An Hybrid Recommender System for Medical Therapies

Matteo Destro (221222)
matteo.destro@studenti.unitn.it
University of Trento
Artificial Intelligence Systems, 2nd year

ABSTRACT

The vast amount of clinical data available on the Internet nowadays makes it difficult for users to find relevant information for their well-being, and also for medical professionals to make accurate and personalized prescriptions. These issues motivated the application of recommender systems to the healthcare domain, to exploit the large amounts of information available and help the final user in making patient-centric decisions.

This paper proposes an *hybrid recommender system* that exploits the correlation between different kinds of data for recommending medical therapies to cure specific conditions. The final system was evaluated using a dataset of more than 900 000 therapies trials and 100 000 patients, and proved to be effective in producing meaningful recommendations.

1 INTRODUCTION

The large volume of data that is becoming available today in healthcare is opening up new possibilities to improve the efficacy of the medical therapies in use. One important way that this data can be used is to develop *recommender systems* that can provide personalized recommendations for treatments, tailored to the individual patient's characteristics.

Recommender systems are a class of algorithms commonly used in a variety of fields, such as e-commerce and music streaming services, able to predict the rating or preference a user would give to a specific item. In the healthcare domain, recommender systems have been used to suggest clinical trials to physicians [9, 10, 16], and to provide personalized recommendations for treatments and drugs [1, 8].

The development of these systems faces a number of challenges, due to the complex nature of the data and the domain-specific knowledge required. For instance, there is a need to ensure that the recommendations are made in the best interest of the patient. Moreover, in order to be effective, these recommender systems need to have a certain degree of explainability, so that users can understand how the recommendations are being made. However, the algorithms used are often complex and opaque, which can make it difficult to provide a meaningful explanation of the predictions.

Despite these issues, the potential benefits of such systems are significant, and there is a growing body of research that is exploring their use in healthcare.

This work provides an overview of the existing methods in this domain and a viable approach based on an hybrid recommender system. The paper is organized as follows: Section 2 gives an overview of the related works, focusing in particular on the use of recommender systems in healthcare, while Section 3 provides a formal statement of the problem. Section 4 describes the data used in the experiments and how it was acquired. Section 5 presents the actual recommender system, while Section 6 provides more details on its implementation. Finally, Section 7 reports the results of the evaluation and Section 8 gives some final consideration on the proposed approach.

2 RELATED WORK

Many techniques exist for recommender systems, based on different algorithms and assumptions on the underlying data. The general objective of these approaches is to produce a list of items to recommend to the final user, based on its profile and preferences. These methods assume that a set of item ratings given by the users is available, and try to exploit the hidden patterns on this data to predict the preference each user would give to unseen items. A large class of approaches are based on nearest neighbors algorithms, i.e. they try to retrieve a subset of users that are most similar to the target user according to some metric, and then try to produce recommendations based on the available ratings of those users.

One approach from this family consists of *Collaborative filtering* [13], where the ratings themselves are used to determine the nearest neighbors set. The idea is that users that gives similar ratings to the same items are more likely to have similar preferences. An alternative approach is to work on the items instead of the users (*item-item* vs *user-user* approaches), since an item usually can be described by a simpler profile [12].

Another widely used family of approaches are based on *matrix factorization*. The *utility matrix* of users-items ratings is factorized into two or more smaller matrices, such that their product approximates to the original matrix (e.g. it minimizes the reconstruction error). Such family includes *Singular Value Decomposition* (SVD) [4], *SVD++* [7], *NMF* [14] and many others [2, 6, 11]. SVD is a factorization techniques that minimizes the sum of squared errors of the reconstructed matrix w.r.t to the original one. It is often applied to reduce the dimensionality of data, but in the context of recommender systems it can be used to predict the missing ratings in a utility matrix by just computing the dot product between the factorized user and the item matrices.

These approaches have been demonstrated to be able to obtain satisfactory results. However, they tend to exhibit a set of common issues. For instance, they all suffer from the *cold start* problem: with these methods it is not possible to produce recommendations for a completely new user or item, since no rating to exploit is available. A possible solution is to use a *demographic recommender*, i.e. a nearest neighbor recommender which uses the demographic information (e.g. age, country of origin, ...) to compare users by similarity. Therefore, similar users are not determined by their behaviour, but by their demographic data.

