Negative-Connection-Aware Tag-Based Association Mining and Service Recommendation

Yayu Ni[1], Yushun Fan[1,*], Keman Huang[2], Jing Bi[1], and Wei Tan[3]

[1] Tsinghua National Laboratory for Information Science and Technology, Department of Automation Tsinghua University, Beijing 100084, China
nyy07@mails.tsinghua.edu.cn, {fanyus,bijing}@tsinghua.edu.cn
[2] School of Computer Science and Technology, Tianjin University, Tianjin 300072, China
victoryhkm@gmail.com
[3] IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598, USA
wtan@us.ibm.com

Abstract. Service recommendation facilitates developers to select services to create new mashups with large-granularity and added value. Currently, most studies concentrate on mining and recommend common composition patterns in mashups. However, latent negative patterns in mashups, which indicates the inappropriate combinations of services, remain largely ignored. By combining additional negative patterns between services with the already-exploited common mashup patterns, we present a more comprehensive and accurate model for service recommendation. Both positive association rules and negative ones are mined from services' annotated tags to predict future mashups. The extensive experiment conducted on real-world data sets shows a 33% enhancement in terms of F1-Score compared to classic association mining approach.

Keywords: Service Recommendation, Negative Mashup Patterns, Tag Collaboration Rules.

1 Introduction

In Web 2.0 era, a growing number of interactive web services have been published on the Internet. By combing chosen web services, web developers now are able to create novel mashups (i.e., composite services derived from services) to meet specific functional requirements from web users. This programmable paradigm produces an enlarging ecosystem of web services and their mashups [1].

Rapid increasing of available online services makes manual selection of suitable web services a challenging task. Automatic service integration architecture, like SOA [2], and service recommendation algorithms [3-6] are proposed to facilitate service selection and integration in mashup completion. Recently, an increasing number of

* Yushun Fan is with Tsinghua National Laboratory for Information Science and Technology, the Automation Department Tsinghua University, Beijing, 100084 China. (Corresponding, E-mail: fanyus@tsinghua.edu.cn).

X. Franch et al. (Eds.): ICSOC 2014, LNCS 8831, pp. 419–428, 2014.

researchers are becoming interested in analyzing and summarizing common service composition patterns from historical mashups. In order to instruct future mashup creation, a plenty of mashup pattern models are proposed to enable automatic service integration [1,7-9]. There are also techniques that focus on finding collaboration rules between services [4,10,11] and complete service composition based on rule reasoning.

Besides these common patterns or rules, there are also relationships indicating improper service combinations. Consider two very popular services, "*Google App Engine*" and "*Yahoo Video Search*", on ProgrammableWeb.com[1], the largest services/mashups repository on the Internet. Even if both of them have been frequently used in the creation of thousands of mashups, they've never been co-used in the same mashup even once (until Dec. 31th, 2013). In the collaboration network of programmable ecosystem, there exist a large number of never-collaborate connections between services, formally named negative connections in this paper. Straightforwardly, the ever-collaborate connections among services can be named as positive connections. Based on the previous work [12,13], the collaboration network is a rare sparse network. Hence the number of the negative connections are much greater than the positive connections, which brings the data imbalance problem. On the other hand, some current automatic integration techniques just use the positive connections but tend to ignore all these negative connections, which will lose some useful information.

In order to consider the negative connections in a suitable way to improve the recommend performance, this paper proposes a novel method to collect negative connections between services as well as positive ones. The main contribution of this paper is a novel association mining model based on both positive and negative connections for service recommendation. More specific:

- We introduce a training dataset generation strategy to collect both positive and negative connections between services from the collaboration network.
- We propose a rule-based decision tree algorithm, i.e., *RuleTree*, to mine both positive and negative tag collaboration rules from the annotated tags of service connections in the training dataset. A rule scoring strategy, named *RuleScore*, is presented to score the collaboration rules. Combing RuleTree and RuleScore, the novel service recommendation approach is introduced to recommend service for the user.

Experiments on the real-life ProgrammableWeb dataset shows that, compared with the classic association-based approach, our negative-connection-aware approach gains a 33% improvement in terms of F1-Score for service recommendation.

The rest of the paper is organized as follows. Section 2 gives the concepts and problem formulation of tag-based service recommendation. Section 3 presents the details of tag-based association model for service recommendation. Section 4 presents experimental results on mashup ecosystem. Section 5 summarizes the related work and section 6 concludes the paper.

