Neural Computing and Applications (2022) 34:16049–16060
https://doi.org/10.1007/s00521-022-07231-3

ORIGINAL ARTICLE

Recommendation platform in Internet of Things leveraging on a self-organizing multiagent approach

Agostino Forestiero[1] · Giuseppe Papuzzo[1]

Received: 8 February 2021 / Accepted: 29 March 2022 / Published online: 3 May 2022

Abstract
Identifying user requirements and preferences on the basis of the current context, is one of main challenges of the Internet of Things (IoT) paradigm. Users, services and applications interact maintaining, often unreliable, relations which need of smart management systems to satisfy their demands. Traditional information handling approaches in distributed systems are most often unsuitable for modern Smart Environments due to the huge amount and the extreme dynamism of the entities involved. This paper proposes NARIoT platform that allows building recommendation systems in IoT environment. The approach relies on vector representations of IoT resources obtained by using of a word embedding tool, the Doc2Vec neural model, which, starting from text documents describing the resources, provides real-valued vectors mapping them. The vectors are handled through intelligent agents, which self-organize themselves creating an ordered virtual structure, so enabling *informed* mechanisms of information filtering. In particular, an ordered overlay network emerges from the autonomous, parallel and decentralized work of intelligent agents, thus enabling efficient recommendation operations. The experimental validation confirms the effectiveness of the approach and provides very encouraging results.

Keywords Recommendation platform · Intelligent agents · Self-organization · Internet of Things · Natural Language Processing

1 Introduction

Due to the high dynamism, the heterogeneity and the huge dimensions of the Internet of Things (IoT) infrastructures, traditional management systems are often inadequate [4]. The *information system* is a crucial component of the IoT systems, since it, among other things, allows discovering and selecting available resources and services strongly related to the context [30]. The need to ensure satisfaction of user and service requirements imposes the design of sophisticated and innovative models and algorithms [31]. Approaches based on innovative techniques and methodologies have to be designed to support decision makers, above all, in crucial tasks such as resources selection and filtering. An important aspect concerns the systems able to identify useful items for services and users in a given context, i.e., the recommendation systems. The approaches for building recommendation systems can be basically identified as [6]: (i) *Content-based*, in which similar items previously chosen by the user can be suggested; (ii) *Collaborative Filtering*, where the "rating"—i.e., the level of usefulness for a given user—of all similar users lead the rating of a given item for a given user; and (iii) *Hybrid methods*, combining both approaches, collaborative and content-based. The similarity between users can be computed in several ways; for example, Pearson correlation and cosine distance are two of most popular [7, 32].

This paper proposes NARIoT (*NLP-driven Agents for Recommendation in IoT*), a multiagent platform that, by exploiting semantic concepts, enables efficient recommendations of useful objects (services, devices, etc.) in IoT environment. "Cyber agents" (intelligent agents), each one representing an IoT object, perform cyclically a predefined sequence of simple operations with the aim to create, in

✉ Agostino Forestiero
agostino.forestiero@icar.cnr.it; forestiero@icar.cnr.it

[1] Institute for High Performance Computing and Networking, National Research Council of Italy, CNR - ICAR, Via P. Bucci, 8-9C, Rende, CS, Italy

fully distributed and self-organizing modality, a virtual ordered structure (overlay network) that allows efficient information filtering. Thanks to their useful characteristics, agents are particularly suitable for dynamic systems because they naturally adapt to the environment changing [34]. Agent-based approaches assure: *extensibility*, agents are independent, they can be created or modified, other agents continue to provide services; *stability*, the continuity of the service is assured from other agents sharing the tasks of failing agents; *independent*, they can operate autonomously without user action [20].

The cyber agents work by exploiting a vector representations of associated IoT objects. In peer to peer systems, often, bit vectors are used to index metadata representing the content stored in the host. Each bit, for example, can represent the occurrence of a given *topic,cre*: for contents like documents, this meaning is particularly suitable. Differently, [19] exploits to obtain the vector representation, a hash function locality preserving that allows to assign neighbor vectors to contents having neighbor/similar features. In the approach proposed in this paper, the *Doc2Vec* neural model [23] is utilized to obtain a vector representation of IoT objects. This Natural Language Processing (NLP) model enables to generate a vector representation starting from text documents (metadata), also capturing the semantic meaning. Indeed, device capabilities of IoT objects are often reported in metadata documents as Web Service Descriptions Language (WSDL), HTML Microformat for Describing RESTfulWeb Services (hRESTS) and JSON-based web service descriptions [22]. Moreover, models such as OWL-S[1] or WSMO[2] provide useful semantic descriptions about IoT services functionalities allowing easier discovery and composition operations [12].

