Information Sciences 527 (2020) 159–175

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

An endorsement-based trust bootstrapping approach for newcomer cloud services

Omar Abdel Wahab[a,*], Robin Cohen[b], Jamal Bentahar[c], Hadi Otrok[d], Azzam Mourad[e], Gaith Rjoub[c]

[a] Department of Computer Science and Engineering, Université du Québec en Outaouais, Gatineau, Canada
[b] David R. Cheriton School of Computer Science, University of Waterloo, Canada
[c] Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Canada
[d] Department of ECE, Khalifa University of Science and Technology, United Arab Emirates
[e] Department of Computer Science and Mathematics, Lebanese American University, Lebanon

A R T I C L E   I N F O

Article history:
Received 5 September 2019
Revised 23 January 2020
Accepted 28 March 2020
Available online 5 April 2020

Keywords:
Trust bootstrapping
Cloud computing
Machine learning
Dishonesty discouragement

A B S T R A C T

This paper addresses the challenge of providing trustworthy recommendations on newly deployed cloud services/resources for which little or no evidence about their trustworthiness is available. We also provide a two-level dishonesty discouragement mechanism to fight against unfair recommendations at both the collection and aggregation levels. Our solution consists of a (1) mechanism to allow users to self-assess the accuracy of their recommendations and autonomously decide on whether to participate in the recommendation process or not, (2) machine learning technique that generates reliable endorsements on newcomer items through extracting hidden similarities among the specifications of new and existing ones, (3) dishonesty-aware aggregation technique for endorsements coming from multiple advisors, (4) credibility update mechanism that captures the dynamism in the endorsers' credibility, and (5) incentive mechanism to motivate advisors to participate in the endorsement process. Experiments conducted on the CloudHarmony and Epinions datasets show that our solution improves the accuracy of classifying newly deployed cloud services and yields better performance in protecting the recommendation process against Sybil attacks, in comparison with four existing recommendation approaches.

1. Introduction

With the widespread expansion of online shopping and the booming number of advertisements and promotional messages that target every single person, more and more people are relying on other buyers' opinions and reviews prior to filling their shopping cart. For example, a recent study conducted by *Researchscape International*[1] in 2018 revealed that 45% of consumers are more likely to shop on a Web site that offers personalized recommendations, and that 56% of online shoppers are more likely to return to a site that offers product recommendations. Cloud computing, a revolutionary computing paradigm for offering software and hardware over Internet in the form of virtualized resources, is not immune to this trend.

* Corresponding author.
E-mail addresses: omar.abdulwahab@uqo.ca (O.A. Wahab), rcohen@uwaterloo.ca (R. Cohen), bentahar@ciise.concordia.ca (J. Bentahar), Hadi.Otrok@ku.ac.ae (H. Otrok), azzam.mourad@lau.edu.lb (A. Mourad), g_rjoub@encs.concordia.ca (G. Rjoub).

[1] https://www.evergage.com/wp-content/uploads/2018/04/Evergage-2018-Trends-in-Personalization-Survey.pdf.

https://doi.org/10.1016/j.ins.2020.03.102
0020-0255/© 2020 Elsevier Inc. All rights reserved.

Specifically, with the huge number of cloud services (e.g., network storage, application hosting) that are available online, users are increasingly relying on online reviews to filter out their choices of cloud resources.

This has pushed the research community to design and implement plenty of recommender systems to help users differentiate among online items and services on the basis of their previous performance/quality. Traditionally, recommender systems analyze users' profiles and service/items' specifications and try to find the best matches. It can be argued, however, that the role of recommender systems should go further to investigate the trustworthiness of items prior to recommendation [1,11,30]. Failing to do so might lead to recommending items/services of poor performance or even ones that result in harm, thus decreasing the credibility and reputation of the underlying recommender system for users. Trust is being increasingly adopted to assist recommender systems in providing more reliable decisions for users [33,38,39], especially in contexts where peer advice is employed [9,11], both to prevent reliance on properly measuring user similarity [27] and to address cold start challenges by inferring user preferences from trusted neighbours [8]. Nonetheless, current trust solutions leave some problems unsolved, particularly when it comes to recommending newly deployed items for which no evidence about their former behaviour is available. To better highlight this problem and the importance of solving it, we give in the following a real-life motivating example.

Motivating Example. Consider a cloud-based central data analytics service which analyzes patients' data gathered from a number of local cloud-based services around the World. In such a scenario, the overall quality of the analytics process highly depends on the quality of the data provided by each single local server. Therefore, selecting trustworthy servers to get data from is of prime importance to maximize the credibility of the whole analytics process. Imagine now a scenario in which the central server has the choice to get data from service *x* which is newly deployed and has no previous interactions. In case the server decides to get data from *x*, it risks dealing with a completely bad service, thus endangering the quality of the data analytics process. In case the server decides not to get the data from *x*, it might be depriving itself from high-quality data that consist of worthy features. Moreover, if all the parties refrain from collaborating with newly deployed cloud services due to their uncertainty, such newcomer servers will never have the chance to collaborate and build a trust record. Therefore, it is of prime importance for both existing and newcomer cloud services to come up with a trust bootstrapping approach which helps derive accurate initial trust scores for newly deployed services in order to alleviate uncertainty and help both parties make more thoughtful decisions.

1.1. Problem statement

Numerous approaches have been proposed to derive recommendations on cloud services. The main idea of the existing approaches [2] for deriving the recommendation on a certain service is to either map the performance registry of the underlying service to the users' profiles and preferences or to capitalize on the collaborative filtering approach [12] to derive the similarity in terms of users' preferences and capitalize on this similarity to predict the corresponding preferences. However, the performance of these approaches degrades when newly deployed cloud services are encountered. In other words, when no or little data on the cloud services can be found, it becomes quite hard for these approaches to derive meaningful insights on the potential performance of such services. Moreover, in extreme scenarios wherein newly registered users seek recommendations on newly deployed services, these approaches fail to learn the preferences of the newly registered users, which makes them unable to capture potential similarities with existing users. To address this challenge, we propose in this paper a trust bootstrapping approach for newly deployed cloud services. The proposed approach operates effectively even in such challenging scenarios wherein newly registered users seek recommendations on newly deployed cloud services.

Moving to the dishonesty discouragement part, which is a building block in our approach to guarantee the authenticity of the derived recommendations, the current literature can be classified into two categories, i.e., collection-level and processing-level approaches. In the collection-level approaches, the recommender system derives the optimal network of advisors that maximizes trust and propagates the recommendation requests through it. The problem of these approaches is that they overlook the self-willingness and self-confidence of the advisors in submitting recommendations. In other words, although some advisors might be highly trusted in general, this does not mean that they will be providing accurate recommendations for all types of requests. The accuracy here might vary according to the data available to these advisors, the characteristics of the services being recommended, and the technique used to compute recommendations. To tackle this challenge, we leave in this work the choice for the consulted advisors to self-asses their own ability in participating in the recommendation process or not. In addition, we require the advisors to use a common recommendation computation technique (i.e., decision tree) to increase the homogeneity of the received recommendations. The second limitation of collection-level approaches is that when the number of advisors is large, it becomes quite hard to design an efficient optimal network selection algorithm in a reasonable time.

Processing-level approaches [30] employ some aggregation techniques to derive aggregate recommendation decisions that are resilient to dishonesty. The problem with these approaches is that they ignore the collection level and leave all the dishonesty discouragement responsibilities for the aggregation model, which might be vulnerable to manipulation especially in cases in which dishonest advisors form the majority, a case that is likely to arise due to a lack of any countermeasure at the collection level. To address this challenge, our solution operates at the collection level as well to allow honest advisors that have low accuracy for a certain recommendation request to self-withdraw in order to facilitate the detection of dishonest recommendations. Moreover, we propose at the processing level an aggregation technique coupled with a credibility update mechanism to better protect against manipulations.

1.2. Contributions

The proposed solution is composed of four principal phases: endorsements collection, dishonesty-aware aggregation, credibility update, and participation motivation. In the endorsements collection phase, advisors (i.e., cloud users) are asked by the underlying recommender system to run a decision tree machine learning technique on their datasets to derive appropriate recommendations. Specifying the machine learning technique to be used is helpful in our solution to guarantee the homogeneity in terms of opinions origins. Leaving this decision for the advisors might lead to inconsistent decisions that are biased towards some machine learning algorithms, dataset size, number of dimensions in the dataset, etc. Our choice of decision tree stems from its lightweight nature which makes it suitable for situations wherein resource-constrained devices exist[2]. Interested advisors (e.g., those that have enough resources) train the decision tree algorithm on their datasets which record their previous interactions with services of different specifications and behavior and decide, based on the obtained accuracy level, on whether to submit their opinions or not[3].

In the dishonesty-aware aggregation phase, different endorsements from different advisors are aggregated by the recommender system using the Dempster-Shafer Theory (DST) of evidence [36] to arrive at final aggregate initial trust scores. The endorsements aggregation technique takes into consideration the dishonest endorsements that might be submitted to promote/demote some newcomer services. In the credibility update phase, we propose a mechanism to update the credibility scores of the advisors that participate in the endorsements collection phase in order to ensure the authenticity of the bootstrapping process. Finally, in the incentive mechanism, we link the number of endorsement requests that each user is allowed to make regarding newcomer services from any other user to the level of her contribution in responding to that user's requests, proportionally to her credibility score in terms of providing honest endorsements. In summary, the main contributions of this paper are:

- Proposing a trust bootstrapping mechanism to provide recommendations on newly deployed cloud services for which no or little data on their former behavior/performance is available.
- Proposing a dishonesty discouragement mechanism that operates at both the recommendations collection and processing levels to provide effective protection against dishonest endorsements that seek to manipulate the recommendation process.
- Designing our solution in such a way to handle situations wherein users need urgent endorsements as well as non-urgent endorsements.

