A Recommender System for Telecom Users: Experimental Evaluation of Recommendation Algorithms

A Recommender System for Telecom Users

Paolo Falcarin, Antonio Vetrò, Jian Yu

Software Engineering Research Group, Dept. of Control and Computer Engineering,
Politecnico di Torino, 10129, Torino, Italy
{paolo.falcarin, antonio.vetro}@polito.it

Abstract. The increasing flourish of available services in telecom domain offers more choices to the end user. On the other hand, such wide offer cannot be completely evaluated by the user, and some services may pass unobserved even if useful. To face this issue, the usage of recommendation systems in telecom domain is growing, to directly notify the user about the presence of services which may meet user interests. Recommendation can be seen as an advanced form of personalization, because user preferences are used to predict the interests of users for a new service. In this paper we propose a recommender system for users of telecom services, based on different collaborative filtering algorithms applied to a complex data-set of telecom users. Experiments on the recommendation performance and accuracy are conducted to test the different effects of different algorithms in data set coming from the a telecom domain.

Keywords. Recommendation, collaborative filtering, telecom service, recommendation performance.

1. Introduction

Just as on the Internet, there are more and more available services in the telecom domain. Third-party service providers can easily offer their services to end users. Furthermore, telecom services now can be created and provisioned by end users [1], which will result in more services being available. Given this background, one serious question that needs to be answered is how to avoid telecom users getting lost with the huge amount of available services.

Recommendation is the output of a process of analysis on a dataset of users' preferences, whose goal is to extract the most possible related or interesting items for a target user. Recommending web services is not easy because user identity is often unknown and more important the service usage history of a user cannot be easily defined. On the other hand, this problem can be mitigated in the telecom domain where the mobile phone is becoming the main access point for users and its usage can be stored and analyzed by telecom operators in a more profitable way. In fact, the wide availability of users' data in the telecom domain is a good starting point for

applying different analysis in order to suggest the more suitable services to a target user.

A recommender system for the telecom domain is proposed and implemented to combine collaborative filtering algorithms on a data set made of user preferences on different services in telecom domain: the main feature of a telecom data-set is that we have a possibly huge number of users and a relatively small set of services. Experiments on the recommendation performance and accuracy have also been performed to test the effects of different algorithms on such data set of telecom users..

The rest of this paper is organized as follows: we discuss the background and related work in Section 2, then our recommender system in telecom domain is introduced in Section 3; experiments with our recommender system are detailed in Section 4, and conclusions are drawn in Section 5.

2. Background and Related Work

2.1. The OPUCE project

Our recommender system is a part of the OPUCE platform [1][2], which aims at bridging advances in networking, communication and information technology services towards a unique service creation environment where personalized services are dynamically created and provisioned by the end users themselves.

In this context, there will be a large amount of web services and telecom services provided by third-party providers and/or end users. Then, how to recommend a minor service set which is related and interesting for a particular user is an important problem which needs to be solved, because it will ease distribution of services to the most suitable users.

Here follows a possible scenario which shows the main functionality of the recommender system in the OPUCE platform.

John comes from U.S.A and is now travelling in Rome. After visiting a sight spot, he wants to find a nearby restaurant. To do this, he logins in the OPUCE platform through his mobile phone, executes the service recommender, and then get a list of recommended restaurants which is calculated according to his context information (such as location), his preference (such as dinner time, preferred taste), and rates of similar travellers (such as travellers who made similar rates on other dinner services). From the recommendation list, John can enjoyably find his preferred restaurant.

Recommender systems have become an important research area in the last decade and there has been much work done both in the industry and academia [3]. Usually, recommender systems can be classified into three types [3]:

1) Content-based recommendations: The user will be recommended items similar to the ones the user preferred in the past;

2) Collaborative recommendations: The user will be recommended items that people with similar tastes and preferences liked in the past;

3) Hybrid approaches: These methods combine collaborative and content-based methods.

There has been much research work and successful real systems using recommendation systems [3], and recently there are some works on service recommendation [6][7]. They made good foundation for our work, but characteristics of telecom domain should be tackled to apply the recommendation in telecom domain successfully.

