Automatic Academic Advisor

Kamal Taha
Department of Electrical and Computer Engineering
Khalifa University of Science, Technology & Research
Abu Dhabi, UAE
kamal.taha@kustar.ac.ae

Abstract— **One of the problems that face a Distance Education academic advisor *(and for lesser degree local academic advisors)* is to identify courses that best suit a student's *interests and academic skills* from a wide collection of elective courses. This is because an advisor needs to select courses that suit both the interest and academic skills of the student. The student may not be able to know his interest in a course from merely its title or from the description of the course provided in the course catalogue. Also, the advisor needs to advise the student to take a course that suits the student's academic performance and skills. Towards this, the advisor needs to consider the performance of students in all his prior courses, which is time consuming. These problems can be overcome using a course recommender system. We introduce in this paper an XML user-based Collaborative Filtering (CF) system called AAA. The system advises a student to take courses that were taken *successfully* by students, who have the same interest and academic performance as the student. We experimentally evaluated AAA. Results showed marked improvement.**

Keywords-component; Course recommender system; Distance education; Automatic academic advisor; collaborative filering

I. Introduction

Distance Education (DE) *"is a field of education that focuses on the pedagogy/andragogy, technology, and instructional systems design that are effectively incorporated in delivering education to students who are not physically "on site" to receive their education"* [5]. The emergence of DE brought about new economies, increased the spectrum of students' backgrounds, and increased linkages in the international community. The following are some of the advantages of DE: (1) no waste of time in transport, (2) flexibility to study in any location with an Internet connection, (3) education feasibility for those who have full time jobs, families, and limited resources, (4) accessibility for those with restricted mobility. Many universities have turned to DE due to its increased demand and also as a means to increase revenue.

Despite all the advantages of Distance Education (DE), it has also a number of negative effects caused by receiving an education without a classroom. One of these disadvantages is the ineffective student academic advising. One of the key responsibilities of a student academic advisor is to advice students to take courses that suit both the interest and academic skills of the student. Usually, there are several options of elective courses that a student can select from. The student may not be able to know his interest in a course from merely its title or from the description of the course provided in the course catalogue. Also, the advisor needs to advise the student to take a course that suits the student's academic performance and skills. Towards this, the advisor needs to consider the performance of students in all his prior courses, which is time consuming. We introduce in this paper a type of Collaborative Filtering (CF) system called **A**utomatic **A**cademic **A**dvisor (AAA), which overcomes the problems of student advising outlined above. CF [7] is one of the successful recommendation tools. It is the process of filtering for information using the opinion of other people.

AAA aims at predicting a student's academic performance and interest for a course based on a collection of profiles of students who have similar interests and academic performance on prior courses. The framework of AAA identifies a set of course features for every academic major. A course feature is a characteristic skill or attribute that a student needs to possess in order to succeed in the course. For example, some of the course features for Computer Science major can be comprehension skills, memorization skills, programming skills, math skills, inferential thinking skills, problem solving skills, application of strategies skills, etc. Students are categorized based on their similarity on course features. Each category (bicluster) includes students who have close academic skills and interests (i.e., course features) in a number of courses. AAA would return to the active student a ranked list of courses that have been rated high by the majority of the members of the cluster, to which the active student belongs. That is, AAA outputs ranked lists of courses, taking into account not only the initial preferences of the active student, but also the ratings of the bicluster, to which the user belongs. The basic idea is that if the students who have the same academic profile as the active student took a course successfully in the past, it is likely that this student will succeed in this course. That is, the underlying assumption is that those who have similar academic performance and interest on prior courses tend to have the same academic performance and interest on future courses. AAA assigns a bicluster to each student user dynamically on the fly.

A student may belong to more than one bicluster. The results of a query submitted by a student user or his academic advisor will be *filtered* and *ranked* courses based on the union of the interests and academic skills of the biclusters, to which the student belongs. In the framework of AAA, students' characteristics (e.g., biclusters) are inferred *implicitly* by the system without involving the user. That is, the student is not required to reveal the biclusters to which the student belongs. The student is determined whether or not he/she belongs to a bicluster *G* by matching his/her ratings on course features with the ratings of *G*. AAA constructs biclusters and also identifies their interests and academic skills *dynamically* on the fly. We

COLLABORATECOM 2012, October 14-17, Pittsburgh, United States
Copyright © 2012 ICST
DOI 10.4108/icst.collaboratecom.2012.250338

developed formal concepts and algorithms that identify the interests and academic skills of various biclusters dynamically on the fly. These interests and academic skills are determined from the interests and academic skills of the biclusters' member users using a group modeling strategy.

