Ontology-based Web Recommendation from Tags

Nizar R. Mabroukeh and C. I. Ezeife
School of Computer Science
University of Windsor
Windsor, ON. N9B 3P4 Canada
mabrouk@uwindsor.ca

***Abstract*—With the advent of social networks and tagging systems, The Internet has recently witnessed a big leap in the use of Web Recommendation Systems WRS. Based on users' likings of items and their browsing history on the world wide web, these systems are able to predict and recommend items and future purchases to users. They are being used now in various domains, like news article recommendation, product recommendation, and make-friend recommendation.**

WRS are still limited by several problems, of which are sparsity, and the new user problem. They also fail to make full use and harness the power of domain knowledge and semantic web ontologies. In this article, we discuss how an ontology-based WRS can utilize relations and concepts in an ontology, along with user-provided tags, to provide top-*n* recommendations without the need for item clustering or user ratings. For this purpose, we also propose a dimensionality reduction method based on the domain ontology, to solve the sparsity problem.

***Index Terms*—Recommendation Systems; Semantic Web; Domain Ontology; Tagging; Folksonomies; Dimensionality Reduction;**

I. Introduction

A Web Recommendation System (WRS), is a system that is capable of providing real time recommendations to users based on their browsing history and/or similarity to other users' browsing behavior. The recommendation set can include items like merchandise, news articles, movie titles, ...etc. These systems mainly depend on web usage mining as an underlying architecture [7], which is concerned with finding user navigational patterns on the world wide web by extracting knowledge from web logs. A WRS uses this knowledge to predict user's preferences, and recommend certain web sites or products to him. These content-based WRS do not usually follow a domain ontology to extract knowledge about the recommended items or the user history, which results in limitations like the new user problem, who arrives at the system without prior browsing history, and the sparsity problem, which is attributed to the small range of items browsed by few users, as not all items surface in the browsing history. Most WRS use clustering to group users and items that are similar, and provide recommendations to users depending on other users of their group that share the same "taste" in items [2], [3] , or group the items under topic taxonomies that do not really capture any semantic relations, other than the hierarchical *is-a* relation, among the clusters of items [8], [9] . In any case, clustering is an expensive process that results in large matrices of similarity scores.

In this article[1] we propose the utilization of user-provided tags. Systems that allow users to tag items are becoming very familiar (e.g., Delicious[2]. Flickr[3], YouTube[4]). In the proposed system, these tags are mapped to concepts of a domain ontology of the underlying application, bypassing the need to cluster the items. By doing this, the process of recommendation converts to a process of finding concepts that are similar to the active user's tags, and recommending the top-*n* items that share these concepts (the active user is the current user requiring recommendation).

II. Problem Definition and Motivation

Define a *pageview* as the visual rendering of a web page in a specific environment at a specific point in time. A *clickstream* can be defined as a sequential series of pageview requests. The pageview represents a certain product or topic of interest to the user, called an *item*. We will use the terms item and pageview interchangeably. Items are associated with *tags* provided by users as keywords that describe these items, and the users' interests in them.

Definition 1: Given a database of items and their associated tags, the web log can be represented as a clickstream $W = \{< p_i, T_{p_i} >: 1 \leq i \leq |W|\}$, containing pairs of item p_i and set of tags T_{p_i}, where $T_{p_i} = \{g_j : 1 \leq j \leq |\mathbb{G}|\}$, g_j is a tag and $\mathbb{G}$ is the set of all user-provided tags.

Following is an example web log:

$$\begin{aligned} W = \{ &< p_2, [\text{“}close\text{”}, \text{“}quality\text{”}, \text{“}clear\text{”}, \text{“}lens\text{”}] >, \\ &< p_5, [\text{“}tight\text{”}, \text{“}water\text{”}, \text{“}ocean\text{”}, \text{“}dive\text{”}] >, \\ &< p_3, [\text{“}scuba\text{”}, \text{“}mask\text{”}, \text{“}snorkel\text{”}, \text{“}dive\text{”}] >, \\ &< p_1, [\text{“}professional\text{”}, \text{“}video\text{”}, \text{“}flash\text{”}, \text{“}digital\text{”}] >, \\ &< p_4, [\text{“}dry\text{”}, \text{“}toy\text{”}, \text{“}small\text{”}] >\} \end{aligned}$$

The active user u provides a set of tags, either directly as a query or indirectly by providing these tags as keywords of her favorite product features, stored in her profile when she signs up in the web site. Define a utility function $\lambda(u, p_i)$, which measures the interest of u in item p_i. A WRS finds the top-*n* items that maximize this function. In other words, it finds the top-*n* items that the active user could be most interested in.

