MASTER OF SCIENCE
IN ENGINEERING

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts
Western Switzerland

Master of Science HES-SO in Engineering

Major: Information and Communication Technologies

Products recommender system using social network information

Master thesis

Simone Cogno
`simone.cogno@master.hes-so.ch`

Professor

Ghorbel Hatem, *HE-ARC*
`Hatem.Ghorbel@he-arc.chg`

Punceva Magdalena, *HE-ARC*
`magdalena.punceva@he-arc.ch`

Expert

..., ...
...

HES-SO//Master, November 11, 2015, Version 1

Accepted by HES-SO//Master (Switzerland, Lausanne) on a proposal from

Ghorbel Hatem, *HE-ARC*
`Hatem.Ghorbel@he-arc.chg`

Advisor
Schroeter Nicolas,
Nicolas.Schroeter@hefr.ch

Head of MSE
Moghaddam Fariba,
fariba.moghaddam@hes-so.ch

Contents

Analysis

2.1 Related work

2.1.1 Content-based RS

Systems implementing a Content-based (CB) recommendation approach use the information about user ratings on a set of items for building a model describing their interests. The objective is to use these models for propose new items to the users.

For performing such a recommendation we have to computer *Item* and the *User profile* and compare them to retrieve items that are relevant to the user based on his taste.

In the approaches presented on [1, Jeffrey D. Ullman] they use the properties of items for construct a vector. This method is called "Vector model representation" and it is used for construct the *Item* and the *User profile*. For example we can use the category of the item, the frequency of words used in the document [2, Pasquale Lops, Marco de Gemmis and Giovanni Semeraro] or other useful information as the components of the item profile vector.

A book item can have a vector component for their different categories of books. For example, let's take "adventure", "romance", "comedy" as the set of possible classes of books. The vector of a romance and a comedy book will be {0,1,0} and {0,0,1} respectively. Clearly we can describe the items with as many components we need, at least that all vector have the same number of components.

For compare two items, we can use several similarity measures. One of the most used is the Cosine similarity but for simple use cases the Jaccard similarity can be used as well.

The cosine similarity is described in the equation 2.1. As we can see, we compute the product of two item vector A and B divided by their module. This calculation gives the result of the angle between the two vector. If the angle between the two items is small, they are very similar, otherwise is the contrary. The cosine similarity gives values between 0 and 1.

$$sim = cos(\Theta) = \frac{A \bullet B}{\|A\| \bullet \|B\|} \tag{2.1}$$

We have seen how to create the Item profile and how to calculate the similarity between two items. Now we have to construct the user profile for and compare it to items. The method presented in [1, Jeffrey D. Ullman] explain that we can build the user profile according to the previous rating that the user makes on items. The value can be a boolean for describing if the user has bought or not an item or can be an integer number representing the stars that the user gave as a rating for an item.

With these number, we can construct the utility matrix that represent the user ratings on a set of items. The blanks entry describe the situation where the user has not rated the item.

From the utility matrix, we can retrieve the list of items that a user has bought or has given them a rating. Then we can make an average of the components in the item profile for constructing the vector representing the user profile.

Let's say that 20% of items bought by the user have the category "romance". In that situation, the user profile vector will have 0.2 as the value of the component "romance".

When we have computed the user profile vector, we can now compare the user with the items since they have the same component types. The cosine similarity can be a good choice for measuring this similarity.

By performing this technique, we can easily compute the similarity between the preferences of a user compared to particular items in the system. In this way, we can finally recommend some items to the user that he don't have already rated.

The advantage of this approach is that is quite simple to implement and, if there are enough information on the dataset, the quality of the recommendation is pretty good.

The disadvantage of this approach is the complexity of the computation when we want to discover the items similar to a user. In fact for make this calculation we have to traverse all the items in the dataset and to choose the best one. If the dataset is large, this can be an expensive calculation. For avoiding this limitation some solution have been proposed on [1, Jeffrey D. Ullman]. One approach is to cluster similar items in order to have far fewer items to compare. The clusters can be recalculated systematically with a specified interval of time and not on every recommendation to a user. In this case, we make the assumption that the dataset of user ratings varies slowly. Another limit of this algorithms is that if the user doesn't have rated many items (cold start) the system cannot calculate the user interest precisely, therefore the recommendation can be of poor quality. Collaborative filtering can improve the cold start problem by focusing on the similarity between the users. With this approach, the system can recommend items to a user according to the ratings made by similar users (see section 2.1.2).

In conclusion, this approach is very useful if we want to have a RS that work good and not require a complex algorithm. It can be a good starting point to combining it with other approaches.