II. Related Work

There have been a number of works that have addresses on-line automatic advising and predicting student performance in e-learning [6, 9, 10]. In [9], the authors provide techniques for on-line automatic recommendations in e-learning systems using the access history of learners. The work in [6] provides a guide to developing e-advising standards for advisees, advisors, and administrators. The work in [10] uses recommender system techniques for educational data mining and for predicting student performance.

CF [7] is one of the successful recommendation tools. It is the process of filtering for information using the opinion of other people. A number of CF algorithms have been proposed. There are two major classes of these algorithms [2], memory-based and model-based approaches. Memory-based CF (e.g., [2]) predicts a user's preference based on his/her similarity to other users in the database. Model-based CF first learns a descriptive model of the user preferences and then uses it for providing item recommendation.

The advantage of the memory-based methods over their model-based alternatives is that less parameters have to be tuned. Existing memory-based CF methods, mainly user-based (e.g., [2]) and item-based (e.g., [4]) methods, predict new ratings by aggregating rating information from either similar users or items. Given an unknown test rating to be estimated, user-based CF measures similarities between test user and other users. Item-based CF measures similarities between test item and other items.

There have been a number of researches in filtering based on group profiling [1, 8, 11, 14, 13, 15]. In most of these works, a group is formed based on common interests of its members on an item(s)/features. Work in [11] describes how a combination of collaborative and demographic filtering can be used to recommend product bundles. It describes how stored data is used to recommend a combination of tourist services.

In [8], the authors present Caracará, a system for searching and mining information on the World Wide Web, using a dynamic grouping process. Carcará groups Internet users according to their profile. After that, the system makes suggestions of URLs which are likely to be useful for the users of these groups.

Work in [1] creates categories of users having similar demographic characteristics, and tracks the aggregate buying behavior of users within these categories. Recommendations for a new user are issued by applying the aggregate buying preferences of previous users in the category to which the user belongs. In [15], the authors present a model for supporting social groups in an Ubicomp environment. There must be consensus between group members in order for a person to be a member of the group.

III. Outline of the Approach

Notation 1 - Course feature:
A course feature is a characteristic skill or attribute that a student needs to possess in order to succeed in the course.

AAA aims at predicting a student's academic performance and interest for a course based on a collection of profiles of students who have similar interests and academic performance on prior courses. The framework of AAA identifies a set of course features for every academic major.

Students are categorized based on their similarity on course features. Each category (bicluster) includes students who have close academic skills and interests (i.e., course features) in a number of courses. AAA would return to the active student a ranked list of courses that have been rated high by the majority of the members of the cluster, to which the active student belongs. That is, AAA outputs ranked lists of courses, taking into account not only the initial preferences of the active student, but also the ratings of the bicluster, to which the user belongs. The following are outline of the sequential processing steps taken by AAA:

- *Step 1:* Categorizing students into biclusters. Each bicluster includes students with similar academic skills and interests. That is, the simultaneous clustering of students and their ratings on course features discovers biclusters, which correspond to groups of students exhibiting highly correlated ratings on groups of course features. Section IV describes this process in more details.
- *Step 2*: Identifying the academic skills of each bicluster. This is done by identifying the bicluster's scores on course features. Based on the weights of a bicluster's member students on course features, each course feature is given a score. This score reflects the importance of the course feature to the bicluster relative to other course features. Section V describes this process in more details.
- *Step 3:* Identifying the bicluster of a new student user. This is done by matching the student's rating with the biclusters' ratings computed in step 1. That is, the system identifies (*implicitly*) the member students of a bicluster G_x by matching their ratings with the rating pattern of G_x. Section VI describes this process in more details.
- *Step 4*: Ranking and returning recommended courses for new student user. This is done using the scores of the bicluster, to which the student belongs. The courses will be displayed to the student user after being ranked based on their features' scores. Section VII describes this process in more details.

Fig. 1 is an overview of our approach. It shows the sequential processing steps for recommending courses.