[1]This research was supported by the Natural Science and Engineering Research Council (NSERC) of Canada under an operating grant (OGP-0194134) and a University of Windsor grant.

[2]www.delicious.com

[3]www.flickr.com

[4]www.youtube.com

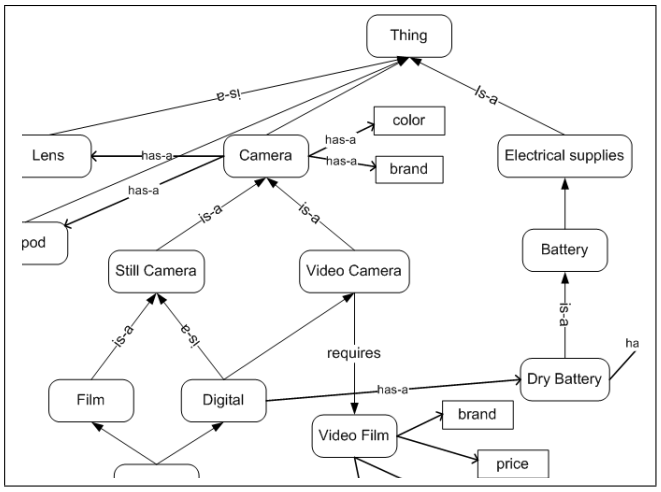

Fig. 1. Part of an example ontology for the domain of Cameras.

Since tags are freely provided by users, there is no limit on their number, which leads to a huge dimensionality problem. On the other hand, domain knowledge is now made available on the world wide web, due to the continuous development and use of Semantic Web and Web 2.0 technologies. It is provided as an underlying ontology for several web applications (like the Internet Movie Database $IMDb$[5]).

A domain ontology is a formal representation of a domain knowledge, by a set of concepts $\mathcal{C}$ within the domain and the relationships $\mathcal{R}$ among them. An ontology $\mathcal{O}$ is defined as a structure $\mathcal{O} := (\mathcal{C}, \leq_{\mathcal{C}}, \mathcal{R}, \sigma, \leq_{\mathcal{R}}, \mathcal{A})$ consisting of:

- two disjoint sets $\mathcal{C}$ and $\mathcal{R}$. $\mathcal{C}$ is the set of concepts/classes, which are entities in the ontology domain, and $\mathcal{R}$ is a set of relations defined among the concepts,
- a partial order $\leq_{\mathcal{C}}$ on $\mathcal{C}$, called concept hierarchy or taxonomy,
- a function $\sigma : \mathcal{R} \rightarrow \mathcal{C}^+$ called *signature* (where $\mathcal{C}^+$ is the set of all finite tuples of elements in $\mathcal{C}$),
- a partial order $\leq_{\mathcal{R}}$ on $\mathcal{R}$, called *relation hierarchy*, and
- a set $\mathcal{A}$ of logical axioms in some logical language $\mathcal{L}$, that can describe constraints on the ontology [11].

Figure 1 shows part of an ontology for the domain of Cameras. Oval shapes represent concepts (also called classes) in $\mathcal{C}$, and edges represent relations in $\mathcal{R}$. Concept hierarchy is represented by *is-a* edges. The figure also shows other relations, like *has-a* and *requires*, e.g., one can notice that "Video Camera" *is-a* "Camera", and that a "Camera" *has-a* "Lens".

Only few WRS use domain knowledge, and those that do, mostly, use it in the form of a topic taxonomy (i.e., categorization of items), which is referred to as a *shallow ontology*. The full power of a domain ontology with relations is still to be utilized in WRS. This power enables the formulation of more accurate similarity measures between concepts, since concepts share several relations besides the *is-a* relation. Use of ontology with relations also provides better interpretability of recommendation results. The proposed system provides this ability by utilizing axioms $\mathcal{A}$ to expand the recommendation set.