Two different overlay networks of IoT objects are allowed, on the basis of the algorithm performed by agents: (i) ordered list and (ii) ordered list of clusters. The organized virtual structure allows the selection operations to become faster and effective, because agents managing similar vectors—representing similar IoT resources—are linked among them and then, can be easily located and recommended. The emerging structure improves IoT resource management, since informed mechanisms can be designed in a non-structured and dynamic environment [18].

The main contributions of this article can be summarized as:

- a platform enabling the construction of a decentralized and self-organized information system in Internet of Things. It does not rely on any centralized mechanism, but it is fully decentralized, self-organizing and scale-free, thanks to agents' characteristics.
- the organized overlay network emerges from the autonomous and simultaneous work of intelligent agents. No preliminary knowledge has to be provided to the agents, but they act executing simple operations using local and limited information.
- the overlay automatically adapts itself to the changing environment (join and departure devices), making it particularly suitable for a highly dynamic and unreliable system such as the IoT.
- the approach allows executing informed discovery operations, typical of structured P2P systems, in a fully unstructured and highly dynamic environment.

The rest of the paper is organized as follows: Sect. 2 analyzes the related works and the state of the art, Sect. 3 describes the infrastructure and the logic layers of the platform, Sect. 3.2 details the multiagent algorithms designed and exploited, and Sect. 4 shows a set of experiments performed to validate the platform proposed.

2 Related works

Many solutions have been proposed in the IoT environment to manage resources by exploiting of different techniques, and peer to peer approaches have been widely leveraged [33]. A method for the construction of an efficient overlay for Internet of Things systems was proposed in [17] . This paper attempts to fill the technical gap and designs an effective p2p-based solution for efficient handling of range queries in IoT (RQIOT). RQIOT relies on a data distribution model based on consistent and order-preserving hash function and a novel scheme for capacity management of devices. The approach enables smart devices to satisfy their storage constraints exploiting a ledger for available network resources to allocate overflow space. [16] proposed new mechanisms aimed to improve IoT-P2P routing protocols for LOADng (Lightweight On-Demand ad hoc Distance-vector routing protocol next generation) and AODV-RPL (ad hoc On-demand Distance Vector routing-based RPL protocol). The improvements deploy Trickle-based forwarding, enhanced smart ring, along with a combination of both. The approach assures scalable, efficient, and reliable dissemination of flooding route request deploying stopping Trickle timers, providing individually and collectively great enhancements to P2P routing protocols, while staying backward compatible with their base specifications.

Recommender systems aim to help users in search operations by providing personalized suggestions or signaling useful services/resources in a large environment.

[1] http://www.w3.org/Submission/OWL-S/.
[2] http://www.w3.org/Submission/WSMO/.

Several current works aim to meet the best needs of users based on agents techniques [2, 28, 35].

Thanks to the enhanced agent's characteristics, recommender systems can be improved in dynamic environments because they automatically adapt to the changes of the context. Agents can operate in a cooperative mode and independently performs a specific task. All agents operate at the same time to achieve overall and complex tasks. Recommender system based on a trust model exploiting a multi-agent approach is proposed in [25]. Each agent with a given trust degree selects another agent wherewith it can exchange the need information to execute its task. In this way, the agents become expert in solving recommend task. MATRES system (Multi-Agent Travel Recommender System) in [26] has been proposed by the same authors. MATRES provides a possible best recommendation using assumptions determined by agents: exists a delay to receive the missing information.

[28] proposed a multi-agent recommender system of web pages to user-based approach that is based on the combination of two algorithms: AR algorithm (Associative Rules) and CF algorithm (Collaborative Filtering). [35] proposed a multi-agent recommender system of games for mobile phones. The recommender system proposed suggests games which adapt with user competences and their abilities in order to maintain longer and increase the time playing some games.

