# On an OptLearn implementation using collaborative filtering[*]

A. Fuster-López[1], P. Guerrero-García[2], E.M.T. Hendrix[3], and J. Martínez-Cruz[1]

[1]Computer Science, Univ. Málaga, {alejandrofuster,jmcruz}@uma.es
[2]Applied Mathematics, Univ. Málaga, pguerrero@uma.es
[3]Computer Architecture, Univ. Málaga, eligius@uma.es

November 8, 2023

**Abstract**

The aim of the iMath project is to develop a recommender system where a student is recommended a next question (or a batch of next questions) in an e-learning platform MathE. This paper describes the systems developed based on collaborative filtering Python application Surprise by the Universidad de Málaga partners of the project.

## 1 Introduction

The aim of the iMath project is to personalise an e-learning path for an individual student in the environment of an e-learning system on mathematics called the MathE portal, cf. [1, 8]. This path does not necessarily coincide with the one a teacher would choose, although this often is considered the best way to go in a teaching environment.

After analysing various recommendation methods, our team decided to develop an automated recommender system (RS) based on collaborative filtering, similar to existing ones, e.g. for recommending films or songs and to adapt them to an educational e-learning environment. Thus, starting for example from a list of 40 exercises (also called questions in this paper) on a given mathematical topic (e.g., matrices and determinants), the objective is to automatically detect to which learners a given student is sufficiently similar to be able to recommend which are the next 5 exercises from a list of most favourably answered exercises among those learners similar to the student.

[*]Funded by iMath project 2021-1-PT01-KA220-HED-000023288.

A fundamental problem is to design a transparent and effective way of obtaining user feedback. An explicit technique is not appropriate for learners, as it is not logical to ask a learner to rate a particular question in an exam as rating a film or a song after watching or listening to it. Therefore, we designed an implicit technique in which the evaluation of an exercise will be given automatically and transparently by the answer provided by the student. This rating consists of an integer number between 0 and 5, where 0 indicates that the student did not answer, 1 indicates that the student marked that he/she does not know the answer, and the rest of values (from 2 to 5) indicate the combinations of correct/incorrect answer to easy/difficult exercise. This ensures that the learner does not have to deal with additional rating in order to have each of the exercises rated and can take advantage of the historical data-file of former answers existing in the MathE project with scores from 2 to 5.

Table 1: Seamless 5-rate question rating scheme

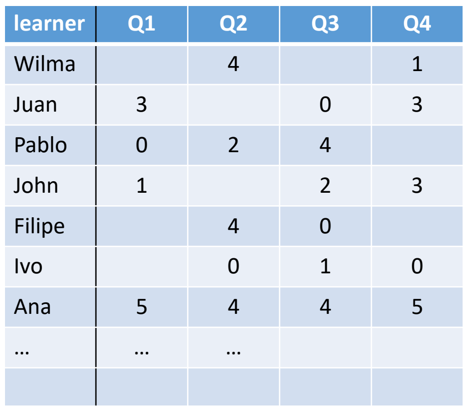
| | |
|---|---|
| $5 \equiv$ | Right Answer for a Difficult Question |
| $4 \equiv$ | Right Answer for a Basic Question |
| $3 \equiv$ | Wrong Answer for a Difficult Question |
| $2 \equiv$ | Wrong Answer for a Basic Question |
| $1 \equiv$ | “I Don’t Know” Answer for a Question |
| $0 \equiv$ | Question has not been answered |

After studying several types of RSs, we chose to use the so-called collaborative filtering (CFs) methodology for its ability to provide predictions about unknown learner/exercise pairs, due to its speed of prototyping and combinability with other types of RSs which other partners of the project are currently working on. Collaborative filtering is able to include graph-based techniques that take advantage of information based on concept maps (CMs). Moreover, we can use easy to use standard metrics (e.g., RMSE and/or MAE) that enable measuring objectively whether a new algorithmic improvement has brought a benefit. Considering different ways of combining RSs to obtain hybrid RSs, we have opted for a non-invasive technique in which, instead of incorporating characteristics of one RS to another, a combination is made on the basis of results obtained by both. In this way, we can measure the progress obtained from combining two RSs even if the metric used by another partner (e.g., based on machine learning classification) is not RMSE and/or MAE. To test the feasibility of our approach, we implemented an algorithm which alternatively recommends with the two most promising CFs among those based on matrix factorization and on clustering, according to tests with randomly generated data.