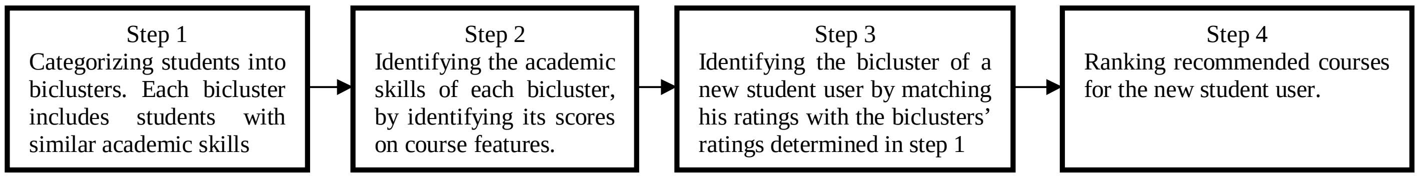
Step 1
Categorizing students into biclusters. Each bicluster includes students with similar academic skills
Step 2
Identifying the academic skills of each bicluster, by identifying its scores on course features.
Step 3
Identifying the bicluster of a new student user by matching his ratings with the biclusters' ratings determined in step 1
Step 4
Ranking recommended courses for the new student user.

Fig 1: A graphical representation showing the sequential processing steps for recommending courses

IV. Identifying Biclusters

We model a student's ratings on course features as a set $D = \{(a_1, w_1), \ldots, (a_m, w_m)\}$, where: a_i denotes course feature i and w_i a weight on a_i. The weight w_i is a value scaled between 1 and 10. A complete set of course features are presented to students to determine their relevance. The system provides the students with a *graphical user interface* (GUI) to reveal their ratings on course features. Table 1 is an example of student's ratings on course features.

The simultaneous clustering of students and their ratings on course features discovers *biclusters,* which correspond to groups of students exhibiting highly correlated ratings on groups of course features. Let T be a table of student ratings on course features. Let the columns of T represent course features and the rows represent students. The biclustering technique finds subgroups of rows and columns in the table T that are similar as possible to one another and as different as possible to the rest. Biclustering has been used in many bioinformatics research works (e.g., [12]). For the biclustering step, we use the xMotif algorithm [12]. The algorithm finds subsets of rows and subsets of columns with coherent values *(i.e., subsets of students who have analogous rating behavior).* Each bicluster is defined on a subset of rows and a subset of columns. Two biclusters may overlap. We now introduce a running example to illustrate some of the concepts in this paper.

Example 1: Consider Table 1, which shows the ratings of nine students on seven Computer Science course features. The rating scale is between [0-10]. After applying the xMotif algorithm to Table 1, three biclusters have been identified as shown in the table. These biclusters are *("B" denotes bicluster, "U" denotes user, and "F" denotes course feature)*:

B_1: $U_{B1} = \{U_1, U_3, U_4, U_6\}$
F_{B1} = {comprehension skills, memorization skills, programming skills, math skills, inferential thinking skills}

B_2: $U_{B2} = \{U_1, U_2, U_4\}$
F_{B2} = {Programming skills, math skills, inferential thinking skills, problem solving skills, application of strategies skills}

B_3: $U_{B3} = \{U_5, U_7, U_8, U_9, U_{10}\}$
F_{B3} = {inferential thinking skills, problem solving skills, application of strategies skills}

TABLE 1: Weighted User-Feature Matrix

	comprehension skills	memorization skills	programming skills	math skills	inferential thinking skills	problem solving skills	application of strategies skills
U_2	4	10	7	8	8.5	7	6
U_4	6	4.5	5.5	5	6	5.5	8
U_1	5	5	6	6	5	8.5	5
U_3	4	5.5	6	5.5	6	9	10
U_6	5.5	4	4.5	4	5.5	10	8.5
U_{10}	6	6	4	4	4	5	5
U_8	7	9.5	8		3.5	3	4.5
U_5	6.5	9	10		5	4	3.3
U_7	6	10	9		4	5.5	3
U_9	10	5	8.5		3.5	4,5	4

V. Identifying the Academic Skills and Interests of a Bicluster

Based on the weights of a bicluster's member students on course features, each course feature is given a score. This score reflects the importance of the course feature to the bicluster relative to other course features. We adopt the following strategy for determining these scores:

> Each *course feature* is assigned a *score*. This score is based on the difference between the *number of times* the course feature *beats* other course features *(i.e., assigned a higher weight by the members of the bicluster)*, and the number of times it *loses*.

