij} = \begin{cases} r_{ui}^{a} - r_{uj}^{a} & \text{if } \Theta = \beta_a, \\ r_{ui}^{b} - r_{uj}^{b} & \text{if } \Theta = \beta_b, \\ \quad ... & \\ r_{ui}^{n} - r_{uj}^{n} & \text{if } \Theta = \beta_n. \end{cases} \tag{14}$$

In this way, after a number of iterations as specified in Algorithm 1, the set of parameters $\beta_a, \beta_b, ..., \beta_n$ will be adjusted according to the importance of each metadata type in the final prediction.

5.3 Step 3: Ensemble and recommendation

As soon as we have computed the weights $\beta_a, \beta_b, ..., \beta_n$, we apply Equation 11 to all (u, i) pairs (and corresponding scores $r_{ui}^{a}, r_{ui}^{b}, ..., r_{ui}^{n}$) that occur in the attribute-specific rankings, resulting in a final score r_{ui}^{final} for each (u, i) pair that will be used to sort the list of recommendations. Consequently, the final score r_{ui}^{final} will be composed of the scores computed by each of the n instances of the recommender algorithm (one for each metadata type). These scores, in turn, contribute to the final score according to the weights $\beta_a, \beta_b, ..., \beta_n$ learned as explained in the previous step.

```
   Input: users, items, users' interactions, items' metadata
   Output: final ranking R'(u, final)
1  for u ∈ users do
2  |   for i ∈ items do
3  |   |   Compute r^a_ui, r^b_ui, ..., r^n_ui
4  |   |   Compute β_a, β_b, ..., β_n
5  |   |   Compute r^final_ui
6  |   |   Aggregate r^final_ui into R(u, final)
7  |   end
8  |   R'(u, final) ← sort_desc(R(u, final))
9  end
```

Algorithm 2: Proposed algorithm to compute personalized recommendations based on ensemble

Algorithm 2 shows the overall process of our proposed approach. Line 3 corresponds to the execution of the n instances of the recommender algorithm, one for each metadata type (Fig. 1c). As a result, there will be n attribute-specific rankings $r_{ui}^{a}, r_{ui}^{b}, ..., r_{ui}^{n}$ for each (u, i) pair (Fig. 1d). Following, line 4 of Algorithm 2 corresponds to the adjustment of weights $\beta_a, \beta_b, ..., \beta_n$, as shown in Sect. 5.2 (Fig. 1e). After this training phase, line 5 consists of computing the final scores for each (u, i) pair, according to Equation 11 (Fig. 1f). Finally, lines 6 and 8 illustrate, respectively, the aggregation of items into the final ranking for each user, and its sorting in descending order according to the final score (Fig. 1g).

6 Empirical evaluation

To evaluate our proposal, we first compare our ensemble technique against each individual attribute-aware collaborative recommender. This comparison was carried out regarding each metadata type. Then, we compare the different types of ensembles among them, in order to demonstrate which one provides the best results. This evaluation was executed for both considered attribute-awararte collaborative recommenders, *k*-Nearest Neighbors and BPR-Mapping, and two datasets, as follows.

6.1 Datasets

The first dataset used in the experiments is from the *Brazilian Embrapa Agency of Technology Information* website.[9] This dataset consists of 4,659 users, 15,037 user accesses, which we use as user relevance, and 1,543 Web pages about agribusiness, written in Portuguese language. The textual content of the pages, crawled from the dataset, was used to obtain the sets of metadata types, i.e., topics, entities and terms. The second one is the HetRec MovieLens 2*k* dataset,[10] introduced by Cantador et al. (2011). It is one of the most widely used datasets for evaluating recommender system performance, and one of the few that contains additional information besides the user-item interactions, such as content features and demographic data. This database consists of 800,000 ratings (ranging from 1 to 5) and 10,000 tag assignments applied by 2,113 users into 10,197 movies. From this dataset, we used as textual information the movies' storylines, which are written in English language. In our experiments, we have not considered any structured metadata from both datasets.

For the topic hierarchies, we considered the topics generated by the BC^2 method as described in Sect. 3.1. For the topic hierarchy construction, we used different runs of the well-known *k*-means algorithm (with random centers initializations and cosine similarity) to obtain several data partitions for the consensus clustering. To analyze the effect of the number of topics used as metadata in the recommendation task, we selected subsets of topics using seven different granularities: $\{50, 100\}$, $\{15, 20\}$, $\{10, 15\}$, $\{10, 50\}$, $\{5, 10\}$, $\{5, 100\}$ and $\{2, 7\}$. In the granularity configuration $\{x, y\}$, the parameter x identifies the minimum number of items allowed in the topic, while the parameter y identifies the maximum number of items per topic. When a topic has a few items associated, usually the topic represents more specific information. On the other hand, topics with many items associated represent more general information about the items. Thus, the seven configurations presented above generate subsets of 26, 44, 101, 210, 305, 510 and 1230 topics, respectively, for the Embrapa dataset. For the HetRec MovieLens 2*k* dataset, we have 61, 187, 490, 902, 1398, 2246 and 5463 topics, respectively.

In the evaluation of the impact of named entities, we considered as metadata the entities related to place, time, and their combination extracted from the textual data by using the tools REMBRANDT and Stanford NER, as described in Sect. 3.2. For the Embrapa dataset, we have 877 different entities related to place, 1,334 related to time, and the combination of both named entities generates a total of 2,211 entities. For the HetRec MovieLens 2*k* dataset, we have 5,986 different entities related to place, 618 related to time, and the combination of both named entities generates a total of 6,604 entities.

[9] http://www.agencia.cnptia.embrapa.br.

[10] http://grouplens.org/datasets/hetrec-2011.

Finally, for the terms extraction, the Embrapa dataset contains 6,826 lemmatized and 5,449 stemmed terms. The HetRec MovieLens 2*k* dataset contains 2,690 lemmatized and 12,774 stemmed terms.

6.2 Experimental setup and evaluation metrics

To measure the predictive ability of the recommender system, we used the All But One protocol (Breese et al. 1998) with 10-fold cross-validation, and calculated the MAP (Mean Average Precision). To do this, the sessions in the dataset were randomly partitioned into 10 subsets, while making sure that each user has at least one interaction in the training set so we can guarantee that the recommendation model can learn the preferences of all users. For each fold, we used $n-1$ of those subsets of data for training and the remaining one for testing. The training set T_r was used to build the recommendation model. For each user in the test set T_e, we randomly hid one item, referred to as the singleton set H. The remaining items represented the set of observables, O, based on which the recommendation was made. Finally, we computed the MAP, as follows.

The Mean Average Precision metric computes the precision considering the respective position in the ordered list of recommended items. With this measure, we obtain a single value accuracy score for a set of test users T_e:

$$MAP(T_e) = \frac{1}{|T_e|}\sum_{j=1}^{|T_e|} AveP(R_j, H_j), \tag{15}$$

where the average precision (AveP) is given by

$$AveP(R_j, H_j) = \frac{1}{|H_j|}\sum_{r=1}^{|H_j|}[Prec(R_j, r) \times \delta(R_j(r), H_j)], \tag{16}$$

where $Prec(R_j, r)$ is the precision for all recommended items up to ranking r and $\delta(R_j(r), H_j) = 1$, iff the predicted item at ranking r is a relevant item $(R_j(r) \in H_j)$ or zero otherwise.

We computed MAP@N, for N equal to 1, 3, 5 and 10 recommendations. For each configuration and measure, we calculate the mean to summarize the 10-fold values. To compare two recommendation algorithms, we applied the two-sided paired t-test with a 95% confidence level (Mitchell 1997).

Regarding the parameters of the algorithms, we defined a set of values which performed well for both datasets, Embrapa and HetRec MovieLens 2*k*. Such definitions were made by cross-validation in the training set. For the *k*-Nearest Neighbors algorithm, we ran experiments for k equals to 10, 40, 70 and 100 neighbors, and chose k equals to 10 as it provided

Table 1 Parameters used in this evaluation

Parameter	Value	Note
$\#Iter$	30	Number of iterations of the LearnBPR algorithm (see Algorithm 1)
α	0.05	Learning rate of the LearnBPR algorithm
Λ_{v_*}	0.025	Regularization constant of the BPR-Mapping algorithm
Λ_{p_*,q_*}	0.025	Regularization constant of user/item factors for the BPR-MF algorithm
Λ_{b_*}	0.0	Regularization constant of user/item biases for the BPR-MF algorithm

Table 2 Comparative evaluation of our ensemble technique using the *k*-Nearest Neighbors algorithm on Embrapa (EM) and MovieLens (ML) datasets

Each Topic Hierarchy *versus* Ensemble_Topics

Topics	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Topics	**0.006**	**0.009**	**0.015**	**0.015**	**0.017**	**0.16**	**0.019**	**0.019**
Topic_2_7	0.005***	0.008***	0.009***	0.014***	0.010***	0.015***	0.010***	0.017***
Topic_5_10	0.005***	0.008**	0.009***	0.014***	0.010***	0.015***	0.010***	0.018***
Topic_5_100	0.005***	0.008***	0.009***	0.014***	0.010***	0.016***	0.010***	0.018***
Topic_10_15	0.005***	0.008***	0.009***	0.014***	0.009***	0.015***	0.010***	0.018***
Topic_10_50	0.005***	0.008***	0.009***	0.014***	0.010***	0.015***	0.010***	0.018***
Topic_15_20	0.005***	0.008***	0.009***	0.014***	0.009***	0.015***	0.010***	0.017***
Topic_50_100	0.005***	0.008***	0.010***	0.014***	0.010***	0.015***	0.011***	0.017***

