OPEN ACCESS Freely available online

Journal of Information Technology & Software Engineering

Research Article

Proposal of a Hybrid Recommendation Algorithm to Support the Discovery for Mashup Applications

Takahiro Koita*, Daiki Takigawa

Department of Science and Technology, Doshisha University, 1-3 Tatara Miyakodani, Kyotanabe, Kyoto, 610-0321, Japan

ABSTRACT

This paper proposes a recommendation algorithm which combines collaborative filtering and content based algorithm. The proposed algorithm provides recommendation list that combines recommendation items generated each algorithms, and improves the novelty and the precision of recommendation. Especially, if the precision is low, the content-based algorithm should have higher priority and if the precision is high, the collaborative filtering should have higher priority. Therefore, this paper discusses and investigates priority rules and priority through the preliminary experiments. The priority rules are some rules to decide priority algorithm when combine two existing algorithms. The priority is a weight for priority algorithm. To decide appropriate priority rules and priority, the proposed algorithm was implemented on the Linked Mash which is our recommendation system of mashup applications and we conducted experiments with Linked Mash. In the experiments, the subjects evaluated some recommended mashup applications. The novelty and precision is calculated based on this evaluation. Changing the priority rules and priority for each subjects, we demonstrated that proposed algorithm can achieve a recommendation which both novelty and precision are high.

Keywords: Web Application program interface; Recommendation algorithm; Mashup; Collaborative filtering

INTRODUCTION

There are a great number of mashup applications currently available on the Web. Mashup applications are Web applications that combine the functions of different web services. The significant number of such applications on the Web creates a situation in which users may not be able to find the application they wish to use, even though it exists. Recommendations to support the discovery of mashup applications have attracted attention [1-4]. Mashup application recommendations use a recommendation algorithm to predict application evaluations. Recommendation algorithms extract features and predict evaluation values using user data, mashup data, and evaluation data as input. By presenting information on mashup applications with high predicted evaluation values, the user can use the recommended applications as-is, and can construct independent mashup applications based on the information on the recommended application. This study proposes a hybrid recommendation algorithm with the objective of supporting the discovery of mashup applications. It combines the existing collaborative filtering and content recommendation algorithms to realize recommendations with high degrees of novelty and recommendation precision.

We discuss issues in existing recommendation algorithms and clarify the issues of the study. Collaborative filtering algorithms and content algorithms are the representative types of existing recommendation algorithms [5-8]. Collaborative filtering has the benefit of having a high degree of novelty. However, it also has the drawback of reduced recommendation precision for low evaluation ratios. The evaluation ratio refers to the ratio of the user's actual ratings among all mashup applications. Another type of existing mashup recommendation algorithm is the content algorithm. This algorithm has the benefit of having a high level of recommendation precision. However, a drawback is that it recommends only similar items, leading to a low degree of novelty. If we can combine these two recommendation algorithms and use their respective benefits to supplement their

Correspondence to: Takahiro Koita, Faculty of Science and Technology, Doshisha University,1-3 Tatara Miyakodani, Kyotanabe, Kyoto, 610-0321, Japan, E-mail: tkoita@mail.doshisa.ac.jp

Received: July 26, 2021; **Accepted**: August 16, 2021; **Published**: August 23, 2021

Citation: Koita T, Takigawa D (2021) Proposal of a Hybrid Recommendation Algorithm to Support the Discovery for Mashup Applications. J Inform Tech Softw Eng. 11:263.

drawbacks, we can increase both novelty and recommendation precision in recommendations.

When the collaborative filtering algorithm recommendation precision is low, we present recommendations prioritizing content. When the collaborative filtering algorithm recommendation precision is high, we present recommendations prioritizing collaborative filtering. In order to fulfill the above requirements, this study addresses the following three issues.

- Investigate comparative methods
- Derive appropriate prioritization rules
- Derive appropriate degree of priority

Prioritization rules refer to the rules that determine which algorithm to prioritize. Degree of priority refers to the weight for recommendation item scores generated by the recommendation algorithms.