2.1.2 Collaborative Filtering

The content-based recommendation systems use the information about the interest of the user for recommending items that he may like. By using Collaborative filtering (CF) we don't consider only the rating of the user but also those of similar users in the systems. In this way, we can recommend items that are very different to the preference of the users, but can be interesting since similar users have rated it positively.

For measure the similarity between two users, in [1, Jeffrey D. Ullman p.320] they use the cosine similarity from the user-rating matrix, also called *utility matrix*. Two users are similar if they rated a lot of items in common, with a similar rating.

From the list of similar users, we can retrieve the items they like the most. Then we can recommend these items to the final user.

In [1, Jeffrey D. Ullman p. 322], they also make a normalization of user rating for better represent the behavior of the user. A 5-stars rating from a user that have rated items only with 4 or 5 stars is different from the same rating from another user that have rated items only with 1-2 stars. For take into account the behavior of a user we normalize the value by subtracting the average rating of the user from the value in the utility matrix. In this way, we have a positive and negative value representing low rating and high rating respectively.

An improvement of this method is to cluster similar items and users before executing the recommendation process. This method can avoid the problem of the sparse utility matrix due to the few rating that users make on items.

In general, the collaborative-filtering approach is very effective because help the people to discover new items that are not similar to their usual interests. In the other hand, this approach uses only the user ratings and don't take into account other information about the content of the items. The utility matrix can be very sparse, especially at the beginning, and the prediction can be imprecise. A hybrid approach can be useful for filing this lack by using both content-based, CF and other methods.

The paper [3, J. O'donovan and B. Smyth] explore another technique to refine the CF technique. Basically, they search to give a measure of "Trust" between users. Let's consider a user v similar to$u$ that it's take into account for the prediction for a item i. The measure of trust between user u and v is calculated by the percentage of correct prediction that user v has made for predicting some items at user u. User v make a correct prediction only if his rating on item i is similar to the average (or other aggregate function) of the ratings made by the group of users similar to u.

For the next prediction the contribution of user v are weighted according to the "trust" that the user u have on him. For example, if the user v make 20 correct prediction for user u aver 100 total contribution, the value of "trust" between u and v is 0.2.

Here the measure of confidence is calculated from implicit information. A more reliable measure of confidence can be acquired by explicit ratings that users made on other users. For example in the Amazon website, we can rate a seller from 0 to 5 stars, for denote how much trust we have on him. Using implicit information, however, can enhance the recommendation quality and can be useful for filtering some "fake" users that have made ratings in an inconsistent way. In conclusion can be a good technique for improving the recommendation quality by combining it with the regular collaborative-filtering.

2.1.3 Hybrid RS

In the Hybrid RS, the previous described Content-based and Collaborative filtering are combined to improve the recommendation quality. Often the hybrid system uses the results of CF for run CB on it. In [4, Panagiotis Symeonidis et al.] they make use some notion of Information Retrieval for including the Term Frequency Inverse Document Frequency (TFIDF) as a measure for similarity and weighting scheme in CF. This technique is used for weighting the words of documents according to their rarity. In general, the rarity of a word is proportional to their importance for compare two documents. In fact, two

[0]Amazon,`www.amazon.com`

documents that have both some rare word have a good chance to talks about the same subject. Inversely, two papers that have some very popular words in common have less chance to be written on the same topic.

This principle can also be translated for recommender systems. If, for example, two users have liked a book of the category "Fantasy", doesn't mean that they are similar. This is because the feature "Fantasy" is very popular among users and most of them have rated it (Ex. most of the peoples like Harry Potter or the Lord of the Ring). On the other hand, if two users have both liked books of a very rare category, let's say "Egyptian literature", they have more chance to be similar because only a few users are interested in this category. It is, therefore, reasonable to weighting the item features according to their rarity in the whole dataset.

In [4, Panagiotis Symeonidis et al.] the *TFIDF* is converted into Feature Frequency and Inverse User Frequency (IUF) for a resulting *FFIUF* measure. Where FF(u, f) is the number of time feature f occurs in the profile of user u. Moreover, the UF(f) is the number of users that have feature f at least once. The IUF is the inverse of this value and can be calculated with the equation 2.2.

$$IUF(f) = \log(\frac{\mid U \mid}{UF(f)}) \tag{2.2}$$

Finally the equation 2.3 calculate the weight W of feature f for user u. The algorithms that use this measure for weighting the features is named Featured-Weighted User Model (FWUM).

$$W(u, f) = FF(u, f)IUF(f) \tag{2.3}$$

The evaluations made on [4, Panagiotis Symeonidis et al.] we can see that FWUM performs better that CF and CB approach, also for a small dataset.