Currently the partners in Málaga (Spain) and Hamburg (Germany) are working on a framework for combining CF (or other RS) methods with additional information offered by concept maps. In particular, a novel algorithm for generating learning paths based on both error rates and a concept map was developed and implemented. The concept map may be automatically constructed or provided by a teacher. A massive test with virtual answers in the second semester of 2023 looks promising as a prelude to test the entire system with real students, preferably in the first semester of 2024.

The rest of this manuscript describes the applied methodology and refers to literature on which the algorithms are based.

## 2 Rating scheme, notation and evaluation metric



| learner | Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|---|
| Wilma | | 4 | | 1 |
| Juan | 3 | | 0 | 3 |
| Pablo | 0 | 2 | 4 | |
| John | 1 | | 2 | 3 |
| Filipe | | 4 | 0 | |
| Ivo | | 0 | 1 | 0 |
| Ana | 5 | 4 | 4 | 5 |
| … | … | … | | |
| | | | | |

Figure 1: Example of a rating matrix

To translate the recommender system to the usual terminology used in collaborative filtering, see [9], Users are Students, Items are Questions either about a specific subtopic as matrices and determinants, or about any of the subtopics available in the MathE portal. A historical database of ratings is assumed to be available, either that from the very beginning or one constructed during the first months of 2023 by using initially completely random choice among the questions in a given subtopic as recommendation device, i.e. Algorithm 0. The row structure with the fields expected from that historical database is

| | | | | | |
|---|---|---|---|---|---|
| $u_{15}$ | $i_{23}$ | `ans` | `alg` | `StartTime` | `EndTime` |

.

Its interpretation is that student number 15 was recommended question number 23 at `StartTime` by algorithm number `alg` and answered `ans` at `EndTime`. Based on the collaborative filtering recommenders, our RS provides the next five questions as a recommendation offered one after another. The RS quality is measured by standard metrics, i.e. RMSE, MAE, etc.

As students give an answer to the database question, we have a transparent way to rate the question for the student which assumes that the student likes most her recent *dynamic* correct answers. Notice that the characteristic difficult or easy question may change in time. Distinguishing more difficulty levels, will also extend the range of the rating.

We started to study the behaviour of the recommender using artificial answer generation created assuming that teachers should have assigned a specific rating to each of the 5 possible answers. However, we left this idea, because teachers who add questions to the MathE database do not assign a *static* valuation. Thus, as existing historical data does not include the previous static valuation of each choice of all the questions, we have used the coding of Table 1 which is sketched with an example in Figure 1.

One of the challenges we run into is that the historical database does not distinguish between the aforementioned ratings 0 and 1. Moreover, the 0-1 rating information has probably not been saved. We suggest two alternatives in this respect:

(a) Consider an explicit 0 and 1 rating. From the historic data, rank not answered questions as 1 if "I Don't Know" was not among the possible answers for such questions.

(b) Consider both the 0 and 1 rating as a wrong answer. This means they are interpreted as a rate 2 or 3 depending on the difficulty of the question. In this way, one gets rid of the 0 and 1 rating.

Using the terminology introduced by Hug in [4], we define

- $R$: the set of all ratings.
- $R_{\text{train}}$, $R_{\text{test}}$, and $\hat{R}$ denote the training set, the test set, and the set of predicted ratings, respectively,
- $U$: the set of users (students) with indices $u$ and $v$,
- $I$ : the set of items (questions) with indices $i$ and $j$,
- $U_i$ : subset of users that have rated item $i$,
- $U_{ij} \equiv U_i \cap U_j$: subset of users that have rated both items $i$ and $j$,
- $I_u$: subset of items rated by user $u$,
- $I_{uv} \equiv I_u \cap I_v$: subset of items rated by both users $u$ and $v$,
- $r_{ui}$ : the rating obtained by user $u$ for item $i$,
- $\hat{r}_{ui}$ : the estimated rating of user $u$ for item $i$,
- $b_{ui}$ : the baseline rating of user $u$ for item $i$,
- $\mu$ : the mean of all ratings,
- $\mu_u$ : the mean of ratings obtained by user $u$,
- $\mu_i$ : the mean ratings for item $i$,
- $\sigma_u$ : the standard deviation of the rating of user $u$,
- $\sigma_i$ : the standard deviation of the rating of item $i$,
- $N_i^k(u)$: the $k$ nearest neighbour (according to a similarity metric) of user $u$ having rated item $i$,
- $N_u^k(i)$: the $k$ nearest neighbour (according to a similarity metric) of item $i$ that are rated by user $u$.

