View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by University of Brighton Research Portal

A Research Paper Recommender System Using a Dynamic Normalized Tree of Concepts Model for User Modelling

Modhi Al Alshaikh, Gulden Uchyigit, Roger Evans
School of Computing, Engineering and Mathematics
University of Brighton
Brighton, United Kingdom
{m.alalshaikh, g.uchyigit, r.p.evans}@brighton.ac.uk

***Abstract*— The enormous growth of information on the Internet makes finding information challenging and time consuming. Recommender systems provide a solution to this problem by automatically capturing user interests and recommending related information the user may also find interesting. In this paper, we present a novel recommender system for the research paper domain using a Dynamic Normalized Tree of Concepts (DNTC) model. Our system improves existing vector and tree of concepts models to be adaptable with a complex ontology and a large number of papers. The proposed system uses the 2012 version of the ACM Computing Classification System (CCS) ontology. This ontology has a much deeper structure than previous versions, which makes it challenging for previous ontology-based approaches to recommender systems. We performed offline evaluations using papers provided by ACM digital library for classifier training, and papers provided by CiteSeerX digital library for measuring the performance of the proposed DNTC model. Our evaluation results show that the novel DNTC model significantly outperforms the other two models: non-normalized tree of concepts and the vector of concepts models. Further, our DNTC model provides high average precision and reliable results when used in a context which the user has multiple interests and reads a large quantity of papers over time.**

Keywords*— *normalized tree of concepts; recommander system; personalization; user profile; 2012 ACM CCS ontology.

I. INTRODUCTION

The enormous growth of information on the Internet makes finding information both challenging and time consuming. Traditional search engines require the user to manually enter keywords in order to search for relevant web pages or data collections. The results of the search query are displayed to the user based on the order of relevance to the keywords. One of the problems with traditional keyword based search engines is that the user may find it difficult to find the search keywords which will return the best results, especially if the user is searching for information in a new domain. Recommender systems provide a solution to this problem by automatically capturing user interests/preferences and recommending related information the user may also find interesting. There are two ways in which recommender systems are able to capture user preferences: *explicitly*, by enabling the user to enter their preferences, or *implicitly,* by monitoring the user's activities such as browsing the web or reading documents. Collected preferences are stored in a user profile. New items (e.g. documents) are then compared with the user profile and those items which are sufficiently similar are recommended to the user. Existing recommender systems offer efficient personalized services in variety of domains such as movies, music, television, books, documents, e-learning and e-commerce [1].

One of the interesting systems in the document domain is a research paper recommender system. Current research paper recommender systems suffer from a number of limitations that may constrain their recommendation services. One critical limitation in these systems is that they are not compatible with the new advanced ontologies, that have become bigger, more complex and with deeper levels. For example, the 2012 ACM Computing Classification System (CCS) [2] relies on a semantic vocabulary as the source of categories and concepts that reflect the state of the art in the computing discipline. It replaces the previous 1998 version of the ACM CCS (the '98 ACM CSS'), which has served as the de facto standard classification system for the computing field, and has been used by several recent recommender systems (e.g. [3], [8], [9]). The 98 ACM CCS ontology has a three-level hierarchical set of concepts that contains in total 369 concepts [8]. However, to reflect the rapidly developing field of computing research the 98 ACM CCS ontology was updated to the 2012 ACM CCS to include the new deeper level concepts. The 2012 ACM CCS ontology has a poly-hierarchical ontology and maintains a six-level hierarchical tree with more than one thousand concepts [2].

While ontologies are growing bigger and more complex, finding relevant papers related to users' interests becomes a challenging task for the recommender systems. Often dynamic recommender systems use an ontology to create the user's profile as vector of concepts [4]. However, representing the user profile as a vector of concepts assumes that the concepts are independent from each other and does not accurately represent the user's interests. With a complex ontology, there is a need to employ more sophisticated techniques to build a user

978-1-5090-5476-3/17/$31.00 ©2017 IEEE

profile. The tree of concepts model [3] used in conjunction with complex ontologies addresses this problem, but it is static, in that it is unable to dynamically capture new and multiple user's interests. Furthermore, it does not normalize the user model according to the number of papers that are involved in the user profile, which causes its performance to decline significantly if the profile contains larger numbers of papers.

In this paper, we propose a content based recommender system for research papers which addresses these problems. In the proposed system, a user profile is built as a Dynamic Normalized Tree of Concepts (DNTC) by monitoring the user's reading behavior over time. In our DNTC user modelling approach, the parent-child relationships between the concepts from the ontology are maintained whilst computing the similarity between a user profile and the new research papers to be recommended. The DNTC user profile is constructed using the 2012 ACM CCS as a reference ontology. In our offline evaluations we compare the DNTC system with two models: dynamic vector of concepts (DVC) model and non-normalized tree of concepts (NNT) model. We show that our model's performance is equal to or better than previous systems across a range of usage scenarios, and in particular that it is significantly better for the more demanding scenarios (more concepts, more papers) that we are using in our current work on modelling short and long term preferences in recommender systems.

The rest of this paper is organized as follows. Section II presents the related work. Section III discusses our DNTC system. Section IV shows offline evaluation results. Finally, conclusions and future work are given in Section V.

II. Related Work

The accurate representation of user interests and preferences in the form of a user profile is key to the effectiveness of recommender systems [4]. A common profile representation technique is to use weighted feature vectors. We focus on user profiling for document recommendation such as research papers, news and web pages. In this domain, the features can be in the form of keywords automatically extracted from text documents which the user has implicitly or explicitly shown a preference towards. In such representation techniques the keywords are associated with weights to represent the significance of the keyword in the user profile. Lee et al. [5] developed a system that extracts keywords from the user's previously published papers, assuming that the user will be interested in similar papers to their previous research topics. Zeb and Fasli [17] proposed a technique that constructs probabilistic user profiles by subscribing to an RSS (Rich Site Summary) news aggregator. The probabilistic user model is constructed based on implicit user feedback (click response) over a period of time.