MATERIALS AND METHODS

In order to investigate comparative methods for recommended mashup applications and define prioritization rules and the degree of priority, we used evaluation data on IFTTT, applying collaborative filtering and then using the evaluation values and Mean Absolute Error (MAE) of recommended applications to measure recommendation precision. In this study, MAE is expressed by the recommendation precision subtracted from one. In addition, we performed linear regression analysis on the measured results. The results of the evaluation values for the recommended mashup applications are shown in Figure 1.

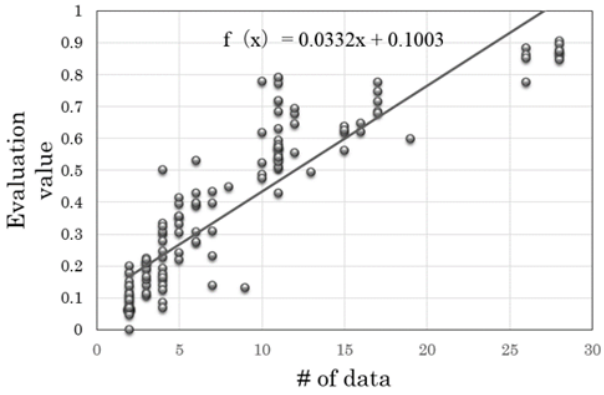

Figure 1: Evaluation value *vs.* #of data.

The results of the experiment showed that the evaluation values and recommendation precision for recommended mashup applications decreased for users with lower evaluation ratios. The regression line shown in Figure 1 shows the standard evaluation value of mashup applications recommended with collaborative filtering algorithms. By plotting the evaluation values for mashup applications evaluated with content algorithms on the regression line, we can compare the applications recommended with the two types of algorithms. We hypothesize that the recommendation precision is low below evaluation value α shown in Figure 1, and define the prioritization rules as detailed below.

Prioritization Rule A

When 0 ≤ evaluation value ≤ α, generate a recommendation list prioritizing mashup applications recommended by the content algorithm.

Prioritization Rule B

When α<evaluation value ≤ 1, generate a recommendation list prioritizing mashup applications recommended by the collaborative filtering algorithm.

In order to prioritize applications recommended by the prioritized algorithm, we must attach weights to the evaluation values of mashup applications recommended with the prioritized algorithm. We define this attached weight as degree of priority λ.

Proposed algorithm

We propose a hybrid recommendation algorithm to support discover of mashup applications, based on the prioritization rules and degree of priority described in the previous section. A flowchart of the hybrid recommendation algorithm is shown in Figure 2.

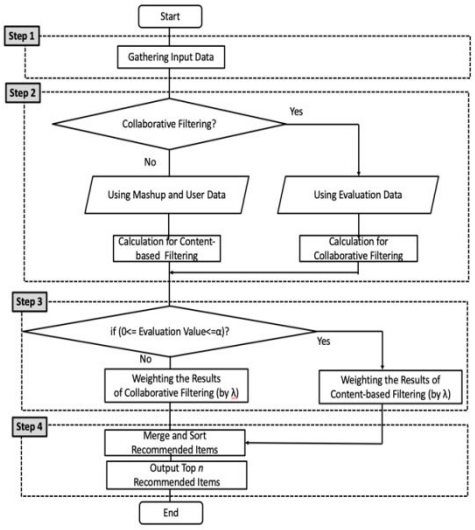

Figure 2: Overview of the proposed algorithm.

The hybrid recommendation algorithm is processed in the following four steps. In step 1, the input data for the proposed algorithm is collected. In step 2, after data collection, a collaborative filtering algorithm is applied to evaluation data, and a content algorithm is applied to user data and mashup data. In step 3, we determine the algorithm to prioritize based on the prioritization rules set up in Chapter 3, and add degree of priority λ to mashup applications recommended with the prioritized algorithm. In step 4, we combine and sort the mashup applications recommended with the two algorithms, generate a recommendation list of the top n results, and output it to the user.