IoT-based systems enable a deeper understanding of user behaviors and preferences which can denote availability of different information sources [3], respect to other recommendation scenarios. The retail environment exploiting Internet of Things systems features the *personalized shopping,mager*, for example. Recommendations related to items and their price are delivered to customers of a store and the suggestions depend of the situation of the offered object. A recommender system in Internet of Things environment not only has to consider the users' preferences, but it has to take into account further information sources.

[24] proposes an IoT infrastructures and related recommendation for the availability of food items, historical food consumption, planned activities, and also historical sports activities. These data provided by IoT devices/services help to increase the quality of recommendation algorithms. In this case, only the data provided by active users are used to generated the suggestions.

Considering recommendation services under some conditions, such as holiday package, personalized Web site, or a movie, it cannot be sufficient to consider only users and items, but it is often necessary to incorporate the contextual data into the recommendation procedure [1]. In fact, additional contextual information, such as location, time, is taking into account to recommend the most relevant items to users. [29] propose a technology market as an application environment of context-aware recommender system. Museum visits of user groups in which recommendations capture aspects, such as time available, accessibility of environments/items at given times, and personal interests, represents and example of this domain. Often the information provided to the visitors of the museum is enormous, therefore, identify relevant items to see in a limited time is a crucial task.

Public services and utilities for clustering and planning of IoT platforms and architectures in recommender systems can be developed and improved using Artificial intelligence (AI) approaches [8, 15, 36], that enhance features and functionalities like recommending points of interest and route planning. A group of smart profiles are produced through machine learning technique applied to a dataset of users by [5]. In this paper, the development of a privacy-setting interface for Internet of Things services/devices exploiting a data-driven approach is proposed. [13] develops some representation learning architectures to derive distributed vector representations of: traces (*trace2vec*), activities (*act2vec*), logs (*log2vec*) and process models (*model2vec*). The architectures proposed allow for the unsupervised learning of distributed representation vectors. They can be considered as general purpose representations since are not tailored for a specific use case.

Compared to existing approaches, our proposed platform relies on mechanism completely decentralized, self-organizing and automatically adapts to the changing environment, making it particularly suitable for dynamic and unreliable systems. Multiagent approaches guarantee service continuity, stability, and limited user action, but, often, a central management mechanism is necessary with all resulting disadvantages, among which a central point of failure. Our approach, in which the final task is performed by the uninformed cooperation of autonomous entities, is fully self-organizing, distributed and with a high level of resilience. Moreover, our approach allows executing *informed* discovery operations, typical of structured P2P systems, in a fully unstructured and highly dynamic environment such as IoT, also allowing range queries. Indeed, unstructured distributed do not manage range queries of data efficiently, which is a common access pattern in IoT applications.

3 NARIoT platform architecture

The logical schema reporting the application context of the NARIoT platform is shown in Fig. 1. The Physical Layer, in which a collection of IoT objects, connected among them collect information related to the physical world, interacts with a Cyber Layer, in which intelligent agents,

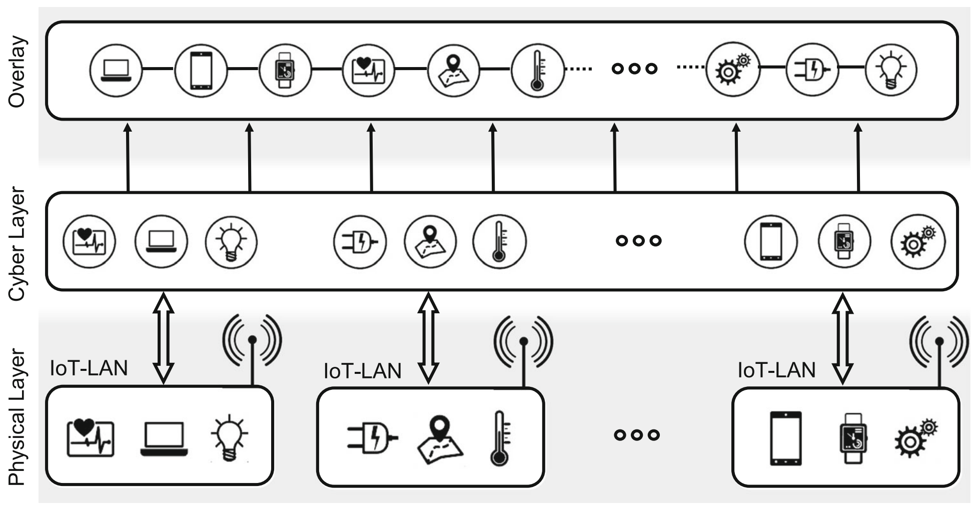

Fig. 1 Logical schema of NARIoT platform. The Overlay layer contains the virtual structure emerging as outcome of the algorithm, while the Cyber layer includes the agents representing the devices/ services of the IoT environment (i.e., the Physical layer)