[1] http://www.programmableweb.com

2 The Problem Formulation

2.1 Preliminary Definitions

Let $S = \{s_1, s_2, \ldots, s_P\}$ denotes all the web services, $M = \{m_1, m_2, \ldots, m_N\}$ denotes all the mashups, and $\langle S, M \rangle$ the collaboration network between services and mashups. If a mashup m is composed by several services $s_m^{(1)}, s_m^{(2)}, \ldots, s_m^{(r)}$, then $m = \{s_m^{(1)}, s_m^{(2)}, \ldots, s_m^{(r)}\}$. Let $T = \{t_1, t_2, \ldots, t_N\}$ denotes all tags that used to annotate services and $\Gamma = \{a | \forall a \subseteq T\}$ denotes all the subsets of T. For an service s, its annotated tags are $T_s = \{t_s^{(1)}, t_s^{(2)}, \ldots, t_s^{(K)}\} \in \Gamma$.

Definition 1: Service Connection.
For two services $s_1, s_2 \in S$, a service connection between them is denoted as $c = \langle s_1, s_2, f(s_1, s_2) \rangle$, where $f(s_1, s_2)$ is a function indicating whether the two services have ever been co-used or not. If s_1 and s_2 have ever been co-used in mashups, then $f(s_1, s_2) = 1$; otherwise $f(s_1, s_2) = -1$. Furthermore, c is called a positive connection if $f(s_1, s_2) = 1$; otherwise c is called a negative connection.

Definition 2: Service's Popularity.
Service's popularity is the function $n(s)$ indicating the number of mashups that contain service s. Hence:

$$n(s) = |\{\forall m \in M | s \in m\}| \tag{1}$$

where $|\cdot|$ means the number of elements in set.

Definition 3: Tag Collaboration Rule.
A tag collaboration rule $r = \langle T_{r1}, T_{r2} \rangle \in \Gamma \times \Gamma$ is a combination of two tag sequences. Given a service connection $c = \langle s_1, s_2, f(s_1, s_2) \rangle$, as well as the annotated tags of services T_{s_1} and T_{s_2}, then c is said to satisfy rule $r = \langle T_{r1}, T_{r2} \rangle$, formally denoted as $c \Vdash r$, if

$$(T_{s_1} \supseteq T_{r1} \text{ and } T_{s_2} \supseteq T_{r2}) \text{ or } (T_{s_1} \supseteq T_{r2} \text{ and } T_{s_2} \supseteq T_{r1}) \tag{2}$$

2.2 Tag-Based Service Recommendation

For most of the popular online repositories, like ProgrammableWeb.com, each web service is annotated with several tags to describe its category, functionalities and properties. Consider two web services published in the repository, denoted as s_1' and s_2'. Each service is annotated by a sequence of tags: s_1' is annotated by tags $T_{s_1'} \in \Gamma$, and $T_{s_2'} \in \Gamma$ for s_2'. The problem of tag-based service recommendation can be defined as follow:

Given two services s_1', s_2' with their annotated tags $T_{s_1'}$ and $T_{s_2'}$, find out whether they will collaborate with each other to construct mashups, based on the potential tag

collaboration rules lying in the historical mashups in the collaboration network $\langle S, M\rangle$.

3 Tag-Based Association Model for Service Recommendation

3.1 Service Connection Generation

Due to the sparsity of collaboration network and the incompleteness of historical mashup dataset, there is an extreme imbalance between positive and negative samples: the number of never-collaboration connections is much greater than that of ever-collaboration ones. Thus, this section proposes a strategy utilizing a service popularity criterion to collect a balanced set of both positive and negative connections.

For two services s_1 and s_2 which have been co-used in at least one mashup, a positive connection between them is generated. Thus given all the historical mashups, a set of positive service connections can be defined as follow:

$$C_p = \{\langle s_1, s_2, 1\rangle | \forall s_1, s_2 \in S, s_1 \neq s_2, f(s_1, s_2) = 1\} \tag{3}$$

For two services which have never collaborate with each other in the same mashup, there exist a negative connection among them. Due to the incompleteness of the dataset, the negative connections cannot be simply considered as the services cannot collaborate to compose a mashup. However, if two services are very popular but never collaborate with each other, it is reasonable to believe that these two services are not capable to construct a mashup in the near future. Hence this paper defines the negative connections between popular services as the credible negative connection. Here the popular service is defined as follow:

Given the popularity threshold k, if the service's popularity is larger than k, i.e. $N(s) \geq k$, then this service is popular.