[5]http://www.imdb.com

III. Contribution and Outline

Previously, we showed how to integrate domain knowledge in web usage mining [6]. This article picks up on such usage of domain ontology and contributes to the area of WRS, by: (1) showing how ontology concepts can be used in place of clustering to provide top-*n* content-based recommendations from user-provided tags, rather than using tags directly. By doing this we provide solutions to several limitations of WRS, the new user problem is solved by the fact that only tags are used that can be extracted from the user's query, with no user browsing history required. (2) The sparsity problem is also solved by a proposed method of dimensionality reduction using the ontology, in which several concepts are combined by their direct subsuming superconcept (the Lowest Common Ancestor – LCA). (3) A method of Spreading Activation is proposed to expand recommendation using all ontology relations, rather than simple taxonomic *is-a* relations.

Next in section IV, we introduce the proposed recommender system in three steps, namely, *clicksream mapping*, *active recommendation*, and *recommendation expanding* with a rough algorithm. Then each step is detailed in section V, section VI and section VII respectively, along with a running example. Related work is presented in section VIII, then testing and experimental results are provided in section IX. Finally, we discuss conclusions and future work in section X.

IV. Using Tags with Domain Ontology for Recommendation

In general, the complete proposed system goes through the following steps:

- *Clickstream mapping*: The web log is represented as clickstream containing pairs of items and their associated tags. Similarity is computed between each item and each ontology leaf concept, and all similarity scores are stored in a matrix of items×concepts, called the PC matrix. Notice that the dimensionality of the matrix depends on the number of leaf concepts in the ontology.
- *Active recommendation*: As the active user arrives at a certain web page (or buys a certain product), the tags associated with this item are retrieved, and a vector is generated, that is similar to one row of the offline PC matrix generated in the previous step. The vector shows the similarity between the active user tags and each concept. This vector is matched against each row in the matrix, and the top-n matching items are used as the recommendation set.
- *Expanding recommendation*: The recommendation set can be expanded by increasing n, and by expanding the active user vector using semantic relations in the ontology to include more concepts, not present in the matrix, from which recommendation can be drawn.

Algorithm 1 *SemAwareIN: Ontology-based WRS with user Tags*

STEP 1: ***Clickstream_mapping***(W, $\mathcal{O}$):
Input: item database W,
domain ontology $\mathcal{O}$,
Output: matrix of items×concepts PC
Algorithm:
1: **for** each pair <item p_i, its tags T_{p_i} > in W **do**
2: **for** each concept c_j in $\mathcal{O}$ **do**
3: calculate $sim(T_{p_i}, c_j)$, which is the average similarity between each tag in T_{p_i} and c_j
4: store in PC, $PC[i, j] = sim(T_{p_i}, c_j)$
5: **end for**
6: **end for**
7: **return** PC
end

STEP 2: ***Active_Recommend***(PC, T_{p^u}, n):
Input: matrix of items×concepts PC,
active user tags T_{p^u},
number of highly similar items required n,
Output: set of top n recommended items, $\mathcal{S}$
Algorithm:
1: [**optional**] Reduce dimensionality of PC matrix as described in section V-A
2: Map user tags to concepts similar to steps 2-3 in ***Clickstream_mapping***()
3: store the mapping result as vector $\vec{p^u}$
4: **for** each item p_i in PC **do**
5: Find $rel(\vec{p^u}, \vec{p_i})$, which is the relatedness of the item to user vector $\vec{p^u}$, using equation (2)
6: **end for**
7: Sort results from previous step and store top n results in $\mathcal{S}$
8: **return** $\mathcal{S}$
end

STEP 3: ***Expand_RecSet***($\mathcal{O}$, $\mathcal{S}$, $\vec{p^u}$):
Input: domain ontology $\mathcal{O}$,
set of top n recommended items, $\mathcal{S}$
active user vector, $\vec{p^u}$
Output: extended set, $\mathcal{S}^+$
Algorithm:
1: Find the concept c_j to which $\vec{p^u}$ is highly similar
2: Rank all relations of c_j according to their relation hierarchy
3: **for** each relation connecting c_j with another concept c_k **do**
4: Instantiate concept c_k to generate items.
5: Add items to $\mathcal{S}$, to get $\mathcal{S}^+$
6: **end for**
7: **return** $\mathcal{S}^+$
end

A rough algorithm (Algorithm 1) illustrates these steps as separate procedures. Next, we detail each step along with a running example.