representing IoT objects, work together following the predefined algorithm. Each object is associated with a single agent, such that the total number of agents N_{ca} is equals to the total number of physical objects N_{po}. The Overlay represents the emergent layer, i.e the ordered virtual structure obtained as result of the algorithm performed the the cyber agents. The algorithm executed continuously by all cyber agents determines the topology of the emerging virtual structure. In particular, the neighbors selection strategy employed by the agents represents the core of the algorithm. Indeed, on the basis of which agents—IoT objects—are linked among them, and then they can be considered neighbors, a particular structure emerges.

Two different algorithms (neighbors' selection strategies), detailed in Sect. 3.2, were designed and implemented that allow emerging overlay networks of devices as: (i) ordered list, and (ii) ordered list of clusters. Metadata, text files containing all physical and behavioral characteristics of IoT objects, are elaborated through *Doc2Vec* model, detailed in Sect. 3.1, so obtaining vector representations of them. The cyber agents exploit these vector representations of IoT objects to perform needed computations in the algorithm execution.

In Fig. 2 is reported a conceptual schema of the logical blocks composing the cyber agent. In particular, the *IoT Connector* takes care of the communication and data exchange with the physical IoT device/service represented. Data coming from the physical layer are stored in a *Metadata Repository* and exploited by the NLP-based library to generate the vector representing the physical IoT object.

The *Overlay Manager* handles all network issues, links, neighbors and connections with the others cyber agents. Moreover, it manages the dynamic building of the *Neighbors List*, which contains the group of linked cyber agents.

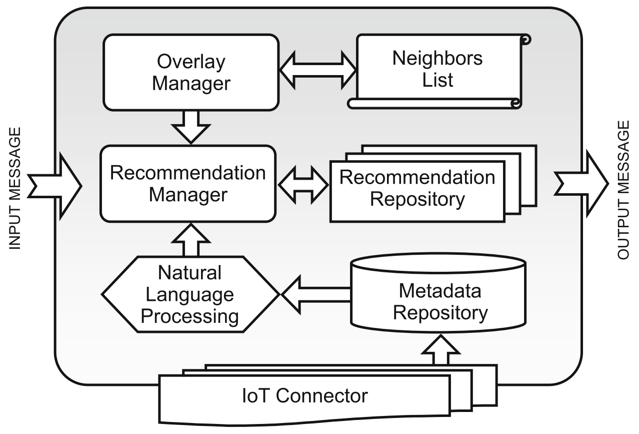

Fig. 2 Conceptual model of the Cyber agent entity

The key element of the schema is the *Recommendation Manager* that, on the basis of the recommendation algorithm defined, drives the others blocks to perform predefined operations. A *Recommendation Repository* keeps a historical data of previous recommendations and possibly exploits them in new similar suggestions. To communicate among them in the cyber layer, a messaging mechanism is exploited, where the messages exchanged contain data regarding to the action to execute.

3.1 The *Doc2Vec* neural model

IoT devices, sensors, services, etc., are represented through *real-valued vectors* obtained with *Doc2Vec* neural model [23], a Natural Language Processing tool for representing sentences/documents, applied to the metadata (text) describing them.

Doc2Vec is an unsupervised algorithm stemming from *Word2vec,mikolov*, a two layer artificial neural network able to process text learning relationships between words. *Word2vec* analyzes a large corpus of text and generates a high-dimensional space—several hundred—in which each word in the corpus corresponds to a single vector in the space. The word embedding method allows to capture different degrees of similarity between two words on the basis of the corpus.