All of the previous mentioned methods have their strengths and weaknesses, and in general there is not a single approach that can be considered better than the others, since their performances strictly depends on the context in which they are applied. For this reason, a common approach is to combine different algorithms in

what is called an *hybrid recommender system* in order to exploit the best method for each situation. There are many ways to combine different systems, for instance:

- *Weighted average*: the predictions from all the models are combined using an average with static weighting.
- *Switching*: a switching component selects a single recommendation system according to a pre-define policy.
- *Feature combination*: features from additional recommenders are combined and given in input to a main recommender (e.g using collaborative filtering features in a content-based system).
- *Mixed*: multiple candidate lists are first produced separately from each model, which are then recombined into the final recommendations (e.g. using *rank aggregation*).
- *Cascade*: a model hierarchy is defined, such that the top recommenders produce the main recommendations, while the secondary models are used as fallback or for minor corrections.

3 PROBLEM STATEMENT

The problem examined in this work can be defined formally as follows: given a set of patients P, where each patient p can be affected by a set of conditions C_p, and a set of therapies T_c that where tested to cure each condition c, the goal is to recommend to the patient a new therapy t which is expected to be the most effective in curing a particular condition. Each therapy trial is characterized by a "success" score, that represents how effective the therapy was in curing a certain condition for a specific patient. Therefore the goal can be also expressed as finding the therapy that maximizes the expected success rate.

In this setting, the recommendation system proposed in this work can be represented as a function f_θ:

$$f_\theta : P \times C \times T \rightarrow R \tag{1}$$

where θ is a set of hyperparameters, and R is the set of possible therapy success scores (in this case $R = [0, 100]$). The goal therefore becomes finding, for a given patient p and condition c, the therapy $\hat{t}$ that maximizes the expected success rate, i.e.:

$$\hat{t} = \underset{t}{\operatorname{argmax}}\, f_\theta(p, c, t) \tag{2}$$

In order to do so, the predicted success scores should be as close as possible to the real value that would be measured if the therapy would be actually tested on the patient. Therefore, the objective becomes finding the configuration $\hat{\theta}$ that minimizes the prediction error ϵ, computed as the sum of squared errors (SSE):

$$\epsilon(f_\theta, u) = \sum_{(p,c,t,r)\in u} (f_\theta(p, c, t) - r)^2 \tag{3}$$

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, \epsilon(f_\theta, u) \tag{4}$$

where u is the set of measured success scores of the therapies trials and r is the score value.

4 DATASET

The dataset used in this work consists of a set of patients and medical conditions. Each patient is characterized by a set of attributes describing its profile and by a set of medical conditions that affect them. Each condition is instead characterized by a *diagnosis time* a *cured time* (if the condition has been cured), and by a set of therapy trials that were applied to cure the patient, each characterized by a *starting date*, an *end date*, and a *success rating*, depicting how successful the trial was in curing the condition. A condition may have no trial associated, meaning that no attempts to cure it has been made yet. Table 1 contains an overview of the attributes available for each entity.

Table 1: The entities with their properties available in the dataset. Only the relevant attributes are reported.

Entity	Property	Example
patient	id	*1*
	name	*Thomas Oswalt*
	gender	*male*
	age	*57*
	blood group	*O+*
	country	*Costa Rica*
	occupation	*teacher*
condition	id	*pc1*
	kind	*Cond01*
	diagnosed	*2008-06-09*
	cured	*2008-10-19*
trial	id	*tr1*
	condition	*pc1*
	therapy	*Th49*
	start	*2008-07-21*
	end	*2008-06-14*
	successful	*86*

The dataset used in the initial experiments was generated randomly using realistic data. Two sets of 322 medical and 204 medical were scraped from multiple sources[1,2], and used to create a realistic dataset of 50 003 patients, affected by a total of 375 252 medical conditions with 1 688 690 tested therapies.

A second dataset, obtained from merging randomly generated datasets from different sources, was also used for the final evaluation over a set of test cases. This dataset is instead comprised of 100 000 unique patients, 540 472 conditions and 939 967 medical trials.