V. Clickstream Mapping

The proposed system starts with mapping the clickstream in a preprocessing step. Items and their associated tags are stored in a database. Then, Wu and Palmer similarity measure [12] is used with WordNet[6] as a thesaurus to compute the similarity between tags of each item and each concept in the ontology, giving the similarity score $sim(T_{p_i}, c_j)$. To elaborate, each tag in p_i is compared against each concept in the ontology, this is done by computing the similarity between the tag and the concept, both the tag and the concept are located as two words in WordNet (say n_1 representing concept c_j, and n_2 representing a tag g from the set T_{p_i}) and the Wu and Palmer similarity measure is computed as

$$sim(n_1, n_2) = \frac{2 \times D_3}{D_1 + D_2 + 2 \times D_3}$$

where n_3 is the Lowest Common Ancestor of n_1 and n_2, D_1 is the distance (in number of nodes on the path) between n_1 and n_3, and D_2 is the distance between n_2 and n_3, while D_3 is the distance between n_3 and the root.

Eventually, each item p_i can be represented as a vector $\vec{p_i}$ of similarity scores over the space of ontology concepts, in which each entry represents the average similarity score of all the tags associated with p_i with respect to one concept. Thus, resulting in a matrix of items×concepts, called the PC matrix, similar to Figure 2.

Definition 2: The items×concepts matrix is the matrix $PC_{I \times J} = \{sim(T_{p_i}, c_j) : I \leq |\mathbb{P}|, J \leq |\mathcal{C}|\}$, where $sim(T_{p_i}, c_j)$ is the similarity score between the set of tags T_{p_i} for item p_i and the concept c_j, $1 \leq i \leq |W|, 1 \leq j \leq |\mathcal{C}|$, where $\mathbb{P}$ is the set of all items in the database.

Assume for the running example, we have the concepts c_1 = "$camera$", c_2 = "$lens$", c_3 = "$battery$", c_4 = "$dive$", to compute the similarity between p_2, from W in section II, and c_1, we use Wu and Palmer measure with WordNet, to get the similarity score between each tag associated with p_2 (the tags 'close','quality','clear','lens') and the concept 'camera'. The result will be 0.346, 0.354, 0.206, 0.517 respectively. The average score of 0.356 is used to represent the overall similarity score $sim(T_{p_2}, c_1)$. By computing the similarity score for each item with each concept in this example, we get the matrix in Figure 2. Such matrices can get large and

$$PC = \begin{bmatrix} & c_1 & c_2 & c_3 & c_4 \\ p_1 & 0.187 & 0.256 & 0.202 & 0.113 \\ p_2 & 0.356 & 0.463 & 0.232 & 0.202 \\ p_3 & 0.156 & 0.116 & 0.098 & 0.491 \\ p_4 & 0.213 & 0.165 & 0.111 & 0.129 \\ p_5 & 0.213 & 0.173 & 0.113 & 0.403 \end{bmatrix}$$

Fig. 2. PC matrix for the running example.

sparse, for this we propose a dimensionality reduction method that utilizes the hierarchy of the ontology, as follows.

A. Using the ontology hierarchy for dimensionality reduction

Matrix sparsity is an inherent problem in recommendation systems. In the proposed model, a good number of items might be highly similar to only a subset of the concepts. Also, another reason to reduce the dimensionality is to decrease the time required to generate a recommendation. As a solution we propose a method that makes use of the *is-a* hierarchy of the ontology, in which two or more leaf concepts, represented by respective columns in the PC matrix, are combined by their LCA into one column. A basic combination formula is proposed using a weighted average, in which the distance from each leaf concept to the LCA is used as a decay factor of the similarity score.

$$sim(p_i, c_{LCA}) = \frac{\sum_j \frac{1}{d_j} sim(p_i, c_j)}{j} \quad (1)$$

[6]http://ai.stanford.edu/∼rion/swn/index.html

such that c_j is a leaf node in $\mathcal{O}$ and $sim(p_i, c_j) > 0$, where c_j is subsumed by c_{LCA} denoted as $c_j \triangleleft c_{LCA}$ and d_j is the number of edges between c_j and its ancestor c_{LCA}. The result of this formula is the similarity score between the LCA and each row in the PC matrix p_i, this formula is applied for each c_{LCA} For example, if concepts c_2 and c_3 in Figure 2 both have c_7 as their LCA, then the two columns representing these concepts in the figure, can be combined by equation (1) into one column representing c_7 and the matrix size is thus reduced by one column. If every two columns in the PC matrix are reduced using this method to one column, then the reduced matrix will have a minimum number of columns half of that of the original PC matrix.