If the prioritization rules and degree of priority are not appropriate, we cannot give recommendations with high degrees of novelty and precision. Appropriate prioritization rules (using

α) and degree of priority λ are set so that the recommendation precision is higher than when using collaborative filtering algorithms and novelty is higher than when using content algorithms. Below, we describe the design and implementation of a recommendation system that finds appropriate values for α and λ, and an evaluation experiment and discussion

Implementation

We implemented the proposed recommendation algorithm and constructed Linked Mash, a mashup recommendation system. The overview of Linked Mash is shown in Figure 3.

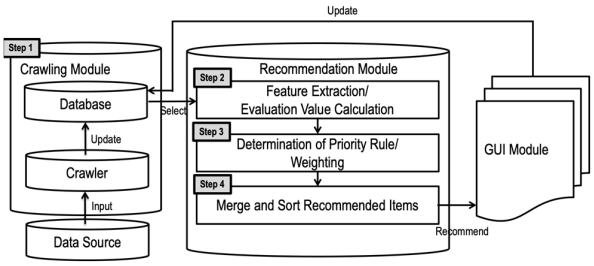

Figure 3: Overview of Linked Mash.

Linked Mash is composed of three modules: a crawl module, a recommendation module, and a GUI module. Step 1 of the proposed algorithm is performed by the crawl module, while steps 2, 3 and 4 are performed by the recommendation module. The crawl module collects and saves data to input into the proposed algorithm. The recommendation module implements the proposed algorithm and generates a list of recommendations. First, collaborative filtering and content algorithms are applied to input data from the crawl module, and then feature extraction and evaluation prediction is performed. Next, we determine the prioritized algorithm using the prioritization rules, then add weights to the mashup applications recommended with the prioritized algorithm. Finally, we combine and sort the recommended mashup applications and generate a recommendation list. The GUI module handles user control of Linked Mash.

RESULTS AND DISCUSSION

Using the Linked Mash implementation, we performed an evaluation experiment for novelty and recommendation precision as well as an experiment for system performance. Using the results of the former experiment, we found values for threshold α and degree of priority λ so that recommendation precision would be higher than using collaborative filtering algorithms and novelty would be higher than using content algorithms.

Using the implemented system, we performed an experiment with nine users. The experiment was performed according to the following procedure.

- While using the recommendation function within the system, the user pressed a "like" button for mashup applications listed in Linked Mash. We obtained the recommendation lists created when participants used the recommendation function.
- Once the user had "liked" a total of 15 times, we had them answer a survey on novelty and recommendation precision for the mashup applications recommended by the recommendation function when using Linked Mash. In the novelty survey, users selected 1 if they knew the mashup application recommended, and selected 2 if they did not. The recommendation precision survey was performed on a five-step scale, where 1 indicated the user wanted to use the application, and 5 indicated the user did not want to use the application.
- We measured novelty and recommendation precision using the user evaluations of mashup applications. Given a set R of mashup applications liked by users within a set L of mashup applications in the recommendation list, recommendation precision is expressed through the following formula.

$$precision = |R| / |L|$$

In addition, given a set C of unknown mashup applications that the user liked within the recommendation list, novelty is then defined in the following formula.

$$novelty = (|L| \cap |C|) / |L|$$

Given the above flow as one set, we performed three sets, changing the Linked Mash recommendation algorithm to collaborative filtering, content, and hybrid in that order. In the recommendation list generated with the hybrid algorithm, the ranges of α and λ for mashup applications recommended by both collaborative filtering and content were $0.2 \le \alpha \le 0.4$ and $1.3 \le \lambda \le 1.5$. Values within this range were allocated to each of the nine users, as shown in Table 1.

	λ= 1.3	λ = 1.4	λ = 1.5
α = 0.2	User A	User B	User C
α = 0.3	User D	User E	User F
α = 0.4	User G	User H	User I

Table 1: Allocation of α and λ.

In the experimental results, increases in novelty and precision were observed for User B and User C, but no increases were seen for other users. Below, we discuss the results for Users B and C, in which increases in novelty and precision were observed. Experiment results for User B are shown in Figures 4, 5 and 6.