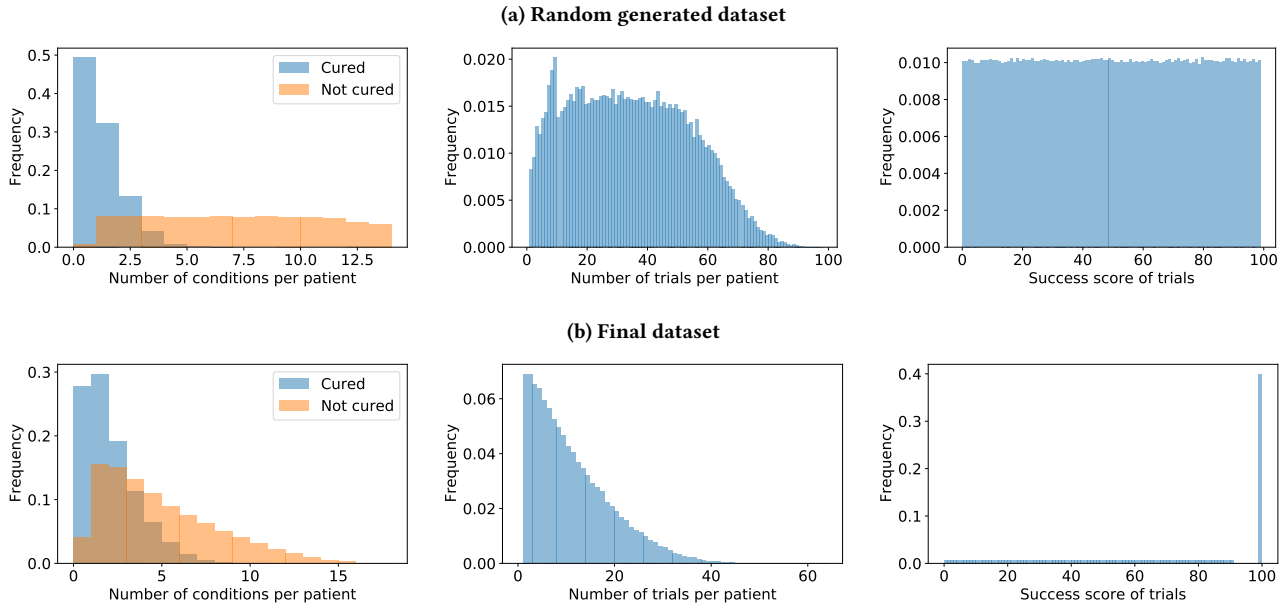
In the following analyses, the first dataset will be referred to as *random*, while the second as *final*. Figure 1 shows an histogram of the number of conditions and trials per patient, and the distribution of the successful rates of the trials for the two datasets. As it can be seen in the plots, the two datasets present different data distributions, that made it possible to evaluate the proposed methods on diverse settings and assumptions.

5 SOLUTION

The solution proposed in this work is an *hybrid recommender system*, comprised by a combination of *collaborative filtering*, *nearest neighbors*, *demographic* and *matrix factorization* systems. Each component was designed to focus on a different kind of relationship between entities, to exploit the available information as much a as possible. It must be noted that a widely used family of approaches, i.e. *content-based* systems, was not considered in this work, since due to the limitedness of attributes (see Table 1) it was not possible to build meaningful items profiles (i.e. conditions profiles).

[1] https://www.nhsinform.scot/illnesses-and-conditions/a-to-z
[2] https://en.wikipedia.org/wiki/List_of_therapies

Figure 1: Histograms of conditions and therapies per patient, and distribution of the success ratings of the therapies trials.



This section will describe why these methods were chosen, their strengths and weaknesses, and how they were adapted to the healthcare domain.

5.1 Collaborative Filtering (CF)

The first approach implemented is based on *collaborative filtering*. As explained in Section 2, in collaborative filtering the ratings given by the target user are used to determine a set of most similar neighbors, and the ratings of these neighbors are then used to estimate the possible ratings of the target user on new items.

In this work, the users are represented by the patients conditions, the items are the possible therapies to apply and the ratings are represented by the success score of the medical trials. The idea behind this approach is to try to exploit the similarities between conditions that respond similarly to the same set of therapies. At the same time, conditions of the same type that afflicts different patients are considered independent entities, since any other existing medical condition on each patient may greatly affect what treatment is applied and its effectiveness.

The utility matrix u has therefore shape $|P||C| \times |T|$, where $|P| \cdot |C|$ represents the size of the patients' conditions set, T is the set of possible therapies, and each element can be an integer in $[0, 100]$. The *Pearson correlation* (or *centered cosine similarity*) was used as similarity function, since it is more robust to skewed success scores.

The assumption behind using the patients' conditions as users instead of the actual patients is that a therapy might be effective in curing a certain type condition, but not another. Therefore with this approach the relationship between conditions that are treated and cured similarly might be better exploited.

The aforementioned approach is also called *user-user* collaborative filtering, since the similarities are computed over the users (i.e. the conditions). As presented in Section 2, another popular approach is the so called *item-item* collaborative filtering, where the similarities are instead computed over the items (i.e. the therapies), which are in principle more simpler to categorize than the conditions. For this reason, the item-item approach was also implemented in the final system.

