Confidence-Aware Reputation Bootstrapping in Composite Service Environments

Lie Qu(✉), Athman Bouguettaya, and Azadeh Ghari Neiat

University of Sydney, Sydney, Australia
{lie.qu,athman.bouguettaya,azadeh.gharineiat}@sydney.edu.au

Abstract. We propose a novel reputation bootstrapping approach for both composite and atomic services in service-oriented environments. We consider multiple factors which may implicitly represent reputations of new services. Our approach does not rely on empirical assumptions. In contrast, we propose a data-driven method to determine how much a factor can represent service reputation. The reputation-related factors are modelled in a layer-based framework. This aims to quantitatively describe the importance of factors in reputation bootstrapping. Furthermore, we define *confidence* to represent how reliable the bootstrapped reputation of a new service is. We evaluate our approach based on a real-world dataset. The experimental results demonstrate the feasibility and outperformance of our approach.

1 Introduction

Reputation is an effective way to determine the performance quality of a service based on prior performance experiences (or records). However, performance experiences may not be always available when a new service emerges. Consequently, its reputation cannot be assessed, and thus trust establishment between consumers and the new service becomes challenging. Reputation bootstrapping is a key enabler to assign appropriate initial reputations for new services.

Reputation bootstrapping has been extensively studied in the literature [1,6,8,10,14,15]. These studies are typically based on particular empirical assumptions in which the reputation of a new service can be extracted from its inherent characteristics. For example, the approach proposed in [11] presents that a new service provided by a reputable provider tends to offer good performance. The approach proposed in [14] assumes that the reputation of a new service may approach to those of its similar services. However, such an assumption may not always hold under various circumstances. Moreover, because a new service usually has multiple characteristics, each of which can relatively reflect its future performance to some extent. How effectively each characteristic can represent the new service's reputation is usually unclear in real-world situations. Therefore, we investigate that reputation bootstrapping should only depend on the assumptions that can be practically validated in particular cases. Otherwise, the validity of the assumptions should be studied, i.e., determining which characteristic can more effectively reflect new services' future reputations.

M. Maximilien et al. (Eds.): ICSOC 2017, LNCS 10601, pp. 158–174, 2017.
https://doi.org/10.1007/978-3-319-69035-3_11

Service composition provides an elegant means to aggregate services to provide a value-added service that meets consumers' complex requirements. Reputation bootstrapping for composite services is a key challenge because the correlation between the performance of a composite service and that of its corresponding component services is unclear and may vary case by case. Whether the reputation of a composite service can be represented by those of its component services may usually be unknown in practice. Therefore, such a correlation should be studied in a particular case, and cannot be taken as a common assumption for reputation bootstrapping. Furthermore, although the reputations of a composite service and its component services are quite correlated, the reputation bootstrapping for composite services is more challenging than that for atomic services. It needs to be addressed in the following three cases: (1) reputations of component services are available; (2) reputations of component services are partially available. (3) reputations of component services are totally unavailable. In the first case, an effective reputation bootstrapping approach should first determine the effectiveness of reputation correlation between a composite service and its component services, and then identify the specific correlation between each other. For the other cases, reputation bootstrapping for atomic services should be performed first to predict the unavailable reputations of component services. The first-case approach is then applied for further bootstrapping. In this paper, we focus on the first case. The other cases will be discussed in our future work.

In this paper, we study reputation bootstrapping in composite service environments. Our main contributions are summarised as follows:

1. We propose a novel service reputation bootstrapping approach by considering multiple characteristic factors[1] which may implicitly represent new services' future reputations. Our approach does not rely on particular empirical assumptions. Instead, a data-driven method is proposed to explore the importance of reputation-related factors in terms of particular cases.
2. A layer-based bootstrapping framework is proposed, which aims to quantitatively model the importance of reputation-related factors. The proposed framework can easily be extended to a general case. That makes the framework compatible with diverse situations in service-oriented environments.
3. We define *confidence* which describes the reliability of new services' bootstrapped reputations. The notion *confidence* would help consumers make a more comprehensive evaluation on reputation bootstrapping.
4. We conduct experiments based on a real-world dataset from GitHub to evaluate the proposed approach. The experimental results demonstrate the feasibility and outperformance of our work.

Motivating Scenario: the problem of reputation bootstrapping is illustrated using a real-world scenario of a mobile application company which provides a location-based review service through an app (e.g., Foursquare App[2]).

[1] To avoid ambiguity, we use the term "factor" to represent "inherit characteristic" in the rest of this paper.

[2] foursquare.com.