In telecom domain, the related work on recommendation is still not much. Ricci et al. [6] discussed a kind of mobile recommender system. Chen et al [7] present a recommendation algorithm in mobile environment, but they do not mention the corresponding architecture of the recommender system. An architecture and implementation of mobile recommender system is proposed in [8][9], but they are not focused on telecom systems, and the analysis of different effect of recommendation algorithms is also missing.

The main characteristics of our work is the evaluation of different correlation algorithms on a typical telecom data set, where there could be millions of users and relatively few services.

In the following sections, we will describe how we have applied main correlation algorithms for predicting user preferences and how such algorithms perform on a large data-set like the preferences of users of a telecom operator.

3. A Recommender System in Telecom Domain

Integration of a recommender in a telecom service platform requires an additional comprehension of recent standards in telecom domain.

There are some specifications of telecom service platform, born to combine telecom resources and IT systems, such as Java Service Logic Execution Environment (JAIN SLEE) [4] and Parlay [5]. JAIN SLEE is chosen in our work for its event-based architecture and the easiness of integrating IT services. A Service Logic Execution Environment (SLEE) is a high throughput, low latency event processing application environment used in telecommunications industry. JSLEE is the Java standard for SLEE and is designed to allow implementations of the standard to meet the stringent requirements of communications applications, such as network signalling applications. The JSLEE specification is designed so that implementations can achieve scalability and availability through clustering architectures.

Unlike enterprise applications which are usually invoked synchronously, telecom applications are always invoked asynchronously. So our architecture and

implementation is also event-based to comply to such an asynchronous communication requirement.
Our recommender system employ JAIN SLEE standard [4] and it is installed inside Mobicents platform which is the first and only Open Source Platform certified for JSLEE 1.0 compliance[1] . This choice has essentially two motivations:

1. Combination of JAIN SLEE environment and event programming approach can bring high performances and low latency;

2. Telecommunication world is intrinsically defined by asynchronous events (example: a phone call), so the event-driven approach is strongly coherent to the environment in which the application will run.

The recommendation system is realized through development of a Service Building Block (SBB). This service manages events defined ad-hoc, and generated by other SBB inside SLEE container. Furthermore, in order to provide accessibility to recommendation service by other applications contained in the platform, the system was anchored to an Activity Context, and Resource Adapters (RA) need to be implemented to utilize the outside resources (such User Information Repository ad Service Repository).

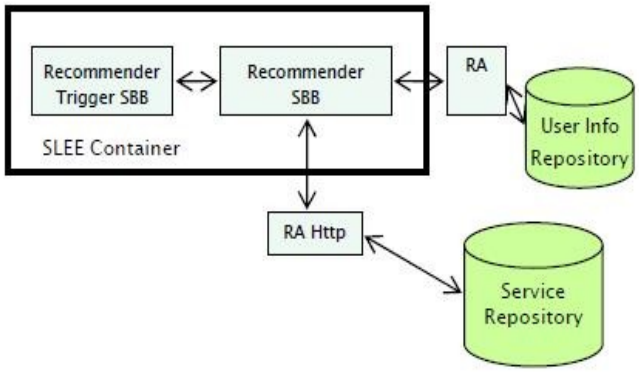

Fig. 1. An architecture of recommender system in telecom domain

Events between recommender and its trigger are defined as (shown in Fig.2): doPredictionsEvent (the event the recommender waits, and in which parameters of request are specified – like user, kind of recommendation, and so on) and ResponsePredictionEvent (prediction are calculated and then sent back to the matcher as an event).

1 Mobicents.org - The Open Source VoIP Middleware Platform https://mobicents.dev.java.net/

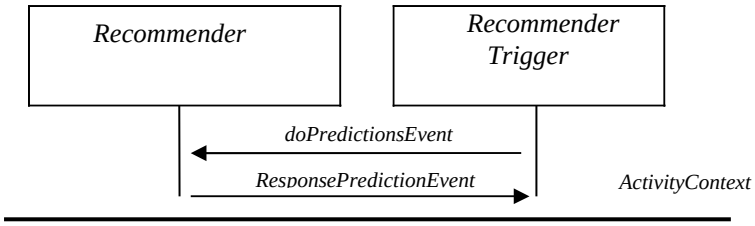

Fig. 2. Communications between recommender and its trigger

Recommender service is also able to receive a synchronous event in order to set the algorithm to be used for the prediction: this operation was thought as a synchronous one because there is the need to have a confirmation before a new prediction request.