1.3. Organization

Section 2 reviews relevant related work and highlights the originality of our work compared to the state-of-the-art. In Section 3, we discuss the details of the proposed recommendation bootstrapping solution. In Section 4, we present experimental results and empirical analysis. Finally, we summarize the main findings of the paper in Section 5.

2. Related work

We present in the following relevant related work and highlight the originality of our solution.

2.1. Recommendation systems for cloud computing

The existing recommendation approaches proposed for cloud computing environments can be classified into two main categories, i.e., collaborative filtering and knowledge-based approaches.

2.1.1. Collaborative filtering approaches

Collaborative filtering approaches seek to find recommendations on newly deployed cloud services based on the similarity between the features of these services and those of existing services. The collaborative filtering process consists of two phases. In the first phase, the recommender system builds a model to derive the similarity between all pairs of services. In the second phase, the recommender system capitalizes on the most similar services to a user's already-rated items to create a list of recommendations for that user. For example, in [26], the authors adopt the collaborative filtering approach to derive recommendations for cloud services. They propose a solution based on the lattice theory in which the description of the cloud environment (e.g., users, ratings, services) is represented via the lattice representation and recommendations are then generated through extracting relevant information from the lattice (e.g., users that are similar to an active user, top services, etc.). In [29], the authors aim to improve the accuracy of cloud services recommendation while considering multi-source quality data coming from multiple cloud provides (e.g., Amazon and IBM). Two challenges are mainly addressed in such a

[2] Examples of resource-constrained environments include cloud users using their battery-constrained smartphones to run the decision tree technique.

[3] The decision of refraining from submitting opinions stems mainly from a low accuracy level obtained by the machine learning classifier. Such a decision both helps advisors to preserve their credibility toward endorsement requestors and the underlying recommender system to maintain the accuracy of its aggregate decisions.

scenario, which are: (1) protecting the privacy of each provider's users (from the rest of the collaborators) and (2) ensuring the scalability and efficiency of the recommendations with the frequent update of quality data for each provider. To answer these challenges, the authors propose a service recommendation approach based on distributed locality-sensitive hashing to ensure privacy-preserving and scalable recommendations in distributed cloud environments. The authors of [5] address the challenge of the frequent change in the QoS of cloud services over time and propose a time-aware service recommendation approach. The approach consists of a (1) similarity-enhanced collaborative filtering technique to capture the time feature of user similarity and address the data sparsity problem in the existing PITs (Point In Time) and (2) autoregressive integrated moving average model (ARIMA) to predict the QoS values in the future PIT under QoS instantaneity.

The main limitation of collaborative filtering approaches is that their performance degrades in extreme scenarios wherein newly registered users seek recommendations on newly deployed services. In such a case, these approaches fail to learn the preferences of the newly registered users and are hence unable to capitalize on the similarities between the specifications of existing and newly deployed services to generate appropriate recommendations. To address this problem, we propose in this paper a bootstrapping approach that does not rely on users' past preferences to derive recommendations on newcomer cloud services.

2.1.2. *Knowledge-based approaches*

As opposed to collaborative filtering systems, a knowledge-based recommendation system does not rely on users or services' rating history to generate recommendations but instead it prompts the user to answer a series of questions based on which the system then searches through its database to return the services that best match with the users' answers. For example, in [45], the authors propose CloudRecommender, a declarative recommendation system for Infrastructure-as-a-Service (IaaS) resources, which automates the process of mapping users' requirements to IaaS configurations. The IaaS resource configurations are represented using ontology and implemented through a structured data model that can be controlled using regular expressions and the Structured Query Language (SQL). A keyword-aware service recommendation approach called *KASR* is proposed in [25]. The approach analyzes keywords to infer users' preferences. Thereafter, an algorithm is proposed to accordingly produce appropriate recommendations. The proposed approach is implemented on Hadoop using the MapReduce programming model to boost its efficiency and scalability. The authors of [10] propose an approach that aids users in selecting services from different cloud providers that meet their requirements. In the proposed approach, the recommendation system relies on Quality of Service (QoS) metrics and Virtual Machine (VM) platform factors of different providers to rank the different cloud services and recommend the appropriate ones to the users. The approach proposed in [43] allows users to specify their perception of quality criteria and then advances a clustering technique from data mining to classify cloud services into several clusters on the basis of the input criteria and rank them accordingly. The authors of [6] advance an agility-oriented and fuzziness-embedded cloud service ontology model. The model captures each service's specifications through analyzing the interactions among service of different categories and abstraction scales, thus enabling prototype-based service recommendation system. In [44], the authors propose a real-time QoS-aware multi-criteria decision-making approach that supports the recommendation of next-generation applications such as online interactive gaming and large-scale sensor analytics. The proposed approach is beneficial for helping select IaaS resources, while allowing users to define various design-time and real-time QoS requirements.

The main limitation of knowledge-based recommendation approaches is that they heavily rely on users' answers to generate recommendations. However, users may sometimes enter some random answers by ignorance or by laziness, which would sway the recommendation decisions generated by the recommender system. Even more importantly, some malicious users might take advantage of knowledge-based systems to intentionally submit malicious answers to attack the underlying recommender system (e.g., SQL injection). To avoid such situations, our approach does not impose any additional burden on users nor does it rely on users' honesty to derive trust scores for newcomer cloud services.

2.2. *Trust bootstrapping*

Trust bootstrapping in the domain of cloud computing refers to the problem of assigning initial trust scores to newly deployed cloud services for which no record about their former behavior is available. The existing bootstrapping solutions in the domain of cloud computing can be categorized into three main classes: (1) Default-value; (2) punishment-based; and (3) adaptive. Default-based approaches assign a default trust score for each newcomer cloud service. The main limitation of this approach is that it might arbitrarily favor either newcomer services (in case the assigned default value is high) or existing services (in case the assigned default value is low). This would encourage malicious providers to constantly deploy services with new identities with the aim of eliminating their past poor trust history. Such a misbehavior is called *whitewashing* attack. To counter whitewashing, the punishment-based approach assigns low trust values for newcomer cloud services. The main drawback of this approach is that it disfavors newly deployed services through depriving them from interacting with other services and building their trust record. The adaptive bootstrapping strategy capitalizes on the concept of collaborative filtering through measuring the similarity between the newly deployed services and some existing services to approximate the initial trust scores for the newcomer ones. For example, the authors of [4] advanced a bootstrapping mechanism that is inspired by human organizational behaviour. In this approach, agents learn stereotypes from interactions with familiar partners and employ them to assign initial trust values for new and unknown partners. The trust prediction problem is modeled as a regression problem whose inputs are sets of observable binary feature variables, where each feature denotes whether the trustee has a particular asset or pertains to a particular organisation. In [21], the authors addressed the trust bootstrapping problem in the Service-Oriented Computing domain and proposed two approaches to compute initial reputation values for the newly deployed services. The first approach capitalizes on the cooperation among services in a community-based context to derive initial reputation values. Specifically, consumers bootstrap newcomer services proportionally to the rate of maliciousness of the community to which services belong. In the second approach, community providers are asked to assess newcomer services for a certain period of time and assign them initial reputation values accordingly. In [37], the authors capitalize on the concepts of trust mirroring and trust teleportation to handle the bootstrapping problem. Using the first concept, agents' similarities, capabilities, and roles are used to predict their future trust. In particular, if agent x realizes, based on previous interactions, that agent y has analogous interests and opinions as those of x, then x is likely to trust y's behavior in the future. Using the teleportation concept, agent x will trust all the agents that have similar capabilities and interests as another agent z that is already trusted by x. The authors of [28] proposed a bootstrapping solution that uses agents' observable features (a.k.a tags). Specifically, the distance between the values of the different tags are computed to quantify how behaviourally similar the new agents are to the existing ones. Once enough interactions and experiences are available, a trust assessment method is advanced to evaluate the trust scores of the agents during their lifetime based on their actual behavior. In [8], the authors assume that users have a network of trusted peers and combine the opinions of these advisors, averaging their ratings on commonly rated items. Trust and social similarity are merged to represent the active user's preferences and generate appropriate recommendations, for the case when little is known about the user. In [23], the authors employ collaborative filtering to ease the recommendation process using a two-stage methodology. In the first stage, users are represented in the form of a social network graph and the task is to collect trust statements regarding newcomer users. In the second stage, all the trust statements are analyzed and aggregated using the averaging technique to predict the trust scores of the newcomer users. In [20], the authors propose a three-phase approach to address the problem of cold-start users. In the first phase, the C4.5 and Naive Bayes techniques are employed to assign new users to specific groups. In the second phase, an algorithm is proposed to explore the neighbors of the new user and an equation is presented to compute the similarity between new users and their neighbors in terms of characteristics. In the final phase, a prediction method is used to estimate the final rating of the new user for every existing item, where the rating is a weighted sum of ratings submitted by the user's neighbors on the corresponding item.

The main drawback of the adaptive approach is that is still vulnerable to whitewashing where malicious providers can inject some well-behaving services to gain high trust values. Thereafter, these providers will deploy malicious services with similar features to the already deployed ones. These malicious services will benefit from the high trust rate of their predecessors to gain high trust scores and initiate their malicious behavior. This type of attacks can be referred to as a *camouflage* attack. To counter such an attack, we propose in this paper a two-level dishonesty discouragement solution that operates at both the trust recommendation collection and aggregation levels. At the collection level, we allow advisors to self-assess their aptitude for participating in the recommendation process. At the aggregation level, we propose a credibility-based trust aggregation techniques that capitalizes on DST to fight against dishonest recommendations. We also evaluate in Section 4 the effectiveness of our solution with regard to camouflage attacks.