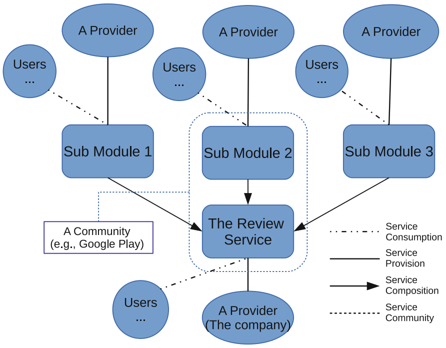

Fig. 1. The Motivating Scenario

The company outsources the sub-functional modules from an open-source software platform (e.g., GitHub) to reduce development cost. By composing these sub modules, the company can offer its own review service to a market. Although the newly developed service has no past performance records, it has multiple characteristic factors to represent its future reputation. The factors include the reputation of its provider (the company), the reputation of the community (e.g., Google Play) it belongs to, the reputations of similar review services, the past performance records of its sub modules, etc. In this scenario, we consider the review service as a composite service whose component services are the sub modules. Figure 1 illustrates the scenario. The review service is composed of three sub modules. In practice, the sub modules of a repository at GitHub are specified in a .gitmodules file. The review service and its sub modules have their distinct providers. On the other hand, there are a number of users for each service. They consume the services and provide feedback (e.g., the star reputation system at GitHub[3]) for service performance assessment. In addition, a service may belong to a community which may be a reputable commercial company or a certified organisation. For example, some open-source repositories at GitHub belong to Google. In this scenario, although the review service has no historical performance records, its reputation can be predicted according to multiple factors, e.g., provider reputation, community reputation and component service reputation. However, which factor is dominant in reputation representing is still unclear. Our proposed approach focuses on determining the importance of factors in reputation bootstrapping, and presents the confidence of every bootstrapped reputation. As the data at GitHub contain all the features which can appear in service-oriented environments, we employ a GitHub dataset to evaluate our proposed approach.

[3] help.github.com/articles/about-stars

2 The Layer-Based Framework

In this section, we propose a layer-based framework to model the importance of reputation-related factors of a service. Considering our motivating scenario, the boostrapped reputation of the new review service can be computed based on some implicit factors. Specifically, the factors are summarised as follows: a reputable provider has a high probability of providing good services; a service belonging to a reputable community may have good quality; a service composed of good-performance component services tends to perform well since the quality of its sub modules is satisfactory. Moreover, similar services may have similar reputations in some cases. In this regard, service similarity can also be used to predict new services' reputations [14]. However, the importance degree of each factor in representing service reputation is still unknown. In practice, the factor importance may change in terms of different circumstances.

We propose a layer-based framework to quantitatively model the importance of these reputation-related factors. Figure 2 describes the proposed framework of reputation transfer among the factors. Each factor is modelled in a layer of the framework. The main reason that we model the reputation-related factors in a layer-based structure is to intuitively illustrate the importance of these factors. According to our motivating scenario, the framework consists of *user layer*, *provider layer*, *community layer*, *similar service layer* and *component service layer*, where the user layer outputs the direct reputation of a service, and all the other layers reflect its indirect reputation. In this paper, we consider that consumers' feedback is the most reliable information to assess service reputation. That is because feedback is generated based on the actual experiences of service performance. Although there may exist biased or malicious feedback, user feedback is still the most direct way to evaluate service reputation. Furthermore, some studies [7,13] focus on credibility evaluation of user feedback to improve its

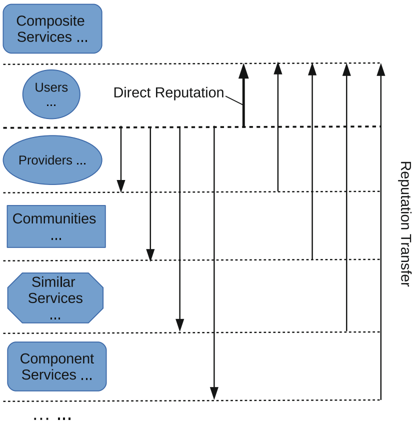

Fig. 2. The Layer-based Framework

reliability. Consequently, the feedback from the user layer is employed to evaluate the direct reputation of a service. The reputations computed from other layers are taken as the indirect reputations of the service. Except for the user layer, each of the other layers contains a reputation transfer process. The reputation transfer processes are shown via arrow lines in Fig. 2. In a reputation transfer process, the direct reputation of a new service is estimated through the indirect reputations from other layers.

In particular cases, some reputation-related factors may be unavailable. For example, if a service does not belong to any community, then community information cannot be used to estimate its reputation. On the other hand, there may also exist new factors, which are not included in Fig. 2. A new factor can be modelled in a new layer of the framework. This guarantees the generality of our reputation bootstrapping approach. The proposed framework can also be applied for reputation bootstrapping of atomic services. This is equivalent to the case of removing the component service layer from the framework.