Another point need to mention in our implementation is the reuse of a set of open-source libraries that provide a set of algorithms and data models. Libraries adopted are part of Taste project (now moved to Apache Mahout)[2].

3.1. Recommendation Algorithms

The interface we specified has different methods: the *n* highest recommendations of services for a given user, the *n* most similar users to a given user, the neighbourhood of a given users with a specified similarity threshold, and finally a special recommendation that provides a neighbourhood of most similar services to a given service.

All recommendations are computed through the use of algorithms provided by Taste libraries. Here a short list of algorithms we used (see the Taste documentation for more details):

1. *GenericItemBased*: A simple Recommender which uses a given DataModel and ItemCorrelation to produce recommendations.
2. *GenericUserBased*: A simple Recommender which uses a given DataModel and UserNeighborhood to produce recommendations.
3. *ItemAverage*: A simple recommender that always estimates preference for an Item to be the average of all known preference values for that Item. No information about Users is taken into account.

2 http://taste.sourceforge.net/ , now moved to Mahout project: http://mahout.apache.org/

4. *ItemUserAverage*: Like ItemAverageRecommender, except that estimated preferences are adjusted for the Users' average preference value. For example, say user X has not rated item Y. Item Y's average preference value is 3.5. User X's average preference value is 4.2, and the average over all preference values is 4.0. User X prefers items 0.2 higher on average, so, the estimated preference for user X, item Y is 3.5 + 0.2 = 3.7.
5. *TreeClustering*: A Recommender that clusters Users, then determines the clusters' top recommendations. This implementation builds clusters by repeatedly merging clusters until only a certain number remain, meaning that each cluster is sort of a tree of other clusters.
6. *SlopeOne*: A basic "slope one" recommender. This Recommender is especially suitable when user preferences are updating frequently as it can incorporate this information without expensive re-computation.

Algorithms can use correlations (Pearson, cosine, and Spearman) [3] between both users and services (items) in order to build neighbourhoods.

4. Experiment

To know the different effects of recommendation algorithms in telecom domain, we implement the recommender system in OPUCE platform according to the above architecture and do the corresponding experiments.

4.1. Datasets

Database of preferences is hosted in MySQL data base system. The OPUCE project is not finished and we have not full access to real data on telecom. As the goal of our work is to assess if correlation algorithms may scale on a large dataset like the one of users of a telecom operator, we employed a real dataset provided by GroupLens research group[3] which can be comparable in size. Dataset consists of 2,811,983 ratings expressed inside the range 1-5, entered by 72,916 users for 1628 movies (which can simulate services in our experiment) : we resized datasets in our experiments to evaluate scalability with increasing number of users, which is the real constraint in a telecom domain.

4.2. Performance Experiment

Experiments on performances were executed with dataset of 5000, 10000, 20000, 40000 records. We perform replications with recommendations of 20 different users,

3 http://www.grouplens.org

and we proposed 2 different iterations with 250 and 2500 items (services) involved. These are results (in ms) for 5 algorithms (the Tree Clustering Recommender was not considered because it did not work with more than 5000 records):

Table 1. Recommendation time experiment results

records/items	generic item	generic user	item avrg	item user avrg	slop one
5k / 250	658.6	110.15	14.25	16.55	142.5
5k / 2500	4356.2	309.25	41.45	42.1	735.75
10k / 250	1161.6	93.1	24	24.3	234.95
10k / 2500	6957.15	216.95	51.05	53.75	837.05
20k / 250	2237.75	129.4	45.45	45.55	295.9
20k / 2500	12449.2	247.35	70.15	74.15	957.75
40k / 250	8680	427.7	172.4	176.5	974.9
40k / 2500	44975.7	557.6	198.85	202.85	1635.1