A major shortcoming of keyword based user profile representation techniques is that they are not suitable for representation of complex user profiles [4]. This is because representing user interests as simple keywords increases the ambiguity as it lacks semantic information. One way of semantically enriching the user profile representation is through the use of abstract concepts drawn from an ontology instead of words. By mapping between words and concepts in a reference ontology it is possible to build more robust user profiles with reduced user feedback and monitoring. Examples of ontologies that are used in recommender systems include the Open Directory Project (ODP)[1] and the ACM CCS[2]. Such approaches have been shown to provide a significant improvement in the performance of user profiling models in recommender systems [4]. Gauch et al. [4] noted that most of researchers who used ontologies for user profile representation use them in a similar way to weighted keywords in that the concepts are represented as vectors of weighted features, but the features represent concepts rather than words. For example, Agarwal and Singhal [6] employed OWL (Web Ontology Language)[3] to build the user profile. Their system periodically gathers visited news pages by using unique features of RSS feed news items and arranges them in chronological order. The user profile consists of concepts that are interesting to the user. Concepts are given weights based on number of clicks in a session, recency of the session and active session duration. Tang and Zeng [7] used an ontology that is defined by the Sciencepaper Online[4]. Fig.1 shows the concepts of the subject "computer science" in this ontology. The user profile model in [7] computes weights of concepts for the papers that are read by the user and represents them as vectors of concepts. Kodakateri et al. [8] designed a recommender system that recommends potential research papers of interest to users from the CiteSeer database. The 98 ACM CCS is used as a reference ontology. They developed a dynamic user profile that is updated each time the user visits a new research paper.

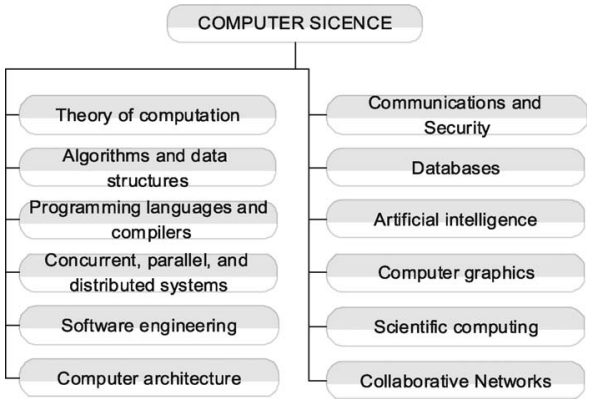

Fig. 1. The classification of "computer science" in the Sciencepaper ontology [7].

[1] http://www.dmoz.org/docs/en/about.html
[2] http://www.acm.org/about/class/
[3] https://www.w3.org/TR/owl-ref/
[4] http://www.paper.edu.cn/en

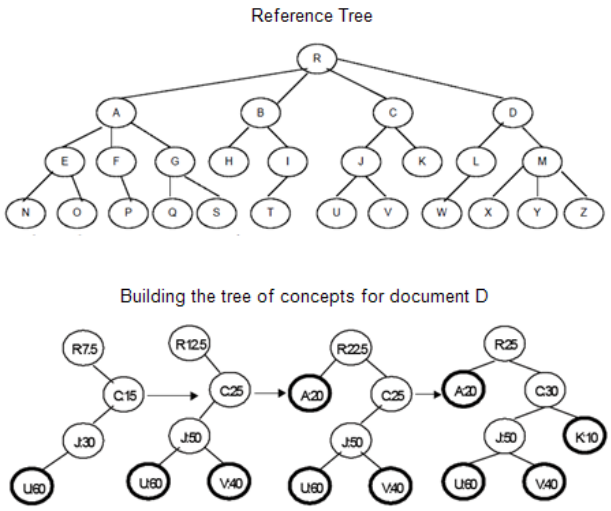

Fig. 2. Tree of Concepts Technique [9].

The concept vectors technique may be sufficient with a simple ontology that consists of two levels of classification, such as primary and secondary subjects as shown in Fig.1. However, with complex ontologies such as ACM CCS ontology that maintains multiple level hierarchies, there is a need to employ more sophisticated techniques to build a user profile. An interesting technique is developed in [3] which represents a user profile as a tree of concepts. Their recommender system is for the research paper domain using the 98 ACM CCS ontology. A user profile is created based on the user's previously published papers. The tree of concepts is created as follows. A paper is submitted to a classifier to determine the list of top concepts with the highest weights. For example, a document D (in Fig.2) have the following concept vector (conceptID, weight):

D = { (U, 60), (V, 40), (A, 20), (K, 10) }

Then, the concept vector is input to the Tree Builder Module [9] to create a (weighted) tree of concepts for the document D based on a reference tree as in Fig. 2. The output tree is a subtree of the reference tree spanning all the concepts in the concept vector. Weights are assigned to leaf nodes from the input vector, and then percolated upwards, reducing by 50% on each step (see Fig. 2). The user profile is constructed by combining the trees of concepts from the user's publications.

The concept vector technique assumes that the elements of the vectors being compared are independent, which is not an accurate representation of the user's preferences [3]. In order to exploit the relationships between the concepts it is more efficient to use the tree of concepts technique, because it can exploit inter-relationships between the concepts through the ontology [3]. However, their user profiling model using the tree of concepts technique is static over time, whereas user preferences and needs are not static but they usually change over time. Moreover, this user profiling technique does not normalize the concept weights. Without normalization, the weights in the user's tree of concepts profile representation are too big to compare accurately with the weights in a tree of concepts for a paper in the recommendation phase. To overcome these problems, we have developed a Dynamic Normalized Tree of Concepts (DNTC) model for user profiles, which we introduce in the next section.