Each Named Entity *versus* Ensemble_Entities

Named Entities	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Entities	**0.006**	**0.008**	**0.016**	**0.014**	**0.018**	**0.016**	**0.019**	**0.018**
Entity_Place	0.004***	0.007***	0.009***	0.012***	0.010***	0.014***	0.019***	0.017***
Entity_Place_Time	0.005***	0.007***	0.011***	0.013***	0.011***	0.015***	0.012***	0.017***
Entity_Time	0.004***	0.008***	0.010***	0.014***	0.010***	0.015***	0.011***	0.017***

Each Domain Term *versus* Ensemble_Terms

Domain Terms	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Terms	**0.010**	**0.010**	**0.021**	**0.015**	**0.023**	**0.016**	**0.025**	**0.018**

Table 2 continued

Each Domain Term *versus* Ensemble_Terms

Domain Terms	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Term_Lemmatized	0.007***	0.006***	0.012***	0.014***	0.013***	0.015***	0.013***	0.017***
Term_Stemmed	0.008**	0.007***	0.017***	0.013***	0.018***	0.014***	0.019***	0.017***

Ensemble_Topics, Ensemble_Entities and Ensemble_Terms *versus* Ensemble_All_Metadata

Ensembles	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_All_Metadata	**0.011**	0.008	**0.022**	**0.016**	**0.024**	**0.017**	**0.026**	0.018
Ensemble_Topics	0.006***	0.009***	0.015***	0.015***	0.017***	0.016***	0.019***	**0.019****
Ensemble_Entities	0.006***	0.008***	0.016***	0.014***	0.018***	0.016***	0.019***	0.018***
Ensemble_Terms	0.010***	**0.010**	0.021***	0.015	0.023***	0.016	0.025***	0.018

Ensemble_Topics, Ensemble_Entities and Ensemble_Terms *versus* Ensemble_Best_Metadata

Ensembles	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Best_Metadata	**0.011**	0.008	**0.021**	**0.016**	**0.023**	**0.017**	**0.025**	0.018
Ensemble_Topics	0.006***	0.009***	0.015***	0.015***	0.017***	0.016***	0.019***	**0.019****
Ensemble_Entities	0.006***	0.008***	0.016***	0.014***	0.018***	0.016***	0.019***	0.018***
Ensemble_Terms	0.010***	**0.010**	**0.021****	0.015	0.023***	0.016	**0.025****	0.018

Table 2 continued

Ensemble_Topics, Ensemble_Entities and Ensemble_Terms *versus* Ensemble_All_Ensembles								
Ensembles	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_All_Ensembles	**0.011**	0.009	**0.022**	**0.015**	**0.023**	**0.016**	**0.027**	0.017
Ensemble_Topics	0.006***	0.009***	0.015***	**0.015*****	0.017***	**0.016*****	0.019***	**0.019*****
Ensemble_Entities	0.006***	0.008***	0.016***	0.014***	0.018***	0.016***	0.019***	0.018***
Ensemble_Terms	0.010***	**0.010**	0.021***	0.015**	0.023***	0.016	0.025***	0.018

The following notation is used for statistical significance: 1 star (*) when p value<0.1; 2 stars (**) when p value<0.05; and 3 stars (***) when p value<0.001. The best results are in boldface

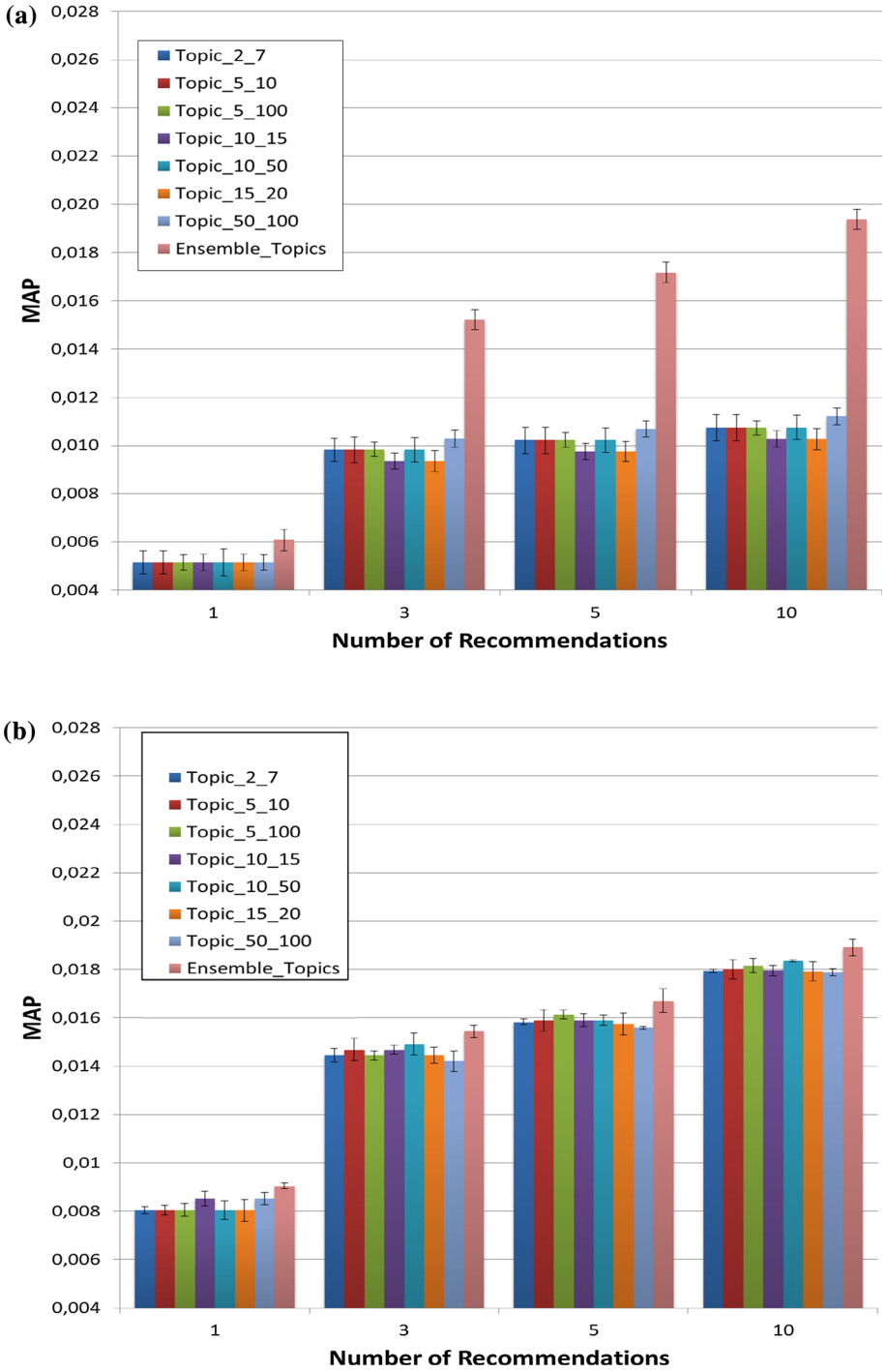

Fig. 2 Comparing the ensemble of topics against the individual topic-based recommenders for the *k*-Nearest Neighbors algorithm. **a** Embrapa dataset. **b** HetRec MovieLens 2*k* dataset