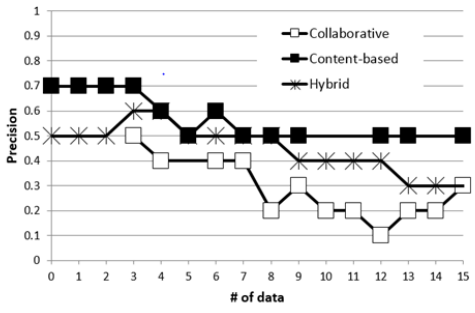

Figure 4: Precision *vs.* #of data (for User B).

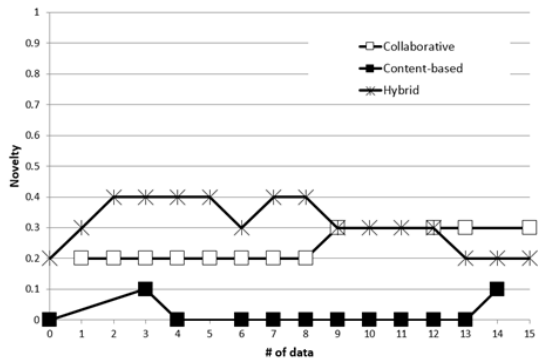

Figure 5: Novelty *vs.* #of data (for User B).

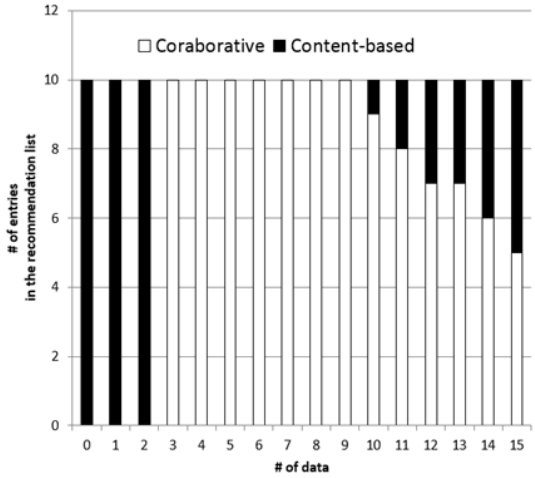

Figure 6: #of entries for each algorithm *vs.* #of data.

In the results of the experiment measuring User B's recommendation precision, the hybrid algorithm maintained an accuracy result above 0.5 for up to eight evaluations, but precision gradually decreased with further evaluations, finally declining to 0.3. In the results of the experiment measuring novelty shown in Figure 5, the results for User B show a novelty of approximately 0.4 maintained for up to eight evaluations, but novelty gradually decreased with further evaluations, finally declining to 0.2. In the results of the experiment that obtained the structure of User B's recommendation list, shown in Figure 6, with all evaluations, the list was composed of 70% applications from the cooperative filtering algorithm, and 30% applications from the content algorithm. Broken down by the number of evaluations, 100% of applications from 0 to 2 evaluations were recommended by the content algorithm, and 100% of applications from 3 to 9 evaluations were recommended by the cooperative filtering algorithm. The ratio of applications from the content algorithm gradually increased after ten evaluations, finally composing 50% of the recommendation list.

The results of the experiment measuring recommendation precision and novelty for User B are shown in Figures 4 and 5. The recommendation precision of the hybrid algorithm showed an average increase of 0.166 and a maximum increase of 0.3 from 0 to 15 evaluations in comparison with the collaborative filtering algorithm. Hybrid algorithm novelty showed an average increase of 0.299 and a maximum increase of 0.4 from 0 to 15 evaluations in comparison with the content algorithm. The results of the experiment obtaining the recommendation list of User B suggested that for up to ten evaluations, applications recommended by the content algorithm were reflected in the recommended application list. It is thought that the increase in novelty was because the prioritized algorithm changes at three evaluations, and more than three evaluations applications recommended by the collaborative filtering algorithm are often reflected in the list.