Another way to reduce the dimensionality of the PC matrix is to use *Feature Subset Selection* (FSS), by removing columns of concepts that have the lowest similarity scores. This method is more simple but it is not guaranteed to reduce the matrix to half of its original size. In section IX, we experiment with FSS and compare results with LCA-based reduction.

VI. Active Recommendation

The recommendation process takes place online. This section describes how recommendation is produced.

Tags are provided by the active user u. Let's call the user profile, the *active page* p^u. These tags T_{p^u} are then mapped to ontology concepts, in a way similar to the offline clickstream mapping process. So, p^u can now be modeled as a vector of similarity scores in the space of concepts, $\vec{p^u} = [s_1^u \;\; s_2^u \ldots \;\; s_k^u]$, where $k = |\mathcal{C}|$ and $s_j^u = sim(T_{p^u}, c_j)$. For the adopted running example, assume $T_{p^u} = \{\text{“}camera\text{”}, \text{“}digital\text{”}, \text{“}zoom\text{”}\}$, so one can compute $\vec{p^u} = [0.400\ 0.234\ 0.266\ 0.220]$. To find the top-$n$ recommendations, $\vec{p^u}$ is matched against every row in PC to find their relatedness, using cosine similarity,

$$rel(\vec{p^u}, \vec{p_i}) = \frac{\vec{p^u} \cdot \vec{p_i}}{\left\|\vec{p^u}\right\| \left\|\vec{p_i}\right\|} \tag{2}$$

such that $1 \leq i \leq I$, I from Definition 2 is the number of rows in the PC matrix.

The results are ranked in descending order and the items with top-n relatedness scores are added to the recommendation set $\mathcal{S}$.

$$\mathcal{S} = \arg\max_{p_i \in \mathbb{P}}^{n}(rel(\vec{p^u}, \vec{p_i})) \tag{3}$$

Applying this to the example here, requires that $\vec{p^u}$ be matched with every row of the matrix in Figure 2, using equations (2) and (3). If $n = 3$, the recommendation set will be $\mathcal{S} = \{p_4 : 0.987,\ p_1 : 0.940,\ p_2 : 0.936\}$, where the score associated with each recommended item is the score from equation (2).

VII. Recommendation Expanding Using The Ontology

In addition to the top-*n* recommendations, the recommendation set is expanded by incorporating items from other concepts that are related to the ones represented in $\vec{p_u}$. A full ontology plays a better role here than a shallow ontology, in the sense that all relations in the ontology are utilized to provide a recommendation, in what is referred to as *Spreading Activation*. The process starts from the concept c_j which has the highest s_j^u similarity score in the $\vec{p^u}$ vector of similarities. Then, the recommendation is spread over all relations from this concept to other concepts, generating recommendations of items belonging to those concepts using object properties. A detailed procedure is provided in Algorithm 1. This is not possible in a shallow ontology, because only *is-a* relations are present. For example, if a user is recommended a video camera product, to expand this recommendation, the system will utilize the relations (follow with Figure 1):

$$requires(\text{“}VideoCamera\text{”}, \text{“}VideoFilm\text{”})$$
$$is-a(\text{“}VideoCamera\text{”}, \text{“}Camera\text{”})$$
$$has-a(\text{“}Camera\text{”}, \text{“}Lens\text{”})$$

So, object instances of Lens and Video Film concepts are generated and added to the recommendation set. Such relations, like *has-a* and *requires*, are not present in a shallow ontology as used in [8], [13] where only *is-a* relation is used to climb up the taxonomy to infer user interest. The recommendation expansion presented here is different from query expansion (e.g. [13]), in that all ontology relations are utilized to expand the recommendation set after active recommendation. While in query expansion, the thesaurus is used to add more related tags to the user's original query tags, then active recommendation is performed.