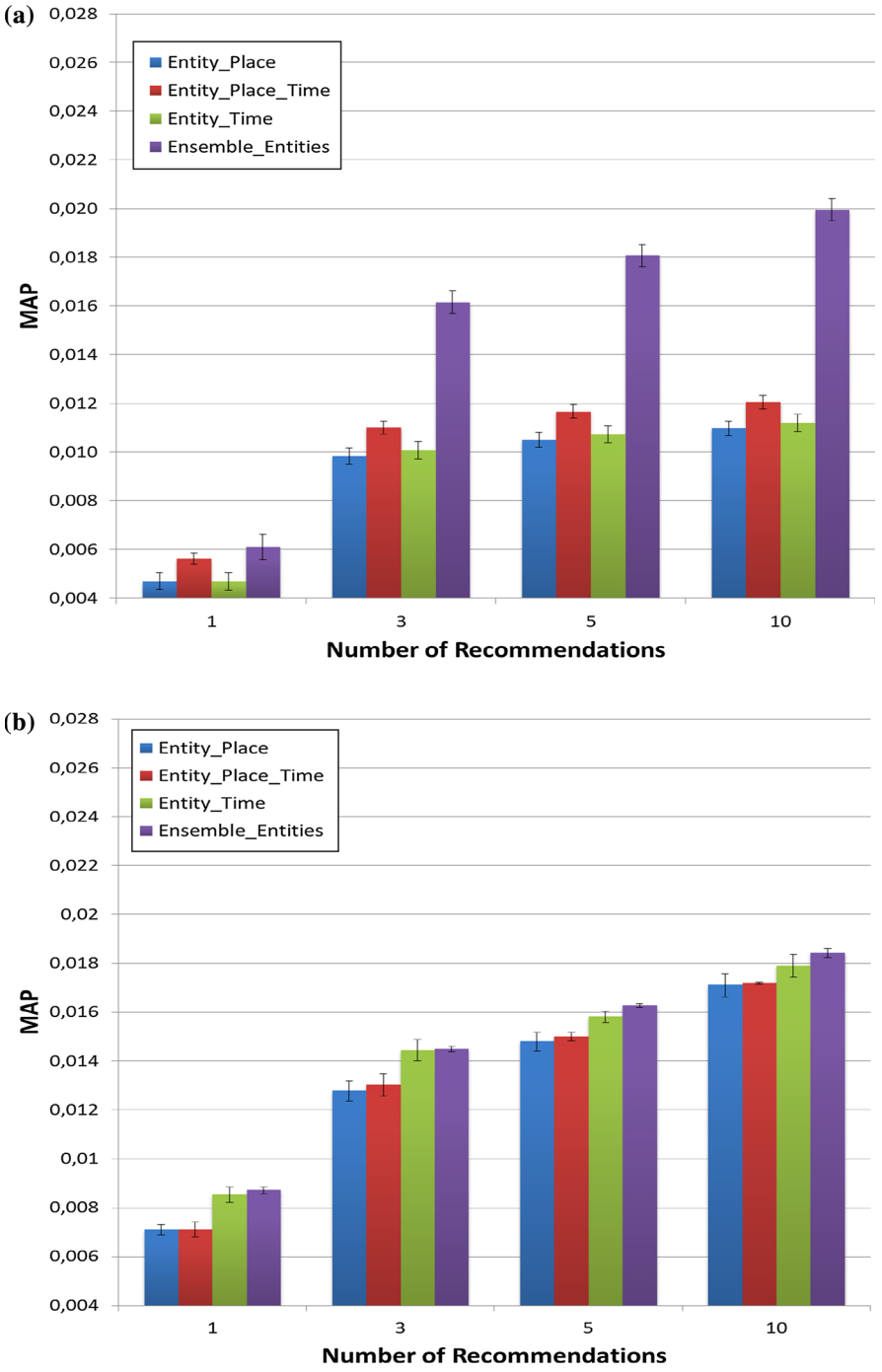

Fig. 3 Comparing the ensemble of entities against the individual entity-based recommenders for the *k*-Nearest Neighbors algorithm. **a** Embrapa dataset. **b** HetRec MovieLens 2*k* dataset

the best results. For the BPR-Mapping algorithm, we ran experiments for the latent factors 10, 40, 70 and 100, and chose the number of 100 latent factors as it presented the best results for this algorithm. Other parameters and corresponding values are presented in Table 1.

6.3 Results

The results of the experiments are presented for both algorithms, *k*-Nearest Neighbors and BPR-Mapping, used by our proposal.

6.3.1 Ensembles with the k-Nearest Neighbors Algorithm

In this section we show the results for the *k*-Nearest Neighbors algorithm. Table 2 presents the results for both datasets, Embrapa and HetRec MovieLens 2*k*. We first compare each individual attribute-aware collaborative recommender (i.e. regarding each metadata type) against our ensemble technique. Then, we compare the first three ensembles (Ensemble_Topics, Ensemble_Entities and Ensemble_Terms) against the Ensemble_All_Metadata, Ensemble_Best_Metadata and Ensemble_All_Ensembles, which are used as baseline in the last three analyses. For the Embrapa dataset, all ensembles present results which are statistically significant (i.e. p value<0.05). For the MovieLens dataset, most of the results are statistically significant. These facts demonstrate that our hypothesis of improving attribute-specific rankings with ensembles is valid for the *k*-Nearest Neighbors algorithm.

We also present the same results, with their respective standard deviation, in bar graphics for both datasets. In Fig. 2a, b, the graphic illustrates the results for the topics at different granularities and for the ensemble of topics. The graphic represents in x-axis the number of recommendations and in the y-axis the values of MAP, with the error bars (standard deviation). In both figures, we see that the values of MAP for the individual recommenders (i.e. using topics at different granularities) are quite close to each other, while the ensemble of topics presents values of MAP higher than all individual recommenders– which means that the ensemble of topics provides better recommendations.

Similarly to the results obtained with the ensemble of topics, the ensemble of named entities also presented better results when compared with the entities separately (Fig. 3a, b). In Fig. 3a, we note that the results of the ensemble are better when compared with the entities place and time together and these, in turn, are better than the entities of place and time separately.

The ensemble approach also led to the best results for metadata containing the domain terms. Analyzing in Fig. 4a the results for the use of lemmatized terms, stemmed terms and the ensemble of terms, we observe that the best results are presented by the ensemble, followed by stemmed terms and, finally, lemmatized terms. On the other hand, analyzing in Fig. 4b, we observe that the best results are presented by the ensemble, followed by lemmatized terms and, finally, stemmed terms in most cases.

Finally, in Fig. 5a, b we compare the six different ensembles built using the *k*-Nearest Neighbors algorithm (Ensemble_Terms, Ensemble_Entities, Ensemble_Topics, Ensemble_All_Ensembles, Ensemble_Best_Metadata and Ensemble_All_Metadata). Note that the Ensemble_Best_Metadata, for this algorithm, was built with Term_Stemmed, Entity_Place_Time and Topic_50_100.

Comparing all ensembles, in Fig. 5a, the results show that the values of MAP for the ensemble of all metadata, the ensemble of best metadata, the ensemble of all ensembles and the ensemble of terms present values of MAP close among themselves, values which are better than the ensembles of topics and named entities. In Fig. 5b, the results show that the values of MAP for all ensembles are close among themselves.

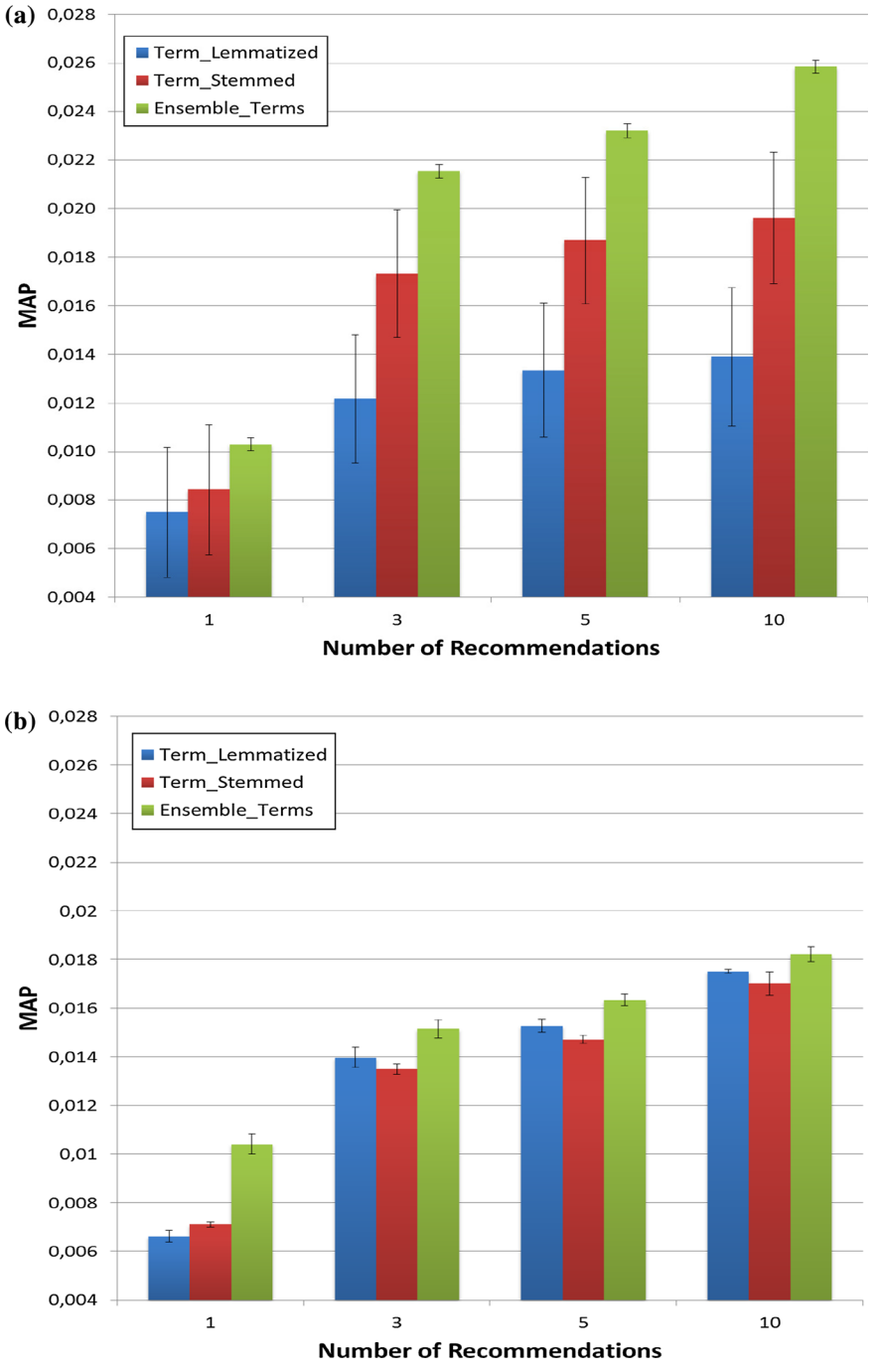

Fig. 4 Comparing the ensemble of terms against the individual term-based recommenders for the *k*-Nearest Neighbors algorithm. **a** Embrapa dataset. **b** HetRec MovieLens 2*k* dataset