Two models are available to generate the representation: (i) Continuous Bag of Words (CBOW), starting from the context and (ii) Skip-gram, that starts the target word. In Skip-gram, the input is a word and the output is a context word. Each input word defines the number of right or left context words to predict by the window size hyper-parameter. Continuous Bag of Words (CBOW) differs to skip-gram mainly for one aspect: the input is multiple words that are merged via vector addition to predict the context word. The word embedding model aims to maximize the log probability, reported in formula 1, for all words in the device corpus.

$$\log \quad probability(word|context) = \frac{S_d(word_i|context)}{\sum_k S_d(word_k|context)} \tag{1}$$

$S_d(word_i|context)$ represents the similarity degree of the *i-word* in the current *context*.

Doc2Vec is an extension of Word2Vec for handling document embeddings. There are two approaches within Doc2Vec: *dbow* and *dmpv*. *dbow* works similarly to skip-gram, but the input is represented by a special item representing the document. The order of words in the document is ignored; hence, the name distributed bag of words. *dmpv* works similarly to *CBOW*. *dmpv* input is an additional document item in addition to multiple objective words. Differently from CBOW, these vectors are concatenated and, using a document and word vectors, the content word is predicted.

3.2 Multiagent algorithms allowing overlay network to emerge

The neighbors selection strategy adopted by the agents determines the topology of the virtual structure emerging. Two different algorithms allowing two different virtual structures to emerge were designed. The first algorithm, detailed in Sect. 3.3, allows building of a virtual ordered list, while the second algorithm, specified in Sect. 3.4, enables the emerging of an ordered list of clusters, i.e., groups of devices similar among them. The designed algorithm assures that in a finite number of steps a *steady situation* is reached, i.e., no agent executes any operation, and the ordered structure emerges. The agents are always in the execution even if they do not make operations because IoT objects leave and join the system dynamically, and the virtual structure has to be updated.

The cosine distance, as reported in formula (2), allowing the distance computation between two vectors, was exploited for agents' similarity comparison.

$$\cos(\overrightarrow{u}, \overrightarrow{v}) = \frac{\overrightarrow{u} \cdot \overrightarrow{v}}{|\overrightarrow{u}| \times |\overrightarrow{v}|} = \frac{\sum_{i\epsilon n} u_i v_i}{\sqrt{\sum_{i\epsilon n} u_i^2}\sqrt{\sum_{i\epsilon n} v_i^2}} \tag{2}$$

u and *v* represent the vectors mapping of two IoT objects in *n-dimensional* space. The designed algorithms with different selection strategy of neighbors are proposed in the following.

3.3 Topology as ordered list

A useful topology enabled by agents' work is represented by a ordered list. The similarity degree with a given IoT object having fixed characteristics (see Overlay layer in Fig. 1) determines the position in the list. The first element of the list is the IoT object, among all, with the highest value of similarity. The second element will have the second highest value of the similarity with the target object, and so on. Algorithm 1 reports the algorithm performed autonomously by the generic agent A_{cyber} with the associated vector v_c.

Algorithm 1 Pseudo code allowing ordered list to emerge.

```
1:  while true do
2:      $computeLists(H_{ca}, L_{ca})$;
3:      if $H_{ca}.length() \geq 2$ then
4:          $[A_{max}, A_{submax}] = identifyMaxSubmax(H_{ca})$;
5:          $createLink(A_{max}, A_{submax})$;
6:          $remove(H_{ca}, A_{max})$;
7:      else if $L_{ca}.length() \geq 2$ then
8:          $[A_{min}, A_{submin}] = identifyMinSubmin(L_{ca})$;
9:          $createLink(A_{min}, A_{submin})$;
10:         $remove(L_{ca}, A_{min})$;
```

Through the function *computeList*(...), the agent compute H_{ca} and L_{ca}, the lists containing the connected agents—representing initially the connected IoT objects in the physical layer—with vector value higher and lower than $v\ c$, respectively. The function *identifyMaxSubmax*(*list*) returns the linked cyber agents contained in *list* having max and sub-max similarity value with the current cyber agent, while the function *identifyMinSubmin*(*list*) provides those having the min and sub-min similarity value. The function $createLink(ca_a, ca_b)$ produces a virtual link between cyber agent ca_a and cyber agent ca_b. Simply, ca_a adds ca_b in its neighbors list and notifies ca_b of including ca_a in its neighbors list. *remove(list, ca)* function removes the cyber agent *ca* from the list *list*. This function is exploited to eliminate a neighbor of a cyber agent, simply removing it from the neighbors list.