2.3. Discussion and unique features of our solution

The main idea of the aforementioned approaches for deriving recommendations is to either map the performance history of the cloud services to the profiles of the users or to employ the collaborative filtering approach to measure the similarity in terms of users' preferences. Nonetheless, the performance of these approaches degrades when newly deployed cloud services are encountered. In other words, when no or little data on the cloud services are available, these approaches struggle to derive meaningful insights on the potential performance of these services. Moreover, in extreme scenarios wherein newly registered users seek recommendations on newly deployed services, these approaches fail to learn the preferences of the newly registered users and are hence unable to capture potential similarities with existing users. To address this problem, we propose in this paper a trust bootstrapping approach for newly deployed cloud services that works effectively even in such challenging scenarios when newly registered users seeks recommendations on newly deployed cloud services.

Moving to the dishonesty discouragement part, the primary contrast with the aforementioned models can be outlined in three main points. First, the discussed approaches deal with dishonest recommendations at either the collection or processing level. Our solution operates on both levels to increase the protection against dishonest recommendations. Second, in the approaches that operate at the collection level, the choice of evaluating the adequacy of the advisors (e.g., cloud users) in participating in the recommendation process is left only to the recommender system through computing the optimal network of advisors that maximize trust, without accounting for the self-willingness and self-confidence of the advisors themselves. In other words, although some advisors might be highly trusted in general, this does not mean that they will be providing accurate recommendations for all types of requests. The accuracy here might vary according to the data available to these advisors, the characteristics of the services being recommended, and the technique used to compute recommendations. To tackle this challenge, we leave in this work the choice for the advisors to self-assess their own ability in participating in the recommendation process or not. In addition, we ask the advisors to use a common recommendation computation technique (i.e., decision tree) to increase the homogeneity of the received recommendations. Third, different from the literature which employs aggregation techniques that might be vulnerable to manipulation [11], we take advantage of DST to perform the aggregation in a manner that is sensitive to possible dishonesty even in extreme cases wherein dishonest agents might be the majority. This formulation is described in Section 3.3.

3. Trust bootstrapping for newcomer cloud services

In this section, we give the details of our bootstrapping solution and explain its main phases.

3.1. Solution overview

The proposed solution can be summarized as follows. Users maintain a dataset which corroborates its previous interactions with cloud services of different specifications and behavior. These users can be companies, hospitals, smartphones, Internet of Things (IoT) devices or any other type of devices that are charged, plugged-in, and on an unmetered wi-fi connection. For example, in a cloud-based healthcare management application, data owners are mainly hospitals, doctors' offices, home-based devices and patients' smartphones. This scenario complies well with the future trend in machine learning which is increasingly heading toward federated distributed learning [24] in which users volunteer to distributively participate in the training process on their own local data. This dataset is assumed to be labelled in the sense that the user would classify each interaction as being either trustworthy or untrustworthy based on the degree of his/her satisfaction on the behavior of the items dealt with during the interaction. Upon the receipt of a bootstrapping request from the recommender system to submit an endorsement on a newly deployed cloud service i in favor of user u_1, user u_2 has the choice to decide on whether to participate in the bootstrapping process or not. If user u_2 accepts to participate, she will train a decision tree machine learning classifier on her dataset to predict the trustworthiness of service i based on the potential similarities between the specifications of i and the specifications of the services that user u_2 has previously dealt with. (i.e., content-based recommendation). The output of the decision tree is a label indicating whether the service being bootstrapped is trustworthy or untrustworthy. Based on the results of the decision tree classifier, user u_2 endorses service i as being (potentially) either trustworthy or untrustworthy. To avoid biased endorsements, the recommender system collects endorsements from multiple users and aggregates them using DST to come up with a final aggregate decision that is resilient to dishonesty. Since the performance of DST is greatly dependent on the credibility of the parties giving the judgements, the final step involves updating the credibility scores of the participating users on the basis of the convergence/divergence of their opinions w.r.t the final judgement given by DST. To incentivize users to submit endorsements, we restrict each user to a limited number of inquiries it can make initially, where this number is increased whenever the user participates in an endorsement process, provided that it also maintains a good credibility score. Thus, over time, the users that refrain from reporting will get their inquiries drained and **will** be unable to make further requests. The proposed bootstrapping solution is depicted in Algorithm 1.

3.2. Endorsements collection

Whenever a user encounters a newly deployed cloud service for which no reviews about its behavior and performance are available, she sends a bootstrapping request to the underlying recommender system. This recommender system runs Algorithm 1 to obtain appropriate trust values on that service. The computational complexity of the proposed solution (i.e., Algorithm 1) can be divided into two parts, i.e., complexity on the recommender system and complexity on the users participating in the bootstrapping process. Starting with the complexity on the recommender system, steps 5–8 can be executed by the recommender system in constant time, i.e., $\mathcal{O}(1)$. The main complexity on the recommender system's side lies in steps 16 and 17, which can be executed with a complexity of $\mathcal{O}(2^m)$ [19], with m being the number of basic probability assignments being combined. Steps 18–21 can be executed in constant time, i.e., $\mathcal{O}(1)$.Steps 23–24 can be executed in linear time, i.e., $\mathcal{O}(|U|)$ with |U| being the number of users participating in the bootstrapping process. Thus, the overall complexity on the recommender system's side is $\mathcal{O}(2^m) + \mathcal{O}(1) + \mathcal{O}(|U|) = \mathcal{O}(2^m + |U|)$. Moving to the users' side, step 12 entails a computational complexity of $\mathcal{O}(n^2 p)$ for training the decision tree model and step 13 entails a complexity of $\mathcal{O}(p)$ for predicting the class labels, where n is the number of training observations and p is the number of features in the user's dataset. Step 14 can be executed in a constant time, i.e., $\mathcal{O}(1)$. Thus, the overall complexity on each users is $\mathcal{O}(n^2 p) + \mathcal{O}(p) + \mathcal{O}(1) = \mathcal{O}(n^2 p + p)$.

We distinguish between two types of bootstrapping requests, i.e., *urgent* requests and *non-urgent* requests. This classification is important to cover both users who need immediate recommendations to complete their transactions while online and those who prefer to make their choices at later stages. Specifically, urgent requests are those requests for which the user wants a prompt answer to complete her transaction immediately. On the other hand, for non-urgent requests, the user can wait for a certain time before receiving the bootstrapping endorsements. In the case of urgent requests wherein a prompt answer is needed, only online or active users at the moment when the request is sent are consulted by the recommender system. In the case of non-urgent requests, the online and offline users that are stored in the recommender system's database are consulted.

When a user receives a bootstrapping request, she has the choice either to participate in the process or not. This voluntary aspect of participation is a building block in our bootstrapping mechanism to ensure the fairness of the bootstrapping process for both bootstrapping users and bootstrapped service. Specifically, some users might not be willing to spend some

Algorithm 1: Trust bootstrapping algorithm.

```
1:  **Input**: Newly cloud service $i$
2:  **Input**: Request type $T \in \{\text{urgent}, \text{non-urgent}\}$
3:  **Input**: Set $U$ of users participating in the bootstrapping process
4:  **Output**: Endorsement $R(r, i)$ of recommender system $r$ on item $i$
4:  **procedure** TrustBootstrapping
5:      **if** $T = \text{urgent}$ **then**
6:          Send bootstrapping request to currently online users
7:      **else**
8:          Broadcast bootstrapping request to online and offline users
9:      **end if**
10:     // Bootstrapping process
11:     **for** each user $u \in U$ **do**
12:         User $u$ builds a decision tree classifier using Eqs. (1), (2), and (3)
13:         User $u$ derives the endorsement of cloud service $i$ using decision tree
14:         User $u$ submits the endorsement if the obtained accuracy is sufficient
15:     **end for**
16:     Use Eq. (4) to compute belief in $i$'s trustworthiness $\theta_r^i(T)$
17:     Use Eq. (5) to compute belief in $i$'s untrustworthiness $\theta_r^i(N)$
18:     **if** $\theta_r^i(T) > \theta_r^i(N)$ **then**
19:          $R(r, i)$=trustworthy
20:     **else**
21:          $R(r, i)$=untrustworthy
22:     **end if**
23:     Update the credibility score of each user $u \in U$ using Eq. (8)
24:     Update the number of inquiries that each user $u \in U$ is allowed to make using Eq. (9) (8)
25: **end procedure**
```

time helping other users make choices. Moreover, some users might have insufficient accuracy (determined by the machine learning technique) due to lack of any similarity between the specifications of the item dealt with and those of the item being bootstrapped[4] Therefore, refraining from participating would be the best choice for these users instead of giving inaccurate endorsements (thanks to the credibility update mechanism proposed in Section 3.4). In case users agree to participate in the bootstrapping process, they first use the decision tree technique to predict the behavior of the newly deployed items and derive the appropriate endorsements. Note that decision tree has been chosen for the bootstrapping problem due to its lightweight nature which requires no heavy computations nor long training time. This is important to (1) incentivize users to participate in the endorsements collection process since they are not required to use large amounts of resources (e.g., smartphone battery) to derive endorsements, (2) minimize the time required to collect endorsements to fit urgent bootstrapping requests which require prompt answers, and (3) support different platforms that users can use to derive endorsements (e.g., mobile phones).