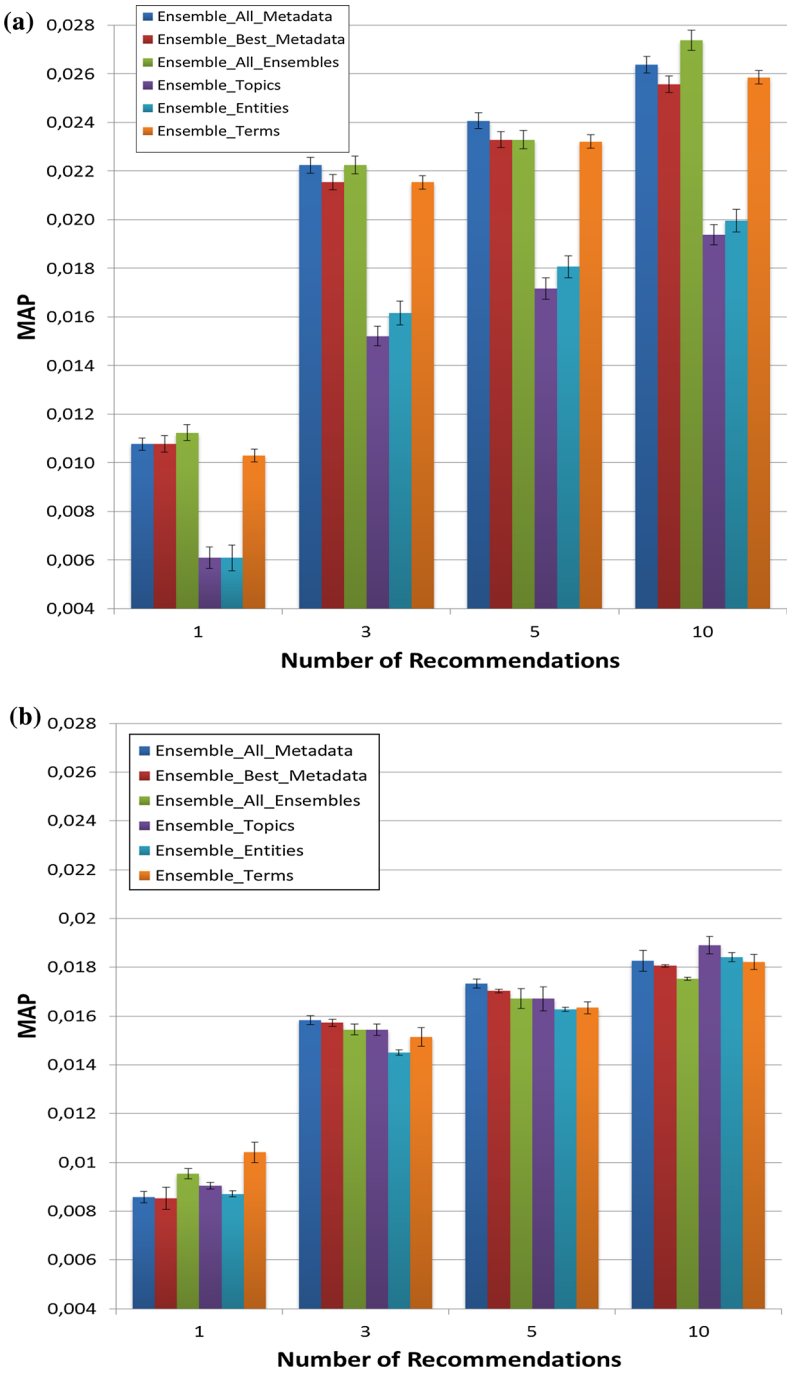

Fig. 5 Comparing all ensembles for the *k*-Nearest Neighbors algorithm. **a** Embrapa dataset. **b** HetRec MovieLens 2*k* dataset

6.3.2 Ensembles with the BPR-Mapping Algorithm

The results for the BPR-Mapping algorithm are presented in Table 3 and Fig. 6a, b. Again, we first compare each individual attribute-aware collaborative recommender (i.e. regarding each metadata type) against our ensemble technique. Then, we compare the first three ensembles (Ensemble_Topics, Ensemble_Entities and Ensemble_Terms) against the Ensemble_All_Metadata, Ensemble_Best_Metadata and Ensemble_All_Ensembles. For the BPR-Mapping we also have some results which are not statistically significant (i.e. p value>= 0.05). Regarding only the values which are statistically significant (i.e. p value< 0.05), we can see that our ensemble technique provides better results in 15 out of 24 evaluations (6 types of comparison x 4 values of MAP), i.e., in more than 50% of the experiments for each dataset. In the table, we can also see that Ensemble_Best_Metadata and Ensemble_All_Ensembles provide the same value for MAP.

The results for the use of each topic and for the ensemble of topics are also presented in Fig. 6a, b. For the Embrapa dataset, the ensemble presents better values for 1, 5 and 10 recommendations. Note that though the figure shows that Topic_10_15 presents better results for 1 recommendation, we know by the Table 3 that this value is not statistically significant. For the HetRec MovieLens 2k dataset, the ensemble presents better values for 3, 5 and 10 recommendations.

Figure 7a shows that the results for the use of named entities and for the ensemble of named entities are very similar. However, the ensemble has values which are a little better than the individual named entities for 1, 3 and 10 recommendations. For the HetRec MovieLens 2k dataset (Fig. 7b), we see that the ensemble of terms provide better results only for 1 and 10 recommendations.

The Fig. 8a must be analyzed together with the Table 3. If we only look to Fig. 8a, we will observe that the stemmed term provides better results than lemmatized term and the ensemble of terms. However, taking a look at the Table 3, we will see that some values are not statistically significant, and that the ensemble of terms provide better results for 1 and 3 recommendations. On the other hand, the results in Fig. 8b show clearly that the ensemble of terms provides better results for 3, 5 and 10 recommendations.

Finally, we also compare the six different ensembles built using the BPR-Mapping algorithm (Ensemble_Terms, Ensemble_Entities, Ensemble_Topics, Ensemble_All_Ensembles, Ensemble_Best_Metadata and Ensemble_All_Metadata). Note that the Ensemble_Best_Metadata, for this algorithm, was built with Term_Stemmed, Entity_Place_Time and Topic_10_50. By comparing all ensembles, in Fig. 9a, the results show that the values of MAP for the ensemble of all ensembles and the ensemble of best metadata present the best results for 3, 5 and 10 recommendations. For the HetRec MovieLens 2k dataset (Fig. 9b), the same fact occurs only for 1 and 5 recommendations.

6.3.3 Comparing ensembles for k-Nearest Neighbors against BPR-Mapping

In this section, we compare the ensembles built by using the k-Nearest Neighbors algorithm against the ones built by using the BPR-Mapping algorithm. In Fig. 10a, b, we can see that for all ensembles, the ones built with the BPR-Mapping algorithm present the highest values of MAP considering 10 recommendations.

We also compare Ensemble_All_Ensembles against a set of baselines: BPR-MF (Rendle et al. 2009), a collaborative filtering based matrix factorization algorithm which consists of providing personalized ranking of items to a user according only to

Table 3 Comparative evaluation of our ensemble technique using the BPR-Mapping algorithm on Embrapa (EM) and MovieLens (ML) datasets

Each Topic Hierarchies *versus* Ensemble_Topics

Topics	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Topics	0.039	0.013	0.087	**0.033**	**0.098**	**0.038**	**0.106**	**0.044**
Topic_2_7	0.038***	0.009**	0.085***	0.019***	0.095***	0.025***	0.102	0.030***
Topic_5_10	0.038***	0.013*	0.085***	0.027	0.094***	0.031	0.103***	0.037
Topic_5_100	0.038***	0.012***	0.084*	0.027***	0.095	0.031***	0.102	0.037***
Topic_10_15	**0.040**	0.011	0.087***	0.025	0.096***	0.031	0.104**	0.037
Topic_10_50	0.038	0.013	0.086***	0.027	0.097***	0.032	0.105***	0.037
Topic_15_20	0.038**	**0.014*****	0.082***	0.026***	0.093	0.031***	0.102***	0.037***
Topic_50_100	0.038***	0.011***	**0.088*****	0.029***	0.096***	0.033***	0.105***	0.038***