Autonomously and in parallel, each cyber agent adds and removes other cyber agents in its neighbors lists, following the rules of the algorithm. When an agent is added/removed to/from one of the lists, ideally, a virtual link is created/deleted between the current agent and the added/removed agent. After a transition phase, i.e when a steady situation is reached, each cyber agent will be connected (with a real or virtual link, but always belonging to the virtual structure) with two cyber agents: the cyber agent having the vector value immediately less similar (contained in L_{ca}) and the cyber agent with the vector value immediately more similar (contained in H_{ca}) of the all cyber agents.

Once a steady situation is reached, the cyber agents will be arranged in a sorted list organized on the basis of the similarity degree with the target agent representing the required IoT object. Then, an organized virtual structure of IoT objects is generated on the basis of the similarity with a given IoT object. The needed number of IoT devices/services can be selected starting from the head of the list and, thanks to the ordering, they will be the IoT objects with highest similarity degree among those available. The selected IoT objects are very close, similar and most likely useful for users, so allowing very effective recommendation operations.

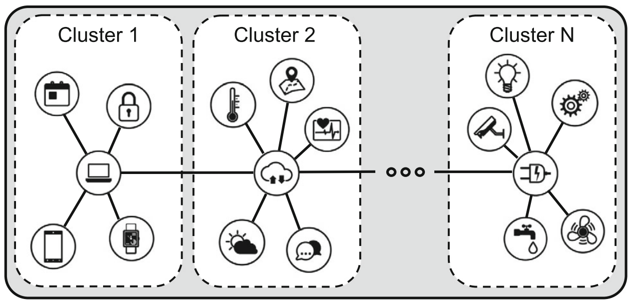

Fig. 3 An example of orderly linked clusters structure

3.4 Topology as ordered list orderly linked clusters

A further useful organization strategy, engineered in this paper, consists in grouping cyber agents on the basis of the similarity degree with the target IoT object in a given number of clusters C_l. The clusters are connected among them as an ordered list in this way a fast localization/recommendation operation is guaranteed. In Fig. 3 is shown a graphic example of overlay network in which all agents are grouped in *N* clusters, on the basis of the similarity with a given IoT object.

All the agents belonging to the cluster *k* have a similarity value higher than all the agents belonging to the cluster *k-1* and lower than all the agents belonging to the cluster *k+1*. The cyber agents having the highest value of similarity for each cluster are orderly linked among them.

The algorithm autonomously performed by each cyber agent A_{cyber} having vector v_c to bring out the overlay network is reported in Algorithm 2. Here, $List^h_{noclus}$ and $List^l_{noclus}$, the lists containing the linked cyber agents with vector value higher and lower, respectively, than v_c, not belonging to the same cluster of A_{cyber}, $List^h_{clus}$ and $List^l_{clus}$ containing the cyber agents belonging to the same cluster of A_{cyber} with vector value higher and lower, respectively, than v_c, are computed through the function *computeLists*(....).

Algorithm 2 Pseudo code allowing orderly linked clusters to emerge.

```
1: while true do
2:     $computeLists(List^h_{noclus}, List^l_{noclus}, List^h_{clus}, List^l_{clus})$;
3:     if $(List^h_{clus}.length() > 1)$ then
4:         $[A_{max}, A_{submax}] = identifyMaxSubmax(List^h_{clus})$;
5:         $createLink(A_{max}, A_{submax})$;
6:         $remove(List^h_{clus}, A_{max})$;
7:     else if $(List^l_{clus}.length() > 0) \wedge (List^h_{clus}.length() > 0)$ then
8:         $A_{min} = identifyMin(List^l_{clus})$;
9:         $A_{max} = identifyMax(List^h_{clus})$;
10:        $createLink(A_{min}, A_{max})$;
11:        $remove(List^l_{clus}, A_{min})$;
12:    else if $(List^h_{noclus}.length() > 1)$ then
13:        $[A_{max}, A_{submax}] = identifyMaxSubmax(List^h_{noclus})$;
14:        $createLink(A_{max}, A_{submax})$;
15:        $remove(List^h_{noclus}, A_{max})$;
16:    else if $(List^l_{noclus}.length() > 1)$ then
17:        $[A_{min}, A_{submin}] = identifyMinSubmin(List^l_{noclus})$;
18:        $createLink(A_{min}, A_{submin})$;
19:        $remove(List^l_{noclus}, A_{min})$;
20:    else if $(List^l_{noclus}.length() > 0) \wedge (List^h_{clus}.length() > 0)$ then
21:        $A_{min} = identifyMin(List^l_{noclus})$;
22:        $A_{max} = identifyMax(List^h_{clus})$;
23:        $createLink(A_{min}, A_{max})$;
24:        $remove(List^l_{noclus}, A_{min})$;
```