III. OUR SYSTEM

The DNTC recommender system consists of three main phases: papers classification phase, DNTC user profiling phase and recommendation phase. The first phase is responsible for preparing papers and classifying them. The second phase is responsible for tracking user reading activities for papers, and using the papers read by the user to build a user profile as a dynamic normalized tree of concepts. The third phase is recommendation phase that uses a dynamic tree edit distance technique to recommend a set of papers to the user that match his/her preferences. Fig. 3 shows an overview diagram for the proposed system. The next subsection discusses our ontology model. After that, our system's phases will be explained in detail in subsections III.B, III.C and III.D.

A. Ontology in our system

In our system, a reference ontology is used for three main purposes:

1) Mapping a paper to the correct concepts using a classification algorithm.
2) Representing a paper profile as a tree of concepts.
3) Representing a user profile as a normalized tree of concepts.

We use the ontology for the 2012 ACM CCS. It relies on a semantic vocabulary as the source of categories and concepts that reflect the state of the art of the computing discipline and is receptive to structural change as it evolves in the future. The usage of the 2012 ACM CCS in our classification phase is explained in the next section.

B. Research papers classification phase

In the first phase in our system, we build a classifier using the ACM training dataset and classify a set of research papers from the CiteSeerX [18] dataset (see section IV for more information about these datasets) to the reference ontology. All the papers in the dataset are mapped to the reference ontology by classifying each paper to the correct concepts using the Term Frequency-Inverse Document frequency (TF-IDF) technique [16] and cosine similarity [13]. The classification process consists of two phases:

1) Training phase: During this phase, papers in the ACM training set, which are pre-assigned to one or more concepts in the reference ontology, are used to learn a vector of features for each concept in the ontology.

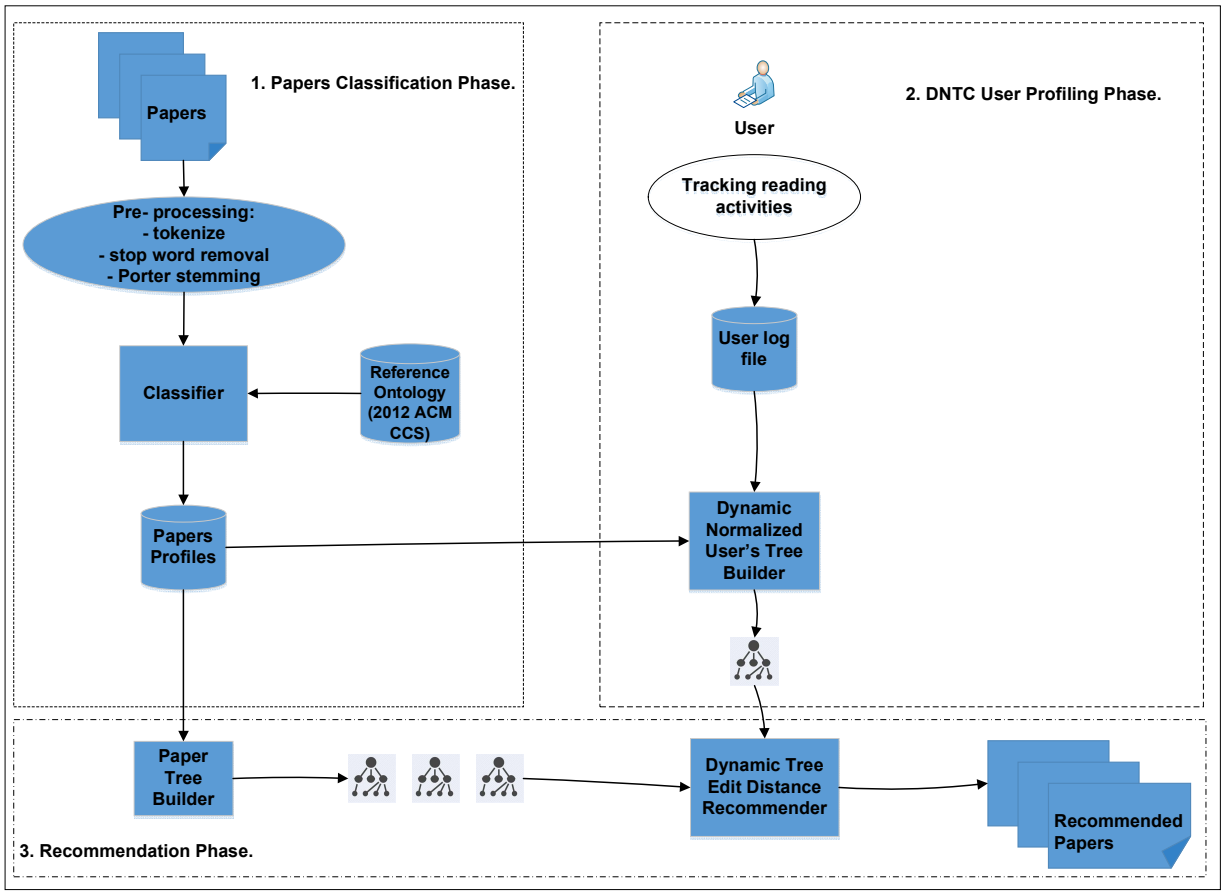

Fig. 3. The DNTC system architecture.

2) Classification phase: In this phase the cosine similarity classifier uses the vectors learnt in the training phase to classify papers in the CiteSeerX dataset. The output is a list of concepts for each input paper along with their corresponding weights which indicate the degree of association between the concept and the paper. The top N concepts for each paper are retained and stored in the research paper profile.