Definition 1 – A score of a course feature:
Let $a \succ b$ denote: the *number of times* the *members* of a bicluster rated their academic skills on course feature *a greater* than that of course feature *b*. Let *c(a)* denote the score of course feature *a*. Given the dominance relation $\succ$ on a set F of course features rated by the bicluster, the score *c(a)* of course feature *"a"* equals:
$$|\{b \in F : a \succ b\}| - |\{b \in F : b \succ a\}|.$$

The following are some of the characteristics of this scoring strategy:

(1) The sum of the scores of all course features is always zero.

(2) The *highest and lowest possible* scores are (n-1) and – (n-1) respectively, where n is the number of course features.

We normalize the scores by first adding the *absolute of the most negative score* to all scores and then normalizing the resulting values.

Example 2: Table 2 shows bicluster 1 of example 1 (recall Table 1). Based on the ratings in Table 2, the "beats" and "looses" of each course feature are shown in Table 3. The symbol "+" denotes that a feature beat a corresponding one (i.e., rated higher by the majority of users), while "-" denotes it lost. For example, feature "comprehension skills" beat feature "memorization skills". A zero means: two features beat each other the same number of times and also lost to each other the same number of times. The raw before the last one in Tables 3 shows the score of each feature computed using the strategy described in Definition 1. The last raw shows the normalized scores.

TABLE 2: BICLUSTER 1 OF EXAMPLE 1

	comprehension skills	memorization skills	programming skills	math skills	inferential thinking skills
U_4	6	4.5	5.5	5	6
U_1	5	5	6	6	5
U_3	4	5.5	6	5.5	6
U_6	5.5	4	4.5	4	5.5
U_{10}	6	6	4	4	4

TABLE 3: BEATS/LOOSES, SCORE, AND NORMALIZED SCORE OF EACH FEATURE BASED ON THE RATINGS IN TABLE 2

	comprehension skills	memorization skills	programming skills	math skills	inferential thinking skills
comprehension skills	0	-	-	-	0
memorization skills	+	0	+	+	+
programming skills	+	-	0	-	+
math skills	+	-	+	0	+
inferential thinking skills	0	-	-	-	0
Score	+3	-4	0	-2	+3
Normalized Score	0.35	0	0.2	0.1	0.35

VI. IDENTIFYING A NEW MEMBER OF A BICLUSTER IMPLICITLY

The system identifies (*implicitly*) member students of a bicluster G_x by matching their ratings with the rating pattern of G_x. Let *sim*(u_m , G_x) be the *similarity* between the ratings of user u_m and bicluster G_x. We measure *sim*(u_m , G_x) using the *cosine-similarity measure* shown in equation 1:

$$sim(u_m, G_x) = \frac{\sum_{\forall i \in I} \left(\left(r_{u_m,i} - \bar{r}_{u_m} \right) \left(r_{G_x,i} - \bar{r}_{G_x} \right) \right)}{\sqrt{\sum_{\forall i \in I} \left(r_{u_m,i} - \bar{r}_{u_m} \right)^2} \sqrt{\sum_{\forall i \in I} \left(r_{G_x,i} - \bar{r}_{G_x} \right)^2}} \quad (1)$$

- I: Set of features rated by bicluster G_x and *co-rated* by u_m.
- $r_{u_m,i}$: Weight of user u_m on course feature *i*.
- $r_{G_x,i}$: Normalized score of bicluster G_x on feature *i*.
- $\bar{r}_{u_m}$: Normalized *mean* weight of u_m on set *I*.
- $\bar{r}_{u_m} = \frac{\sum_{\forall i \in I} r_{u_m,i}}{|I|}$
- $\bar{r}_{G_x}$: Normalized *mean score* of G_x on set *I*;
- $\bar{r}_{G_x} = \frac{\sum_{\forall i \in I} r_{G_x,i}}{|I|}$

Equation 1 considers *each* feature rated by bicluster G_x and co-rated by user u_m even if the feature was rated by only one student of G_x. Therefore, the equation *may give misleading similarity results*, since some features in set *I* may not reflect the actual academic performance of G_x. A feature that has been rated very low or by few members of bicluster G_x is most likely rated by a member(s) of G_x who belongs also to another bicluster G_y. Therefore, when measuring the similarity between an active user and G_x, we should consider only the features that reflect the preferences of G_x. That is, we need to consider only the dominant features of G_x *(i.e., the features that have been rated high and by the majority of the members of* G_x*).*

We adopt the following strategy for determining the set of dominant features for a bicluster. From the set *F* of all features, the subset F' is the dominant features for a bicluster, if every feature in F': (1) dominates every feature not in F' *(i.e., has a greater score)*, and (2) acquires a score greater or equal to a threshold *z*. For example, recall Table 3 and consider that *z* is set to "0". Accordingly, the set F' of dominant features for the bicluster would be {comprehension skills, programming skills, inferential thinking skills}. We now formalize the concept of dominant features Definition 2.