The reputation-related factors may have different degrees of importance in representing a service's reputation. We model these factors in the framework by following this rule: "the more important a factor is in reputation representing, the higher layer it stays in". Therefore, as aforementioned, we put the user layer in the first place as it represents the direct reputation. The order of other layers is not fixed, and depends on particular cases. The layer order is determined by our data-driven bootstrapping approach introduced in the next section.

3 The Reputation Bootstrapping Approach

In this section, we first introduce the details of the proposed reputation bootstrapping approach in Sects. 3.1 and 3.2, and then introduce *confidence* in Sect. 3.3.

3.1 Reputation Evaluation

In this paper, the evaluation of service reputation is assumed to be performed by aggregating users' feedback during a particular period. We assume that user feedback is converted into normalised numerical values in this paper. In [7], Malik and Bouguettaya propose that service reputation evaluation should consider multiple metrics, including rater credibility, personal preferences, temporal sensitivity, majority ratings and past rating history. As reputation evaluation is not the main contribution of this paper, we only consider users' credibility of giving feedback since it is the most influential factor in reputation evaluation. In this regard, the direct reputation of a service is computed as follows:

$$r_j = \frac{\sum_u (f_j^u \times c_u)}{\sum_u c_u}, \tag{1}$$

where r_j denotes the direct reputation of service j, f_j^u denotes the feedback given by user u to service j, and c_u is the credibility of u.

User credibility is usually computed in different ways under various circumstances. As we apply a GitHub dataset to evaluate our work (see Sect. 4), we consider how to compute the credibility of GitHub users and reputations of repositories. At GitHub, a user gives stars to other users to express his/her appreciation on their work. Through this starring system, all users are connected as a directed graph. Figure 3 illustrates an example of the star network. In this network, if a user obtains more stars from others, he/she is considered more capable of giving fair feedback since many other users recognise his/her expertise. In addition, a star given by a user who has more stars should be considered more important than a star given by a user who has fewer stars. Given a star network, we compute user credibility of giving feedback using PageRank [12], which is a well-known approach to identify the importance of Web pages. In PageRank, a Web page has a high rank if the sum of the ranks of the pages which cite it is high. As a result, it is a recursive process to compute the importance of all pages. Due to the space limitation, we omit the detailed process of applying PageRank to compute the user credibility c_u at GitHub. Furthermore, the GitHub users also give stars to repositories. The direct reputation of a repository is computed based on the number of stars it obtains. Every star is weighted by the credibility of the user who gives the star.

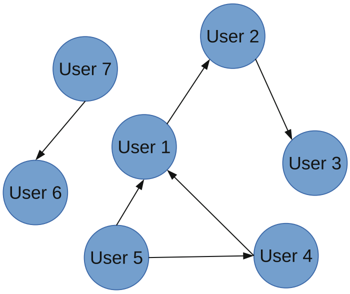

Fig. 3. User Starring Network

3.2 Reputation Bootstrapping

Suppose a complete framework denoted as $\mathbb{L}$ contains all possible reputation-related factors. $L \subseteq \mathbb{L}$ is a subset of $\mathbb{L}$. L describes the situation where some of the reputation-related factors for services may be unavailable. Let r denote the direct reputation of a service. Our reputation bootstrapping approach aims to identify a function $R(L) = \hat{r}$ to make $\hat{r} \approx r$, where $\hat{r}$ is the bootstrapped reputation of a new service, and is computed based on the reputation-related factors modelled in L. Furthermore, our approach quantitatively determines the importance of factors in reputation bootstrapping. Given the complete framework $\mathbb{L}$, a function $I(\mathbb{L}) = \overrightarrow{i}$ outputs a vector $\overrightarrow{i}$ which contains the importance value of every factor in $\mathbb{L}$. The functions R and I are learned based on the historical records of existing services. A set of features are extracted based on the factor modelled in each layer of $\mathbb{L}$. For example, the factor "provider" is modelled in the provider layer of $\mathbb{L}$, in which several features related to service providers are extracted. These features may include the reputation of a provider, the reputations of the provider's past services, the number of its services, etc. The features in every layer of $\mathbb{L}$ are then collected and trained through a learning method to compute functions R and I.

Function Learning: we apply Random Decision Forest [2], which is an ensemble learning algorithm based on Decision Tree, to determine the functions R and I. The standard Random Forest algorithm is modified to apply to our work. The reason to adopt Random Forest is: (1) in comparison with the complete framework $\mathbb{L}$, some reputation-related factors may be unavailable in a particular case L, i.e., $L \subseteq \mathbb{L}$. Random Forest can naturally handle various cases of incomplete factors through a feature bagging process [2]; (2) Random Forest can easily compute feature importance in a learning process. As a result, the importance of each reputation-related factor can be computed by summing up the importance value of every feature in each layer of $\mathbb{L}$; and (3) the efficiency of Random Forest is very high in training and prediction processes, compared to most of other learning algorithms.