Decision tree reasoning is a machine learning technique which learns decision rules from historical (training) data to predict the classification or value of target variables [16]. The basic idea of decision tree is to recursively and constantly split the training data into subsets based on an attribute importance test until all samples for a given node belong to the same class or until there are no remaining attributes for further partitioning. The topmost decision node (a.k.a root node) represents the attribute which gives the best prediction results, whereas the bottommost nodes (a.k.a leaf nodes) correspond to the decisions generated by the decision tree algorithm. The most critical part of building decision trees is non-leaf attribute nodes selection. The challenge here is to select the most informative attributes, i.e., the attributes that give the best clue about the output attribute which helps minimize the number of intermediate nodes and hence minimize the decision tree size. To measure the degree of informativeness of the attributes, the *information gain* statistical measure is commonly used [34]. It measures how much knowing the value of a certain input attribute can give us hints about the potential value of the output attribute. Thus, the attribute which gives the highest information gain is selected as being the attribute that is the most appropriate to start building the decision tree with. In order to compute the information gain metric, we need first to calculate the *entropy* which measures the impurity or un-orderedness in the class distribution in a certain dataset D (i.e., how good an attribute is in differentiating among samples belonging to different classes). The entropy

[4] Allowing advisors to opt out is especially valuable towards specifying how to perform Sen's proposed "engage" step in trust processing [35] - limiting which peers to consult, to improve the performance of the trust modeling.

Table 1
Snapshot of the CloudHarmony dataset.

Provider	*Deployment_Area*	*Actual_Availability*	Trusted?
Amazon EC2	ap-northeast-1	100	Yes
Amazon EC2	ap-northeast-1	100	Yes
Agile Cloud	dublin	99.9848	Yes
Agile Cloud	dallas	100	Yes
LunaCloudCompute	dublin	99.7222	No
dediserve	eu-west	99.5554	No
dediserve	eu-west	100	Yes
dediserve	eu-west	99.9949	Yes
dediserve	dallas	84.9422	No
dediserve	dublin	85.0587	No
dediserve	dallas	-	?

is computed as per Eq. (1).

$$E(D) = \sum_{i=1}^{n} -p_i \log_2 p_i \tag{1}$$

where D is a (sub)set of examples, $n > 1$ is the number of class labels and p_i is the proportion of samples in D that belong to the class label i. Note that a value of 0 for the entropy means that all the samples of D belong to the same class label and that the data is perfectly classified (into one single class), whereas a value of 1 means that the data is classified evenly among the different class labels. This entropy measure quantifies only the quality of a single set of examples but cannot measure the quality of the complete split. Therefore, we need to compute the weighted average entropy over all the sets forming a certain split as per Eq. (2).

$$I(D, attr) = \sum_{k \in K_{attr}} \frac{|D_k|}{|D|} \cdot E(D_k), \tag{2}$$

where K_{attr} is the set of distinguished values of attribute *attr* and $E(D_k)$ is computed following Eq. (1) restricted to the value k of the attribute *attr*. Having computed both the entropy and weighted entropy, the next step is to derive the information gain of each attribute *attr* as per Eq. (3).

$$Gain(attr) = E(D) - I(D, attr), \tag{3}$$

To better clarify how decision tree can be practically used to obtain endorsements on newly deployed items, we give in the following an illustrative example on a subset of the CloudHarmony dataset, which is used in Section 4 for our experimental analysis. The dataset records information about cloud services possessed by well-known providers and running in disparate parts of the World. In particular, the availability and throughput (we show only availability in this example for the sake of simplicity) of the services have been measured for a period of thirty days and compared with the availability and throughput promised by the providers in the Service-Level Agreement (SLA) in order to get an idea on how trusted are these providers in committing to their promises. More details about the dataset are provided in Section 4. A snapshot of the dataset is also given in Table 1.

Having this (sub)dataset at hand, suppose now that we are asked to endorse a newcomer service whose provider is *dediserve* and which is deployed in the area of *dallas* (last line in Table 1). The first step is to compute the entropy of the dataset D using Eq. (1). In the class label **Trusted?** (rightmost column) of Table 1, we have 6 services out of 10 classified as *Yes* and 4 services classified as *No*. Thus the entropy of the dataset would be: $E(D) = -\frac{6}{10}\log_2(\frac{6}{10}) - \frac{4}{10}\log_2(\frac{4}{10}) = 0.971$. Then, the next step is to compute the entropy of each attribute (i.e., provider and *deployment_country*) as per Eq. (1) as well. Note that the *actual_availability* attribute is omitted from the following calculations since the *actual_availability* attribute is not supposed to be known for any newcomer service being bootstrapped. However, we show this attribute in Table 1 to show how the class label attribute (i.e., **Trusted?**) has been decided. The methodology used to compute the entropy value of the **Provider** attribute is illustrated in what follows:

- **Provider**=Amazon EC2: We have 2 examples classified as *Yes* and 0 example classified *No*: $E(D_{AmazonEC2}) = -1 \cdot \log_2(1) - 0 \cdot \log_2(0) = 0$
- **Provider**=Agile Cloud: We have 2 examples classified as *Yes* and 0 example classified as *No*: $E(D_{Agile\ Cloud}) = -1 \cdot \log_2(1) - 0 \cdot \log_2(0) = 0$
- **Provider**=LunaCloudCompute: We have 0 examples classified as *Yes* and 1 example classified as *No*: $E(D_{LunaCloudCompute}) = 0$
- **Provider**=dediserve: We have 2 examples classified as *Yes* and 3 examples classified as *No*: $E(D_{dediserve}) = -\frac{3}{5} \cdot \log_2(\frac{3}{5}) - \frac{2}{5} \cdot \log_2(\frac{2}{5}) = 0.971$

Table 2
Snapshot of the filtered CloudHarmony dataset according to the provider attribute (i.e., dediserve).

Provider	*Deployment_Country*	*Actual_Availability*	Trusted?
dediserve	eu-west	99.5554	No
dediserve	eu-west	100	Yes
dediserve	eu-west	99.9949	Yes
dediserve	dallas	84.9422	No
dediserve	dublin	85.0587	No

Now, we compute the weighted average entropy for the **Provider** attribute over all the sets as per Eq. (2): $I(D, provider) = \frac{2}{10} \cdot 0 + \frac{2}{10} \cdot 0 + \frac{1}{10} \cdot 0 + \frac{5}{10} \cdot 0.971 = 0.4855$. Having derived the entropy and the weighted average entropy, the information gain of the **Provider** attribute (Eq. (3)) is: $Gain(Provider) = 0.971 - 0.4855 = 0.4855$.

Next, we move to computing the information gain of the **Area** attribute using the same methodology used for the **Provider**:

- **Area**=ap-northeast-1: We have 2 examples classified as *Yes* and 0 example classified as *No*: $E(D_{ap-northeast-}1) = 0$
- **Area**=dublin: We have 1 example classified as *Yes* and 2 examples classified as *No*: $E(D_{dublin}) = -\frac{2}{3} \cdot \log_2(\frac{2}{3}) - \frac{1}{3} \cdot \log_2(\frac{1}{3}) = 0.917666$
- **Area**=euro-west: We have 2 examples classified as *Yes* and 1 example classified as *No*: $E(D_{euro-west}) = -\frac{1}{3} \cdot \log_2(\frac{1}{3}) - \frac{2}{3} \cdot \log_2(\frac{2}{3}) = 0.917666$
- **Area**=dallas: We have 2 examples classified as *Yes* and 3 examples classified as *No*: $E(D_{dallas}) = -\frac{1}{2} \cdot \log_2(\frac{1}{2}) - \frac{1}{2} \cdot \log_2(\frac{1}{2}) = 0$

Now, we compute the weighted average entropy for the **Area** attribute over all the sets as follows: $I(D, Deployment\_Area) = \frac{2}{10} \cdot 0 + \frac{3}{10} \cdot 0.917666 + \frac{3}{10} \cdot 0.917666 + \frac{2}{10} \cdot 0 = 0.5505996$. Having derived the entropy and the weighted average entropy, the information gain of the *Deployment_Area* attribute is: $Gain(Deployment\_Area) = 0.971 - 0.5505996 = 0.4204004$.

Since the information gain of the **Provider** attribute is higher than that of the **Deployment_Area** attribute, the former attribute is selected to be the root of the decision tree. Since there are four possible values for the **Provider** attribute (Table 1), the root of the decision tree would have exactly four branches, namely those of *Amazon EC2, Agile Cloud, LunaCloudCompue*, and *dediverse*. Now, the mission is to find the successor of the root attribute for each of those four branches. Starting with the *Amazon EC2* branch, since all of the services whose provider is *Amazon EC2* are classified as trusted in the **Trusted?** attribute (Table 1), this branch is terminated with a leaf node of value *Yes*. Similarly, the *Agile Cloud* branch is terminated with a leaf node of value *Yes* since all of the two services whose provider is *Agile Cloud* are classified as *Yes*. In contrary, since the only service whose provider is *LunaCloudCompute* is rated as untrustworthy, the *LunaCloudCompute* branch is terminated with a value of *No*. Finally, for the *dediverse*, we have conflicting classifications (i.e., two *Yes* and three *No*) as shown in Table 2. Therefore, the **Deployment_Area** is used as another split attribute in the decision tree. For the *dallas* and *dublin* values of the **Deployment_Area** attribute, the services whose provider is *dediverse* are always classified as untrustworthy. On the other hand, when the **Deployment_Area** is *eu-west*, we have two cases where the services have been classified as trusted (i.e., *Yes*) and one case where a service has been classified as untrustworthy (i.e., *No*). These cases are schematically displayed using the filtered dataset given in Table 2. To settle this dispute, the majority voting scheme [14] is employed, where the branch $\textbf{Provider} \xrightarrow{\text{dediserve}} \textbf{Deployment\_Area} \xrightarrow{\text{eu-west}} ?$ is terminated with a *No* leaf node. The complete decision tree for this example is given in Fig. 1.