Definition 2 – Dominant features for a bicluster:
Let F be a set of n features and c(f) be the score of feature f. The subset $F' \subset F$ *of dominant features with maximal scores for a* bicluster *is given by: {a* $\in$ *F: c(a)* $\geq$ *c(b), for all b* $\in$ *F}* and *{c(a)* $\geq$ *z: (n-1) > z < –(n-1)}*

We adjusted equation 1 so that only the subset $F' \cap I$ is considered, as shown in Equation 2.

$$sim(u_m, G_x) = \frac{\sum_{\forall i \in F''} \left(\left(r_{u_m,i} - \bar{r}_{u_m} \right) \left(r_{G_x,i} - \bar{r}_{G_x} \right) \right)}{\sqrt{\sum_{\forall i \in F''} \left(r_{u_m,i} - \bar{r}_{u_m} \right)^2} \sqrt{\sum_{\forall i \in F''} \left(r_{G_x,i} - \bar{r}_{G_x} \right)^2}} \quad (2)$$

- F': Set of dominant features rated with maximal scores by bicluster G_x

- F'' : Subset of F' co-rated by user u_m (i.e., $F'' \subseteq F'$).
- $\bar{r}_{u_m} = \frac{\sum_{\forall i \in F''} r_{u_{m,i}}}{|F''|}$ and $\bar{r}_{G_x} = \frac{\sum_{\forall i \in F''} r_{G_{x,i}}}{|F''|}$

From the set F'', equation 2 overlooks the subset $F' - F''$ *(i.e., the subset that has not been co-rated by student user u_m).* Therefore, the equation may give inaccurate similarity results. We observe that we can consider user u_m assigned a weight of zero to each of the features in the subset. The reason is that users usually have either no or very little interest on features they do not rate. We adjusted equation 2 to consider the subset $F' - F''$ as shown in equation *3*.

$$sim(u_m, G_x) = \frac{\sum_{\forall i \in F''} \left(r_{u_m,i} - \bar{r}_{u_m}\right)\left(r_{G_x,i} - \bar{r}_{G_x}\right) + \sum_{\forall j \in P} \left(\bar{r}_{u_m}\left(r_{G_x,j} - \bar{r}_{G_x}\right)\right)}{\sqrt{\sum_{\forall i \in F''} \left(r_{u_m,i} - \bar{r}_{u_m}\right)^2 + |P|\left(\bar{r}_{u_m}\right)^2} \sqrt{\sum_{\forall i \in (F'' \cup P)} \left(r_{G_x,i} - \bar{r}_{G_x}\right)^2}} \quad (3)$$

- $P = \{F' - F''\}$

Let F_u be the set of features rated by student user u_m. As a final improvement of the similarity equation, we consider each feature $f_k \in \{F_u - F'\}$, if the weight of bicluster G_x on f_k beat other features' weights at least k number of times, where $k > 0$. However, we need to penalize each expression operand in the equation involving f_k to ensure that it will have a lower impact on the similarity result. Moreover, we need to scale down these expressions appropriately to account for the rank specificity of f_u among the list of features ranked by G_x to ensure that that lower ranked features indeed get higher penalty. Towards this, we *penalize and scale down* each expression operand involving f_u by a factor $decay^{t-1}$, where *decay* is a parameter that can be set to a value in the range 0 to 1. We set the exponent t to account for the rank of f_u among the list of features ranked by bicluster G_x. *We adjusted equation 3 accordingly as shown in equation 4.*

$$sim(u_m, G_x) = \frac{\sum_{\forall i \in F''} \left(r_{u_m,i} - \bar{r}_{u_m}\right)\left(r_{G_x,i} - \bar{r}_{G_x}\right) + \sum_{\forall j \in P} \left(\bar{r}_{u_m}\left(r_{G_x,j} - \bar{r}_{G_x}\right)\right) + N}{\sqrt{\sum_{\forall i \in \{F'' \cup F_k\}} \left(r_{u_m,i} - \bar{r}_{u_m}\right)^2 + |P|\left(\bar{r}_{u_m}\right)^2} \sqrt{\sum_{\forall i \in (F'' \cup P)} \left(r_{G_x,i} - \bar{r}_{G_x}\right)^2 + M}} \quad (4)$$