Algorithm 1. Forest Building for Reputation Bootstrapping

```
Input:
    the training set $N$ containing $n$ samples;
    the complete layer-based framework $\mathbb{L}$;
    the set $\{L_i\}$ containing all possible subsets of $\mathbb{L}$, where $L_i \subset \mathbb{L}$;
    the feature set $F$ containing all features in $\mathbb{L}$;
    the feature set $F_i$ for $L_i$, where $F_i \subset F$.
Output: the structure of a decision forest.
for each $L_i \in \{L_i\}$ do
    for $t = 1 \ldots T$ ($T$ is the number of times of bagging.) do
        Sample Bagging: randomly select samples from $N$ with replacement for $n$ times to form a
        new sample set $N_i^t$;
        Feature Bagging: randomly select features from $F_i$ to form a sub feature set $f_i^t$;
        Tree Building: build an unpruned decision tree $tr_i^t$ based on $N_i^t$ and $f_i^t$.
    end for
end for
Build a standard random forest $\{tr^{t'}\}$ containing trees denoted as $tr^{t'}$ based on $N$ and $F$;
return a decision forest $FR$ based on the combination of $\{tr_i^t\}$ and $\{tr^{t'}\}$.
```

In the standard tree bagging process [2] of Random Forest, data samples and data features are randomly selected with replacement from the original dataset. The random selection process is repeated several times to form a number of subsets of data. A decision tree is built based on each subset. All of these trees form a forest. The result of a prediction is the aggregation of the results obtained from all the trees in the forest. The randomness and aggregation in Random Forest improve prediction accuracy and effectively control overfitting. In our work, we modify the standard bagging process in order to deal with various cases of L. For every possible L, a corresponding sub forest is built only based on the reputation-related factors modelled in L before the standard bagging process. If a service can only be modelled in a particular L according to its factors, its future reputation is predicted only through the sub forest that is built on L. For example, a new composite service only has the information of its provider and component services. It is reasonable to bootstrap its reputation only based on a particular L consisting of a provider layer and a component service layer. On the other hand, the importance of all possible reputation-related factors is learned based on the complete framework $\mathbb{L}$.

Compared to standard Random Forest, the bagging process of our modified forest consists of two steps:

1. Build a sub forest for every possible $L \subset \mathbb{L}$, where each sub forest is a standard random forest which is built on L only.
2. Build a standard random forest based on $\mathbb{L}$.

The whole forest for reputation bootstrapping is the combination of the decision trees built in Steps 1&2. Algorithm 1 presents the details of the forest building process. Lines 1–7 describe the sub forest building process in Step 1, where Lines 2–7 is a standard bagging process in Random Forest. Lines 8 and 9 describe the process in Step 2. In the end, a forest-based reputation prediction model is built by training actual data. The sub forest building in Step 1 effectively addresses the real-world situations where only partial reputation-related factors are available for service reputation bootstrapping.

Note that we apply classification trees in our proposed approach rather than regression trees. The reason is that a reputation value of a service usually needs to be mapped into a *trust* degree to describe how possible the service performs satisfactorily. A trust degree is typically represented by a probabilistic tuple (*belief, uncertainty, disbelief*) [5]. In this regard, classification trees are more suitable and easier to map reputation values into a trust degree.

Factor Importance: Random Forest has the ability to rank feature importance in a training process. Given a decision forest FR, the importance of every reputation-related factor is determined by aggregating the importance values of all the features belonging to the factor. In a ready-trained decision tree, every node of the tree contains a part of samples. Except for leaf nodes, every node is split into two child nodes in order to make similar samples stay in the same node. Every split is performed according to a condition on a single feature. The optimal condition is determined by sample "*impurity*", which describes the confusion degree of samples in a node. The training process of a decision tree is to determine how quickly each feature can reduce sample impurity until similar samples stay in the same node. Therefore, in a single decision tree, the unnormalized importance of a feature can be defined as follows:

$$importance = im_n \times s_n - im_l \times s_l - im_r \times s_r, \tag{2}$$

where im denotes impurity values, s denotes the number of samples in a node. n, l and r respectively denote the current node, its left child node and right child node. The impurity values are typically computed through Gini impurity or information gain [2]. The unnormalised importance values are then normalised, i.e., make the sum of the importance of all features equal to 1. The global importance of features in a decision forest is the average of the importance of features computed in every single tree. Let $importance_f$ denote the importance of a factor f in FR built on $\mathbb{L}$; $importance_f^i$ denotes the importance of a feature i belonging to f. The importance of f in reputation bootstrapping is the sum of the importance of all the features belonging to it:

$$importance_f = \sum_i importance_f^i. \tag{3}$$

Through factor importance, the order of layers in $\mathbb{L}$ can be determined. Consequently, the bootstrapped reputations computed based on more important factors are considered more reliable.