The cyber agents contained in *list* having the max and sub-max similarity value with the current cyber agent are returned from the function *identifyMaxSubmax*(*list*), and the cyber agents having the min and max similarity value are provided by the functions *identifyMin*(*list*) and *identifyMax*(*list*), respectively.

The function *identifyMinSubmin*(*list*) provides the linked cyber agents contained in *list* with the vector value having the min and sub-min similarity value with the current cyber agent. The function $createLink(ca_a, ca_b)$ produces a virtual link between cyber agent ca_a and cyber agent ca_b. Essentially, to create a virtual link between ca_a and ca_b, ca_a adds ca_b in its neighbors list and notifies ca_b of including ca_a in his neighbors list. *remove(list, ca)* function removes the cyber agent *ca* from the list *list* and implicitly it removes the link between *ca* and the current agent.

In Fig. 4 is reported, as example, the outcome obtained by the application of the algorithm to a very simple IoT system, in which $N_{\text{po}} = N_{\text{ca}} = 50$. We can note that, in the resulting overlay network:

– all agents, and consequently all IoT objects existing in the environment, are grouped in clusters;
– all the agents belonging to the cluster k have a similarity degree higher than all the agents belonging to the cluster $k-1$ and lower than all the agents belonging to the cluster $k+1$;
– in each group, the agent representing the device with highest similarity degree of the cluster acts as a *star point* and all other agents belonging to the same cluster are linked to it;
– all the star points are connected among them in order to form an ordered list ensuring fast discovery operations.

Once a steady situation is reached, the cyber agents will be arranged in a sorted list of groups of IoT objects, organized on the basis of the similarity degree with the target agent representing the required IoT object. Therefore, an organized virtual structure of IoT objects is generated on the basis of the similarity with a given IoT object. The cyber agents are organized in clusters on the basis of the similarity with the target object, and it is possible selecting one or more clusters, and consequently, a number of similar, and probably useful, IoT objects can be suggested/recommended.

4 Experimental results

In order to validate the proposed system, a prototype of the platform NARIoT, written in Java, was implemented. The prototype enables a simulation environment in which a group of cyber agents, N_{ca}, equals to the number of IoT objects, $N\,po$, executes the proposed algorithms allowing to the virtual ordered structure to emerge. As test bench, a set of interconnected IoT networks, in which the components (devices/services) of each one are connected among them through a local area network (LAN), were designed and implemented.