$$N = \sum_{\forall i \in V} \left(r_{u_m,i} - \bar{r}_{u_m}\right)\left(r_{G_x,i} - \bar{r}_{G_x}\right) \times decay^{t-1}$$

$$M = \sum_{\forall i \in V} \left(r_{G_x,i} - \bar{r}_{G_x}\right)^2 \times decay^{t-1}$$

$$V = \{F_k - F'\}$$

F_k : Set of features that are: (1) rated by G_x, (2) co-rated by u_m, and (3) assigned weights by G_x that beat other features' weights at least k number of times.

Example 3: Recall Table 3. Consider that the threshold k in equation 4 has been set to 1. Thus, feature *"math skills"* will be considered in equation 4, if the active user co-rated it *(even though "math skills" $\notin F'$).* The expressions in equation 4 involving *"math skills"* will be penalized by a factor decay^{t-1}, where t is 5 *(i.e., the rank of "math skills" in the set rated by* bicluster G_x*).* Parameter *decay* can be set to a value from 0 to 1.

As *each* new student user is identified by the system as belonging to a bicluster G_x *(using equation 4)*, the current course features' scores of G_x will be re-optimized and re-updated (*dynamically*) based on: (1) the rating of this new user on these course features, and (2) the rating of the other member students of G_x on these course features. That is, the rating of *each* subsequent user would *update* and *optimize* current course features' scores for the bicluster by updating course features' number of beats/looses and scores *(recall Table 3).*

VII. Ranking Recommended Courses

The system ranks recommended courses using a feature-course matrix *N*. In this matrix, element $N(j, i)$ is *one*, if course C_j requires a student to possess the academic skills of feature f_j and *zero* otherwise. The profile $N(I_j)$ of course C_j is the *j-th* column of matrix *N*. The score of course C_j is the *summation of the normalized scores* (e.g., recall Table 3) of the course features that C_j requires *(see equation 5)*

$$Score\ C_j = \sum_{\forall N(f_i, C_j) = 1} score\, f_i \quad (5)$$

The courses will be displayed to the student user after being ranked based on their scores.

Example 4: Let *c* and *f* denote course and feature respectively. Table 4 shows an example data set of Matrix *N* for our running example. Element $N(c_j, f_i)$ is one if course c_j requires the student to have the skills of feature f_i and zero otherwise. For example, the score of c_1 is the sum of the normalized scores of features comprehension skills, math skills, and, inferential thinking skills, which is 0.35 + 0.1 + 0.35 = 0.8 (*recall Table 3).* Therefore, the courses will be ranked for the active user as follows: c_2, c_1, c_4, c_3, and c_5.

TABLE 4: FEATURE-COURSE MATRIX *N*. c_i DENOTES COURSE *i*

feature	c_1	c_2	c_3	c_4	c_5
comprehension skills	1	1	0	1	1
memorization skills	0	0	1	0	1
programming skills	0	1	0	1	0
math skills	1	1	0	0	0
inferential thinking skill	1	1	1	0	0
Score	0.8	1	0.35	0.55	0.2

VIII. EXPERIMENTAL RESULTS

We implemented AAA in Java and ran it on an Intel(R) Core(TM)2 Dup CPU processor, with a CPU of 2.1 GHz and 3 GB of RAM, under Windows Vista. We evaluate AAA using real-user evaluation conducted by 43 students from the University of Texas at Arlington-USA and Khalifa University-UAE. Each student was asked to: (1) rank and list the courses he/she received at least grade B on, (2) rate the features of these courses and provide them to AAA.

A. *Measuring the Distance between the Lists Ranked by the Students and the Lists Ranked by AAA*

We measured the distance $d(\sigma_u,\sigma_s)$ between each list[1] of courses ranked by a student u and the corresponding list ranked by AAA, using the Euclidean distance measure shown in equation 6.

$$d(\sigma_u,\sigma_s)=\sum_{x\in X}|\sigma_u(x)-\sigma_s(x)| \qquad (6)$$

- s: Refers to the AAA system.
- *X*: Set of courses.
- $\sigma_u \in [0,1]^{|X|}$: List of courses ranked by student u .
- $\sigma_s \in [0,1]^{|X|}$: A list ranked by AAA.
- $\sigma_u(x)$ and $\sigma_s(x)$: *position* of course $x \in X$ in the lists σ_u and σ_s respectively (*a ranking of a set of n courses is represented as a permutation of the integers 1, 2, . . . , n*).