Then a set of credible negative service connections can be generated as follows:

$$C_n = \left\{\langle s_1, s_2, -1\rangle) \middle| \begin{matrix} \forall s_1, s_2 \in S, n(s_1) \geq k, n(s_1) \geq k, \\ s_1 \neq s_2, f(s_1, s_2) = -1 \end{matrix} \right\} \tag{4}$$

3.2 Mining Tag Collaboration Rules: RuleTree Algorithm

This section presents a rule-based decision tree algorithm, named *RuleTree*, to mine tag collaboration rules from the training set of positive service connections and credible negative connections. RuleTree algorithm constructs a decision tree which contains collaboration rules as its non-leaf nodes and composability labels, positive or negative, as its leaf nodes. The RuleTree algorithm contains two steps, which is shown as follows.

Step 1: Generate Candidate Rules.

A rule r can only be considered as a candidate rule that can probably be used to construct the tree, if there is at least one service connection in the training set $C = C_p \cup C_n$ satisfies it. Hence the set of candidate rules can be defined as follows:

$$R = \left\{ \forall r \in \Gamma \times \Gamma \middle| \exists c \in C_p \cup C_n, c \Vdash r \right\} \tag{5}$$

Step 2: Construct Decision Tree.
The construction step of RuleTree algorithm employs a procedure similar with ID3 [14]: at each iteration of the algorithm, the most significant rule is chosen from candidate rule set, which could distinguish between positive service connections and negative connections to achieve the minimal entropy. Hence the detail of training procedure is given as follows:

Algorithm 1. RuleTree: Construct Decision Tree
Require: the connection set C
Require: candidate rule set R
Require: pre-defined maximal depth n of decision tree
RuleTree(C, R, n):
1. if all samples in C are of one class, **or** $R = \phi$, **or** $n = 0$ **do**
2. **if** the majority of connections are positive **do**
3. **return** the single-node tree with label=1
4. **else do**
5. **return** the single-node tree with label=-1
6. else do
7. $r \leftarrow$the rule from R that best classifies C according to minimal entropy metric
8. $C_+ \leftarrow \{\forall c \in C | c \Vdash r\}$, $C_- \leftarrow C - C_+$, $R \leftarrow R - \{r\}$
9. **return** a RuleTree with root node r, left branch Rule_Tree$(C_+, R, n-1)$ and right branch Rule_Tree$(C_-, R, n-1)$

3.3 Scoring Collaboration Rules: RuleScore Algorithm

RuleScore algorithm scores the tag collaboration rules previously found out by RuleTree. RuleScore employs the classic Adaboost algorithm [15] to construct a sequence of shallow rule-based decision trees whose depths are restricted to 1 and assign each of these trees with a coefficient indicating its contribution to composability. At each iteration of main loop, RuleScore constructs a tree over current training dataset of service connections with current weights and then updates the weights of all training samples according to the fact whether they are correctly classified by generated rule stumps. The detail of RuleScore algorithm is given as follows:

Algorithm 2. RuleScore: Score Collaboration Rules
Require: the connection set C
Require: candidate rule set R
Require: number of maximal iterations q
RuleScore(C, R, q):
1. Initialize every connection i of C with weight $D_1(i) = 1/|C|$
2. Initialize every rule r of R do with $Score(r) = 0$
3. for $j = 1, \ldots, q$ **do**

```
4.    training a tree h_j(r) ←RuleTree(C, R, 1) over C with respect to
   the weight distribution D_j where r is the root rule of h_j(r)
5.    ε_j ←the misclassification rate of h_j(r) over C
6.    α_j ← 0.5 ln((1 − ε_j)/ε_j)
7.    Score(r) ← Score(r) + α_j
8.    for connection i that is correctly classified by h_j do
9.          D_{j+1}(i) ← D_j(i)e^{−α_j}
10.   for connection i that is misclassified by h_j do
11.         D_{j+1}(i) ← D_j(i)e^{α_j}
12. return {∀r ∈ R|Score(r) ≠ 0} as Scored_Rules
```

3.4 Rule-Based Service Recommendation

Consider two web services s_1 and s_2. Given the annotated tags T_{s_1} and T_{s_2} of them, as well as the scored collaboration rules $Scored\_Rules$ generated by RuleScore, then the set of scored rules that can be satisfied by $\langle s_1, s_2, f(s_1, s_2)\rangle$ is denoted as follows:

$$R(s_1, s_2) = \{\forall r \in Scored\_Rules | \langle s_1, s_2, f(s_1, s_2)\rangle \Vdash r\} \quad (6)$$

Hence the composability score of service s_1 and s_2 can be calculated by summing up the scores of all satisfied rules $R(s_1, s_2)$:

$$F(s_1, s_2) = \sum_{\forall r \in R(s_1, s_2)} score(r) \quad (7)$$

The more positive rules and less negative rules are satisfied by $\langle s_1, s_2, f(s_1, s_2)\rangle$, the greater $F(s_1, s_2)$ is, and two services are more probably composable. Therefore if the estimated value $F(s_1, s_2) > 0$, then services s_1 and s_2 are regarded composable, and if $F(s_1, s_2) < 0$, then s_1 and s_2 are considered as none-composable in mashups.