Each Named Entity *versus* Ensemble_Entities

Named Entities	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Entities	**0.039**	**0.014**	**0.087**	0.029	0.095	0.034	**0.103**	**0.040**
Entity_Place	0.038***	0.009	0.083***	0.027***	0.094**	0.032***	0.103***	0.038***
Entity_Place_Time	0.038***	0.013	0.085***	**0.030*****	**0.096*****	**0.035*****	0.103***	0.039***
Entity_Time	0.036***	0.012	0.085***	0.024***	0.094***	0.028***	0.102***	0.034***

Table 3 continued

Each Domain Term *versus* Ensemble_Terms

Domain Terms	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Terms	**0.040**	0.012	0.086	**0.030**	0.095	**0.036**	0.103	**0.042**
Term_Lemmatized	0.039	0.008***	0.084***	0.020***	0.092	0.026***	0.100**	0.033***
Term_Stemmed	0.039	**0.013*****	**0.087**	0.029***	**0.097****	0.034***	**0.104****	0.040***

Ensemble_Topics, Ensemble_Entities and Ensemble_Terms *versus* Ensemble_All_Metadata

Ensembles	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_All_Metadata	0.039	0.012	**0.088**	**0.034**	0.097	**0.038**	0.105	**0.045**
Ensemble_Topics	0.039***	0.013**	0.087***	0.033***	**0.098****	0.038***	**0.106*****	0.044***
Ensemble_Entities	0.039***	**0.014*****	0.087***	0.029***	0.095**	0.034***	0.103***	0.040***
Ensemble_Terms	**0.040*****	0.012***	0.086***	0.030***	0.095**	0.036***	0.103***	0.042***

Ensemble_Topics, Ensemble_Entities and Ensemble_Terms *versus* Ensemble_Best_Metadata

Ensembles	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Best_Metadata	0.039	**0.015**	**0.091**	0.032	**0.104**	**0.039**	**0.110**	0.043
Ensemble_Topics	0.039***	0.013**	0.087***	**0.033*****	0.098***	0.038***	0.106***	**0.044*****
Ensemble_Entities	0.039***	0.014***	0.087***	0.029***	0.095**	0.034***	0.103***	0.040***

Table 3 continued

Ensemble_Topics, Ensemble_Entities and Ensemble_Terms *versus* Ensemble_Best_Metadata								
Ensembles	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_Terms	**0.040*****	0.012***	0.086***	0.030***	0.095***	0.036***	0.103***	0.042***

Ensemble_Topics, Ensemble_Entities and Ensemble_Terms *versus* Ensemble_All_Ensembles								
Ensembles	MAP@1		MAP@3		MAP@5		MAP@10	
	EM	ML	EM	ML	EM	ML	EM	ML
Ensemble_All_Ensembles	0.039	**0.015**	**0.091**	0.032	**0.104**	**0.039**	**0.110**	0.043
Ensemble_Topics	0.039***	0.013**	0.087***	**0.033*****	0.098***	0.038***	0.106***	**0.044*****
Ensemble_Entities	0.039***	0.014***	0.087***	0.029***	0.095**	0.034***	0.103***	0.040***
Ensemble_Terms	**0.040*****	0.012***	0.086***	0.030***	0.095***	0.036***	0.103***	0.042***

The following notation is used for statistical significance: 1 star (*) when p value<0.1; 2 stars (**) when p value<0.05; and 3 stars (***) when p value<0.001. The best results are in boldface

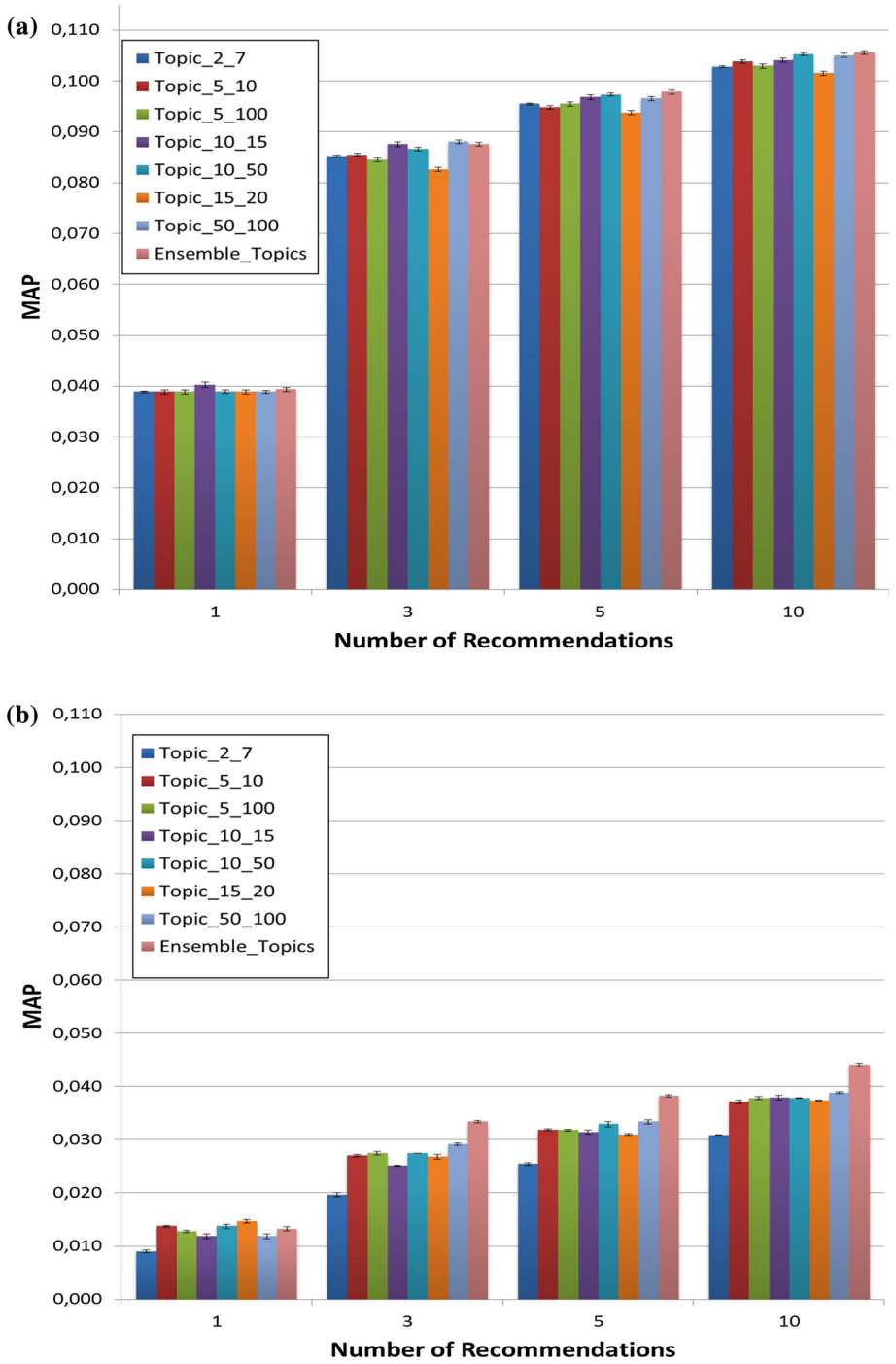

Fig. 6 Comparing the ensemble of topics against the individual topic-based recommenders for the BPR-Mapping algorithm. **a** Embrapa dataset.**b** HetRec MovieLens 2*k* dataset