3.3. Dishonesty-aware aggregation

The purpose of this phase is to aggregate the different endorsements collected as per the previous section in a non-collusive manner, i.e., in such a way that is resilient to the bootstrappers that submit misleading endorsements to promote/demote some newly deployed items. To do so, the Dempster-Shafer theory of evidence, which is known for its power in combining observations coming from multiple sources having different levels of credibilities, is employed. It is true that DST has been used in many proposals for trust establishment purposes [41,42]; however, the main difference between our endorsements aggregation mechanism and the existing DST-based trust establishment techniques is that our solution requires no predefined thresholds to make a final decision on whether to trust the newly deployed item or not. Specifically, contrary to the existing approaches whose performance is greatly dependent on a certain threshold, we propose to compute both the belief in an item's trustworthiness and untrustworthiness and compare them to arrive at a final decision. Moreover, the existing approaches adopt a static approach for assigning weights for the witnesses. Contrary to the literature, each witness (endorsement) is weighted according to the credibility score of its issuer in our approach as was discussed in [40], where credibility scores are continuously updated to reflect endorsers' honesty dynamism.

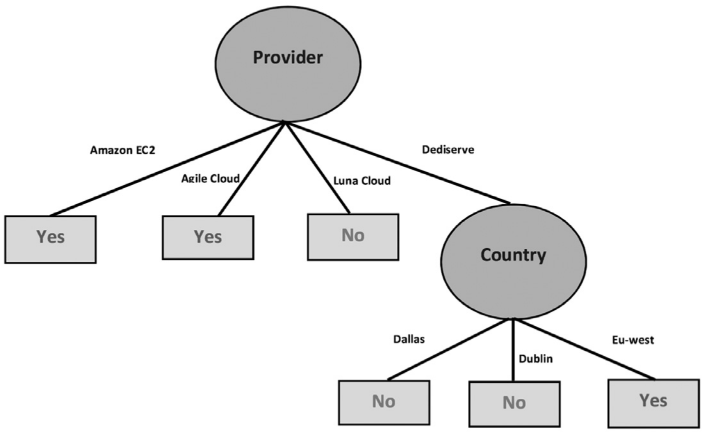


Fig. 1. Classification results.

Formally, let $\Omega = \{T, N, U\}$ be a set composed of three hypotheses representing the possible endorsements regarding a newly deployed item, where T means trustworthy, N means untrustworthy, and U means uncertainty between trust and distrust. The basic probability assignment (bpa) $m_b^i(H)$ of a particular hypothesis H given by bootstrapper b regarding an service i is proportional to the credibility score of b. Specifically, if bootstrapper b having a credibility score equal to λ has bootstrapped item i as being trustworthy, then the bpa's of the different hypotheses are computed as follows: $m_b^i(T) = \lambda$, $m_b^i(N) = 0$, and $m_b^i(U) = 1 - \lambda$. Otherwise, if b bootstraps i as being untrustworthy, then the bpa's of the hypotheses are computed as follows: $m_b^i(T) = 0$, $m_b^i(N) = \lambda$, and $m_b^i(U) = 1 - \lambda$.

Having defined the bpa's, the final aggregate belief function regarding a certain hypothesis H is computed through summing up all the bpa's coming from different bootstrappers upholding this hypothesis H. The belief function that recommender system r computes regarding item i's trustworthiness after having consulted two bootstrappers b and b' is given in Eq. (4).

$$\theta_r^i(T) = m_b^i(T) \oplus m_{b'}^i(T) = \frac{1}{K}[m_b^i(T)m_{b'}^i(T) + m_b^i(T)m_{b'}^i(U) + m_b^i(U)m_{b'}^i(T)] \tag{4}$$

Similarly, the belief function computed by r regarding item i's untrustworthiness after having consulted two bootstrappers b and b' is given in Eq. (5).

$$\theta_r^i(N) = m_b^i(N) \oplus m_{b'}^i(N) = \frac{1}{K}[m_b^i(N)m_{b'}^i(N) + m_b^i(N)m_{b'}^i(U) + m_b^i(U)m_{b'}^i(N)] \tag{5}$$

Finally, the belief function computed by r on the cloud service i being either trustworthy or untrustworthy (i.e., uncertainty) after having consulted two bootstrappers b and b' is given in Eq. (6).

$$\theta_r^i(U) = m_b^i(U) \oplus m_{b'}^i(U) = \frac{1}{K}[m_b^i(U)m_{b'}^i(U)], \text{ where:} \tag{6}$$

$$K = \sum_{h \cap h' = \emptyset} m_b^i(h)m_{b'}^i(h') \tag{7}$$

Note that the values produced by the different belief functions are real-valued numbers between 0 and 1, i.e., $\theta_r^i(T)$, $\theta_r^i(N)$, $\theta_r^i(U) \in [0, 1]$. Finally, the decision of the recommender system regarding a certain newcomer item i is taken by computing the beliefs in i's trustworthiness $\theta_r^i(T)$ and untrustworthiness $\theta_r^i(N)$ and comparing them, i,e., if $\theta_r^i(T) > \theta_r^i(N)$, i is deemed trustworthy; otherwise, i is considered as being untrustworthy.

3.4. Credibility update mechanism

The credibility scores of the bootstrappers need to be constantly updated in order to maintain the authenticity of the bootstrapping process. Specifically, honest bootstrappers should get their credibility values increased and dishonest bootstrappers should undergo a decrease in their credibility values [40]. We propose in Eq. (8) a credibility update mechanism by which the recommender system r updates its credibility belief $\phi(r \to u)$ toward every user u that has submitted a bootstrapping endorsement regarding a newcomer cloud service i upon the request of r.

$$\phi(r \to u) = \begin{cases} \min(1, \phi(r \to u) + X), & \text{if C1} \\ |\phi(r \to u) - Y|, & \text{if C2} \end{cases} \tag{8}$$

where $X = \max(\theta_r^i(T), \theta_r^i(N))$, $Y = \min(\theta_r^i(T), \theta_r^i(N))$, and C1 and C2 are two conditions such that:

C1. $R(u,i) \in \{T\} \& \theta_r^i(T) > \theta_r^i(N)$ or $R(u,i) \in \{N\} \& \theta_r^i(T) < \theta_r^i(M)$
C2. $R(u,i) \in \{T\} \& \theta_r^i(T) < \theta_r^i(N)$ or $R(u,i) \in \{N\} \& \theta_r^i(T) > \theta_r^i(N)$

The basic idea of Eq. (8) is to update the credibility score of each bootstrapper u proportionally to the difference between her submitted endorsement $R(u, i)$ on cloud service i and the final decision yielded by the DST-based aggregation mechanism. In this way, the users whose endorsements converge to the final decision of the recommender system receive an increase in their credibility scores and those whose endorsements are far from the final decision undergo a decrease in their credibility scores. This process is of prime importance to guarantee the honesty of the bootstrapping process since the performance of the DST aggregation technique is highly dependent on the credibility scores of the endorsers.

3.5. Incentive mechanism and participation motivation

In order to motivate the users to participate in the recommendation process, we propose in this section an incentive mechanism which links the participation of the users with the number of inquiries that they are allowed to make. Initially, all users have an equal amount of inquiries that they are allowed to make from any other user. This amount is then updated during the recommendation process as shown in Eq. (9). Specifically, every certain period of time, the number of inquiries that a user x is allowed to make from any other user s get increased in terms of the number of inquiries coming from s that x has answered, as well as the credibility score of x believed by s, i.e.,

$$Inq(x \to s) = Inq(x \to s) + (|E(x \to s)| + \lceil |E(x \to s)| \times Cr(s \to x) \rceil + 1) \tag{9}$$

In Eq. (9), $Inq(x \to s)$ denotes the total number of inquiry requests that x is allowed to make from s, $|E(x \to s)|$ denotes the number of recommendations that x has answered in favor of s, and $Cr(s \to x)$ denotes the credibility score of user x believed by user s. In this way, the users that refuse to participate in the recommendation process would, over time, end up being unable to make any request from any other user. In addition, by linking the number of inquiries with the credibility score of the recommender user, we aim at motivating those users to provide honest opinions.

4. Experimental evaluation

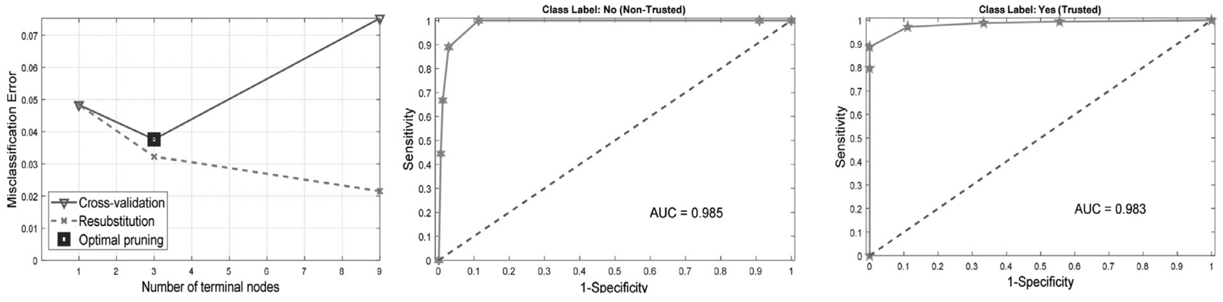
4.1. Experimental setup and datasets

To evaluate the performance of our solution, we conduct a series of experiments on real trust and cloud services datasets. The first part of the experiments is dedicated to measuring the accuracy of the proposed approach in terms of generating credible initial trust scores on a real cloud services dataset. The second part provides a comparison between our proposed bootstrapping solution and two existing approaches, namely the MoleTrust algorithm proposed in [22,23] and the users grouping approach proposed in [20]. The third part is introduced to evaluate the performance of our solution in the presence of Sybil, camouflage, and whitewashing attackers that try to manipulate the endorsement process compared to the MET model proposed in [13]. To carry out the first set of experiments, we employed a real cloud services dataset obtained from CloudHarmony[5]. The dataset records 187 transactions of 53 cloud services owned by renowned providers such as Amazon and Google. To derive recommendations on the services, several factors such as provider's name, deployment area, promised availability, actual availability, promised throughput, actual throughput, and number of outages are taken into consideration. Specifically, the availability, throughput, and number of outages have been evaluated for a period of thirty days and the average of each metric has been computed and recorded in the dataset. The purpose is to quantify the truthfulness of the cloud providers in committing to their Quality of Service (QoS) promises agreed upon in the Service-Level Agreement (SLA) made with customers and hence decide on whether the underlying service should be recommended to users as trustworthy or untrustworthy.