Once the set of most similar conditions (or therapies in case of item-item) N has been determined using the similarity function, the known success scores of the tried therapies can be used to estimate the success scores of the untested therapies for the target condition. To do so, the ratings of the neighbors are weighted by their similarity to the target condition and averaged, as shown in Eq. 5:

$$\hat{r}_{ct} = \frac{\sum_{k \in N} \text{sim}(k, c) \cdot r_{kt}}{\sum_{k \in N} \text{sim}(k, c)} \tag{5}$$

To further improve the results, a *global baseline estimate* was computed and used to account for any deviation in the therapies effectiveness and correct the predictions accordingly (for instance caused by a patient being more healthy and therefore more responsive to any therapy on average). The equation for computing the ratings prediction therefore become:

$$\hat{r}_{ct} = b_{ct} + \frac{\sum_{k \in N} \text{sim}(k, c) \cdot (r_{kt} - b_{kt})}{\sum_{k \in N} \text{sim}(k, c)} \tag{6}$$

where b_{ct} is the baseline estimate of condition c over the success of therapy t, computed as:

$$b_{ct} = \mu + (\overline{u}_c - \mu) + (\overline{u}_t - \mu) \tag{7}$$

where μ is the global average scores, and $\overline{u}_c$ and $\overline{u}_t$ are the average of the known success scores of condition c and therapy t respectively.

5.2 Nearest Neighbors (NN)

The previous presented collaborative filtering approaches are rather simple but proved to be already effective in producing satisfactory results, as shown in Section 7. However, they focus only the therapies success scores, disregarding a lot of additional information that might contain useful patterns.

To solve this, three other simple methods, all based on a *nearest neighbors* approach, were included in the final system to take into account other properties, in particular the global medical condition of a patient, its demographic profile and the order of the therapies tried to cure a particular condition.

5.2.1 Global medical condition

The approaches presented in Section 5.1 are effective in exploiting the relationship between similar conditions and therapies, but since each condition of a patient is treated as a independent entity, the relationship between different conditions of the same patient are completely disregarded. The assumption made is that this might not be optimal, since in presence of multiple medical conditions, past or present, complex interactions might arise, altering the effectiveness of the therapies.

To take this factor into account, for each patient a simple profile was built based on its medical conditions. This profile is represented by a boolean vector $\vec{pc}$, with size $|C|$ (i.e. number of possible medical conditions), and with $pc_i = 1$ if the patient is affected by condition i, or 0 otherwise. The set of N most similar patients to the target can be then computed using the *Jaccard* similarity, to find the patients affected by a similar combination of medical conditions. Since the goal is to compute the therapy which is expected to be more effective in curing a certain condition, the set N is computed using only the patients that are affected by the same condition type of the target one. From this set the final success scores predictions are computed using the same weighted average approach presented in Eq. 6, using the success scores of the therapies tested to cure the same type of conditions as the target one. This approach also solves partially the *cold start problem* that affects collaborative filtering, since it can be applied even if no therapies have been tried to cure the target condition. However, if the target patient has no registered medical condition, this approach is still not applicable and additional methods are needed to cover this case, e.g. the *demographic profile* approach discussed in the next section.

5.2.2 Demographic profile

Collaborative filtering presents some disadvantages, in particular it suffers from the *cold start* problem, i.e. if no therapy has been tested to cure a condition yet, or if the patient has no other registered medical conditions, these methods cannot be applied since there are no ratings to compute the similarity on. In the dataset used by this work, as much as 10.35% of patients have no previous registered medical conditions and 19.96% of medical conditions are untreated. Moreover, these approaches do not work well when the matrix is highly sparse, since it is difficult to find conditions that were cured with a similar set of therapies, and in the available dataset the utility matrix has an high sparsity of 95.83% (i.e. fraction of missing values).

For these reasons, an approach based on a similarity metric that does not rely on having any existing medical conditions or trials might be effective when producing recommendations. Therefore, a *demographic* recommender system was included in the pipeline. This approach is very similar to the one presented in the previous section, with the only difference on how the patients profiles are computed. Instead of using a vectorized representation of the set of medical conditions (which might be empty, as explained before), the profile is built using a set of demographic information which characterizes each patient. The assumption is that patients with a similar profile will respond similarly to the same therapies for a certain medical condition.

Each patient profile is constructed with the data shown in Table 1. To have only discrete value in the profiles, the "age" property is mapped into 6 age groups. The similarity function used in this case to compute the nearest neighbors set N is the *Hamming* similarity, which treat each property as a discrete value and simply computes the fraction of values that are equal. As for the previous approach, N is computed only over the patients affected by the same kind of condition as the target one, and the final predictions are computed with Eq. 6.