Fig. 2 shows the results. We can infer from the experimental results that: the "closeness" between the lists ranked by the students and the corresponding lists ranked by AAA *increases consistently* as the cumulative number of students increases. This is because after the ratings of *each* student are submitted to AAA, it *updates* and *optimizes* the current ratings of the student's Bicluster based on the ratings of this student.

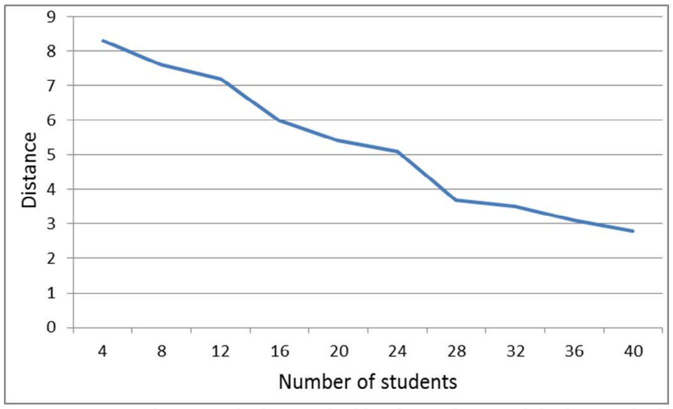

Fig. 2: Distance between the lists ranked by the students and the lists ranked by AAA

B. *Measuring Recall and Precision*

Let: (1) *N* be the number of courses in a list recommended by AAA, (2) R_n be the number of relevant courses for the student in the recommended list, and (3) R_{ALL} be the total number of relevant courses for the student.

- Recall = R_n / R_{ALL}
- Precision = R_n / N.

Fig. 3 shows the recall-precision diagram. As the figure shows, AAA achieved good recall and precision. It achieved good precision because: (1) AAA forms a bicluster based on the rating similarity of its members on the *features* of courses, and (2) of the effectiveness of the AAA's group modeling strategy and similarity equation. AAA achieved good recall because it considers: (1) *all* dominant course features of a student's bicluster, even if the student did not co-rate some of them, and (2) *non-dominant* course features of the student's bicluster, whose assigned weights beat other features' weights at least *k* number of times[2].

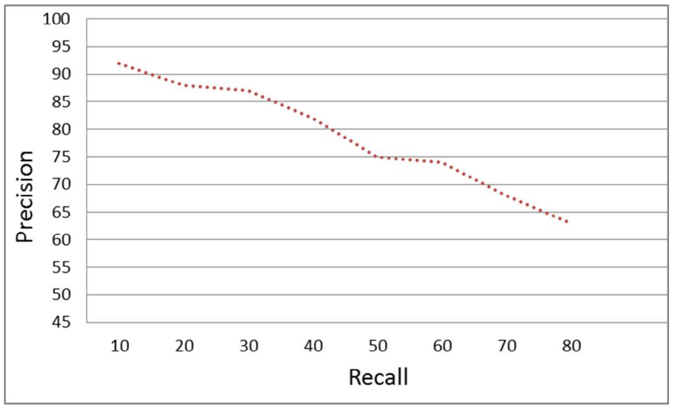

Fig. 3: Recall vs. precision

[1] The list of courses, which the student had received at least grade B on.

[2] In the experiments, we set the threshold *k* to (*number of features*) / 2.

C. Measuring Explain Coverage

Explain coverage measures the number of course features that are: (1) rated by a student to a value greater or equal to a threshold *p*, and (2) covered by the features of the courses recommended by the AAA. We set the threshold *p* to 5. Fig. 4 shows the explain coverage versus the number of course features. As the figure shows, intuitively, explain coverage increases as the number of course features increases.