To conduct the second set of experiments which aims at comparing the performance of our solution w.r.t the MoleTrust approach [22,23] and the users grouping approach [20], real-world trust data from the Epinions[6] large Web community are employed. Epinions allows users to express their opinions regarding a wide variety of items and services (e.g., movies, cars, etc.) in the form of numeric ratings from the interval [1,5], with 1 being the rating which represents the least satisfaction level and 5 being the rating which represents a full satisfaction level. Epinions allows users as well to rate each other on the basis of the meaningfulness of their submitted ratings. The dataset consists of approximately 140,000 items and services rated by 50,000 users, where a total of $\approx 660,000$ reviews are collected [23]. We chose to use Epinions when comparing with other approaches due its large-scale nature, which allows us to better test the generalizability of the studied solutions.

In the third series of experiments, we compare our solution with the MET recommendation-based trust model proposed in [13] under several attack scenarios. The objective of MET is to derive the optimal trust network that provides the most accurate estimation of sellers' reputation scores in duopoly environments. To carry out these experiments, we consider a similar environment to that considered in [13] by simulating three types of attacks that can be launched by consulted advisors to mislead the recommendation-based trust establishment process (i.e., Sybil, camouflage, and whitewashing) and

[5] http://cloudharmony.com/.
[6] http://www.epinions.com/.



(a) Optimal pruning level (b) Accuracy (untrustworthy) (c) Accuracy (trustworthy)

Fig. 2. Bootstrapping accuracy: Our bootstrapping mechanism achieves high accuracy rate (CloudHarmony).

setting the percentage of attackers to 30% of the total number of advisors. We compare both approaches in terms of Mean Absolute Error (MAE), which is computed as follows: $MAE(i) = \frac{|Trust(i) - \hat{Trust}(i)|}{|A|}$, where $Trust(i)$ is the actual trust score of item i, $\hat{Trust}(i)$ is the trust value of item i estimated by the trust model, and $|A|$ is the number of consulted advisors. Note that the actual trust score of each item is available in the Epinions dataset. In Sybil attacks, dishonest advisors generate several fake identities in an attempt to manipulate the trust aggregation process by submitting a large number of dishonest recommendations. In camouflage attacks, attackers try to fool the trust system by providing honest recommendations in the beginning to build good credibility scores and then start to submit dishonest recommendations. In whitewashing, dishonest advisors try to clear their bad credibility history through continuously creating new identities.

The machine learning classifier has been trained on the datasets following the k-fold cross-validation approach (with $k = 10$) [31]. According to this approach, the dataset is divided into k subsets, where each of these subsets is selected every time to be the test set and the other $k - 1$ subsets are combined together to serve as the training set. Subsequently, the accuracy of the classifier is derived through computing the average error across all the k trials. The main benefit of this training approach lies in its ability to minimize the bias of the classification results toward the dataset's structure, as each data sample is forced to take part of the test set exactly once and of the training set $k - 1$ times. The experiments have been conducted using Matlab in a 64-bit Windows 7 environment on a machine equipped with an Intel Core i7 − 4790 CPU 3.60 GHz Processor and 16 GB RAM. Note finally that our source code is available at the following link https://github.com/gaith7/Recommendation.

4.2. Results and discussion

Fig. 2 a is introduced to show the optimal pruning level that minimizes the overfitting of the decision tree classifier and hence maximizes the accuracy. Specifically, when the resubstitution error is smaller than the cross-validation error, this means that the structure of the tree overfits the training data (i.e., the accuracy of the classifier is bound to the specific content of the training data and might achieve less performance when new data is encountered). The optimal tree size which best minimizes the overfitting and generalizes to different kinds of data is the one that gives the least cross-validation error. According to Fig. 2a, we notice that a decision tree consisting of three branches achieves the least cross-validation error and hence the best classification performance. Therefore, we adopt such a three-branch decision tree in the rest of our simulation.

To assess the accuracy of the proposed bootstrapping mechanism using the CloudHarmony dataset, we present in Fig. 2b and Fig. 2c the Receiver Operating Characteristic (ROC) curves [7] obtained after having applied our solution to predict the initial trust scores of the involved cloud services. Note that the ground truth (i.e., whether a certain service is trustworthy or not) is determined in the dataset using a binary *status* attribute which is obtained through comparing the availability and throughput promised by the providers in the SLA and those actually received after having run the cloud services for a whole month. In these Figures, specificity indicates the percentage of negatives identified as such and sensitivity indicates the percentage of positives recognized as such. In Fig. 2b, the objective is to examine the accuracy of our bootstrapping mechanism in identifying untrustworthy services as such. Hence, sensitivity represents in this case the percentage of untrustworthy services that are correctly identified as being untrustworthy and 1−specificity represents the percentage of untrustworthy services that are mistakenly classified as being trustworthy. Moving to Fig. 2c where the purpose is to examine the accuracy of our bootstrapping mechanism in identifying trustworthy services as being so, sensitivity denotes in this case the percentage of trustworthy services correctly identified as such and 1− specificity represents the percentage of trustworthy services misclassified as being untrustworthy. An ideal case would be to have a percentage of 100% for both sensitivity and specificity.[7] This case would be graphically translated into a point whose coordinates are (0, 1). In the same context, a di-

[7] Note that a percentage of 100% for specificity is equivalent to a percentage of 0% for 1−specificity.

Table 3
Performance comparison between our solution, the MoleTrust [22,23], and users grouping [20] approaches (Epinions).

	Our Solution	MoleTrust	Grouping
Accuracy	86.23%	59.64%	61.15%
Specificity	91.78%	56.86%	59.84%
Precision	89.66%	21.99%	24.24%
Recall (or sensitivity)	80%	75.61%	68.09%
F-Measure	84.55%	34.07%	35.75%

Table 4
Mean Absolute Error (MAE) comparison of items' initial trust estimation between our solution and MET [13] (Epinions).

	Our Solution	MET
Sybil	0.05 ± 0.02	0.09 ± 0.06
Camouflage	0.18 ± 0.03	0.01 ± 0.00
Whitewashing	0.06 ± 0.03	0.05 ± 0.03

agonal line represents a worthless classifier that is equivalent to a random guess. The Area Under the Curve (AUC) measure [3] quantifies our bootstrapping mechanism's accuracy, where the closest the value of AUC to 1 is, the more accurate our bootstrapping mechanism would be. By observing Fig. 2b and 2 c, we can notice that our solution achieves a high accuracy level up to 98.5% in identifying untrustworthy services as such (Fig. 2b) and a high accuracy level up to 98.3% in identifying trustworthy services as such (Fig. 2c).

Next, we compare the performance of our solution with the MoleTrust [22,23] and users grouping [20] approaches in terms of accuracy, precision, recall (or sensitivity), specificity, and F-measure. The purpose is to study how well each of the compared solutions is able to perform in the presence of such malicious parties. Accuracy is the ratio of correctly classified observations to the total number of observations. Table 3 shows that our solution achieves a high accuracy level of 86.23% in classifying newly deployed items/services compared to 59.64% for the MoleTrust and 61.15% for the grouping approach. As for the specificity (defined earlier), our solution achieves a specificity level of 91.78% compared to 56.86% for the MoleTrust and 59.84% for the grouping approach. Precision is the ratio of correctly classified positive samples to the total number of samples predicted as positive. In simple words, precision attempts to answer the following question: Of all items/services that are classified as trustworthy, how many are actually trustworthy?. We notice from Table 3 that our solution achieves a high precision percentage of 89.66% compared to 21.99% for the MoleTrust and 24.24% for the grouping approach. Recall or sensitivity is the proportion of actual positives that are identified correctly. This metric attempts to answer the following question: Of all the trustworthy items/services, how many did we actually identify as such?. According to Table 3, our bootstrapping solution achieves a recall percentage of 80% compared to 75.61% for the MoleTrust and 68.09% for the grouping approach. F-measure is the harmonic average of precision and recall and is computed as follows: $F-measure = 2 \times \frac{precision \times recall}{precision + recall}$. The F-measure yielded by our solution is 84.55% which is high enough since both the precision and recall percentages are high in our solution, whereas the F-measure yielded by the MoleTrust and grouping approaches is low (i.e., 34.07% and 35.75% respectively) which is due to their low precision (21.99% and 24.24% respectively).

The reasons behind the improvements brought by our solution compared to the MoleTrust and users grouping approaches can be summarized as follows. Both approaches employ simplistic techniques (averaging in the case of MoleTrust and weighted sum in the case of the grouping approach) to combine the collected recommendations. These techniques are known to be vulnerable to malicious recommenders that try to manipulate the aggregate final decisions through submitting misleading recommendations. The situation becomes even worse in case of collusion attacks whereby malicious recommenders collude to provide a large number of deceptive opinions to promote/demote some newly deployed items. Per contra, we propose in our work a comprehensive aggregation technique that employs DST to discourage untruthful reporting and propose also a credibility update mechanism that constantly updates the weights assigned to the collected endorsements on the basis of the credibility scores of the endorsers. The second advantage of our solution lies in the machine learning approach that we propose, which improves the accuracy of the initial endorsements through analyzing the observed features of the newly deployed items and those of the already deployed ones to make the best possible predictions.