5.2.3 Therapies trials order

Another important information left out from the previous approaches is the order of the tried medical therapies. As shown in Section 4 each therapy trial is characterized by a start and end date. This information might be relevant in predicting the success rate of a therapy, since its effectiveness might be affected by the order in which the previous therapies were applied. Moreover, it might be possible that to cure two different conditions similar sets of therapies are used but with a different order, which might be fundamental for a successful outcome. This is not taken into account by any of the previous approaches, therefore another nearest neighbors method was introduced to account for this.

The proposed approach is the same as the one presented in Section 5.1, with the only difference on how the similarity between conditions is computed. In this case, a profile for each condition of each patient is built using the ordered sequence of therapies that were tried to cure it. To determine the nearest neighbors set N, the *Levenshtein* distance is computed over the built sequences t_i and t_j, and then converted into a similarity metric as:

$$\text{sim}(t_i, t_j) = \max(|t_i|, |t_j|) - \text{dist}(t_i, t_j) \tag{8}$$

The therapies success scores for the target condition are then computed using Eq. 6, as for the previous methods. Additional tricks were later implemented to reduce the computational complexity o this approach, as discussed in Section 7.2.

5.3 Latent Factor Models

Another widely used approach for recommendation systems is based on *Matrix Factorization* (MF). The same utility matrix u already presented in Section 5.1 is factorized into two smaller matrices, P and Q, such that their product minimizes the sum of squared errors on the known ratings w.r.t. the original matrix. Once P and Q have been computed, their product can be used to estimate the values of the missing ratings, as:

$$\underset{C\times T}{\hat{R}} = \underset{C\times l}{P} \cdot \underset{T\times l}{Q^T} \tag{9}$$

where l is the *latent space* size, a pre-defined hyperparameter.

In this work two approaches based on matrix factorization where used: *SVD* [4] (also known as *Funk SVD*) and *SVD++* [7], already briefly presented in Section 2. Alg. 1 provides the pseudocode of the algorithm used to compute the factorization into the P and Q matrices using *Stochastic Gradient Descent* (SGD).

SVD++ is an improvement over SVD. It takes into account also the global effects of the users and items ratings to normalize the predictions using a global baseline estimate, as seen in Section 5.1. Note that with this approach the global baselines are estimated during the optimization process, instead of computing them directly. The estimated success score $\hat{r}_{ct}$ for a therapy t in

Algorithm 1 SVD algorithm for matrix factorization.

Input:
E: total number of epochs;
R: list of (condition, therapy, successfull score) tuples;
f: latent space size;
lr: learning rate;
λ: regularization parameter;
Initialize:
$\mathbf{P}, \mathbf{Q} \leftarrow \mathcal{N}(0, 0.1)$
for $1 : \mathrm{E}$ **do**
  for $c, t, r_{ct} \leftarrow R$ **do**
    $\vec{p}_c \leftarrow \mathbf{P}[c]$
    $\vec{q}_t \leftarrow \mathbf{Q}[t]$
    $err \leftarrow r_{ct} - (\vec{p}_c \cdot \vec{q}_t)$
    for $k \leftarrow 1 : f$ **do**
      $p_{ck} \leftarrow p_{ck} + lr \cdot (q_{tk} \cdot err - \lambda p_{ck})$
      $q_{tk} \leftarrow q_{tk} + lr \cdot (p_{ck} \cdot err - \lambda q_{tk})$
    end for
  end for
end for

curing a condition c can be then computed as:

$$\hat{r}_{ct} = \mu + b_c + b_t + \sum_{k=1}^{f} p_{ck} \cdot q_{tk} \tag{10}$$

As for SVD, the optimization problem is solved with SGD, minimizing the SSE.

5.4 Hybrid

The methods presented in the previous sections all have their weaknesses and strengths, therefore using them togheter can be a successful strategy to increase accuracy and robustness of the predictions [3, 5, 10, 15]. In this work two approaches were tested to combine the previous algorithms: an *average-based* and a *cascade* method.

In the average method, the predicted success scores from the different recommender systems are averaged to obtain the final predictions. In this case, if one method is not able to make a prediction for a particular therapy (e.g. when using collaborative filtering on untreated conditions), it is simply ignored.

The cascade approach is based on a predefined priority list over the recommender systems. The model first tries to generate a prediction using the method with the highest priority. If it fails, it tries with the method with the second highest priority, and so on, until it reaches the lowest priority method, which should be guaranteed to produce a prediction. In general the highest priority should be assigned to the most accurate and robust methods. It is also possible to assign the same priority to multiple methods, in which case the average of the predictions is used. Different priority assignments were tested, and the final one used for the final evaluation has the following order (from highest to lowest priority):

(1) *SVD* and *SVD++ MF*
(2) *Item-item CF*
(3) *User-user CF* and *trials-sequence NN*
(4) *Conditions-profile NN*
(5) *Demographic NN*

Note how the *demographic* approach has the lowest priority, since no pre-existing conditions or therapies are needed to produce recommendations, making this approach always applicable.