This method is very interesting due to its implementation of both CB and CF technique with some improvements on it. Even id adds some complexity on the construction of the algorithm it recovers this complexity in term of quality of the prediction

It is good to note that this algorithm doesn't take into account any social information. By integrating social information can possible to improve the algorithm for example for the cold-start problem.

In the next section, we will resume some article in the context of Social Recomender Systems and explore the principles the algorithms used.

2.1.4 SRS

In this section, we describe some techniques for SRS that we have explored in several articles. The principal goal in SRS is to take advantage of the social network information for enhancing the recommendation quality.

In [5, W. Carrer-Neto M.L. Hernández-Alcaraz R. Valencia-Garc ıa F. Garc ıa- Sánchez] they make the assumption that user friends have similar interest and similar behavior. This assumption has obviously some limits if we compare it on real data. Then they enhance the amount of data by combining the friend network with other similar users in the whole systems. In the final system, they also provide that the user can weight the importance of the recommendation based on the network of friends and the recommendation based on knowledge-based approach. An open-minded user can give more relevance to the friends recommendation. Otherwise, he can decrease their weight.

This technique can be combined with CB and CF approaches and has the advantage to be simple. The disadvantage is that can be imprecise when confronted with real data. The user's friends can be very different to the user himself. For example on Facebook, a user can have hundreds or even thousands of friends and is oblivious that not all of them have similar taste.

In [6][7] they take into account the differences between user and his friends. This is done by weighting the contribution of a user in the recommendation process according to the similarity between the user and his friends. The similarity measure is based on the previous ratings made by users. By taking into account the contribution of the network of friends it is possible to enhance the recommendation quality, while maintaining the simplicity of the algorithm. In fact, this method has a similar computational complexity of CF algorithms. Besides, compared to a CF approach, this can be useful for cold start problem when we have few data (ex. ratings) for a new user. By considering his friends, we can certainly increase the amount of data available.

In [7, Alexandra Olteanu et al.] they explore some social affinities that can be useful for the recommendation. They estimate the social affinity in a friends graph by using Random Walks (RWs). This technique has been used for both friend recommendations (paper [8], [9]) and item recommendation (paper [10],[11],[12]). In short, for predicting item i for user u they perform several RWs starting from u until they reach an user v that have rated item i or when performing a maximum number of steps k-max. In [11] they also take into account when user v have rated an item similar to i. The similarity between items are calculated with the Pearson Correlation Coefficient (PCC) on the user ratings, but also Vector Space Similarity (VSS) can be used as well. They call this algorithm *TrustWalker*. As presented in [11] TrustWalker has a good improvement compared to other method like CB RS or CF, especially in the cold-start situation. The downside is the complexity of the computation for a large graph, but we can limit the steps made by RWs by set k-max=6, according to the "six-degrees of separation" rule [13], without loose important information.

A different approach has been presented in [14]. They partition the user-items rating matrix according to contextual information associated with each rating (data, hour, physical position, etc.). By doing this, they increase the prediction accuracy of relevant items. They also use an additional social regularization term that is calculated based on the similarity between users. The similarity measure that they choose is the PCC, computed on the previous rating made by users combined with contextual information that are associated with each rating.

The disadvantage of this method is that our dataset doesn't have a lot of information on the context in which ratings have been made.

Until now we have explored algorithm based on explicit information on the social network as the network of friends. In **DBLP:journals/corr/abs-1112-2774** they explore some techniques for calculate the social affinity between two users based on the activity that they perform in the system. In this case will be build an *implicit* social network based on tie strength calculation and represented as a bipartite graph. This graph contains two type of node and links between them. For example, "users" and "events". Where an "event" can be any activity that two or more user have made together. They define some axioms that the measure of strength must follows, and they expose some functions that satisfy the most of them. Some of the more interesting measure function are the *Common Neighbors*, *Katz Measure* and *Random Wlk with Restarts*. But several other function can be used as well.

2.1.5 Sentiment analysis

In the previous sections, we have made the assumption that the evaluations made by the users are effectively their real thought about an item. In reality, the opinion of users can be quite different for the value of the rating. For discovering the real opinion of users we can take advantage of the text *reviews* that they made on items. The process that measure the feeling of a user text starting from a text or a sentence is called *Sentiment Analysis* and refer to the use of *Natural Language Processing.*

Four our particular use case this technique can be useful for improving the reliability of the average rating of items. In fact, the mean of all the rating can be a very significant value but sometimes don't reflect the real evaluation of the items. We can measure the opinion of a user on an item by analyzing the terms and the sentences wrote in the reviews. The resulting opinion of users on an item can be very useful for recommending quality items.