Now, we compare our solution with the MET trust model [13]. The objective of MET is to derive the optimal trust network that provides the most accurate estimation of sellers' reputation scores in duopoly environments. This framework is selected in order to independently verify the effectiveness of our solution against a richer set of possible attacks. We notice from Table 4 that our solution decreases the MAE compared to MET in the presence of Sybil attacks. In fact, MET is based on the idea of generating various networks of advisors with different trust values using evolutionary operators and then keeping the best network that consists of advisors having the highest trust values. However, even in the optimal network of advisors, there is still some chance of encountering a minority of advisors that might provide inaccurate endorsements. Worse, in the case of Sybil attacks, such a minority might even become a majority by creating a large number of fake

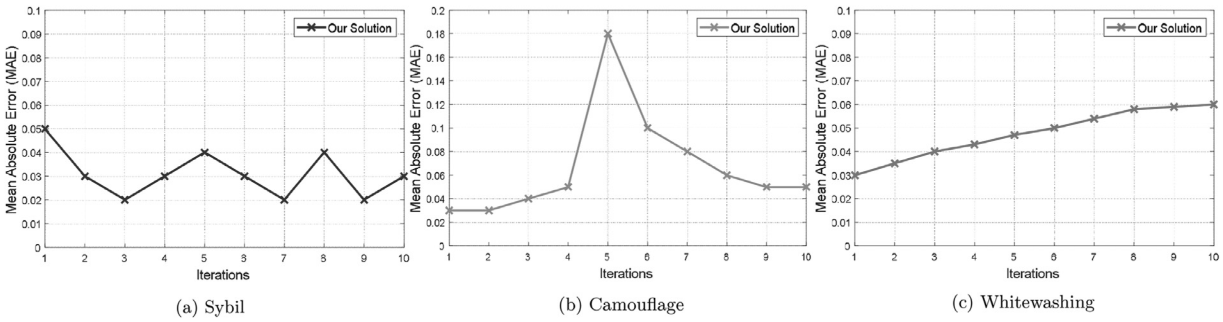


Fig. 3. The effects of Sybil, camouflage, and whitewashing attacks on the MAE enatailed by our solution.

identities. In such cases, MET provides no countermeasures against such advisors. On the other hand, in addition to operating at the endorsements' collection level, our solution operates as well at the trust aggregation level to further improve the protection against dishonest opinions. Specifically, in our approach, we broadcast the endorsement requests to the set of available advisors and allow these advisors to self-assess their degrees of accuracy prior to submitting their endorsements. Thereafter, we aggregate the endorsements coming from advisors having different levels of credibility using the Dempster-Shafer method, which is mainly influenced by the credibility of the advisors, rather than their number. This makes our solution quite resilient to Sybil attacks and efficient even in scenarios in which dishonest advisors might form the majority [40].

On the other hand, in camouflage attacks, MET entails lower MAE compared to our solution. The reason is that in such a type of attacks, dishonest advisors initially provide honest endorsements to build up good credibility scores, prior to starting their dishonest endorsements. This makes our trust aggregation method, which is mainly influenced by the credibility scores of the advisors, to be vulnerable to such attackers for a short period of time (i.e., the period at which dishonest advisors switch their behavior and start providing misleading endorsements). When it comes to whitewashing attacks, our solution and MET show relatively similar resilience to such attacks. In fact, MET keeps only the network of advisors with the most suitable trust values according to some evolutionary operators, which makes it hard for whitewashing advisors to get into these networks. In our solution, even though some dishonest advisors might clear their bad credibility history and rejoin the network again, such newcomer advisors aren't expected to have high credibility values as compared to those honest advisors who have strived to build and retain high credibility scores. Consequently, the presence of camouflage attackers does not have a significant impact on the performance of our solution.

In Fig. 3, we study in more detail the effects of each of the three types of attacks on our solution. We can notice from Fig. 3a that Sybil attacks have no effect on our solution, for the reasons mentioned above. From Fig. 3b, we can see that up to the fifth iteration (the period during which camouflage attackers provide honest endorsements to gain high credibility scores), the MAE entailed by our solution is low. At the fifth iteration (the time moment at which camouflage attackers start to change their behavior by providing dishonest endorsements), the MAE of our solution is reported to be relatively high (i.e., 0.18). Starting from the sixth iteration, our solution starts to recognize the camouflage attackers and decrease their credibility, thus leading to gradually improving the performance and decreasing the MAE. Finally, from Fig. 3c, we notice that whitewashing attacks have a small effect on the performance of our solution for the reasons mentioned in the previous paragraph.

Finally, in Fig. 4, we compare our solution with that proposed in [26] in terms of Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) on the CloudHarmony dataset. We inject in this set of experiments a number of newly registered users (15% of the total number of users seeking for recommendations) to study the performance of the compared approaches in such a complex scenario. As explained earlier, MAE quantifies the average difference between the predicted trust scores of the cloud services being bootstrapped and the actual trust scores of these services, proportionally to the number of consulted advisors. RMSE quantifies the standard deviation of the prediction errors and is computed as follows: $RMSE(i) = \sqrt{\sum_{i=1}^{n}(\frac{|Trust(i)-Tru\hat{s}t(i)|)^2}{|A|}}$, where Trust (i) is the actual trust score of item i, $Tru\hat{s}t(i)$ is the trust score of item i estimated by the bootstrapping model, and |A| is the number of consulted advisors. The first observation that can be drawn from the figure is that increasing the number of advisors leads to decreasing both the MAE and RMSE. The second observation is that our solution outperforms that of [26] in minimizing the prediction errors. More specifically, the main limitation of [26] is that it relies heavily on the similarity between existing users in the lattice representation to derive appropriate recommendations. This degrades its performance when extreme (yet realistic) scenarios in which newly registered users seek recommendations on newly deployed services. Our solution, on the other hand, does not rely on users' past preference to derive recommendations on newcomer cloud services, which makes it powerful in such scenarios.

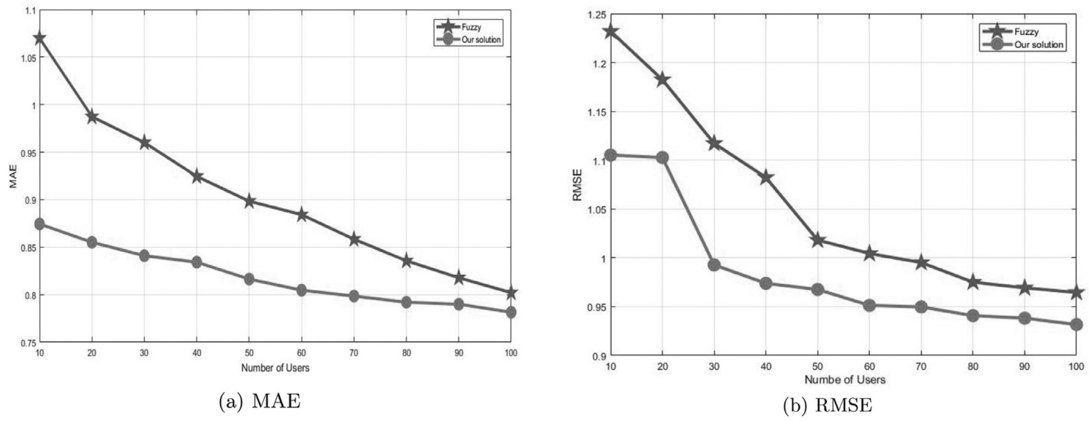


Fig. 4. Our solution decreases the MAE and RMSE compared to [26] in the presence of newly registered users seeking recommendations on newcomer cloud services.

5. Conclusion and future work

Trust bootstapping is still an open research challenge in the context of recommender systems and beyond. We proposed in this paper a solution based on endorsements from peers. Specifically, we advanced trust bootstrapping mechanism which consists of (1) a machine learning strategy to obtain endorsements regarding newcomer cloud services for which no evidence of their former behaviour is available (2) an endorsements aggregation technique that is robust to dishonesty and (3) a credibility update mechanism for the endorsers. The proposed solution can be integrated into any existing recommender system context to improve and secure **its** decisions. Experiments conducted using the CloudHarmony cloud services dataset and the Epinions trust dataset reveal that our proposed bootstrapping mechanism improves the accuracy of assigning initial trust scores for newcomers items up to $\approx 26\%$ compared to the ModelTrust [22,23], users grouping [20] and fuzzy formal concept analysis [26] approaches and is thus a promising solution. We have also studied the performance of our solution in comparison with the MET model [13] in the presence of Sybil, camouflage and whitewashing attackers. The results revealed that our solution shows more resiliency to Sybil attacks, similar resiliency to camouflage attacks and slightly less resiliency to whitewashing attacks compared to MET.

Several issues are important to continue to consider for future work. The first is that of identity management, which is important to verify the identities of the endorsers and prevent identity impersonation and/or duplication. To address this issue, we plan to integrate blockchain-based solutions such as uport[8] into our solution. Thus, prior to submitting endorsements, users would be asked to create uport identities (if not already done). Uport then generates a new asymmetric key pair (i.e., private and public keys) for users, which allows the recommender system (having a uportID as well) to verify users' identities. The ethereum blockchain would include controller contract, proxy contract and application contract and would interact both with the Uport user device and the recommender system. Another issue is addressing cases where peers have subjective differences. In the future, we plan to extend our solution to support situations wherein users might use different evaluation functions, thus learning these functions in a manner similar to that of the BLADE system [32]. This would expand the applicability of our solution and improve the endorsements' accuracy. Another promising direction would be to investigate the effectiveness of applying a multi-criteria clustering approach [17,18] that groups users based on several criteria such as credibility scores, resource availabilities and geographical location. Such an approach is anticipated to help the recommender system better select the appropriate groups of users to participate in each particular bootstrapping request. Finally, we plan to implement the proposed bootstrapping solution using the federated learning concept [15] where users can use their mobile phones to collaboratively learn a shared recommendation prediction model.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

[8] https://www.uport.me/.