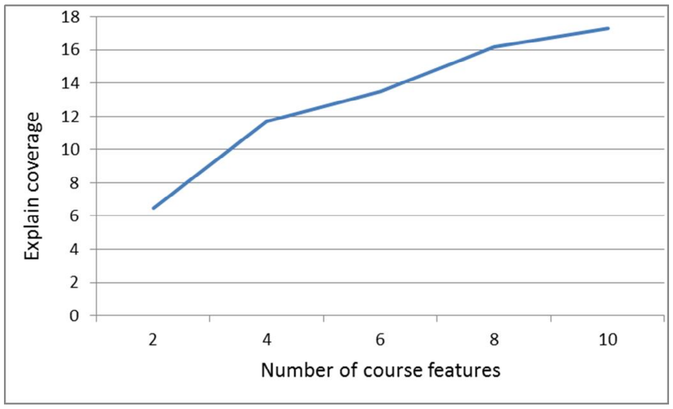

Fig. 4: Explain coverage vs. number of course features

IX. Conclusion

In this paper, we proposed an XML-based Collaborative Filtering (CF) recommender system, called AAA, which overcomes the problems of student advising. The system advises a student to take courses that were taken *successfully* by students, who have the same interest and academic performance as the student. AAA aims at predicting a student's academic performance and interest for a course based on a collection of profiles of students who have similar interests and academic performance on prior courses. The framework of AAA identifies a set of course features for every academic major. A course feature is a characteristic skill or attribute that a student needs to possess in order to succeed in the course.

AAA would return to the active student a ranked list of courses that have been rated high by the majority of the members of the cluster, to which the active student belongs. That is, AAA outputs ranked lists of courses, taking into account not only the initial preferences of the active student, but also the ratings of the bicluster, to which the user belongs.

We experimentally evaluated AAA using real-user evaluation conducted by 43 students from the University of Texas at Arlington-USA and Khalifa University-UAE. The results showed that the distances between the lists of courses ranked by the students based on their prior academic performance and the corresponding lists ranked by AAA are small. Moreover, the results showed good recall, precision, and explain coverage of the AAA.

References

[1] Aimeur, E., Brassard, G., Fernandez, J., and Mani Onana, F. *Priva* cy preserving demographic filtering. Proc. SAC'06.

[2] Breese, J., Heckerman, D. and Kadie, C. "Empirical analysis of predictive algorithms for collaborative filtering," in Proc. 14th UAI, 1998

[3] Chamberlin, D., Fankhauser, P., Florescu, D. and Robie, J. XML Query Use Cases, W3C Working Draft'07

[4] Deshpande, M., and Karypis, G. Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst., 22(1):143–177, 2004.

[5] Edward P. BaileyFocus on Distance Education Developments. Nova publisher

[6] E. Vance Wilson, "A standards framework for academic e-advising services", International Journal of Services and Standards, Vol. 1, Issue 1, 2004.

[7] Herlocker, J., Konstan, J. A., and Riedl, J. "Evaluating Collaborative Filtering Recommender Systems". ACM TOIS, 22 (1), 2004.

[8] Junior, M., Canuto, A., "Carcara: A Multi-agent System for Web Mining Using Adjustable User Profile and Dynamic Grouping". Proc. IEEE/WIC/ACM IAT'06, Hong Kong, China, 2006.

[9] Mohamed Koutheaïr Khribi and Mohamed Jemni and Olfa Nasraoui. "Toward a Hybrid Recommender System for E-Learning Personalization Based on Web Usage Mining Techniques and Information Retrieval", World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education, 2007

[10] Nguyen Thai-Nghe, Lucas Drumond, Artus Krohn-Grimberghe, Lars Schmidt-Thieme. "Recommender System for Predicting Student Performance". Workshop on Recommender Systems for Technology Enhanced Learning. 2010.

[11] O'Connor, P., Höpken, W., and Gretzel, U. *Dynamic Packaging* using a Cluster-based Demographic Filtering Approach. In Proc. the International Conference in Innsbruck, Austria, 2008.

[12] T. Murali and S. Kasif, "Extracting conserved gene expression motifs from gene expression data," in *Proc.* Pacific Symp. Biocomputing Conf., 2003, vol. 8, pp. 77–88.

[13] Tang, L., Liu, H., Zhang, J., Agarwal, N., Salerno, J. Topic Taxonomy Adaptation for Group Profiling. ACM Transactions on Knowledge Discovery from Data, 2008, Vol. 1, No. 4.

[14] Symeonidis, P., Nanopoulos, A., Manolopoulos, Y., Providing Justifications in Recommender Systems. IEEE Transaction Systems, Man, and Cybernetics, Vol. 38, No. 6, 2008.

[15] Wang, B., Bodily, J., Gupta, S., "Supporting Persistent Social Groups in Ubiquitous Computing Environments Using Context Aware Ephemeral Group Service". Proc. IEEE PerCom, 2004.