6 IMPLEMENTATION

The final system was implemented in Python 3.9, with a large use of the numpy and pandas libraries. The algorithms were implemented by exploiting numpy vectorized functions and parallelization to maximize performances. The numba library was used in some computational heavy parts of the system to further increase performances, in particular for the matrix factorization algorithm. Moreover, for matrix factorization the utility matrix is converted to a sparse format, to optimize memory consumption and increase the factorization performances.

7 EXPERIMENTAL EVALUATION

To understand how the different proposed algorithms performed and to tune the hyperparameters of the system, different evaluation experiments were executed. This section will present the results obtained regarding both the quality of the predictions and the computational requirements of the final system.

7.1 Prediction quality

All the evaluations presented in this section were conducted over the randomly generated dataset (see Section 4), using 80% of the data for fitting the algorithms and 20% for validation. In particular, the split was performed over the applied therapies, so that the algorithm could be evaluated by predicting the success rate of the therapies in the validation split. To avoid any prediction bias, the split was done only over a subset of patients and conditions, so that the train split would still contain the full original data for most of the patients. Moreover, when splitting the trials history of a patient, the order was preserved, so that the algorithm could try to predict the next therapies considering all and only the previous ones, without creating any gap in the medical history. For all the experiments the same split was used for fitting and validation.

To understand whether a method was performing better than another one, different metrics were computed. In the following experiments these metrics are reported w.r.t to predicted therapies success scores:

- $RMSE = \sqrt{\frac{1}{|\tilde{T}|} \sum_{t \in T} (\hat{r}_t - r_t)^2}$
- $MAE = \frac{1}{|\tilde{T}|} \sum_{t \in T} |\hat{r}_t - r_t|$

where $\tilde{T}$ is the set of therapies trials in the validation set. *RMSE* is a standard metric when evaluating recommender systems, since it gives more weight to predictions with large residuals. The *Mean Absolute Error* (*MAE*) is also reported, since it provides a more human-readable metric as it can be directly interpreted as the average prediction error.

To understand how the different methods performed and to obtain a performance baseline, each algorithm was first tested separately, without combining the predictions with other methods. The results can be found in Table 2. Two basic approaches are also included as baselines, which use as predictions either randomly generated values or the mean success score of the therapies used to cure the target condition.

Table 2: Baseline performances of each method run separately.

Method	RMSE	MAE
Random baseline	55.81	47.82
Mean baseline	45.28	38.73
CF user-user[5.1]	31.64	20.61
CF item-item[5.1]	29.75	18.58
NN conditions-profile [5.2.1]	36.16	25.27
NN demographic[5.2.2]	37.50	27.88
NN trials-sequence[5.2.3]	33.12	21.52
MF SVD[5.3]	27.10	16.47
MF SVD++[5.3]	26.91	15.62

Looking at the performances, it is clear that matrix factorization methods obtain the best performances, followed by collaborative filtering. The nearest neighbors approaches seem to perform worse in general, with the demographic one obtaining the lowest performances. This can be a sign that the data used for the similarity functions does not provide major benefit w.r.t collaborative filtering. For instance, the demographic approach is very basic and therefore we should expect lower results, but it is still a useful method to include in the system as fallback. In any case, all the presented methods consistently perform better than the two baselines, meaning that there is an underlying pattern being exploited.

After this initial experiment to obtain a set of baseline performances, the actual hybrid recommender system was tested. First, the two approaches presented in Section 5.4 were compared and the results can be found in Table 3. The *average* approach seems to perform better than the *cascade* one, presumably because combining the scores of multiple methods is a better strategy than just using a chain of fallback systems, since if the method with the highest priority produces a valid prediction the other methods are completely ignored.

Table 3: Performances obtained with different hybrid recommender methods.

Hybrid approach	RMSE	MAE
average	23.26	13.87
cascade	27.83	15.91

An ablation study was then executed to better understand the contribution of each method to the global performances of the hybrid system. Each recommender system was excluded from the system one-by-one and the predictions of the other remaining methods were combined with the *average* approach. The results are reported in Table 4. As expected, the methods providing the most relevant contributions are the ones that also obtained the best performances in Table 2.