To compare the relative accuracy of the approaches, we adopt frequently used standards. In particular, we performed a cross-validation RMSE-based procedure consisting of averaging RMSE calculated after splitting $L$ times the set $R = R_{\text{train}} + R_{\text{test}}$ into two disjoint sets TR and TS. One then computes prediction $\hat{r}_{ui}$ using only $R_{\text{train}}$ for all $(u, i)$ pairs in test set TS and finally compares it to the observed ratings $r_{ui}$ available at $R_{\text{test}}$. So

$$RMSE_\ell = \sqrt{\frac{1}{|TS|} \sum_{(u,i) \in TS} (r_{ui} - \hat{r}_{ui})^2}, \qquad RMSE = \frac{1}{L} \sum_{\ell=1}^{L} RMSE_\ell.$$

This computation is implemented in the cross validation procedure `cross_validate` in Surprise [4] with `rmse` as the `measures` parameter.

## 3 Main algorithm and its (sub)algorithms

This section outlines the three (sub)algorithms that are used as building blocks for the OptLearn recommender we develop, described in Algorithm 4.

**Algorithm 4** Developed OptLearn recommender

```
0. IF the list of questions from which the recommendation is done is restricted by a so-called top-layer (sub)algorithm (dealing with item-item relationships and thus tackling learning indicators) THEN take such a restriction into account ELSE consider all questions as eligible END.
1. Call either (sub)Algorithm 1 or (sub)Algorithm 2 to select (on a so-called bottom layer, dealing with user-user relationships) the next (or batch of five next) best question(s) from among the eligible ones.
2. IF several outputs are going to be combined THEN proceed as described in (sub)Algorithm 3 END.
3. Compute cross-validated RMSEs as stated at the end of previous section.
```

### 3.1 SVD$^{++}$ (Algorithm 1)

The matrix-factorization based model is also known as a latent factor model. SVD$^{++}$ is a modification of the SVD algorithm, in which the prediction $\hat{r}_{ui}$ is given by

$$\hat{r}_{ui} = b_{ui} + q_i^T p_u \equiv \mu + b_u + b_i + q_i^T p_u,$$

as described in [6, Eq.(5)] and [5, Eq.(5.2)]. In the SVD$^{++}$ implicit ratings are considered by

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T \left( p_u + \frac{1}{\sqrt{|I_u|}} \sum_{j \in I_u} y_j \right),$$

where a new set of item factor vectors $y_j$ captures implicit ratings, cf. [5, Eq.(5.3)]. An implicit rating describes the fact that a user $u$ rated an item $j$ regardless of the rating value, as described in [5, §5.3.1–5.3.2].

The estimation of all unknowns $b_u$, $b_i$, $p_u$ and $q_i$ is performed, as usual, by minimizing a Funk's regularized squared error function using Alternating Least Squares (ALS) as an alternative to the usual Stochastic Gradient Descent (SGD) to solve the so-called regularized SVD with missing values problem. This is done by selecting `SVDpp()` as the `algo` parameter in Surprise [4].

### 3.2 KNN with Means (Algorithm 2)

The collaborative filtering $k$-NN approach is also known as a user-user neighbourhood. It takes the mean ratings of each user into account. The prediction $\hat{r}_{ui}$ is given by

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i^k(u)} \mathrm{sim}(u,v) \cdot (r_{vi} - \mu_v)}{\sum_{v \in N_i^k(u)} \mathrm{sim}(u,v)},$$

as described in [3, Eq.(1)] and [2, Eq.(4.15)]. The maximum number $k$ of neighbours to take into account for aggregation is 40 and the similarity weights are computed by using the MSD (mean squared differences), namely

$$\mathrm{sim}(u,v) = \frac{|I_{uv}|}{|I_{uv}| + \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2},$$

cf. [2, Eq.(4.22)]. This is done by selecting `KNNWithMeans()` as the `algo` parameter in Surprise [4].