3.3 Confidence of Bootstrapped Reputations

We propose $confidence$ to describe how much a bootstrapped reputation is reliable. The confidence of a bootstrapped reputation is denoted as a tuple (a, e), where a represents the overall accuracy of reputation bootstrapping in a particular case (i.e., for a particular L), and e describes the uncertainty of the bootstrapped reputation of a particular service. For example, suppose a new service s only has its provider information and community information. Its reputation-related factors are modelled in a framework L_{pc} which is composed of a provider layer and a community layer. After training a decision forest FR through Algorithm 1, a is the prediction accuracy computed from the sub forest which is built on L_{pc}. The accuracy a describes the general accuracy of reputation bootstrapping in the case of L_{pc}. a is computed as follows:

$$a = \frac{The\ number\ of\ correctly\ predicted\ samples}{The\ total\ number\ of\ samples}. \tag{4}$$

On the other hand, e is computed based on the probability estimate of the bootstrapped reputation of s. In a decision tree, given a particular sample, the probability of every class to which the sample belongs can be estimated. Suppose there exist c classes. After the tree training, a sample s (i.e., service s) is finally classified into a leaf node l. The probability of s belonging to a particular class C is computed using Laplace estimate [9] as follows:

$$Probability\ Estimate = \frac{The\ number\ of\ samples\ belonging\ to\ C\ in\ l + 1}{The\ total\ number\ of\ samples\ in\ l + c}. \tag{5}$$

This probability estimate is suitable for balanced datasets, i.e., the number of samples belonging to each class is approximately equal. In this paper, we use a balanced dataset to evaluate the proposed approach.

In a random forest, the overall probability estimate of every class to which s belongs is the mean probability estimate of all the trees. e describes how

Table 1. Dataset Statistics

Data Type	**Statistic**
The number of repositories	4715
The number of repositories with sub modules	417
The average number of sub modules per repository	1.28
The minimum number of stars a repository obtains	1
The maximum number of stars a repository obtains	9992

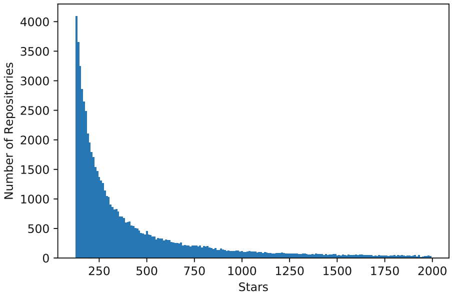

Fig. 4. The Distribution of Stars

certainly the bootstrapped reputation of s belongs to a particular reputation class, and is a necessary amendment for a. For example, suppose there are three reputation classes for services, which are represented by "*bad*", "*fair*" and "*good*". The probability estimate for s is $(0.2, 0.6, 0.2)$. Another new service s' is bootstrapped under the same circumstance. The probability estimate of s' is $(0.1, 0.9, 0)$. Although the predicted reputation class of s and s' is the same (i.e., "*fair*"), the uncertainty of their predictions is not equal. The prediction of s is less reliable than that of s' since the probability of s belonging to "*fair*" is smaller than that of s'. To this end, we use the entropy of a probability estimate to quantitatively describe the uncertainty e:

$$e = -\sum_j p(C_j) \log p(C_j), \tag{6}$$

where $p(C_j)$ denotes the probability estimate of a new service's bootstrapped reputation belonging to a particular reputation class C_j. The higher e is, the more unreliable the bootstrapped reputation is. It should be noted that, the effectiveness of e representing uncertainty of reputation bootstrapping is influenced by the bootstrapping accuracy a. If a is quite low, the effectiveness of e is low as the bootstrapping model learned via Algorithm 1 cannot output correct probability estimates. Even if such a situation occurs, a is still an effective metric to evaluate the confidence of reputation bootstrapping.

4 Experimental Results

We conduct a set of experiments to evaluate the proposed reputation bootstrapping approach. These experiments show: (1) the importance levels of reputation-related factors; (2) reputation bootstrapping accuracy; and (3) the effectiveness of reputation bootstrapping uncertainty e.

4.1 Experiment Setup

Dataset: we collect the data from GitHub via its RESTful API[4] that provides an access to all public repositories. The collected information contains the reputation-related factors which can appear in a general composite service environment. We assume that a repository is a service. If a repository has sub modules, its sub modules are considered as its component services. The multiple contributors of a repository is considered as a whole entity, which is the provider of the repository. At GitHub, every repository has an owner that can be a user or an organisation. An owner can have multiple repositories. We assume that the owner of a repository is a community. In addition, similar repositories with the same keywords can be identified through the semantic search function provided by the API. The keywords are extracted from repository descriptions by removing stop words and duplicated words.