It's important to note that the branch of Natural Language Processing is very vast, and it's still under development and research. Building an algorithm for performing Sentiment Analysis on text can be a very expensive task, and it is not the purpose of this work. In our project, we want to retrieve that information only for improve the quality of our dataset and for testing the usefulness of using it on a RS. Therefore, we will use some tools (ex *textblob*[1].) that can help us to retrieve these information.

2.1.6 Evaluate a Recomander analysis

Evaluating a RS is a necessary task that have to be accomplished for testing the correct behaviour of our algorithm.

One of the criterias for evaluating an RS is to calculate the prediction quality by using one or multiple of these metrics[15, J. Bobadilla et al. cp. 4]:

- Mean Absolute Error (MAE)
- Root Mean Square (RMSE)
- Normalized Mean Average Error (NMAE)

[1]TextBlob, `https://textblob.readthedocs.org/en/dev/quickstart.html#textblobs-are-like-python-strings`

These measures are calculated by measuring the error between the real data and the predicted ones.

For evaluate recommendation quality we have some commonly used metrics.

- Precision
- Recall
- Receiver Operative Characteristics (ROC)
- F1 score

Most of these measures can be retrieved starting from the Confusion Matrix that it is determined in the recommendation process. This matrix provides some value about True Positive, True Negative, False Positive and False Negative prediction. With these measures, we can retrieve all the metrics listed above.

Other useful metrics for evaluating an RS can be the consideration of the Diversity and Novelty [15, J. Bobadilla et al. cp. 4]. Where the diversity is the measure of the recommended items presented to the user and Novelty is the measure of the promotion of less widely known (so-called long tail) items in the whole dataset [16, Saúl Vargas Sandoval].

For measuring all these metric the k-fold cross validation has been used because its advantages on low size dataset and for avoiding the over-fitting problems [15, J. Bobadilla et al. cp. 4].

2.2 Conclusion

The algorithms explored in previous sections are all useful for accomplishing the main objectives of this work. However, for the final algorithm, we have to choose only a few of these algorithms. The most interesting ones.

The purpose is also to build an algorithm that can be used with a various type of data. In our dataset, we have structured data of items, social activity information and friends relations. The algorithm has to work by taking advantages of these data but also has to consider possible unstructured data that can be present in other datasets.

For structured data we can use CB and CF or an Hybrid approach. As discussed in section 2.1.1 and 2.1.2 CF and CB takes into account only user ratings data and uniformly weighted features respectively. The Hybrid approach can combine this two information in addition to weighting the item features thanks to a principle of Information Retrieval. As resulting from our analysis, we can conclude that this algorithm is the most effective, particularly since ti takes into account a lot of information presents in our dataset. Furthermore, it can be used for structured and unstructured data. Besides, for taking advantage of social information, we can use some of the approaches discussed in section 2.1.4. As explained previously, [5, W. Carrer-Neto M.L. et al.] make too naive assumption about the similarity of friends. The approach described in [14] use, instead, contextual information for enhancing the prediction in the recommendation process. This process can be interesting but, unfortunately, we don't have a lot of such information in our dataset. As resulting from the analysis, the remaining and most useful approach, is the one described in [6][7] where they weight the contribution of friends in the recommendation process based on their similarity. In addition, we can use some techniques examined in [22, Mangesh Gupte and Tina Eliassi] for enhancing the reliability of the similarity measure between users using the tie-strength metrics.

Then we will test some tools of Sentiment Analysis for helping us to describe in a more authentic way the taste of users on items.

In the next chapter, we will explain better the details of the algorithm chosen as well as the technologies that can be used for implement it.

Acronyms

CB	Content-based	6, 8–10, 12
CF	Collaborative filtering	7–10, 12
FWUM	Featured-Weighted User Model	9
IUF	Inverse User Frequency	9
PCC	Pearson Correlation Coefficient	10
RS	Recommender Systems	i, 6–8, 10–12
RWs	Random Walks	10
SRS	Social Recommender System	i, 9
TFIDF	Term Frequency Inverse Document Frequency	8
VSS	Vector Space Similarity	10

Bibliography

[1] J. Ullman,

[2] P. Lops, M. de Gemmis, and G. Semeraro, *Recommender Systems Handbook, Chapter 3*. 2005, pp. 73–105. DOI: `10.1007/978-0-387-85820-3_3`. [Online]. Available: `http://google.ch`.