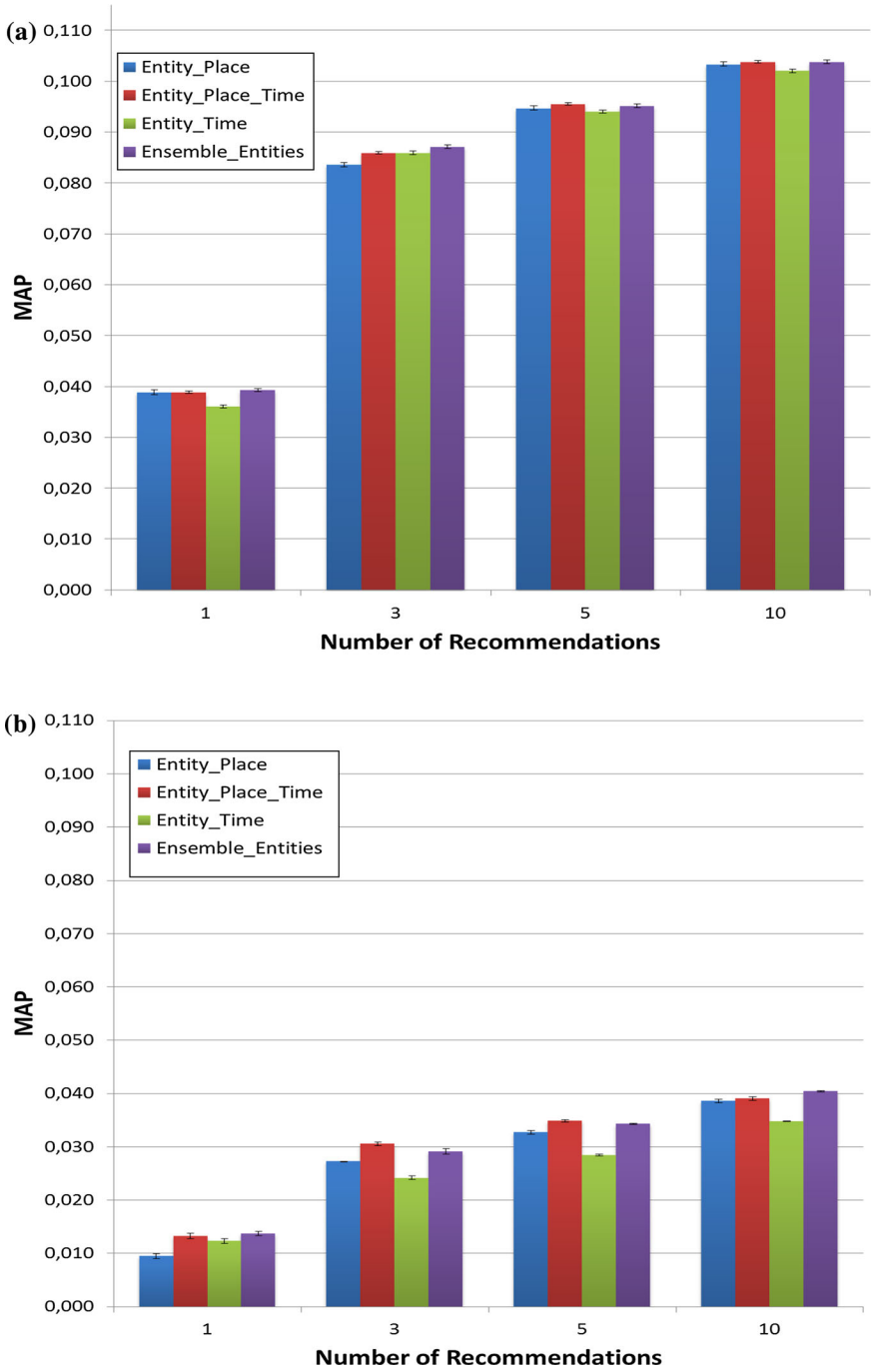

Fig. 7 Comparing the ensemble of entities against the individual entity-based recommenders for the BPR-Mapping algorithm. **a** Embrapa dataset.**b** HetRec MovieLens 2*k* dataset

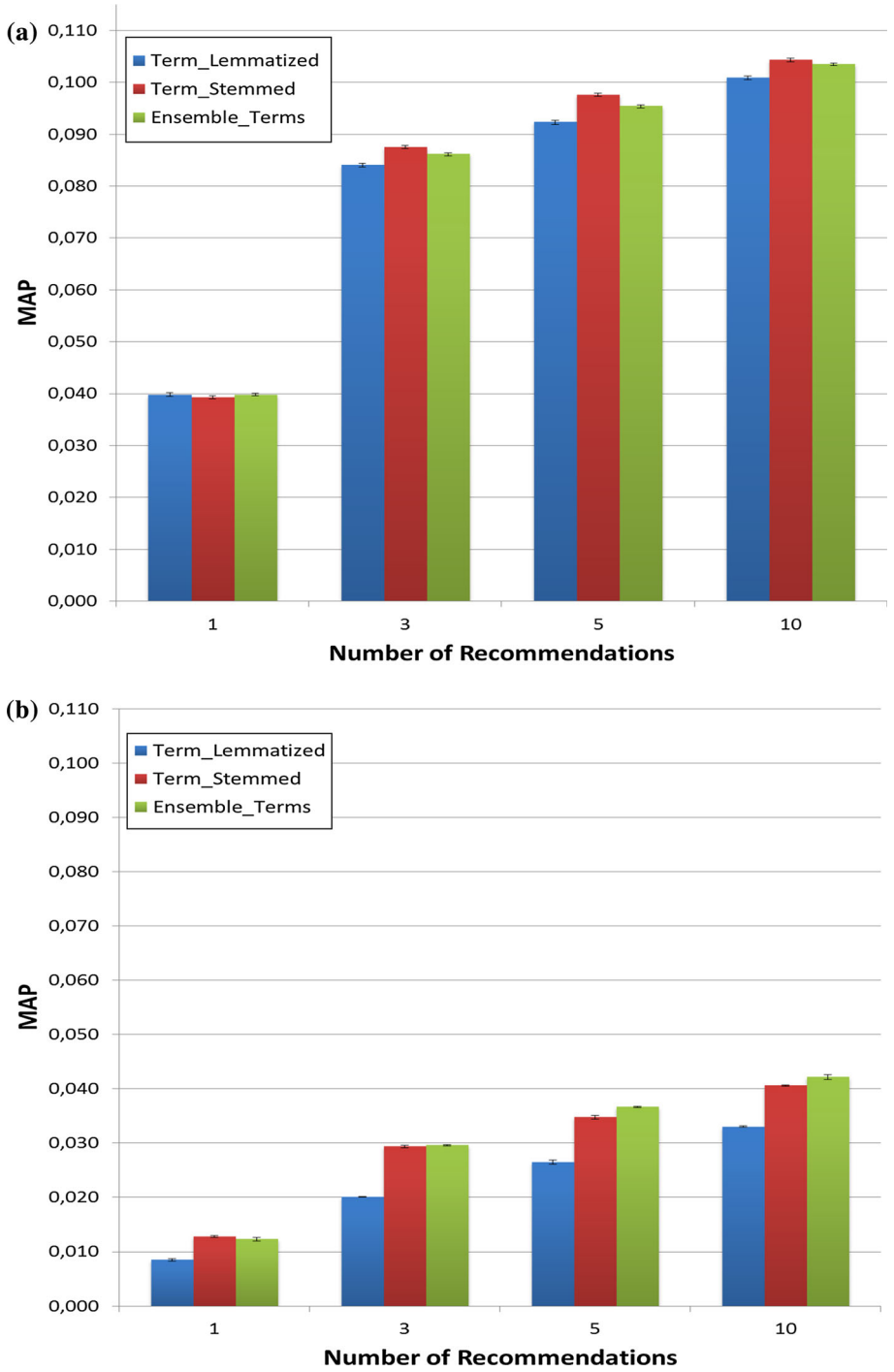

Fig. 8 Comparing the ensemble of terms against the individual term-based recommenders for the BPR-Mapping algorithm. **a** Embrapa dataset.**b** HetRec MovieLens 2*k* dataset

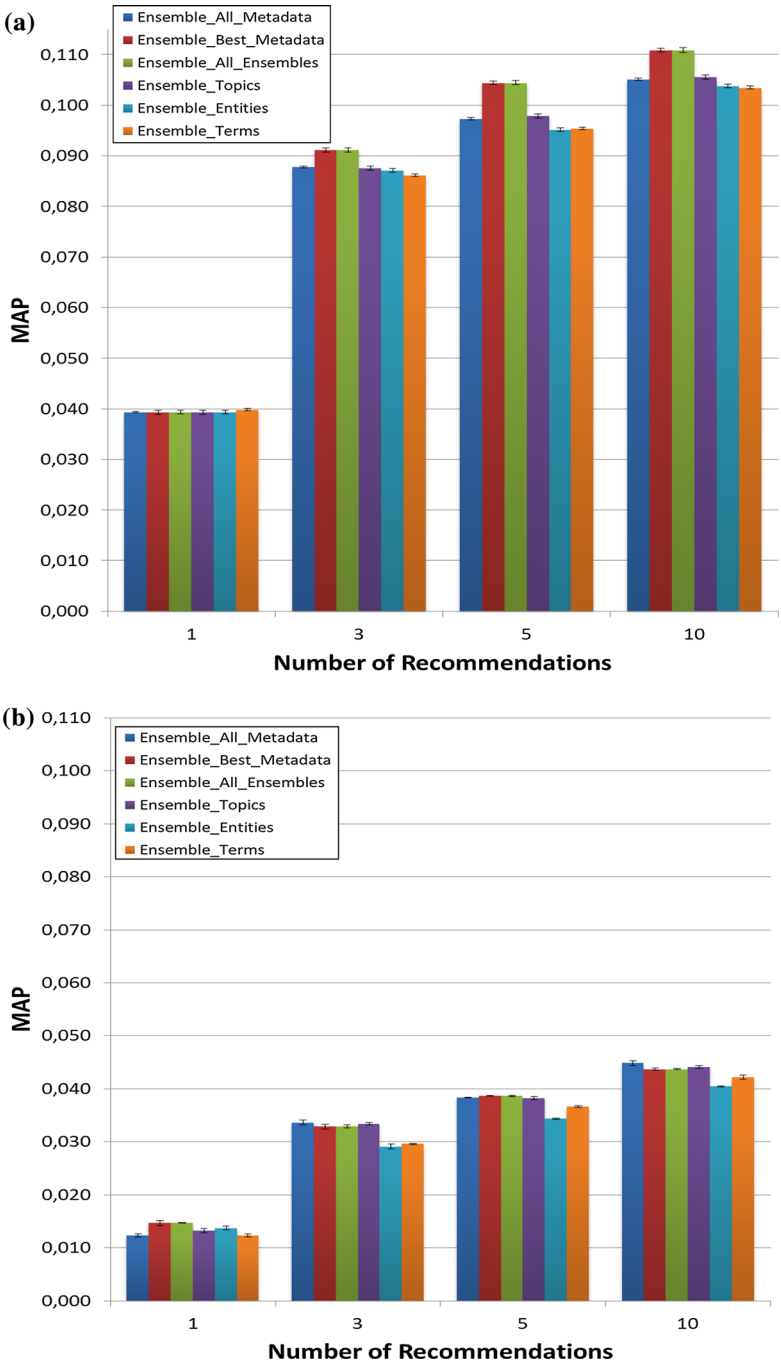

Fig. 9 Comparing all ensembles for the BPR-Mapping algorithm. **a** Embrapa dataset.**b** HetRec MovieLens 2k dataset