### 3.3 Non-invasive hybrid CF (Algorithm 3)

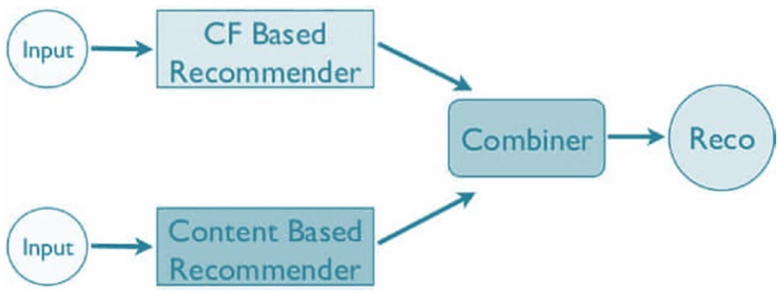


Figure 2: Non-invasive Hybrid RS [7, p. 9]

As pointed out in [7, pp. 8–9], to overcome the drawbacks of two different types of RSs (e.g., a content-based RS and a CF) or two RSs of the same type (e.g., two CFs), one may design a hybrid RS. There are two ways to construct such an hybrid RS:

- An **invasive hybrid RS** incorporates the algorithmic features of one RS into the other.

- A **non-invasive hybrid RS** uses the *results* of one RS to be combined by the *results* obtained by the other. Hence you can combine the results obtained by different RSs although they are using different metrics.

We have chosen a non-invasive technique (Algorithm 3) in which, instead of incorporating characteristics of one RS to another, a combination is made on the basis of results obtained by both after alternatively recommending with Algorithm 1 and Algorithm 2. In this way, we can measure the progress obtained from combining two RSs even if the metric used by another partner (e.g., based on machine learning classification) is not RMSE and/or MAE.

To test the feasibility of our approach, we simply append the results obtained after recommending with Algorithm 2 to the results obtained after recommending with Algorithm 1 to check that the cross-validated RMSE for the whole rating matrix has improved for any (or both) of the combined algorithms.

## 4 Summary

This note described the concepts of collaborative filtering (CF) applied in an e-learning environment where the next question to deal with is recommended by a system. A matrix factorization and a $k$-NN approach have been implemented and tested on about 70 students going through a data set of mathematical questions. Furthermore, a way to combine several RSs in a non-invasive way is described as well. We argue that it is very feasible to lay a recommended path among concepts over a standard recommender based on collaborative filtering.

## References

[1] Consortium 2018-1-PT01-KA203-047361. MathE, a toolkit for students for self-evaluation of their knowledge on selected math topics. `https://mathe.pixel-online.org/`.

[2] Christian Desrosiers and George Karypis. A comprehensive survey of neighborhood-based recommendation methods. In Ricci et al. [9], pages 107–144.

[3] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In F. Gey, M. Hearst, and R. Tong, editors, *Proc. 22nd ACM SIGIR Conference*, pages 230–237. ACM Press, August 1999.

[4] Nicolas Hug. Surprise: A Python library for recommender systems. *Journal of Open Source Software*, 5(52):2174, 2020.

[5] Yehuda Koren and Robert M. Bell. Advances in collaborative filtering. In Ricci et al. [9], pages 145–186.

[6] Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[7] Akshay R. Kulkarni, Adarsha Shivananda, Anoosh Kulkarni, and V. Adithya Krishnan. *Applied Recommender Systems with Python*. APress, 2023.

[8] Maria F. Pacheco, Ana I. Pereira, and Florbela Fernandes. MathE—improve mathematical skills in higher education. In M. Della Ventura, M. Vavalis, and D. Sampson, editors, *Proc. 8th ACM ICEIT Conference*, pages 173–176. ACM Press, March 2019.

[9] F. Ricci, L. Rokach, B. Shapira, and P. Kantor, editors. *Recommender Systems Handbook*. Springer, 1st edition, 2011.