B.1. Training phase

The training set was provided by ACM[1]. The training set contains papers which are pre-assigned to one or more concepts in the 2012 ACM CCS ontology manually by the authors of the papers. Hence, the concepts in the ontology reference are associated with training papers that represents each concept. We combine all the papers to one document (d_j) to represent a concept (c_j). Each document is tokenized and represented as a set of terms constructed from the papers' title, keywords and abstract. We applied some heuristics functions to pre-process the text, these functions are stop words removal and then Porter stemming algorithm [10] which reduces each word (term) to its shortest stem. The documents are then represented as weighted feature vectors by using the TF-IDF weighting algorithm. The TF-IDF is used to determine the importance of a word in a document within a collection or corpus (the corpus in our system is the training set). The importance increases proportionally to the number of times a term appears in a document but is offset by the frequency of the term in the corpus. The TF-IDF is calculated as follows:

$$TF\text{-}IDF\ (t_{ij}) = TF(t_{ij}) * IDF_i \qquad (1)$$

where $TF(t_{ij})$ is Term Frequency that measures how frequently a term t_i occurs in a document d_j. Since the documents are different in length, it is possible that a term would appear more times in longer documents than shorter ones. Thus, the term frequency is normalized using the document length:

$$TF\left(t_{ij}\right) = \frac{\text{Number of times term } t_i \text{ appears in a document } d_j}{\text{Total number of terms in a document } d_j} \qquad (2)$$

The IDF_i is Inverse Document Frequency which measures the importance of a term t_i across all documents in the training set:

$$IDF(t_i) = \log\left(\frac{\text{Total number of documents in the training set}}{\text{Number of documents with term } t_i \text{ in the training set}}\right) \qquad (3)$$

The TF-IDF weighted terms are calculated between 0 and 1 for each document in the training set. Therefore, all the concepts in the reference ontology are associated with training documents that have TF-IDF weighted terms, which can be used to measure a vector similarity between a concept represented by the document and a paper that we want to classify.

[1] https://www.acm.org/

B.2. Classification phase

In this phase, papers from the CiteSeerX dataset are classified to create a database of paper profiles for the recommender system to make recommendations from. The cosine similarity method is used to assign an input paper to appropriate concepts in the reference ontology. In our system, the cosine similarity algorithm [13] is applied to classify an input paper to the correct concepts:

$$SW_j = CosinSim\ (d_j, P) = \frac{\sum_{i=1}^{n}(w_{ij} * w_{iP})}{\sqrt{\sum_{i=1}^{n} w_{ij}^2} * \sqrt{\sum_{i=1}^{n} w_{iP}^2}} \quad (4)$$

where d_j is a document that represents a concept c_j in the reference ontology, P is an input paper, w_{ij} is the TF-IDF weight for term t_i in d_j and w_{iP} is the TF-IDF weight for term t_i in P. The cosine similarity is computed between all concepts' documents and paper P. The output from the classification phase is a profile for representing the research papers, composed of a decreasing ordered list of concepts' IDs along with their cosine similarity (c_j, SW_j) for each input paper P in the dataset. The Cosine Similarity (SW_j) is the degree of association between a paper P and a concept c_j. The resulting profile of papers is stored in a database which is used to build the tree of concepts model for the users and the papers.

C. DNTC user profiling phase

The main goal of the user profiling phase is to build the user profile as a dynamic normalized tree of concepts. Building a user profile as a tree of concepts maintains parent-child relationships between the concepts in the ontology. These relationships can be useful while computing the similarity between a user profile and a paper profile. Normalizing the user's tree of concepts by the number of papers read by the user provides a more accurate comparison between a paper profile and the user profile (which generally involves more than one paper).

All papers read by a user are stored in a user's log file as paper ID associated with a time sequence of the paper's reading order. Hence, the user profile is dynamically updated each time the user reads a new interesting paper (we assume if the user reads a paper, then this is a paper of interest to the user). We added this new feature (time sequence of the paper's reading order) to make the proposed tree profiling model dynamic and changeable because user preferences and interests change over time.

For each paper that is read by the user, the top N related concepts and their corresponding cosine similarity weights are retrieved from the paper's profile, which results from the classification phase. In order to exploit the relationships between concepts in a hierarchical concept ontology, a user tree of 2012 ACM CCS ontology is initiated with zero weights for all concepts. Then, the user tree is updated each time a new paper is read by the user as follows. For every new paper, the top N concepts and their corresponding Cosine Similarity (SW) weights are used to update the existing user tree. First, the SW weights for the top N concepts are updated by adding the new SW weights to old weights values in the user tree. Then, new weight values recursively propagate to the parent nodes until the root node is reached. We assign weights to parents according to the following equation:

$$SW_{Parent} = \propto \times SW_{Child} \quad (5)$$

Where SW_{Parent} is the weight of the parent, SW_{Child} is the weight of the child and α is the weight propagation factor. α is

```
Build Dynamic Normalized User Tree (UserID, UserTree, PapersProfiles, CurrentTime, Alpha, TopN)
{
    CurrentNumberOfUserPapers =0;
    Foreach Paper Pi in user's log file in CurrentTime do
        {
           CurrentNumberOfUserPapers = CurrentNumberOfUserPapers + 1;
           Get the TopN concepts and their corresponding weights from Paper Pi Profile;
           Foreach concept cj in the TopN concepts do
             {
                Find the concept cj in the UserTree;
                Update the concept cj weight: SWj += Pi_SWj;
                If the concept cj is not root do
                {
                       currentConcpet = cj ;
                       CurrentConcept_SW = SWj;
                       Loop until UserTree's root reached
                       {
                         Get currentConcpet.Parent;
                         Update currentConcpet.parent weight: SWP += CurrentConcept_SW * Alpha;
                         currentConcpet = currentConcpet.Parent;
                         CurrentConcept_SW = SWP;
                       }
                }
           }
        }

  //Divide all the concepts' weights by the current total number of user's reading papers.
  Foreach concept cj in UserTree do
    {
        Divide the concept cj weight: SWj = SWj / CurrentNumberOfUserPapers;
    }
}
```

Fig. 4. DNTC user profiling algorithm.

used to maintain the parent-child relationships between the concepts in the user's tree and its value varies between 0 and 1. If α is given the value zero, then the parents will not be assigned any part of the child's weight and there will be no actual tree structure in the user profile, which means a user profile is created as a vector of concepts without any parent-child relationships in a tree structure. Otherwise, if α is given non zero value ($0 < \alpha < 1$), then a user profile will be created as tree of concepts. α is used to determine how much of a child's weight is propagated to its parent. The value of α will be discussed in section IV.B.