We discuss the derivation of appropriate prioritization rules. In this study, we defined two prioritization rules that used the evaluation value α predicted by the collaborative filtering algorithm as a threshold to toggle the prioritized algorithm. Performing an experiment based on these rules, we found that for all users, priority algorithm toggling took place within 15 evaluations. The number of user evaluations that toggled the prioritized algorithm was around the point where the regression line expressing the standard value for the collaborative filtering algorithm and the straight evaluation value α line intersected. Setting evaluation value α allowed the prioritized algorithm to be toggled. In order to derive appropriate prioritization rules, we must set the evaluation value α so that the user feels that the recommendation precision of applications recommended by the collaborative filtering algorithm is high. Setting a high value for α prevents toggling to collaborative filtering for all but users with many evaluations and reduces the novelty of recommendations, thus necessitating that we set the value of α as low as possible. In this experiment, we confirmed that high precision can be maintained even at an evaluation value of 0.2 when the user has 3 or 4 evaluations. Thus, it is best to set a suitable value for α that toggles the priority algorithm at around 3 or 4 user evaluations.

We discuss the derivation of an appropriate degree of priority. In this study, we defined the degree of priority as a weight λ to be added to the evaluation value predicted with the prioritized algorithm. By adding this weight, we made it possible to prioritize mashup applications recommended with the prioritized algorithm to be reflected in the recommendation list. The results of an experiment based on this degree of priority showed three trends in values of λ. First were cases in which the value for λ was too low. If λ is too low, mashup applications [5-8] recommended by the prioritized algorithm are prioritized even after the prioritized algorithm is toggled. Second were cases in which the value for λ were too high. When λ is too high, only recommendation items recommended by the prioritized algorithm are reflected in the list. Third was an appropriate value for λ, in which case recommendation items from both lists were appropriately mixed in the list after the prioritized algorithm was toggled. Overall results indicated that λ=1.4 was the proper value in this study. This study performed an actual experiment and found the value for the degree of priority heuristically. However, because we cannot obtain evaluations from the user each time we determine λ, we need a rule for the degree of priority that differs from that used in this study. One suitable method for deriving the weight of the degree of priority is to use the MAE of mashup applications that have been evaluated. We prepare candidates for the weight, then use the

weight for which the MAE of evaluated mashup applications is smallest. We can set an appropriate degree of priority by dynamically setting the weight.

CONCLUSION

This study proposed a hybrid recommendation algorithm with the objective of supporting the discovery of mashup applications. We implemented the proposed algorithm into a recommendation system called Linked Mash, and participant experiments using Linked Mash indicated that the recommendation precision of the proposed algorithm was an improvement over collaborative filtering algorithms, and that the novelty was an improvement over content algorithms.

REFERENCES

1. Bizer C, Heath T, Idehen K, Berners-Lee T. Linked data on the web (LDOW2008). In proceedings of the 17th international conference on world wide web. 2008;1265-1266.
2. McIlraith SA, Son TC, Zeng H. Semantic web services. IEEE intelligent systems. 2001;16(2):46-53.
3. Elmeleegy H, Ivan A, Akkiraju R, Goodwin R. Mashup advisor: A recommendation tool for mashup development. In 2008 IEEE international conference on web services. IEEE. 2008;337-344.
4. Tapia B, Torres R, Astudillo H, Ortega P. Recommending APIs for mashup completion using association rules mined from real usage data. In2011 30th international conference of the chilean computer science society. IEEE. 2011;83-89.
5. Maaradji A, Hacid H, Skraba R, Vakali A. Social web mashups full completion via frequent sequence mining. In2011 IEEE world congress on services. IEEE. 2011;9-16.
6. Cremonesi P, Picozzi M, Matera M. A comparison of recommender systems for mashup composition. In 2012 third international workshop on recommendation systems for software engineering (RSSE). IEEE. 2012;54-58.
7. Cao B, Liu J, Tang M, Zheng Z, Wang G. Mashup service recommendation based on user interest and social network. In 2013 IEEE 20th international conference on web services. IEEE. 2013;99-106.
8. Xu W, Cao J, Hu L, Wang J, Li M. A social-aware service recommendation approach for mashup creation. In 2013 IEEE 20th international conference on web services. IEEE. 2013;107-114.