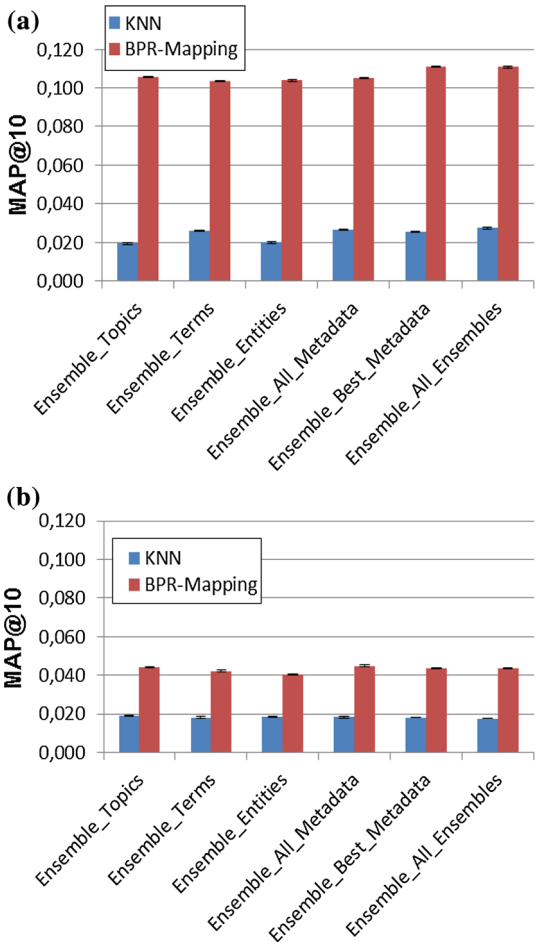

Fig. 10 Comparing ensembles for *k*-Nearest Neighbors against BPR-Mapping. **a** Embrapa dataset.**b** HetRec MovieLens 2*k* dataset

implicit feedback (e.g. navigation, clicks, etc); most popular items, which recommends the most viewed items to a particular user; and random, which simply chooses a random item to recommend to the user. These three algorithms do not exploit metadata from the items. In general, we can see in Fig. 11a that our Ensemble_All_Ensembles using the BPR-Mapping algorithm provides better results than the other algorithms. In Fig. 11b, the Ensemble_All_Ensembles provides the best results for 1 and 3 recommendations. This fact shows that by using different types of metadata to describe items, we can provide better recommendations.

6.4 Discussion

As detailed in the previous sections, the use of ensembles provides better results than individual metadata for most cases, which indicates that our proposal has potential to improve results provided by recommender systems.

For the topics in Fig. 2a, b, we have not found a pattern between the granularities and the values of MAP. However, we can see that by using all the granularities in an ensemble,

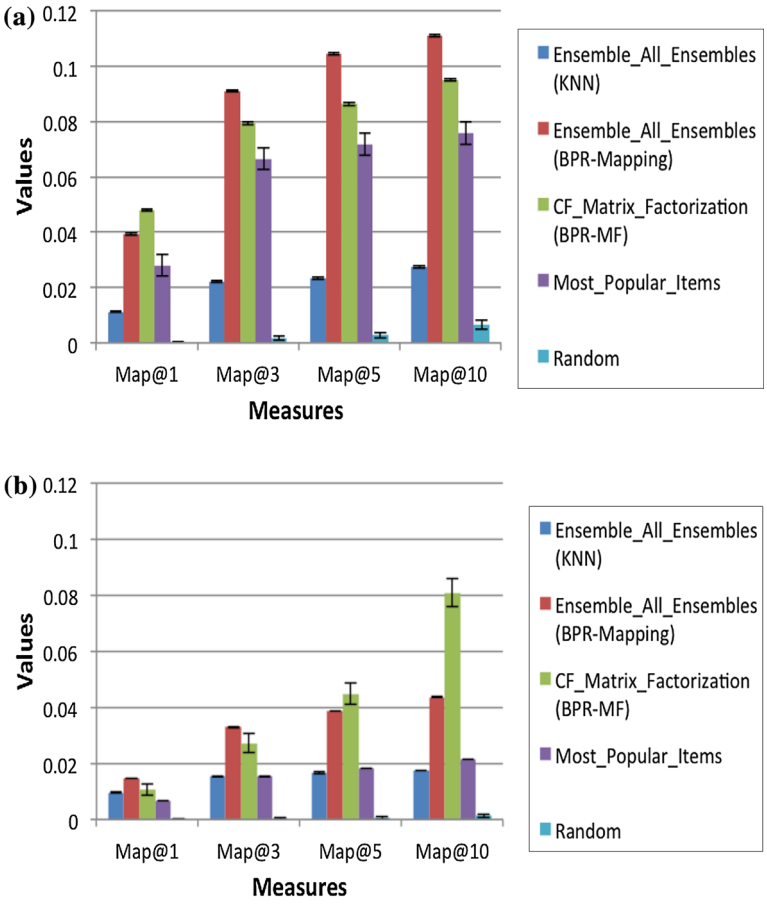

Fig. 11 Comparing Ensemble_All_Ensembles against baselines. **a** Embrapa dataset.**b** HetRec MovieLens 2*k* dataset

we can provide better results. In Fig. 3a, we see that by using the entities place and time together, we obtain better values of MAP than using each metadata separately. But, if we combine both metadata by using our proposed approach, the results are even better than the previous combination. Regarding the terms, our proposed approach provides the highest values of MAP (Fig. 4a, b). Finally, we see in Fig. 5a, b that the more data we have the better results we obtain with our ensemble approach.

Regarding the BPR-Mapping algorithm, we see in Table 3 and Fig. 6a, b that our proposed approach has provided improvements in more than 50% of the cases. In Fig. 9a, b, we see again that the more data we have the better results we obtain with our ensemble approach.

In Figure 10a, we see that the Ensemble_All_Ensemble provided the best results for both algorithms, while Ensemble_Terms provided the worst results for the BPR-Mapping algorithm and Ensemble_Topics provided the worst results for the *k*-Nearest Neighbors algorithm. Finally, we can see in Figure 11a that the Ensemble_All_Ensemble for the

BPR-Mapping provides better results than a collaborative filtering based matrix factorization algorithm (BPR-MF), demonstrating that by using different types of metadata we can provide better recommendations.

Contrary to the *k*-Nearest Neighbors, for the BPR-Mapping algorithm there is often not much difference between the ensembles and individual metadata, and sometimes the individual metadata are better than the ensembles. Based on Koren (2008), a possible explanation for this fact is: 1) By applying our ensemble on the *k*-Nearest Neighbors, we provide much better results because each individual result exploits a particular aspect of content, according to a specific metadata, and by combining the various individual results, we obtain a better overall representation from the metadata; 2) On the other hand, we believe that the BPR-Mapping algorithm is detecting relationships between similar items for any type of metadata, since it has an overview of all data. Thus, the ensemble ends up not having so much impact.

Other interesting fact is that by using the BPR-Mapping algorithm with individual metadata, we sometimes obtain better results than using our proposed ensemble technique with the *k*-Nearest Neighbors algorithm. Although this fact is true, there are some scenarios more favorable to using the *k*-Nearest Neighbors than the BPR-Mapping. For example, it is easier to explain recommendations generated with the *k*-Nearest Neighbors than with the BPR-Mapping algorithm. As the algorithm computes the most similar items to a particular one, a recommendation can be explained simply by saying that item was recommended because it is close to other liked by the user. In the case of BPR-Mapping, as it is based on matrix factorization (MF), the system does not know which items are close to one preferred by the user.

Another advantage of using *k*-NN-based approaches is the ease of incremental updates, as additional items may be incorporated in the system simply by computing its correlation to other items. In MF-based approaches, however, all relationships among items have to be re-computed periodically.

Therefore, in such conditions where *k*-Nearest Neighbors is favorable, the proposed ensemble model can be used to incorporate distinguished features from the content using simple and efficient text mining techniques. As shown in the experiments, the approach is flexible and extensible to different combinations of metadata types, recommender algorithms and datasets, although some of these configurations result in marginal improvements over the baselines (in particular MF-based algorithms).