Finally, all concept weights are divided by the total number of papers that are read by the user in order to normalize the concept weights. Without normalizing the user's tree of concepts, the concept weights are too large in comparison to the weights in a tree of concepts for a single paper in the recommendation phase. Fig. 4 presents our DNTC user profiling algorithm. The output of the DNTC user profiling phase is a normalized tree of concepts and their corresponding weights. This tree contains all the concepts in the reference ontology. It implicitly encodes a subtree of the sort described in section II above, by eliminating concept nodes with zero weight. However, retaining these nodes in the tree simplifies the Tree Edit Distance algorithm we use below. This dynamic normalized tree is used in the recommendation phase in next section.

D. Dynamic recommendation phase

In this phase, the trees of concepts for all papers that the user has not read (unread papers) are created. Then, the user profile, represented as a dynamic tree of concepts, is compared with the unread papers' trees of concepts to recommend the most relevant papers to the user's interests. The details are as follows.

The outputs from the papers classification phase and the DNTC user profiling phase are used as inputs to this phase. These inputs are: the papers' profiles and the user's DNTC profile. First, a tree of concepts is built for each unread paper in our dataset collection. A tree of concepts for an unread paper is built based on the top *N* concepts and their weights from the paper's profile, stored in the database which resulted from the papers classification phase. The process for building the tree of concepts for a paper is as described above: a tree of 2012 ACM CCS ontology is initiated with zero weights for all concepts, the top N concepts and weights for this paper are retrieved from the profile database, and the weight values are propagated recursively to the parent nodes according to the equation (5). Once the user profile and the papers profiles are represented as trees of concepts, Tree Edit Distance [9] is used to calculate the distance between two trees (the user's tree and a tree of concepts for an unread paper). This distance is the cost of transforming one tree into another with the minimum number of operations. There are three types of operation: insertion, deletion and substitution. Insertion operation is the cost of inserting a new concept into the tree with a given weight. Deletion operation is the cost of deleting an existing concept with a given weight from the tree. Substitution operation is the cost of changing a concept's weight to another weight.

In our 2012 ACM CCS trees we suppose that the concept with zero weight is non existing node. Hence, the cost of deletion or insertion of a concept is equal to the weight associated with the concept. Whereas the substitution cost is the difference between weights of an existing concept in both trees. Thus, the cost of modifying a tree of concepts for a paper to match the user tree is calculated. The two most similar trees are those which have the lower total cost of transformations between them. After calculating the total cost between all trees of concepts for the papers and a user tree, the total cost with its associated id of the paper (PaperID) are stored a list and sorted in increasing order. Hence, the closest papers to user's preferences appear first and the most distant papers appear last. The final output of the recommendation phase is a list of ordered recommended papers. In our system the Tree Edit Distance technique runs dynamically every time the user reads a new paper from the system dataset collection. Fig. 5 presents the algorithm for our Dynamic Tree Edit Distance technique.

```
Dynamic Tree Edit Distance (UserTree, UnreadPapersTrees, CurrentTime)
{
    //m is the number of unread papers in CurrentTime.
    Create an array (ECosts [m]) to save the edit distance costs for each paper;
    Foreach UnreadPaperTree PTi do
        {
           W1=0, W2=0;
           Foreach concept cj in UserTree do
           {
                Get the concept cj weight in UserTree SWUj;
                Find the concept cj in UnreadPaperTree PTi and its weight SWPTij;
                W1 = SWUj;
               W2 = SWPTij;
               Absolute = |W1-W2|;
               Ecost [PTi] += Absolute;
           }
         }

    Sort the array Ecosts [m] in increasing order;
  }
```

Fig. 5. Dynamic Tree Edit Distance technique algorithm.

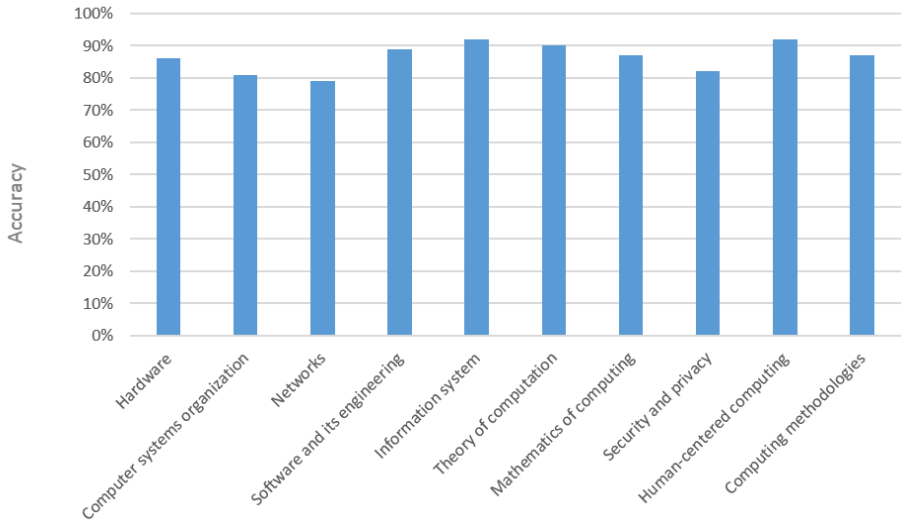

Fig. 6. Accuracy for papers classification phase.

IV. EVALUATION

In order to measure the performance of the proposed system, we evaluate:

1) The accuracy of the classifier model.
2) The performance of our DNTC user profiling and recommendation method.