4 Experiments

Experiments is conducted to evaluate our proposed model compared with some baselines. This paper employs the real-life dataset of web services and mashups obtained from ProgrammableWeb.com. By removing mashups containing less than two services and services that never collaborate with others, we obtain a filtered collection of 1301 services and 3557 mashups (Dec. 31th, 2013).

4.1 Baseline Algorithms

Baseline I: Apriori-based Service Recommendation.
In this approach [11], each mashup is represented as the union of annotated tags of its component services. Apriori [16] mines positive rules of tags from the transactions of mashups. Then composability of any two services s_1 and s_2 is estimated as:

$$F(s_1, s_2) = \sum_{r \in R(s_1, s_2)} \beta * support(r) + (1 - \beta) * confidence(r) \quad (8)$$

where $R(s_1, s_2)$ denotes the rules satisfied by $\langle s_1, s_2 \rangle$, and β is a weighting coefficient. If $F(s_1, s_2)$ is greater than a predefined threshold, s_1 and s_2 are considered as composable; otherwise not.

Baseline II: RuleTree-based Service Recommendation.
In this approach, service recommendation is made by utilizing decision tree constructed by RuleTree algorithm. The maximal height of constructed tree is restricted to 1000. For every two services with their annotated tags, decision tree returns a binary value, 0 or 1, indicating whether they can collaborate in mashups.

Baseline III: Subsampling for Negative Service Connections.
In this approach, subsampling technique [17] is utilized to handle the dataset imbalance, instead of popularity-based selection strategy in our proposed model. By randomly sampling without replacement, a portion of negative service connections is selected for model training to equal the size of positive connection dataset.

4.2 Performance of Service Recommendation

Table 1 shows the average precision, recall and F1-Score values of proposed service recommendation model in a ten-fold cross validation experiment compared with baseline algorithms for dataset introduced above. Empirically, the threshold of popularity for credible negative connections generation is set 10 in the proposed model, and run 20000 iterations for it to stop. The weighting coefficient of Apriori-based approach is set $\beta = 0.5$, and threshold to determine composability is set 0.4.

Table 1. Performance of Four Recommendation Approaches

Recommendation Approach	**Precision**	**Recall**	**F1-Score**
Apriori-based Recommendation	17.44%	83.58%	0.2885
RuleTree-based Recommendation	70.23%	46.83%	0.5619
Subsampling For Negative Connection Generation	76.12%	62.57%	0.6868
The Proposed Model	***78.85%***	***66.03%***	***0.7187***

It can be observed that our proposed model has the best performance in terms of F1-Score, which can be interpreted as a weighted average of the precision and recall. The classic Apriori algorithm results in an extreme low prediction precision because of its incapability of modeling negative associations and it misclassifies a large portion of negative connections as positive ones. Our proposed model outperforms RuleTree-based approach because of the additional adoption of RuleScore, which enhances the accuracy of RuleTree by utilizing Adaboost meta-algorithm. Our model also outperforms the baseline method that utilizing subsampling for negative connection generation, because popularity-based selection strategy of our model produces a more credible set of negative connections than random sampling.

5 Related Works

In the researches of automatic service composition, service association learning is emerging as a new technique for automatic mashup creation. Most current service association models is on the basis of Apriori algorithm: [4] presents a global Co-utilization service Ranking (gCAR) strategy using association rules inferred by Apriori from historical service usage data, to recommend best services for mashup completion. [7] defines a three-level model of service usage data, from which service association rules is mined to model service usage patterns based on Apriori algorithm. [11] uses Apriori to find association rules of social tags and predict future mashups based on mined rules. [18] utilizes Apriori to mine service association rules from service logs to build a knowledge repository to instruct mashup creation.

Some researches try to extend Apriori algorithm by including negative rules. [19] presents a pruning strategy to reduce the size of candidate negative rules, so that the generation of negative rules can be achieved in an acceptable time. [20] defines a more generalized form of negative rules and generates both positive and negative rules based on the correlation metric. However, these researches merely take the positive samples into consideration, ignoring the latent existence of negative samples.