After analysing the GitHub data, we discover that the number of stars of a repository follows a Pareto (long tail) distribution that is illustrated in Fig. 4. As can be seen, most of repositories have been given quite few stars. The number of repositories having a particular number of stars is quite imbalanced. An imbalanced dataset would bring bias in prediction accuracy evaluation. To avoid such bias, we collect the approximately same number of repositories from five different reputation intervals. Every reputation interval represents a reputation class to which a repository can belong. The dataset contains 4715 repositories, in which 417 repositories have sub modules. At GitHub, most of repositories with sub modules have zero star. The repositories with zero star cannot provide effective information. Therefore, we only keep the repositories having at least one star in the dataset. This also indicates the small proportion of repositories with sub modules in the dataset (i.e., only 417 from 4715 repositories). In addition, we find that most of the repositories with sub modules have only one sub module. Table 1 reports the statistics of the GitHub dataset.

We apply the reputation evaluation approach introduced in Sect. 4.1 to compute the reputations of either repositories or users. The actual reputations of repositories are taken as a ground truth to evaluate whether the proposed approach can accurately bootstrap service reputation.

Model Learning: we build a framework based on Fig. 2. The features are extracted from each layer of the framework:

- There is one feature in the provider layer: the average reputation of top-10 contributors of a repository. The reputations of these contributors are weighted by the numbers of their commits.
- There are three features in the community layer: the reputation of the owner of a repository, the average reputation of the owner's other repositories, and whether the owner is a user or an organisation.
- There are two features in the similar service layer: the average reputation of Top-5 similar repositories, and the average reputation of the owners of similar

[4] developer.github.com/v3.

repositories. The reputations of the similar repositories and their owners are weighted by the similarity scores computed by the search API.

- There are two features in the component service layer: the average reputation of sub module repositories, and the average reputation of the owners of sub module repositories.

80% of the dataset forms a training set. The rest forms a test set. We apply Algorithm 1 to build a decision forest. We evaluate the proposed approach by comparing the predicted reputations of repositories and their corresponding actual reputations.

Table 2. The Importance Level of Factors

Features	Normalised Importance
Provider Layer:	**0.238**
Average reputation of contributors	0.238
Community Layer:	**0.722**
Reputation of the owner	0.127
Average reputation of the other repositories of the owner	0.532
User or organisation	0.063
Similar Service Layer:	**0.024**
Average reputation of similar repositories	0.013
Average reputation of the owners of similar repositories	0.011
Component Service Layer	**0.016**
Average reputation of sub modules	0.010
Average reputation of the owners of sub modules	0.006

4.2 Results

Factor Importance: in the first experiment, we explore which factor plays an important role. Table 2 reports the normalised importance of the factors on the GitHub dataset through model learning. The results demonstrate that the factor *community* is the dominant factor to predict service reputation. Consequently, it is more reliable for reputation bootstrapping of new services. In contrast, the other factors are insignificant. An interesting finding is that, in the provider layer, the reputations of the contributors of a repository do not directly influence the reputation of the repository. The importance of the factor *provider* is only 0.238. In addition, the reputation of the owner of a repository also has low importance (0.127). On the contrary, the reputations of the other repositories of the owner has very high importance (0.532). This phenomenon may be caused by the starring system at GitHub. The stars given to a user may more greatly reflect his/her social relations (following or followed at GitHub) rather than his/her reputation on project development. Instead, his/her past experiences more effectively reflect his/her ability on providing valuable repositories.

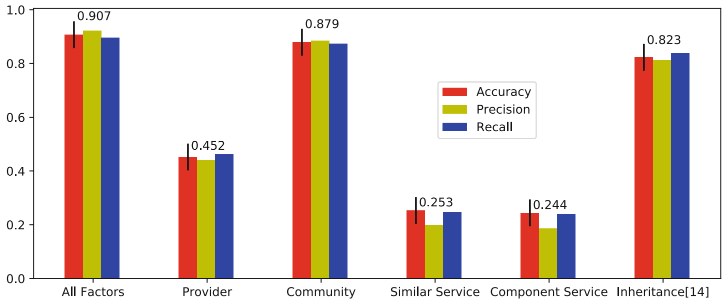

Fig. 5. User Starring Network

The experimental results also show that the factors *similar service* and *component service* have quite low importance in reputation prediction. The possible reasons may include: (1) semantic similarity cannot be applied to group repositories with similar reputations; and (2) the reputation of a composite repository is more influenced by its own developers rather than its sub module repositories.

Bootstrapping Accuracy: In the second experiment, we evaluate the reputation bootstrapping accuracy of our approach. We apply the proposed approach in five cases, i.e., consider all the reputation-related factors and consider every single factor (*provider, community, similar service* and *component service*). In addition, we compare our approach to a baseline approach that applies the *inheritance mechanism* proposed in [11]. The baseline approach uses the past reputations of the existing services of a provider to bootstrap the reputation of the provider's new service.