For these purposes we introduce two evaluation experiments. The first experiment aims to evaluate the classification performance for mapping papers in a dataset. The second experiment evaluates the performance of our DNTC recommendation method. We conduct our evaluations using ACM and CiteSeerX datasets. ACM dataset contains 16,307 mapped papers for 2012 ACM CCS ontology, and is used as the training set. CiteSeerX is a search engine and digital repository of scientific and research papers. It is a collection of over 5 million papers primarily in the field of computer and information science. We used 100,000 papers as a subset of that collection. This subset of CiteSeerX's papers are entered to our classifier to classify them according to 2012 ACM CCS ontology and we then use them as our dataset to evaluate the performance of our DNTC recommender system.

A. Evaluation of the classification phase

ACM provided us with a dataset that contains mapped papers for 2012 ACM CCS ontology. The main categories in 2012 ACM CCS that are evaluated are: Hardware, Computer systems organization, Networks, Software and its engineering, Information system, Theory of computation, Mathematics of computing, Security and privacy, Human-centered computing, Computing methodologies. The total number of the concepts under these categories is 1,329 concepts, and the number of leaf concepts is 986 concepts. The papers are mapped by the authors of the papers. The authors of the papers are allowed to assign their papers to more than one leaf concept. To evaluate the accuracy of our classifier, 50% of ACM dataset is used as training set and the other 50% as the test set. The papers from the training set used to learn a concept (c_j) are all combined into one training document file (d_j). All terms in this file are converted to vectors with their weights using the TF-IDF as explained in section III.B.1.

Following this training phase, papers in the test set are classified as explained in section III.B.2. The output for each paper is stored as the paper's profile. If the highest weighted concept resulted from the classifier is one of the concepts that are chosen by the paper's authors, then we consider it as positively classified. We evaluate the performance using the following equation:

$$\text{Accuracy} = \frac{\text{Positive classified papers}}{\text{All papers}} \quad (6)$$

Fig. 6 shows the accuracy results for our classifier for the main categories in ACM CCS 2012. The accuracy results may depend on distribution of concepts in the training set. For example, the concepts with significant high accuracy result (92%) under Information systems and Human-centered computing may have good representation among the training papers. Whereas the concepts with low accuracy results, such as (79%) under Networks, may have poor representation among the training papers. The average of the classification results in accuracy for all categories is 87%.

After evaluating the accuracy of our classifier, we retrained the classifier using all papers in ACM dataset as training set. We used this classifier to classify the CiteSeerX papers to create the paper profile database which serves as our dataset in all the subsequent experiments.

B. Evaluating the performance of the DNTC recommender system

The evaluation process of recommender system algorithms is known to be difficult and expensive as these systems are typically complex and have many components, properties and parameters which have to be examined in order to provide the optimum performance [14]. To establish a preliminary

indication of performance, offline evaluations are attractive because they require no interaction with real users, and thus the measuring of performance is allowed at a low cost in terms of time and effort [15]. Therefore, offline evaluation methodology is used for our evaluation to measure the performance of the proposed DNTC recommender system. We opted to use a user behaviour simulation approach to test specific scenarios for multiple concepts and variant range of papers quantity. We have to create user scenarios that simulate users' interests and preferences. We created 9 main templates for user scenarios to simulate different numbers of concepts that represent multiple user interests. We have 3 main types of template scenarios that consider different number of concepts during user's reading: three concepts, four concepts and five concepts as follow:

- Users' template scenarios 1, 2 and 3 consider three concepts (as shown below):

Scenario 1: small quantity of papers (15)

Concepts	Number of papers	Time sequence for user reading
Concept 1	5	1-5
Concept 2	4	6-9
Concept 3	6	10-15

Scenario 2: medium quantity of papers (30)

Concepts	Number of papers	Time sequence for user reading
Concept 1	10	1-10
Concept 2	11	11-21
Concept 3	9	22-30

Scenario 3: large quantity of papers (50)

Concepts	Number of papers	Time sequence for user reading
Concept 1	15	1-15
Concept 2	18	16-33
Concept 3	17	34-50

- Users' template scenarios 4, 5 and 6 consider four concepts (as shown below):

Scenario 4: small quantity of papers (15)

Concepts	Number of papers	Time sequence for user reading
Concept 1	3	1-3
Concept 2	4	4-7
Concept 3	3	8-10
Concept 4	5	11-15

Scenario 5: medium quantity of papers (30)

Concepts	Number of papers	Time sequence for user reading
Concept 1	6	1-6
Concept 2	9	7-15
Concept 3	7	16-22
Concept 4	8	23-30

Scenario 6: large quantity of papers (50)

Concepts	Number of papers	Time sequence for user reading
Concept 1	14	1-14
Concept 2	11	15-25
Concept 3	13	26-38
Concept 4	12	39-50

- Users' template scenarios 7, 8, and 9 consider five concepts (as shown below):

Scenario 7: small quantity of papers (15)

Concepts	Number of papers	Time sequence for user reading
Concept 1	3	1-3
Concept 2	4	4-7
Concept 3	3	8-10
Concept 4	2	11-12
Concept 5	3	13-15

Scenario 8: medium quantity of papers (30)

Concepts	Number of papers	Time sequence for user reading
Concept 1	5	1-5
Concept 2	7	6-12
Concept 3	6	13-18
Concept 4	4	19-22
Concept 5	8	23-30

Scenario 9: large quantity of papers (50)

Concepts	Number of papers	Time sequence for user reading
Concept 1	9	1-9
Concept 2	11	10-20
Concept 3	10	21-30
Concept 4	12	31-42
Concept 5	8	43-50

Each type has three different scenarios that involve different quantity of papers during user's reading. There are small quantity (15 papers – scenarios 1, 4 and 7), medium quantity (30 papers – scenarios 2, 5 and 8) and large quantity (50 papers, scenarios 3, 6 and 9).