7 Conclusion and future work

This article proposed an ensemble approach for the combination of different text mining algorithms applied to the recommendation of textual items. Instead of designing an integrated model composed of different features to extract rich and detailed information from unstructured content, we developed a post-processing module based on ensemble learning, which combines a number of attribute-specific rankings generated by content-based recommenders using a variety of items' descriptions types. The evaluation experiments show the effectiveness of our proposal.

The main advantages of our proposal is extensibility and flexibility. The ensemble module allows developers to use different text mining algorithms and attribute-aware recommenders, as it uses only the output from these techniques to ensemble the partial rankings (produced by the execution of those techniques). This is an important feature of

our approach, because it gives the freedom to developers to use any number of metadata extraction algorithms and recommender techniques as they have available. Furthermore, this flexibility avoids additional efforts to combine the techniques in an efficient way, as our model has a training phase which will learn the contributions of each method for the final recommendations.

Indeed, our approach could be extended to the recommendation of non-textual items, such as video, songs, photos, etc., being necessary, in this matter, the use of adequate indexing mechanisms. The proposed ensemble module, in turn, allows to use simple and straightforward feature extractors, as opposite to more complex and integrated indexing algorithms.

As future work, we plan to evaluate the system with additional datasets from other domains and different recommendation algorithms in order to check the accuracy with different information types. Furthermore, we plan to extend the ensemble module to add a user-based clustering procedure, so that computational power could be saved with data constraints.

Acknowledgments Thanks to the support from Grants 2012/13830-9, 2013/16039-3, 2013/10756-5, 2013/22547-1, 2014/08996-0, São Paulo Research Foundation (FAPESP); CNPq (311659/2011-0) and CAPES/Brazil.

References

Aggarwal, C. C., & Zhai, C. X. (2012). *Mining Text Data*. Incorporated: Springer Publishing Company.

Bar, A., Rokach, L., Shani, G., Shapira, B., & Schclar, A. (2013). Improving simple collaborative filtering models using ensemble methods. *Multiple Classifier Systems*, Lecture Notes in Computer Science, vol 7872, Springer Berlin Heidelberg pp. 1–12.

Bobadilla, J., Ortega, F., Hernando, A., & GutiéRrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, *46*, 109–132.

Breese, J. S., Heckerman, D., Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, pp. 43–52.

Cantador, I., Brusilovsky, P., Kuflik, T. (2011). 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In: Proceedings of the 5th ACM conference on Recommender systems, ACM, New York, NY, USA, RecSys '11.

Cardoso, N. (2012) Rembrandt-a named-entity recognition framework. In: Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12), pp. 1240–1243.

Conrado, M. S., Rossi, R. G., Pardo, T. A. S., & Rezende, S. O. (2013). Applying transductive learning for automatic term extraction: The case of the ecology domain. In: *Proceedings of the IEEE 2nd INT CNF on informatics and applications (ICIA)* Lodz, Poland, pp. 264–269.

Conrado, M. S., Di Felippo, A., Pardo, T. A. S., & Rezende, S. O. (2014). A survey of automatic term extraction for Brazilian Portuguese. *Journal of the Brazilian Computer Society (JBCS)*, *20*(1), 12.

D'Addio, R. M., Conrado, M., Resende, S., Manzato, M. G. (2014). Generating recommendations based on robust term extraction from users' reviews. In: Proceedings of the 20th Brazilian Symposium on Multimedia and the Web, WebMedia 2014, João Pessoa, Brazil, November 18-21, 2014, pp 55–58.

Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook* (pp. 107–144). New York: Springer. doi:10.1007/978-0-387-85820-3_4.

Domingues, M. A., Sundermann, C. V., Manzato, M. G., Marcacini, R. M., & Rezende, S. O. (2014). Exploiting text mining techniques for contextual recommendations. In: 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), Warsaw, Poland, August 11-14, 2014—volume I, pp. 210–217.

Domingues, M. A., Sundermann, C. V., Barros, F., Manzato, M. G., Pimentel, M. G. C., Rezende, S. O., & Oliveira, S. (2015). Applying multi-view based metadata in personalized ranking for recommender

systems. In: Proceedings of the 30th ACM/SIGAPP Symposium On Applied Computing, pp. 1105–1107.
Finkel, J. R., Grenager, T., Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '05, pp 363–370, doi:10.3115/1219840.1219885.
Gantner, Z., Drumond, L., Freudenthaler, C., Rendle, S., Schmidt-Thieme, L. (2010). Learning attribute-to-feature mappings for cold-start recommendations. In: 2010 IEEE 10th international conference on data mining (ICDM), pp. 176–185.
Ganu, G., Kakodkar, Y., & Marian, A. (2013). Improving the quality of predictions using textual information in online user reviews. *Information Systems*, *38*(1), 1–15.
Hariri, N., Mobasher, B., Burke, R., & Zheng, Y. (2011). Context-aware recommendation based on review mining. In: Proceedings of the 9th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems, ITWP '11, pp. 1–7.
Jahrer, M., Töscher, A., Legenstein, R. (2010). Combining predictions for accurate recommender systems. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, KDD '10, pp. 693–702.
Kim, H., Han, K., Yi, M., Cho, J., & Hong, J. (2012). Moviemine: Personalized movie content search by utilizing user comments. *IEEE Transactions on Consumer Electronics*, *58*(4), 1416–1424.
Koren, Y. (2008). 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, New York, NY, USA, KDD '08, pp. 426–434.
Koren, Y. (2010). Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *4*(1), 1:1–1:24.
Korkontzelos, I., Klapaftis, I. P., Manandhar, S. (2008). Reviewing and evaluating automatic term recognition techniques. In: Nordström B, Ranta A (eds) Proceedings of the 6th international conference on advances in natural language processing, Springer-Verlag, Berlin, Heidelberg, Lecture Notes in Computer Science, vol 5221, pp. 248–259.
Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the eighteenth international conference on machine learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ICML '01, pp. 282–289, http://dl.acm.org/citation.cfm?id=645530.655813.
Li, Y., Nie, J., Zhang, Y., Wang, B., Yan, B., & Weng, F. (2010). Contextual recommendation based on text mining. In: Proceedings of the 23rd international conference on computational linguistics: Posters, COLING '10, pp. 692–700.
Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook* (pp. 73–105). New York: Springer. doi:10.1007/978-0-387-85820-3_3.
Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. New York, NY, USA: Cambridge University Press.
Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In: Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations, pp. 55–60.
Manzato, M. G., Domingues, M. A., Marcacini, R. M., & Rezende, S. O. (2014). Improving personalized ranking in recommender systems with topic hierarchies and implicit feedback. In: 22nd international conference on pattern recognition, ICPR 2014, Stockholm, Sweden, August 24–28, 2014, pp. 3696–3701.
Marcacini, R. M., Hruschka, E. R., Rezende, S. O. (2012). On the use of consensus clustering for incremental learning of topic hierarchies. In: Proceedings of the 21st Brazilian conference on Advances in Artificial Intelligence, Springer, Berlin, Heidelberg, SBIA'12, pp. 112–121.
Mikheev, A., Moens, M., & Grover, C. (1999). Named Entity Recognition without Gazetteers. *EACL '99: Proceedings of the Ninth Conference on European Chapter of the Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 1–8.
Mitchell, T. M. (1997). *Machine learning*. New York: McGraw-Hill.
Nothman, J., Ringland, N., Radford, W., Murphy, T., & Curran, J. R. (2013). Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, *194*, 151–175.
Porter, M. F. (1997) Readings in information retrieval. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, chap An Algorithm for Suffix Stripping, pp. 313–316. http://dl.acm.org/citation.cfm?id=275537.275705.

Qumsiyeh, R., Ng, Y. K. (2012). Predicting the ratings of multimedia items for making personalized recommendations. In: Proceedings of the 35th international ACM SIGIR conference on research and development in information retrieval, New York, NY, USA, SIGIR '12, pp. 475–484.

Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L. (2009). BPR: bayesian personalized ranking from implicit feedback. In UAI '09 Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, pp. 452–461.

Ricci, F., Rokach, L., Shapira, B. (2011). Introduction to recommender systems handbook. In: Ricci F, Rokach L, Shapira B, Kantor PB (eds) Recommender Systems Handbook, Springer US, pp. 1–35, doi:10.1007/978-0-387-85820-3_1.

Ristoski, P., Loza Mencía, E., & Paulheim, H. (2014). A hybrid multi-strategy recommender system using linked open data. In A. Tordai, V. Presutti, M. Stankovic, E. Cambria, I. Cantador, A. Di Iorio, T. Di Noia, C. Lange, & D. Reforgiato Recupero (Eds.), *Semantic web evaluation challenge, communications in computer and information science* (Vol. 475, pp. 150–156). New York: Springer International Publishing.

Sekine, S. (2004). Named entity: History and future. http://cs.nyu.edu/∼sekine/papers/NEsurvey200402.

Semeraro, G., Lops, P., Basile, P., de Gemmis, M. (2009). Knowledge infusion into content-based recommender systems. In: Proceedings of the third ACM conference on recommender systems, ACM, New York, NY, USA, RecSys '09, pp. 301–304.

Socher, R., Bauer, J., Manning, C. D., Ng, A. Y. (2013). Parsing with compositional vector grammars. In Proceedings of the 51st annual meeting of the association for computational linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers, pp. 455–465.