We use the metrics *accuracy*, *precision* and *recall* to illustrate the comparison results. Figure 5 demonstrates that the accuracy of our approach is quite low in terms of the factors *similar service* and *component service*. The accuracy is slightly higher than that of random guessing (approximate 0.2 due to five classes with approximately equal size). The accuracy in terms of the factor *provider* is higher, but only 0.452. As the dominant factor, the accuracy in terms of *community* is much higher and reaches approximate 0.88. In addition, the baseline approach is equivalent to reputation bootstrapping based on the feature *average reputation of the other repositories of the owner*. This feature is in the community layer, and its importance is very high (0.532). Therefore, the bootstrapping accuracy of the baseline approach reaches approximate 0.82. However, it is still 8% lower than the accuracy of our approach in terms of all the factors (0.907). The comparison results demonstrate that the more important a reputation-related factor is, the more accurate the reputation bootstrapping in terms of the factor is. Compared to the baseline approach which only takes a single factor into account, our reputation bootstrapping approach considers multiple factors and is able to identify the most important factor. Therefore, our approach is more adaptable under diverse circumstances.

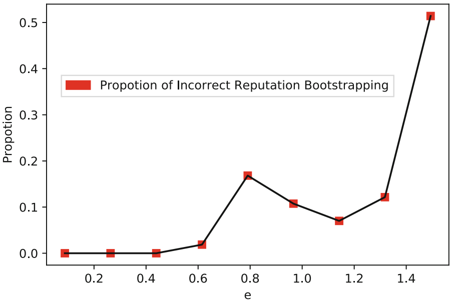

Fig. 6. Evaluation of e

Evaluation of e: In the last experiment, we evaluate the effectiveness of the proposed bootstrapping uncertainty e. For every sample in the dataset, we conduct reputation bootstrapping and compute its e using Eq. (6). All the e values are sorted in a descending order. The maximum and minimum values of e are used to build an uncertainty interval $[Min(e), Max(e)]$. The interval is equally divided into several sub intervals. We collect all the samples whose reputations are not correctly bootstrapped. The number of incorrectly reputation bootstrapping in every uncertainty sub interval is counted. We compute the proportion of incorrectly reputation bootstrapping in each sub interval over the total number of incorrectly bootstrapping. The proportions are shown in Fig. 6. The results demonstrate that most of incorrectly bootstrapping has a high value of e. Over 99% of incorrectly bootstrapping has an e over 0.5. The overall trend indicates that the higher an uncertainty e is, the more unreliable a bootstrapped reputation is. Although the trend fluctuates in particular cases due to reputation prediction errors, the overall trend remains stable.

5 Related Work

The approaches of reputation or trust bootstrapping are typically classified into three categories: *characteristic*-based, *guarantee*-based and *trial*-based approaches. We briefly overview the principal related work in these three areas.

Characteristic-based Approaches: this category of approaches focuses on predicting a new service's future reputation via its reputation-related characteristics. In [11], a reputation bootstrapping model is proposed through three mechanisms: *inheritance*, *referral* and *guarantee*. The inheritance mechanism uses provider reputation to predict service reputation; the referral mechanism uses community reputation to estimate service reputation; the guarantee mechanism is a guarantee-based approach which allows a new service to provide a commitment for its future performance. In [1], a trust bootstrapping approach is proposed in a multi-agent environment based on the notion *stereotype*. A stereotype is learned from past experiences to describe the correlation between an agent's characteristics and its expected probability of good performance. This approach does not consider concrete characteristics and their corresponding importance in trust bootstrapping. In [14], a trust bootstrapping approach is proposed for Web services based on a tagging system. The system allows users to tag different services which they are interested in. Therefore, similar services can be identified through the tagging system. The trustworthiness of a new service is predicted according to the similarity of other services with common tags. In addition, some approaches assign a single population statistic as the bootstrapped reputation or trust for every new entity. In [3] and [16], the mean trust value and the minimum trust value of the whole system is assigned to every newcomer respectively. None of the above studies take factor/characteristic importance and bootstrapping confidence into account.

Guarantee-based Approaches: this category of approaches allows a newcomer to provide evidence to guarantee that it will offer good performance. The guarantee can be the referral from other trustworthy parties [4,8,10]. The referral also requires past transactions between newcomers and the trustworthy parties. However, this requirement can be hardly meet in practice as newcomers may be quite new without any historical transaction records. Another way to obtain a guarantee is to ask a newcomer to offer a monetary commitment before transactions [6,11]. In such a case, if the newcomer performs unsatisfactorily, it will lose money. This also requires a centralised authority to manage monetary commitments.