Each scenario template is applied on the ten main categories in ACM CCS 2012: Hardware, Computer systems organization, Networks, Software and its engineering, Information system, Theory of computation, Mathematics of computing, Security and privacy, Human-centered computing, Computing methodologies. Hence, we have 90 virtual users (i.e. 9 templates*10 main categories). The concepts are selected randomly from the main categories in ACM CCS 2012 to create each individual user's scenarios using the templates. The papers for each concept are chosen randomly from our classified CiteseerX dataset that resulted from the classification phase.

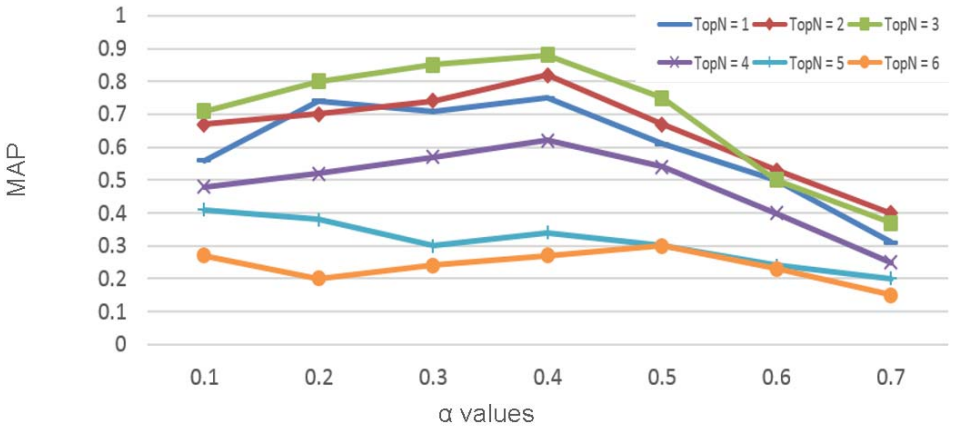

Fig. 7.The MAP results using different α and *TopN* values.

1) *Evaluating α and TopN parameters*

In this section we evaluated different values of α (the propagation factor) and *TopN* (the number of the top related concepts for a paper) parameters to find the optimal values that provide the best overall performance for our recommender system. The measurement that is used for evaluation is Mean Average Precision (MAP). The MAP for all users is the average of the average precision of each user [11]:

$$MAP = \frac{\sum_{i=1}^{M} AVG\ P}{M} \quad (7)$$

We calculated the average precision (AVG P) for each user as follows [12]:

$$\text{AVG P} = \frac{P_{10} + P_{20} + P_{30}}{3} \quad (8)$$

Where P_{10}, P_{20} and P_{30} are precisions for cut-off results for top 10, 20 and 30 recommended papers. The precision for cut-off results at position *k* (P_k) is used to evaluate the top k recommended papers as follows [12]:

$$P_k = \frac{\text{Number of relevant recommended papers to a user}}{k} \quad (9)$$

Fig.7 shows the MAP results of applying our recommender system on all the users' scenarios using different α and *TopN* values. It can be clearly seen that the MAP results for *TopN*= 6 are relatively low. This is because the top 6 related concepts are a very large number of concepts to be included during build user and paper trees of concepts. The MAP results increase whenever the *TopN* value decreases until *TopN*=3. When *TopN*=3, we have the best results because the top 3 similar concepts to a paper might hold the most essential concepts that are expected to be related to this paper, while considering just the top 1 or 2 concepts may omit some of very significant concepts.

We tested our system with different values for α in the range of [0.1 to 0.7]. Fig.7 shows that the MAP results improve when α value comes close to 0.4 and *TopN* values decrease, and clearly MAP results tend to decrease when reach the smallest or largest values (i.e. 0.1 and 0.7 respectively). The results are very low when α = 0.7, because the propagation value is very large, and then large values are propagated over the reference ontology that makes recommending the correct interests is difficult. When α = 0.1, most of the papers were mapped to leaf concepts from the reference ontology which make the recommendations to be too specific to represent all the users' interests. When α=0.4, the MAP results improve considerably as this value maintains a balance between general and specific concepts. According to these results, we assign α to be 0.4 and *TopN* to be 3 in our system.

2) *Comparing our system against baselines*

In this section we compared our DNTC system against two baselines. Baseline 1 is recommender system using dynamic vector of concepts (DVC) where there is no propagation of weights to parents (i.e. α=0). Baseline 2 is recommender system using non-normalized tree of concepts (NNT) [3].

Fig.8 shows the AVG P for our DNTC system against the two baselines. For user scenarios 1, 2 and 3 that consider only three concepts, we can see that vector of concepts system results are comparable with our DNTC model. However, when the number of concepts are increased in the other scenarios to be more than three concepts, our DNTC system outperforms the DVC method. This is because with multiple concepts the task of user profiling and recommendation is more difficult for the recommender system based on vectors of concepts. For instance, scenarios 7, 8, and 9 consider five concepts during users' reading and there is a substantial improvement in the average precision for these scenarios by using our system. Therefore, when a user reads multiple concepts, our system based on tree of concepts significantly outperforms the system that based on vectors of concepts.

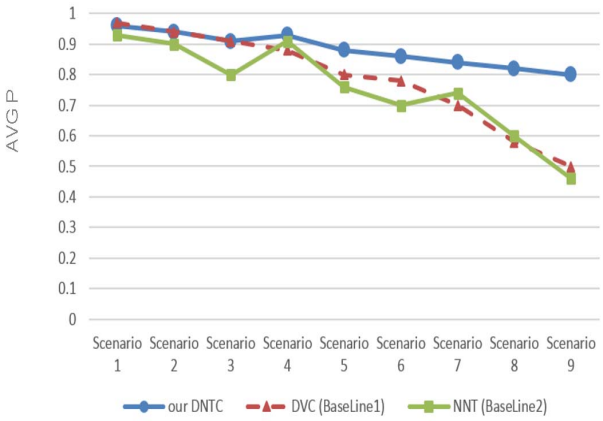

Fig. 8. Comparing average precision for each scenario with the three recommender systems.