Finally, the system was run over a small test set, for which the 5 most relevant therapies were recommended for each condition. The test set comprises 10 and 3 conditions for the *final* and the *random* dataset respectively. For this experiment no ground truth was available, so only the produced recommendations are reported in Table 5, ordered from most to least highly advised.

7.2 Computational complexity

In order to be usable in a real setting, the system should be able to scale well w.r.t. the size of the data used to fit the recommenders.

Table 4: Ablation study over the different components of the hybrid recommender system.

Excluded method	RMSE	MAE
CF user-user[5.1]	26.74	16.53
CF item-item[5.1]	28.52	17.37
NN conditions-profile [5.2.1]	24.13	13.98
NN demographic[5.2.2]	25.87	14.38
NN trials-sequence[5.2.3]	26.75	15.85
MF SVD[5.3]	30.41	19.94
MF SVD++[5.3]	32.88	20.79
Full	23.26	13.87

Table 5: Therapy recommendations computed over the two test sets.

(a) Final dataset

Patient	Condition	Recommended therapies				
6	pc32	Th26	Th6	Th35	Th40	Th49
51345	pc277636	Th41	Th18	Th17	Th27	Th32
82486	pc445475	Th33	Th24	Th28	Th23	Th26
51348	pc277652	Th1	Th10	Th49	Th24	Th9
51358	pc277696	Th45	Th31	Th29	Th27	Th23
51362	pc277711	Th48	Th17	Th43	Th1	Th21
51366	pc277723	Th51	Th14	Th43	Th6	Th8
51387	pc277825	Th49	Th10	Th14	Th43	Th16
51416	pc277986	Th2	Th14	Th31	Th30	Th10
51453	pc278191	Th51	Th44	Th10	Th24	Th32

(b) Random generated dataset

Patient	Condition	Recommended therapies				
50000	pc375245	Th169	Th198	Th41	Th106	Th42
50001	pc375248	Th153	Th8	Th183	Th40	Th172
50002	pc375251	Th155	Th109	Th66	Th141	Th89

As already explained, the final system is a combination of seven different methods, each with different requirements in terms of computational complexity. Each algorithm can be characterized by the complexity of the initial *fit* and the *prediction* steps. An overview of the computational complexity of each method is given in Table 6.

Table 6: Summary of the computational complexity of each method.

	Complexity	
Method	Fit	Predict
CF user-user	$O(1)$	$O(\|C\|)$
CF item-item	$O(1)$	$O(\|C\| \cdot \|T\|)$
NN conditions-profile	$O(1)$	$O(\|C\|)$
NN demographic	$O(1)$	$O(\|C\|)$
NN trials-sequence	$O(1)$	$O(\|C\| \cdot T_{max}^2)$
MF SVD	$O(E \cdot \|R\| \cdot f)$	$O(1)$
MF SVD++	$O(E \cdot \|R\| \cdot f)$	$O(1)$

The user-user collaborative filtering and the nearest neighbors approaches (see Sections 5.1 and 5.2) do not require any particular pre-processing in the fit step, which therefore has constant complexity. However, to generate a new prediction the set of most similar neighbors N must be computed first. Therefore, the computational cost of the predict step is $O(|C|) \cdot O(\text{sim})$,

where C is the set of patients' conditions and $O(\text{sim})$ is the computational cost of the similarity function used. In most cases, $O(\text{sim}) = O(1)$ since the cost of the similarity function is linear w.r.t. the number of variables involved, which is constant. For instance, that is the case with collaborative filtering, where the Pearson correlation is computed over the therapies success scores, and where the set of possible therapies is fixed.

The only exception is given by the *trials-sequence* approach since it uses the Levenshtein similarity, which has a complexity of $O\left(|t_{c_i}| \cdot |t_{c_j}|\right)$, where t_{c_i} is the sequence of therapies tested to cure condition c_i. Given an upper bound t_{max} to the number of trials a condition can have, the computational complexity becomes $O\left(|C| \cdot t_{max}^2\right)$. Considering that the given dataset already has $t_{max} = 20$, this factor can become an issue when scaling the system to bigger datasets. As a solution, an additional filtering based on the *Jaccard* similarity was implemented. In particular, first a subset of $10 \cdot |N|$ nearest neighbors is determined using the Jaccard similarity, then the final set N is computed using the Levenshtein similarity over this smaller filtered set.

The item-item collaborative filtering has instead of a prediction complexity of $O(|C| \cdot |T|)$, where T is the set of all possible therapy types. The prediction for each therapy must be computed separately, and to compute each prediction the set of most similar therapies must be first determined. This can be partially improved by limiting the number of therapies for which the predictions are actually computed. To do so, only the therapies that are most likely to be recommended are considered. This is determined by looking at the success scores predicted by the other recommenders in the hybrid system. By doing this, the complexity becomes $O(|C|)$ if the predictions are computed for a small enough number of candidate therapies (i.e. $\ll |T|$). All the computational performances of the aforementioned methods could be further improved with techniques such as *locality-sensitive hashing*, to optimize the computation of the nearest neighbors set.