Trial-based Approaches: this category of approaches gives a newcomer a trial period to build its reputation. In this period, newcomers are allowed to make transactions with other parties under some restrictions. In [8], a newcomer can only make transactions with the selected parties that have high credibility. In addition, the full transaction payment can be obtained only when the trial period finishes. The newcomer's reputation is then computed based on its performance during the trial period. In [15], the trust patterns of service performance are first modelled through Hidden Markov Model (HMM) based on the prior observations of the entire service population. The performance of a new service is then evaluated during a trial period to obtain its specific trust pattern.

Our proposed approach is classified into the characteristic-based category. Compared to the other two categories, it requires no extra process (e.g., commitment management or a trial period). As a result, it is more practical and easier to achieve in real-world situations.

6 Conclusion

This paper proposed a novel reputation bootstrapping approach in composite service environments. The proposed approach is based on a number of factors which may implicitly reflect new services' future reputations. We introduced a layer-based framework where the importance of these factors are modelled. A data-driven approach based on a modified version of Random Forest was proposed to quantitatively determine the importance of the factors and predict new services' reputations. The proposed framework can also be extended to a general case, and thus can effectively deal with diverse reputation-related factors in real-world situations. In addition, the notion *confidence* was proposed to describe the reliability of bootstrapped reputations. In our experiments, we demonstrated the effectiveness of our approach using a GitHub dataset.

Acknowledgement. This research was made possible by DP150100149 grant from Australian Research Council. The statements made herein are solely the responsibility of the authors.

References

1. Burnett, C., Norman, T.J., Sycara, K.P.: Bootstrapping trust evaluations through stereotypes. In: 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 241–248 (2010)
2. Ho, T.K.: Random decision forests. In: 3rd International Conference on Document Analysis and Recognition (ICDAR), pp. 278–282 (1995)
3. Huang, K., Liu, Y., Nepal, S., Fan, Y., Chen, S., Tan, W.: A novel equitable trustworthy mechanism for service recommendation in the evolving service ecosystem. In: Franch, X., Ghose, A.K., Lewis, G.A., Bhiri, S. (eds.) ICSOC 2014. LNCS, vol. 8831, pp. 510–517. Springer, Heidelberg (2014). doi:10.1007/978-3-662-45391-9_43
4. Huynh, T.D., Jennings, N.R., Shadbolt, N.R.: Certified reputation: how an agent can trust a stranger. In: 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1217–1224 (2006)
5. Ismail, R., Jøsang, A.: The beta reputation system. In: 15th Bled eConference: eReality: Constructing the eEconomy, pp. 324–337 (2002)
6. Jiao, H., Liu, J., Li, J., Liu, C.: A framework for reputation bootstrapping based on reputation utility and game theories. In: IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 344–351 (2011)
7. Malik, Z., Bouguettaya, A.: Rateweb: reputation assessment for trust establishment among web services. VLDB J. **18**(4), 885–911 (2009)
8. Malik, Z., Bouguettaya, A.: Reputation bootstrapping for trust establishment among web services. IEEE Internet Comput. **13**(1), 40–47 (2009)
9. Margineantu, D.D., Dietterich, T.G.: Improved class probability estimates from decision tree models. Nonlinear Estimation Classif. **171**, 173–188 (2003)
10. Maximilien, E.M., Singh, M.P.: Reputation and endorsement for web services. SIGecom Exchanges **3**(1), 24–31 (2002)
11. Nguyen, H.T., Yang, J., Zhao, W.: Bootstrapping trust and reputation for web services. In: 14th IEEE International Conference on Commerce and Enterprise Computing (CEC), pp. 41–48 (2012)
12. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. In: 7th International World Wide Web Conference, pp. 161–172 (1998)
13. Qu, L., Wang, Y., Orgun, M.A., Liu, L., Liu, H., Bouguettaya, A.: CCCloud: Context-aware and credible cloud service selection based on subjective assessment and objective assessment. IEEE Trans. Serv. Comput. **8**(3), 369–383 (2015)

14. Skopik, F., Schall, D., Dustdar, S.: Start trusting strangers? bootstrapping and prediction of trust. In: Vossen, G., Long, D.D.E., Yu, J.X. (eds.) WISE 2009. LNCS, vol. 5802, pp. 275–289. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04409-0_30
15. Yahyaoui, H., Zhioua, S.: Bootstrapping trust of web services through behavior observation. In: Auer, S., Díaz, O., Papadopoulos, G.A. (eds.) ICWE 2011. LNCS, vol. 6757, pp. 319–330. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22233-7_22
16. Zacharia, G., Moukas, A., Maes, P.: Collaborative reputation mechanisms for electronic marketplaces. Decis. Support Syst. **29**(4), 371–388 (2000)