With baseline 2 NNT the results in Fig.8 shows that when the quantity of papers is small as in scenarios 1, 4 and 7 that involve 15 papers, the results for NNT system are slightly lower than our system. However, when the quantity of papers is increased to be 30 papers with more than three concepts in scenarios 5 and 8, the results for NNT decline significantly compared with our system. NNT results dramatically drop when the number of papers becomes 50 papers in scenarios 3, 6 and 9. This is because the NNT system does not normalize the concepts' weights in the user's tree of concepts to be appropriate to compare them with the concepts' weights in a paper's tree of concepts. Hence as the user reads more papers, the weights in the user profile grow and become less and less comparable with the weights in the profile of a single paper.

Finally, both the DVC and NNT system achieved the lowest average precision performance in Fig. 8 at scenario 9, where the scenario consider five concepts of interests and 50 papers. The average precision at scenario 9 for DVC is 0.5 and for NNT is 0.46. Whereas the average precision for our DNTC system is 0.8, DNTC system did not drop dramatically as DVC and NNT systems. Therefore, when a user reads multiple concepts and large quantity of papers, our system significantly outperforms both of the baselines systems. Table 1 shows the MAP that reflect the results of those of Fig.8. Our DNTC system has the highest MAP of 0.88, while DVC scored the second best MAP (i.e. 0.78) while NNT achieved the lowest MAP of 0.76.

TABLE I. MAP RESULTS FOR THE THREE SYSTEMS.

Recommender system	MAP
Our system (DNTC)	0.88
Baseline 1 (DVC)	0.78
Baseline 2 (NNT)	0.76

Overall, it can be clearly seen that the proposed DNTC system effectively outperforms the other two systems, and is able to provide high average precision when a user has multiple concepts and read large quantity of papers.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a novel recommender system for research papers which used a Dynamic Normalized Tree of Concepts (DNTC) user modelling technique. Our system utilizes the ontology for 2012 ACM CCS, which is far richer and more complex than the previous 1998 ACM CCS ontology. The user profiling phase creates a user profile as a dynamic normalized tree of concepts which is used with a dynamic tree edit distance method to compare between the user profile and the new unseen research papers that are also represented as tree of concepts. We performed offline evaluations to find the optimal values for α and *TopN* parameters that can produce the best overall performance for our system. According to our evaluative results, the optimal values are $\alpha = 0.4$ and $TopN = 3$. As part of evaluations we compared our DNTC model with two baselines: recommender system using dynamic vector of concepts (DVC) and recommender system using non-normalized tree of concepts (NNT). Our results show that our novel DNTC model significantly outperforms both DVC and NNT systems when simulating user's reading behavior of large quantity of papers and of multiple topics (concepts). In our future work, we will improve our system to be able to determine multiple concepts reflecting user's long-term and short-term interests.

REFERENCES

[1] G D. H. Park, H. K. Kim, I. Y. Choi, and J. K. Kim, "A literature review and classification of recommender systems research," Expert Systems with Applications, vol. 39, no. 11, pp. 10059-10072, 2012.

[2] "The 2012 ACM Computing Classification System — Association for Computing Machinery," https://www.acm.org/about/class/2012 .

[3] K. Chandrasekaran, S. Gauch, P. Lakkaraju, and H. P. Luong, "Concept-based document recommendations for citeseer authors," In International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems. Springer Berlin Heidelberg, pp. 83-92, 2008.

[4] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli, "User profiles for personalized information access," The adaptive web, In: Brusilovsky P., Kobsa A., Nejdl W. (eds) The Adaptive Web. Lecture Notes in Computer Science, vol 4321. Springer, pp. 54-89, 2007.

[5] J. Lee, K. Lee, and J. G. Kim, "Personalized academic research paper recommendation system," arXiv preprint arXiv:1304.5457, 2013.

[6] S. Agarwal, and A. Singhal, "Handling skewed results in news recommendations by focused analysis of semantic user profiles. " 2014 International Conference on Reliability Optimization and Information Technology (ICROIT), pp. 74-79, 2014.

[7] X. Tang and Q. Zeng, "Keyword clustering for user interest profiling refinement within paper recommender systems," Journal of Systems and Software, pp.87-101, 2012.

[8] A. Kodakateri Pudhiyaveetil, S. Gauch, H. Luong, and J. Eno, "Conceptual recommender system for CiteSeerX," In Proceedings of the third ACM conference on Recommender systems, pp. 241-244, 2009.

[9] P. Lakkaraju, S. Gauch, and M. Speretta, "Document similarity based on concept tree distance," In Proceedings of the nineteenth ACM conference on Hypertext and hypermedia, pp. 127-132, 2008.

[10] K. Sparck Jones, and P. Willett, Readings in Information Retrieval. Morgan Kaufmann, San Mateo, US. 1997.

[11] C. D. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval." Cambridge University Press, 2008.

[12] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, "Recommender systems: an introduction," Cambridge University Press, 2010.

[13] B. Ricardo and R. Berthier, "Modern Information Retrieval the concepts and technology behind search second edition," Addision Wesley, 2011.

[14] L. Li, L. Zheng, F. Yang, and T. Li, "Modeling and broadening temporal user interest in personalized news recommendation," Expert Systems with Applications, Elsevier, pp. 3168-3177, 2014.

[15] G. Shani and A. Gunawardana, "Evaluating recommendation systems," in Recommender systems handbook, Springer, pp.257-297, 2011.

[16] G. Salton and M. McGill, "Introduction to modern information retrieval,", McGraw-Hill, 1983.

[17] M. Zeb and M. Fasli, "Adaptive user profiling for deviating user interests," IEEE 3rd Conference in Computer Science and Electronic Engineering (CEEC), pp. 65-70, 2011.

[18] "CiteseerX, scientific literature digital library and search engine," http://citeseerx.ist.psu.edu/index.