Impact of a growth of items (services) in computational time (ratio):

Table 2. Computational time experiment results with the growth of items and services

	generic item	generic user	item based	item user	slop one
servicesX10	4.03	0.56	0.52	0.53	1.18
recordsX10	9.6994695701	1.349308536	5.6651705566	5.46803069	1.971818958

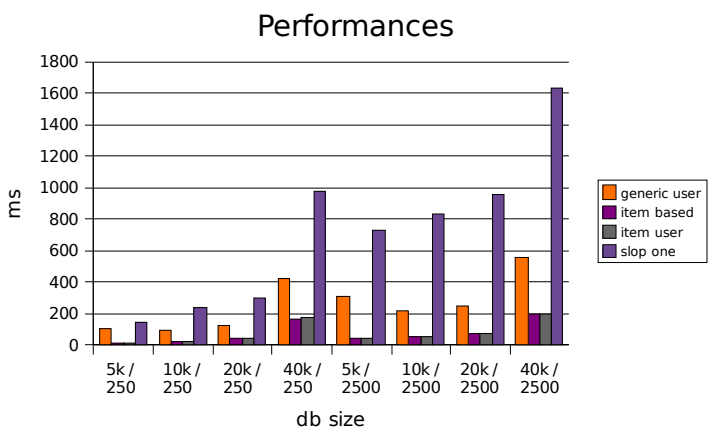

Fig. 3. User neighbourhoods calculation time experiments

We also analyzed time needed to calculate user neighbourhoods, which means the set of similar user to a given user. We analyzed how performances vary relating to the sample size (in % of the total size) and comparing the use of Pearson and Spearman correlation.

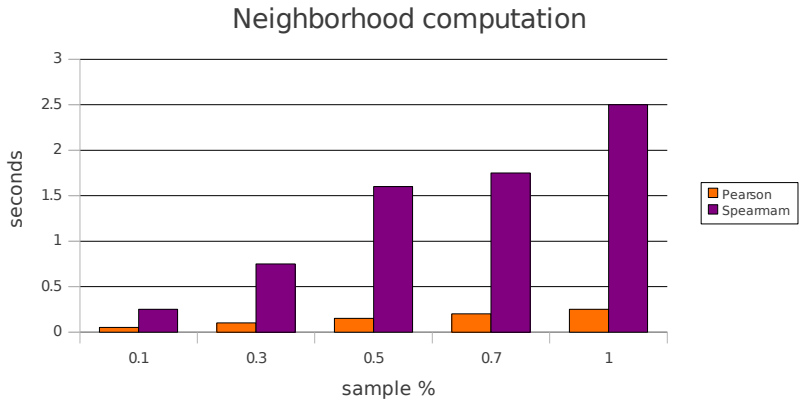

Fig. 4. Pearson and Spearman correlation calculation time experiments

Main results and considerations can be obtained from the above experiments:

- Slop one and Generic Item Based algorithms are generally the worst in performances.
- When the number of services is low (250 in the experiment), growth of records means growth of computation time. When the number of services is high (2500), growth of records means decrease of computational time.
- Impact of growth of number of services seems to be weaker than effect of growth of db size (= number of evaluations).
- Neighbourhood computation with Spearman coefficient is always worse than the computation with Pearson coefficient.

4.3. Accuracy Experiment

Taste provides a useful instrument that permits to evaluate accuracy of an algorithm. This evaluator estimates the precision of the prediction using a portion of data as training set, and the remaining part to evaluate predictions. For each evaluation predicted, if the return value is between 0 the prediction is perfect and equal to the real value, while higher return values mean higher distances.

We tested accuracy experiment with the different sizes of the database: 5000, 10000, 20000 and 80000 records.