VIII. Related Work

Content-based WRS still rely on text mining of web pages, or assume that items are already annotated with ontology concepts [2], [4]. In Quickstep and Foxtrot [8] a pageview is represented as a *k*-dimensional feature vector of keywords, and user-based CF is used. The disadvantage of this approach is its high dimensionality and its incapability of capturing more complex relations among objects, since an external shallow ontology containing only *is-a* relations is used in user profiling.

Tags are used in the area of social networking. Zanardi *et al.* use cosine to compute user-user and tag-tag similarities to propose a social ranking formula for research paper recommendation [13], as opposed to our item-concept similarity. De Gemmis *et al.* rely purely on WordNet such that documents are mapped to synsets to identify semantic concepts behind them [4]. This is augmented with a probabilistic model for learning user profiles. User tags are treated as additional content in documents. This approach might not work for e-commerce applications where semantic relations extend beyond lingual semantics, requiring a domain ontology. On the other hand, Niwa et al. [9] propose a cluster-based algorithm for recommending web pages based on the pages users have tagged. The recommendation is straightforward and is based on the similarity of TF-IDF tag profile vectors. In Guan *et al.* [5] user provided tags are used for document recommendation. While a domain ontology is not utilized, relations between users, documents and tags are found by constructing weighted

bipartite graphs and an algorithm for subspace learning. The reason claimed is that straightforward vector space representation does not consider semantic correlations between tags. Our approach on the other hand, does consider such correlations since tags belong to ontological concepts with relations of several kinds among them in the ontology. As we propose a novel method for dimensionality reduction, Guan *et al.* learning algorithm still suffer from a scalability problem. In a different approach, Sin *et al.* first infer user's tag preferences from user's clickstream and rating information in order to find the user's preferred items for recommendation [10]. But clickstream information requires pattern discovery and rating information is usually absent in social tagging systems.

Our proposed WRS is described as content-based, and it differs from the discussed methods in that, it relies on user-provided tags only, without the need to retrieve keywords from web documents, or the need for rating information. It also relies on a provided domain ontology to which the tags are mapped, without using clustering or annotating the items with ontology concepts.

IX. Experimental Results

For experimental evaluation, we use the MovieLens dataset[7]. The proposed system (*SemAwareIN* from Algorithm 1) is built with the MO-Movie Ontology[8] as the underlying ontology, and WordNet is used for similarity computations. The MovieLens dataset contains 100,000 tags for 10,681 movies, made by 71,567 users. The tags are cleaned by removing non-words, stop words, and keywords with numbers. The number of clean tags extracted from the dataset is 82,454 tags.

The experiments are conducted using 5-fold cross-validation method to test the accuracy of the proposed system, as error metrics methodologies (like RMSE) are not a natural fit for evaluating top-n recommendation systems [1]. MovieLens dataset is divided into five mutually exclusive sets, and at each time four of these sets are used for training (constituting 80% of the original dataset) while the remaining set (20%) is used for testing. At every fold of the cross-validation, the PC matrix is built, such that the items represent movie IDs and the concepts represent leaf concepts in the MO ontology. The tags T_{p^u} from each row in the test set are used as input to the recommendation system. They are mapped to the ontology concepts and the vector $\vec{p^u}$ is created as described in section VI. Then, the recommendation set $\mathcal{S}$ is generated using equation (3) with n=3, and each recommended movie is matched against the movie in the current test row from the test set. If the movie title is the same or the movies are of the same ontology concept, then that is considered a *hit*. Otherwise, it is a *miss*. *Accuracy* is measured at each fold of cross-validation, as the percentage of hits from the total number of trials made on the test set. The average accuracy is finally computed over all of the five folds, and is found to be 82%, which is very good. To confirm this we compare SemAwareIN with non-ontology-based top-of-the-art item-based recommendation algorithms, namely TopPop (which recommends top-n items with highest popularity) and NNCosNgbr (which uses kNN clustering with item-to-item similarity to recommend top-n items), for details of these two algorithms see [1]. Comparisons are made in terms of *recall-at-n*, which is computed as the ratio of hits to the size of the test set at different values of n, and *precision*, which is computed by dividing recall-at-n by n. Figure 3 shows the recall-at-n comparisons for values of n between 5 and 20, which tell that SemAwareIN exceeds TopPop and NNCosNgbr in terms of recall as it reaches, at n=10, a recall of 0.61, compared to 0.28 and 0.45 for TopPop and NNCosNgbr, respectively. This means that about 61% of the top-10 recommended movies are hits (i.e., are the same as or sematically match the expected result). Figure 4 confirms that the proposed algorithm outperforms