From a different perspective, we employ both positive service connections and negative ones generated based on service popularity metric from service collaboration network. Hence a more comprehensive service association mining model is formed for automatic service recommendation and gain a better performance.

6 Conclusion

The paper proposed a tag-based association model for service recommendation, combining positive mashup patterns and negative ones. This combination gives a more comprehensive and meaningful illustration of current trend for mashup creation. To the best of our knowledge, we are the first to mine negative tag collaboration rules from service collaboration network, shedding new light on service usage pattern discovery. Our model also produces a more accurate prediction than the well-known Apriori-based service recommendation approaches, making a great accuracy improvement in service collaboration prediction.

In the future work, we plan to develop a distributed version of our model to improve the efficiency of rule mining, which enables it to scale to a massive number of services and service tags in big data applications.

Acknowledgement. This work is partially supported by the National Natural Science Foundation of China (No. 61174169, No. 61033005), the National Key Technology R&D Program (No. 2012BAF15G01) and Doctoral Program Foundation of Institutions of Higher Education of China (No. 20120002110034).

References

1. Han, Y., Chen, S., Feng, Z.: Mining Integration Patterns of Programmable Ecosystem with Social Tags. Journal of Grid Computing, 1–19 (2014)
2. Erl, T.: SOA: principles of service design. Prentice Hall Upper Saddle River (1) (2008)
3. Keman, H., Yushun, F., Wei, T., Xiang, L.: Service Recommendation in an Evolving Eco-system: A Link Prediction Approach. In: International Conference on Web Services, pp. 507–514 (2013)
4. Tapia, B., Torres, R., Astudillo, H., Ortega, P.: Recommending APIs for Mashup Completion Using Association Rules Mined from Real Usage Data. In: International Conference of the Chilean Computer Science Society, pp. 83–89 (2011)
5. Dou, W., Zhang, X., Chen, J.: KASR: A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Application. IEEE Transactions on Parallel and Distributed Systems PP, 1 (2014)
6. Xi, C., Xudong, L., Zicheng, H., Hailong, S.: RegionKNN: A Scalable Hybrid Collaborative Filtering Algorithm for Personalized Web Service Recommendation. In: International Conference on Web Services, pp. 9–16 (2010)
7. Liang, Q.A., Chung, J.Y., Miller, S., Yang, O.: Service Pattern Discovery of Web Service Mining in Web Service Registry-Repository. In: IEEE International Conference on E-Business Engineering, pp. 286–293 (2006)
8. Chien-Hsiang, L., San-Yih, H., I-Ling, Y.: A Service Pattern Model for Flexible Service Composition. In: International Conference on Web Services, pp. 626–627 (2012)
9. Vollino, B., Becker, K.: Usage Profiles: A Process for Discovering Usage Patterns over Web Services and its Application to Service Evolution. International Journal of Web Services Research 10(1), 1–28 (2013)
10. Spagnoletti, P., Bianchini, D., De Antonellis, V., Melchiori, M.: Modeling Collaboration for Mashup Design. In: Lecture Notes in Information Systems and Organization, pp. 461–469 (2013)
11. Goarany, K., Kulczycki, G., Blake, M.B.: Mining social tags to predict mashup patterns. In: Proceedings of the 2nd International Workshop on Search and Mining User-Generated Contents, pp. 71–78 (2010)
12. Keman, H., Yushun, F., Wei, T.: An Empirical Study of Programmable Web: A Network Analysis on a Service-Mashup System. In: International Conference on Web Services, pp. 552–559 (2012)
13. Huang, K., Fan, Y., Tan, W.: Recommendation in an Evolving Service Ecosystem Based on Network Prediction. IEEE Transactions on Automation Science and Engineering PP, 1–15 (2014)
14. Mitchell, T.: Decision tree learning. Mach. Learn. 414 (1997)
15. Friedman, J., Hastie, T., Tibshirani, R.: Special invited paper. additive logistic regression: A statistical view of boosting. Annals of Statistics, 337–374 (2000)
16. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Databases, pp. 487–499 (1994)
17. Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., Herrera, F.: A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42(4), 463–484 (2012)

18. Bayati, S., Nejad, A.F., Kharazmi, S., Bahreininejad, A.: Using association rule mining to improve semantic web services composition performance. In: International Conference on Computer, Control and Communication, pp. 1–5 (2009)
19. Wu, X., Zhang, C., Zhang, S.: Efficient mining of both positive and negative association rules. ACM Transactions on Information System 22(3), 381–405 (2004)
20. Antonie, M.-L., Zaïane, O.R.: Mining positive and negative association rules: An approach for confined rules. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 27–38. Springer, Heidelberg (2004)