The matrix factorization methods (SVD and SVD++) instead needs a pre-processing step to factorize the utility matrix into the P and Q matrices. Algorithm 1 has a complexity of $O(E \cdot |R| \cdot f)$, where E is the total number of epochs for which the optimization is run, $|R|$ is the number of success trials scores in the dataset, and f is the size of the latent space. After P and Q have been computed, all the missing scores can be predicted at the same time by computing $P \cdot Q^T$. Therefore, each prediction step requires constant time to just retrieve the prediction from the resulting matrix. It must be noted that if we want to produce a prediction for a new patient or if new data becomes available, the fit step must be run again from scratch, making this approach very computationally expensive if the data is updated frequently.

8 CONCLUSIONS

The hybrid recommender system proposed in this work proved to be a viable solution to the problem of medical therapy recommendation. The final model was able to produce satisfactory predictions by exploiting the correlation between different types of information.

However, while the system was validated on a large dataset, it still needs further testing on real data, since the underlying patterns exploited by the model may be quite different than the ones present in the synthetic dataset used for evaluation.

Moreover, further improvements could be introduced, for instance by including other recommender systems to improve the general performances or introducing techniques for improving the computational performances, like locality-sensitive hashing.

REFERENCES

[1] Bhimavarapu, U., Chintalapudi, N., and Battineni, G. A fair and safe usage drug recommendation system in medical emergencies by a stacked ann. *Algorithms 15*, 6 (2022).

[2] Bi, X., Qu, A., Wang, J., and Shen, X. A group-specific recommender system. *Journal of the American Statistical Association 112*, 519 (2017), 1344–1353.

[3] Çano, E., and Morisio, M. Hybrid recommender systems: A systematic literature review. *CoRR abs/1901.03888* (2019).

[4] Funk, S. Netflix update: Try this at home, 2006.

[5] Han, Q., Ji, M., de Troya, I. M. d. R., Gaur, M., and Zejnilovic, L. A hybrid recommender system for patient-doctor matchmaking in primary care, 2018.

[6] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T.-S. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web* (Republic and Canton of Geneva, CHE, 2017), WWW '17, International World Wide Web Conferences Steering Committee, p. 173–182.

[7] Koren, Y. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data 4*, 1 (jan 2010).

[8] Mahmoud, N., and Elbeh, H. Irs-t2d: Individualize recommendation system for type2 diabetes medication based on ontology and swrl. In *INFOS '16* (2016).

[9] Narducci, F., Musto, C., Polignano, M., de Gemmis, M., Lops, P., and Semeraro, G. A recommender system for connecting patients to the right doctors in the healthnet social network. pp. 81–82.

[10] Peito, J., and Han, Q. Incorporating domain knowledge into health recommender systems using hyperbolic embeddings. In *Studies in Computational Intelligence*. Springer International Publishing, 2021, pp. 130–141.

[11] Pu, L., and Faltings, B. Understanding and improving relational matrix factorization in recommender systems. In *Proceedings of the 7th ACM Conference on Recommender Systems* (New York, NY, USA, 2013), RecSys '13, Association for Computing Machinery, p. 41–48.

[12] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web* (New York, NY, USA, 2001), WWW '01, Association for Computing Machinery, p. 285–295.

[13] Schafer, J. B., Frankowski, D., Herlocker, J., and Sen, S. Collaborative filtering recommender systems. In *The Adaptive Web: Methods and Strategies of Web Personalization* (Berlin, Heidelberg, 2007), P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds., Springer Berlin Heidelberg, pp. 291–324.

[14] Sra, S., and Dhillon, I. Generalized nonnegative matrix approximations with bregman divergences. In *Advances in Neural Information Processing Systems* (2005), Y. Weiss, B. Schölkopf, and J. Platt, Eds., vol. 18, MIT Press.

[15] Tran, T. N. T., Felfernig, A., Trattner, C., and Holzinger, A. Recommender systems in the healthcare domain: state-of-the-art and research issues. *Journal of Intelligent Information Systems 57* (08 2021), 1–31.

[16] Waqar, M., Majeed, N., Dawood, H., Daud, A., and Aljohani, N. R. An adaptive doctor-recommender system. *Behaviour & Information Technology 38*, 9 (2019), 959–973.