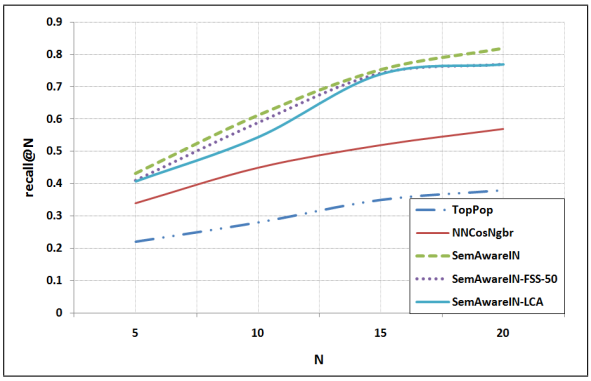

Fig. 3. Recall-at-n comparison.

the two popular algorithms TopPop and NNCosNgbr in terms of precision metrics. Each line in the figure represents the precision of the algorithm at a given recall.

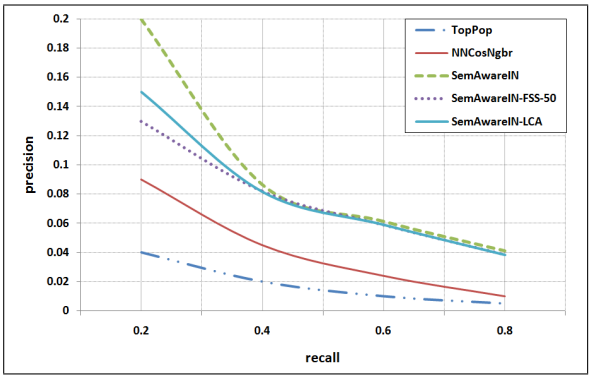

Fig. 4. Precision vs. recall.

To test the effect of dimensionality reduction on solving the sparsity problem of the PC matrix, 5-fold cross-validation is again performed, once using Feature Subset Selection (FSS) and another time using the proposed LCA-based reduction method. In FSS 50% of the concepts are removed, those are the 50% columns in the PC matrix that hold the lowest

[7]available at http://www.grouplens.org/node/73
[8](http://www.movieontology.org

average similarity scores. While in LCA-based, leaf concepts are joined by their LCA as discussed in section V-A, reducing the size of the matrix to its half. The FSS variation is called *SemAwareIN-FSS-50* and the LCA-based variation is called *SemAwareIN-LCA*. Figure 3 shows that FSS and LCA-based reductions (in SemAwareIN-FSS-50 and SemAwareIN-LCA, respectively) do not compromise the accuracy and recall of the proposed algorithm, as only 1%-6% drop in accuracyis observed, which is logical in SemAwareIN-FSS-50 since only low-scoring concepts are removed, and is also encouraging in SemAwareIN-LCA, because it shows that LCA-based reduction that depends on the ontology provides a comparable performance to an algorithm with no dimensionality reduction.

To test the performance scalability of SemAwareIN we conducted cross-validation on datasets of different sizes (extracted from the big MovieLens dataset), as shown in Table I. The

TABLE I
DATASETS EXTRACTED FROM ORIGINAL MOVIELENS DATASET.

Dataset Size	num of Movies	num of Tags	Avg. Rec. time (in seconds)
Small	4,739	11,948	0.973
Medium	7,308	23,895	1.117
Large	9,102	47,790	1.293

TABLE II
ACCURACY OF PROPOSED ALGORITHMS WITH DIFFERENT DATASET SIZES.