CRediT authorship contribution statement

Omar Abdel Wahab: Conceptualization, Methodology, Writing - original draft, Project administration. **Robin Cohen:** Validation, Writing - review & editing, Supervision. **Jamal Bentahar:** Validation, Writing - review & editing, Supervision. **Hadi Otrok:** Supervision. **Azzam Mourad:** Supervision. **Gaith Rjoub:** Project administration.

Acknowledgments

This work has been supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ins.2020.03.102.

References

[1] R. Abdelkhalek, Improving the trustworthiness of recommendations in collaborative filtering under the belief function framework, in: Proceedings of the Eleventh ACM Conference on Recommender Systems, ACM, 2017, pp. 421–425.
[2] F. Aznoli, N.J. Navimipour, Cloud services recommendation: Reviewing the recent advances and suggesting the future research directions, J. Netw. Comput. Appl. 77 (2017) 73–86.
[3] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (7) (1997) 1145–1159.
[4] C. Burnett, T.J. Norman, K. Sycara, Bootstrapping trust evaluations through stereotypes, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1, International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 241–248.
[5] S. Ding, Y. Li, D. Wu, Y. Zhang, S. Yang, Time-aware cloud service recommendation using similarity-enhanced collaborative filtering and arima model, Decis Support Syst. 107 (2018) 103–115.
[6] D. Fang, X. Liu, I. Romdhani, P. Jamshidi, C. Pahl, An agility-oriented and fuzziness-embedded semantic model for collaborative cloud service search, retrieval and recommendation, Future Generat. Comput. Syst. 56 (2016) 11–26.
[7] T. Fawcett, An introduction to ROC analysis, Pattern Recognit. Lett. 27 (8) (2006) 861–874.
[8] G. Guo, J. Zhang, D. Thalmann, Merging trust in collaborative filtering to alleviate data sparsity and cold start, Knowl. Based Syst. 57 (2014) 57–68.
[9] G. Guo, J. Zhang, N. Yorke-Smith, Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems, Knowl. Based Syst. 74 (2015) 14–27.
[10] S.-M. Han, M.M. Hassan, C.-W. Yoon, E.-N. Huh, Efficient service recommendation system for cloud computing market, in: Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human, ACM, 2009, pp. 839–845.
[11] C.-W. Hang, M.P. Singh, Trust-based recommendation based on graph similarity, in: Proceedings of the 13th International Workshop on Trust in Agent Societies (TRUST). Toronto, Canada, 2010.
[12] X.-Y. Huang, B. Liang, W. Li, Online collaborative filtering with local and global consistency, Inf. Sci. (Ny) (2019).
[13] S. Jiang, J. Zhang, Y.-S. Ong, An evolutionary model for constructing robust trust networks, in: Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 813–820.
[14] R. Kohavi, C. Kunz, Option decision trees with majority votes, in: Proceedings of the ICML, 97, 1997, pp. 161–169.
[15] J. Konečnỳ, H.B. McMahan, F.X. Yu, P. Richtárik, A.T. Suresh, D. Bacon, Federated learning: strategies for improving communication efficiency, arXiv:1610.05492 (2016).
[16] S.B. Kotsiantis, I. Zaharakis, P. Pintelas, Supervised machine learning: a review of classification techniques, Emerg. Artif. Intell. Appl. Comput. Eng. 160 (2007) 3–24.
[17] G. Kou, Y. Lu, Y. Peng, Y. Shi, Evaluation of classification algorithms using MCDM and rank correlation, Int. J. Inf. Technol. Decis. Mak. 11 (01) (2012) 197–225.
[18] G. Kou, Y. Peng, G. Wang, Evaluation of clustering algorithms for financial risk analysis using MCDM methods, Inf. Sci. (Ny) 275 (2014) 1–12.
[19] V. Y. Kreinovich, A. Bernat, W. Borrett, Y. Mariscal, and E. Villa: 1994, 'Monte-Carlo Methods Make Dempster-Shafer Formalism Feasible'. In: R. R. Yager, J. Kacprzyk, and M. Fedrizzi (eds.): Advances in the Dempster-Shafer Theory of Evidence. New York: Wiley, pp. 175-191.
[20] B. Lika, K. Kolomvatsos, S. Hadjiefthymiades, Facing the cold start problem in recommender systems, Expert Syst. Appl. 41 (4) (2014) 2065–2073.
[21] Z. Malik, A. Bouguettaya, Reputation bootstrapping for trust establishment among web services, IEEE Int. Comput. 13 (1) (2009) 40–47.
[22] P. Massa, P. Avesani, Controversial users demand local trust metrics: An experimental study on epinions. com community, in: Proceedings of the AAAI, 2005, pp. 121–126.
[23] P. Massa, P. Avesani, Trust-aware bootstrapping of recommender systems, in: Proceedings of the ECAI Workshop on Recommender Systems, 2006, pp. 29–33.
[24] H.B. McMahan, E. Moore, D. Ramage, S. Hampson, et al., Communication-efficient learning of deep networks from decentralized data, arXiv:1602.05629 (2016).
[25] S. Meng, W. Dou, X. Zhang, J. Chen, KASR: a keyword-aware service recommendation method on mapreduce for big data applications, IEEE Trans. Parallel Distrib. Syst. 25 (12) (2014) 3221–3231.
[26] H. Mezni, T. Abdeljaoued, A cloud services recommendation system based on fuzzy formal concept analysis, Data Knowl. Eng. 116 (2018) 100–123.
[27] J. O'Donovan, B. Smyth, Trust in recommender systems, in: Proceedings of the 10th International Conference on Intelligent User Interfaces, ACM, 2005, pp. 167–174.
[28] C.E. Player, N. NGriffiths, Bootstrapping trust and stereotypes with tags, in: Proceedings of the 19th International Workshop on Trust in Agent Societies (Trust@AAMAS 2017), International Foundation for Autonomous Agents and Multiagent Systems, 2017, pp. 1–14.
[29] L. Qi, X. Zhang, W. Dou, Q. Ni, A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data, IEEE J. Sel. Areas Commun. 35 (11) (2017) 2616–2624.
[30] D. Rafailidis, F. Crestani, Learning to rank with trust and distrust in recommender systems, in: Proceedings of the Eleventh ACM Conference on Recommender Systems, ACM, 2017, pp. 5–13.
[31] P. Refaeilzadeh, L. Tang, H. Liu, Cross-validation, in: Encyclopedia of Database Systems, Springer, 2009, pp. 532–538.
[32] K. Regan, P. Poupart, R. Cohen, Bayesian reputation modeling in e-marketplaces sensitive to subjectivity, deception and change, in: Proceedings of the National Conference on Artificial Intelligence, 21, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006, p. 1206.
[33] G. Rjoub, J. Bentahar, O.A. Wahab, Bigtrustscheduling: Trust-aware big data task scheduling approach in cloud computing environments, Future Generat. Comput. Syst. (2019).
[34] S.R. Safavian, D. Landgrebe, A survey of decision tree classifier methodology, IEEE Trans. Syst. Man Cybern. 21 (3) (1991) 660–674.
[35] S. Sen, A comprehensive approach to trust management, in: Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 797–800.

[36] G. Shafer, Dempster-shafer theory, Encycl. Artif. Intell. (1992) 330–331.
[37] F. Skopik, D. Schall, S. Dustdar, Start trusting strangers? bootstrapping and prediction of trust, in: Proceedings of the International Conference on Web Information Systems Engineering, Springer, 2009, pp. 275–289.
[38] R. Urena, G. Kou, Y. Dong, F. Chiclana, E. Herrera-Viedma, A review on trust propagation and opinion dynamics in social networks and group decision making frameworks, Inf. Sci. (Ny) 478 (2019) 461–475.
[39] O.A. Wahab, J. Bentahar, H. Otrok, A. Mourad, A survey on trust and reputation models for web services: single, composite, and communities, Decis Support Syst. 74 (2015) 121–134.
[40] O.A. Wahab, J. Bentahar, H. Otrok, A. Mourad, Towards trustworthy multi-cloud services communities: a trust-based hedonic coalitional game, IEEE Trans. Serv. Comput. 11 (2018) 184–201.
[41] J. Wang, H.-J. Sun, A new evidential trust model for open communities, Comput. Stand. Interf. 31 (5) (2009) 994–1001.
[42] B. Yu, M.P. Singh, An evidential model of distributed reputation management, in: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1, ACM, 2002, pp. 294–301.
[43] T. Zain, M. Aslam, M. Imran, A. Martinez-Enriquez, Cloud service recommender system using clustering, in: Proceedings of the 11th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), IEEE, 2014, pp. 1–6.
[44] M. Zhang, R. Ranjan, M. Menzel, S. Nepal, P. Strazdins, W. Jie, L. Wang, An infrastructure service recommendation system for cloud applications with real-time QOS requirement constraints, IEEE Syst. J. 11 (4) (2017) 2960–2970.
[45] M. Zhang, R. Ranjan, S. Nepal, M. Menzel, A. Haller, A declarative recommender system for cloud infrastructure services selection, in: Proceedings of the International Conference on Grid Economics and Business Models, Springer, 2012, pp. 102–113.