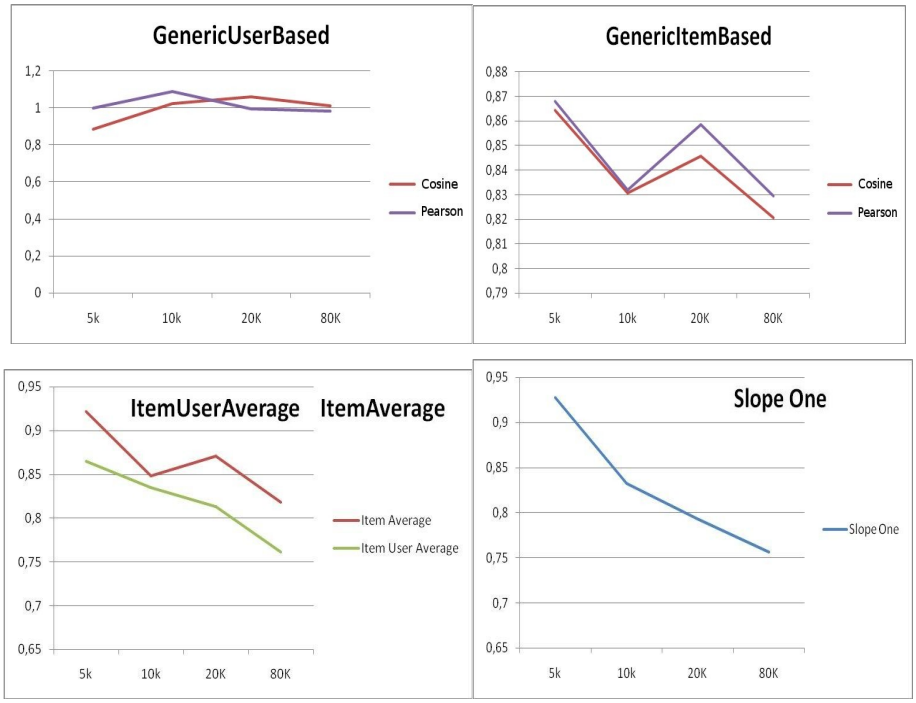

Fig. 5. Accuracy Experiment of different recommendation algorithms

4.3. Scalability Experiment

We also evaluate how the recommendation scales with respect to different amount of services.

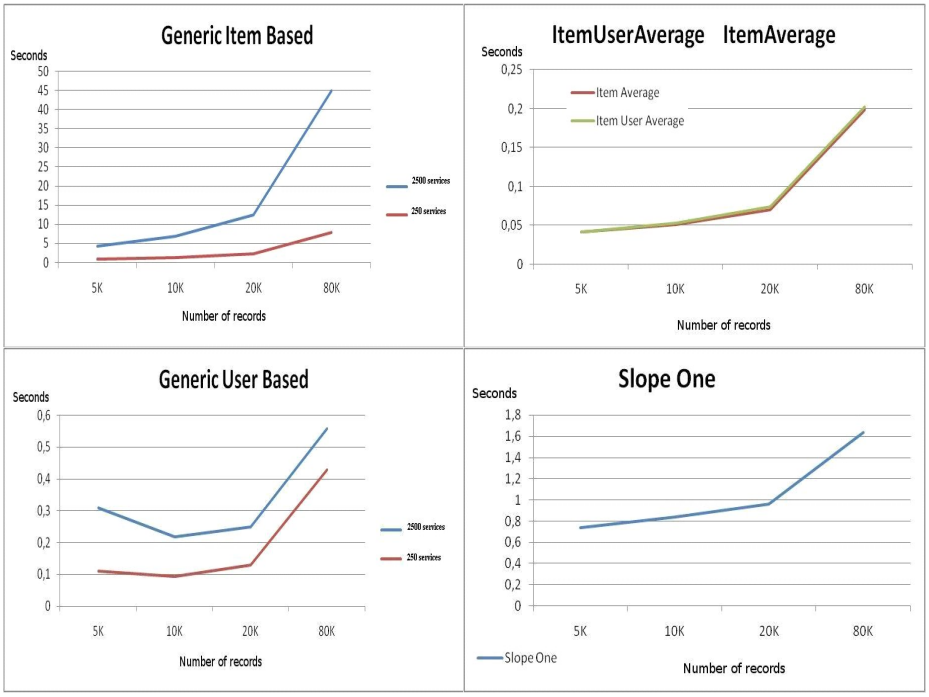

Fig. 6. Scalability Experiment of different recommendation algorithms

From the previous figure, we can know that:

- *GenericItemBased* algorithm becomes worst when items number increases;
- *GenericUserBased* algorithm shows little sensible to the increase of available items and it shows better scalability;
- *ItemAverage* and *ItemUserAverage* Algorithms show linear increment depending on number of records;
- *SlopeOne* algorithm needs processing times 8 times higher than *ItemAverage* and *ItemUserAverage* algorithms.

4.3. Experiment Results Analysis

Below what we learnt from experiments made to evaluate the recommender system.

- The user-based approach has provided bad results, both using Pearson and Spearman coefficients.
- Precisions of algorithms grow with the size of database: the bigger training set is the main reason.

- In the Generic Item Average Algorithm, the cosine correlation brings more precise evaluations.
- Inside the range 10k-20k records some algorithms (Item Average, Generic Item) decrease their performances.
- With datasets with less than 10000 records, Item User Average Recommender and Generic Item Based Algorithm are preferred, while with a greater number of records the Slope One Recommender is better.

We have enough elements to see that computation time and accuracy are not deterministic, but they vary at least in function of database sparseness and numbers of services/users. Providing more detailed tests, it is possible to define for every range of db size/sparseness an associated algorithm that will be used for the recommendation.

5. Conclusions

To realize recommendation in telecom domain, recommender system architecture is proposed to meet the characteristics of telecom systems. Experiments of different recommendation algorithms are also performed to their effect in telecom environment. For future work, we will do more reliable experiments based on user evaluations on telecom services and context aware recommendation on the basis of our architecture.

References

1. Open Platform for User-centric service Creation and Execution, OPUCE, IST FP6 Integrated Project no. IST-034101, http://www.opuce.eu
2. Yelmo. J. C., Trapero. R., et al. : User-Driven Service Lifecycle Management - Adopting Internet Paradigms in Telecom Services. In. 5th Int. Conf. on Service-Oriented Computing (ICSOC 2007), LNCS 4749. pp 342--352 (2007)
3. Adomavicius. G, and Tuzhilin. A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Trans. Knowledge and Data Eng., vol. 17, no. 6, pp. 734--749 (2005)
4. JAIN™ Service Logic and Execution Environment (SLEE): http://jainslee.org/
5. Parlay Group: http://www.parlay.org.
6. Manikrao, U. S., Prabhakar T. V.: Dynamic Selection of Web Services with Recommendation System. In: Int. Conf. on Next Generation Web Services Practices (NWeSP'05). pp. 117--121 (2005)
7. Wen Q., He, J.: Personalized Recommendation Services Based on Service-Oriented Architecture. In: 2006 IEEE Asia-Pacific Conference on Services Computing (APSCC'06) pp. 356--361 (2006)
8. Chen. A., : Context-Aware Collaborative Filtering System: Predicting the User's Preference in the Ubiquitous Computing Environment. In: 1st Int. Workshop on Location- and Context-

Awareness, (LoCA 2005), Oberpfaffenhofen, Germany, 2005, LNCS 3479 pp. 244--253 (2005)
9. Ricci. F., and Nguyen. Q. N.,: Acquiring and Revising Preferences in a Critique-Based Mobile Recommender System, IEEE INTELLIGENT SYSTEMS, MAY/JUNE 2007, pp. 22--29 (2007).
10. van Setten, M., Pokraev, S., Koolwaaij, J.: Context-Aware Recommendations the Mobile Tourist Application COMPASS, In Nejdl, W. De Bra, P. (Eds.). Adaptive Hypermedia 2004, Eindhoven, Netherlands, LNCS 3137, Springer-Verlag, pp 235--244 (2004)
11. Pokraev, S. Koolwaaij, J. et al: Service platform for rapid development and deployment of context-aware, mobile applications. In Proceedings of 2005 IEEE International Conference on Web Service (ICWS 2005) pp 639--646 (2005).