Dataset Size	Accuracy (in %)	
	SemAwareIN	SemAwareIN-LCA
Small	77.00	75.61
Medium	73.00	71.00
Large	70.52	70.30

table shows that the recommendation time scales very well with increasing number of tags used in training. A careful investigation of the numbers will show a linear relation. That is a good indicator, in both cases, with and without using LCA-based reduction. It is worth mentioning here that the average recommendation time using SemAwareIN-LCA is reduced by 22%, because the size of the *PC* matrix is reduced and so is the number of comparison operations. On the other hand, Table II shows the average accuracy versus the dataset size which is consistent with our analysis of Figure 3, in that SemAwareIN-LCA does not compromise accuracy. The ability of the proposed system to expand the recommendation set based on ontology axioms is tested with n=5. This also shows the contribution of reasoning in the final results. We did not adopt any reasoning language, but rather implemented the procedure from Algorithm 1 using C# and OWL libraries with a polynomial complexity. In this case, *SemAwareIN-Ex* is implemented as a variation of SemAwareIN in which the recommendation set of the top-5 items is expanded using spreading activation (as described in section VII) adding 10 more items, resulting in a set of 15 items. Recall of SemAwareIN-Ex is compared with the recall of SemAwareIN at n=15, and found to be 0.862, that is far better than that of SemAwareIN (which is 0.753 at n=15).

X. CONCLUSIONS AND FUTURE WORK

A content-based web recommendation system is proposed based on a domain ontology. It relies on user-provided tags, that are mapped to concepts of this ontology. Similarity measures are used during mapping and a matrix of items×concepts is built offline, which is used later for online top-n recommendation. This system outperforms popular algorithms like TopPop and NNCosNgbr. In Addition, a proposed novel dimensionality reduction solves the sparsity problem, and does not compromise the accuracy of the proposed system. We also show how the recommendation set is expanded using Spreading Activation over the ontology, taking into consideration the several available relations, which raises the accuracy of the proposed model.

There are two benefits in using an ontology over clustering of the tags. First, it saves the costly step of clustering, and second, a full ontology has a far better reasoning power than a topic taxonomy. In a full ontology there are several semantic relations that can be taken into consideration (as opposed to only *is-a* relation in a topic taxonomy) to provide better relatedness measures, and better interpretability. In addition, a similarity measure can be formulated that uses relation hierarchy for recommending only highly similar concepts. This is still an area of research left for future work.

REFERENCES

[1] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *RecSys '10: Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46, New York, NY, USA, 2010. ACM.

[2] H. Dai and B. Mobasher. A road map to more effective web personalization: Integrating domain knowledge with web usage mining. In *Proceedings of the International Conference on Internet Computing (IC'03)*, June 2003.

[3] J. Diederich and T. Iofciu. Finding communities of practice from user profiles based on folksonomies. In E. Tomadaki and P. J. Scott, editors, *EC-TEL Workshops*, volume 213 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

[4] M. de Gemmis, P. Lops, G. Semeraro, and P. Basile. Integrating tags in a semantic content-based recommender. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 163–170, New York, NY, USA, 2008. ACM.

[5] Z. Guan, C. Wang, J. Bu, C. Chen, K. Yang, D. Cai, and X. He. Document recommendation in social tagging services. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 391–400, New York, NY, USA, 2010. ACM.

[6] N. R. Mabroukeh and C. I. Ezeife. Using domain ontology for semantic web usage mining and next page prediction. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1677–1680, New York, NY, USA, 2009. ACM.

[7] N. R. Mabroukeh and C. I. Ezeife. A taxonomy of sequential pattern mining algorithms. *ACM Computing Surveys*, 43:3:1–3:41, December 2010.

[8] S. E. Middleton, D. D. Roure, and N. R. Shadbolt. Ontology-based recommender systems. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, International Handbooks Information System, pages 779–796. Springer Berlin Heidelberg, 2009.

[9] S. Niwa, T. Doi, and S. Honiden. Web page recommender system based on folksonomy mining for itng '06 submissions. In *ITNG*, pages 388–393. IEEE Computer Society, 2006.

[10] S. Sen, J. Vig, and J. Riedl. Tagommenders: connecting users to items through tags. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 671–680, New York, NY, USA, 2009. ACM.

[11] G. Stumme, A. Hotho, and B. Berendt. Semantic web mining: State of the art and future directions. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 4(2):124–143, 2006.

[12] Z. Wu and M. S. Palmer. Verb semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics, 1994.

[13] V. Zanardi and L. Capra. Social ranking: uncovering relevant content using tag-based recommender systems. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 51–58, New York, NY, USA, 2008. ACM.