[3] J. O'donovan and B. Smyth, "Trust in recommender systems", *In: International Conference on Intelligent user Interfaces*, 167–174, 2005.

[4] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "Feature-weighted user model for recommender systems", *Conference: User Modeling 2007 and 11th International Conference*, 2007. DOI: `10.1007/978-3-540-73078-1_13`. [Online]. Available: `http://www.researchgate.net/publication/221261040_Feature-Weighted_User_Model_for_Recommender_Systems`.

[5] W. C.-N. M.L., Hernandez-Alcaraz, R. V.-G. ıa, and F. G. ıa Sanchez, "Social knowledge-based recommender system, application to the movies domain", *Expert Systems with Applications*, vol. 39, 2012. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S0957417412004952`.

[6] H. Ma, D. Zhou, C. Liu, and I. K. Michael R. Lyu, "Recommender systems with social regularization", *Journaltitle*, 2011.

[7] A. Olteanu, A.-M. Kermarrec, and K. Aberer, "Comparing the predictive capability of social and interest affinity for recommendations", in *Web Information Systems Engineering – WISE 2014*, B. B. boualem@cse.unsw.edu.au, A. B. best@cs.bu.edu, Y. M. manolopo@csd.auth.gr, A. V. avakali@csd.auth.gr, and Y. Z. yanchun.zhang@vu.edu.au, Eds., 8786 vols., ser. Lecture Notes in Computer Science. 2014, pp. 276–292, ISBN: 978-3-319-11749-2. DOI: `10.1007/978-3-319-11749-2_22`. [Online]. Available: `http://link.springer.com/chapter/10.1007%2F978-3-319-11749-2_22`.

[8] L. Backstrom and J. Leskovec, "Supervised random walks: Predicting and recommending links in social networks", *WSDM*, 2011.

[9] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks", *Journal of the American society for information science and technology*, 2007.

[10] S.-H. Yang, B. Long, N. S. A. Smola, Z. Zheng, and H. Zha, "Like like alike: Joint friendship and interest propagation in social networks", *WWW*, 2011.

[11] M. Jamali and M. Ester, "Trustwalker: A random walk model for combining trust-based and item-based recommendation", *KDD*, 2009.

[12] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation", *IEEE Trans. on Knowl. and Data Eng.*, 2007.

[13] S. Milgram, "The small world problem", *Psychology Today*, vol. 2, 1967.

[14] X. Liu and K. Aberer, "Soco: A social network aided context-aware recommender system", *WWW '13 Proceedings of the 22nd international conference on World Wide*

Web, pp. 781–902, 2013. [Online]. Available: `http://dl.acm.org/citation.cfm?id=2488457`.

[15] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey", *Knowledge-Based Systems*, 2012.

[16] S. V. Sandoval, "Novelty and diversity enhancement and evaluation in recommender systems", Master final work, Universidad Autonoma de Madrid, 2012, ch. 1, p. 1. [Online]. Available: `http://www.mavir.net/docs/tfm-vargas-sandoval.pdf`.

[17] A. Delley, *Réseaux IP de prochaine génération NGN - IMS - TISPAN*, 2nd ed. Ecole d'Ingénieurs et d'Architectes de Fribourg, 2002.

[18] A. Delley, P. Gaillet, O. Johnsen, M. Rast, and H. Keller, *VoIP - Voix sur IP et Multimédia*, 6th ed. Ecole d'Ingénieurs et d'Architectes de Fribourg, 2014.

[19] J. Cho, K. Kwon, Y. Park, and Q-rater, "A collaborative reputation system based on source credibility theory", *Expert Systems with Applications*, vol. 36, 2009. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S0957417408001486`.

[20] W. Carrer-Neto, M. Hernandez-Alcaraz, R. Valencia-Garcıa, and F. G. ıa Sanchez, "A peer-to-peer recommender system base don spontaneous affinities", *ACM Transactions on Internet Technology*, vol. 9, 1–34, 2009. [Online]. Available: `http://dl.acm.org/citation.cfm?doid=1462159.1462163`.

[21] W. Yuan, D. Guan, Y. Lee, S. Lee, and S. Hur, "Improved trust-aware recommender system using small-worldness of trust networks", *Knowledge Based Systems*, vol. 23, 232–238, 2010. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S095070511000002X`.

[22] M. Gupte and T. Eliassi-Rad, "Measuring tie strength in implicit social networks", *CoRR*, vol. abs/1112.2774, 2011. [Online]. Available: `http://arxiv.org/abs/1112.2774`.