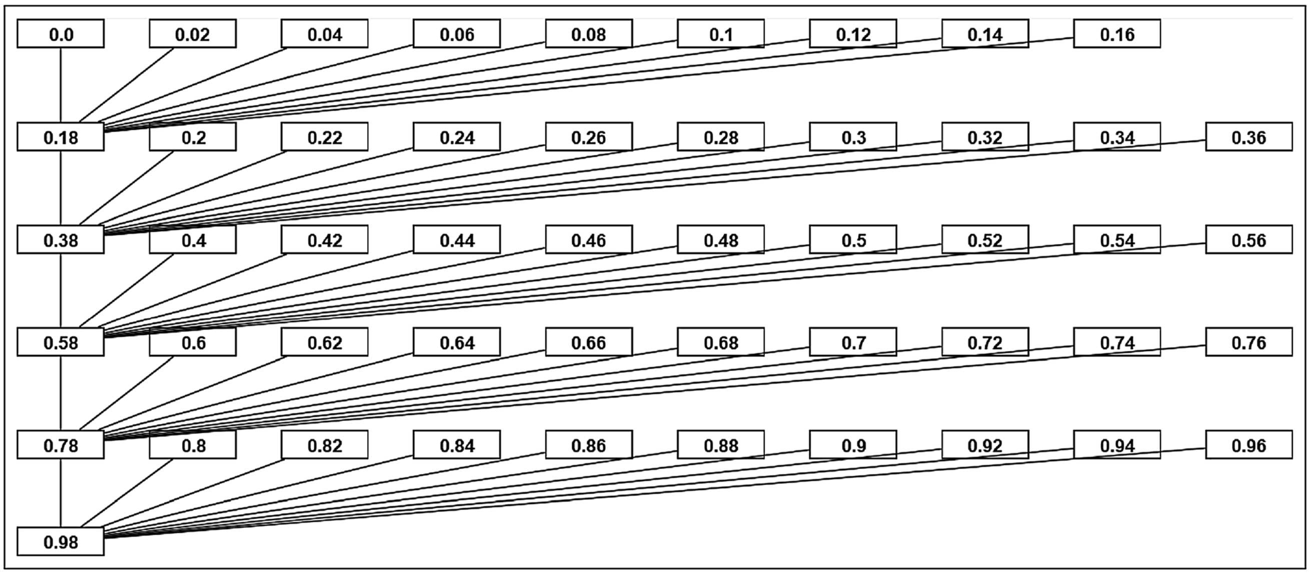

Fig. 4 Example of a virtual structure with $Npo = N_{ca} = 50$. Each agent (a rectangular block) contains the similarity degree with the target IoT object

In order to arrange a simulation scenario as close as possible to the real world, all IoT objects belonging to the smart environment have been represented using real data related to physical characteristics. In order to build the synthetic dataset, the CASAS[3] (*Center of Advanced Studies in Adaptive Systems*) data [10] were exploited. The CASAS project, devised at Washington State University, aims to design a smart home that is small, lightweight in infrastructure, extendable, and ready to perform main capabilities quickly. A consistent set of data related to smart home sensors, consisting of a date, time, sensor identifier, the sensor name, the sensor state, sensor message, etc., was downloaded and employed to obtain a series of events [9]. Each device was described by means of a JSON file, containing its peculiar characteristics and classified on the basis of them to obtain the *ground truth* for the performance evaluation.

By applying the *Doc2Vec* library to this metadata, each IoT object was represented through a *real-valued vector* with dimension equals to 400. *Doc2Vec* model was trained on large external corpora WIKI, the full collection of English Wikipedia.[4] In this paper, we report the experiment results of the platform related to Algorithm 2. The topology as ordered list of linked clusters, indeed, includes the ordered list topology because it is a simplified version in which each cluster is composed by a single cyber agent. The max threshold of the cosine distance to consider a couple of objects similar is set to 0.3 [14].

[3] http://ailab.wsu.edu/casas/datasets/.

[4] Cleaned with WikiExtractor: https://github.com/attardi/wikiextractor.

4.1 Convergence, scalability and computational traffic evaluation

Algorithm convergence and scalability were evaluated by computing the max number of *iterations* in which a steady situation is reached, i.e., no operations are performed by cyber agents since the overlay network is emerged. *iteration* denotes a single execution of the loops of the algorithm, e.g., from `line 2` to `line 24` in Algorithm 2. Results are reported in Fig. 5.

The number of cyber agents considered ranges from 0 to 10.000, while the number of clusters, each with different similarity threshold, ranges from 2 to 100 and the mean number of neighbors for each agent, *MeanNgh*, is set to 6. Notice that the number of iterations needed to obtain the orderly list of clusters is inversely related to the number of clusters.

Figure 6 shows the max number of iterations performed by each cyber agent—with number of cyber agents ranging from 0 to 10.000—to obtain the orderly linked clusters overlay for different initial mean number of neighbors for each cyber agent, i.e., when *MeanNgh* ranges from 4 to 16. The number of clusters, C_l is set to 20. The number of iterations required increases as *MeanNgh* decreases.

In Fig. 7, the computational traffic, computed as the mean number of messages exchanged by each cyber agent to obtain the ordering for an increasing number of cyber agents, is reported. Here, *MeanNgh* is fixed to 6, while C_l ranges from 2 to 100. We can note that, for a number of agents higher than about 2.000, the mean number of messages is almost constant. Figure 8 reports the computational traffic, T_c, to obtain the ordering for an increasing number

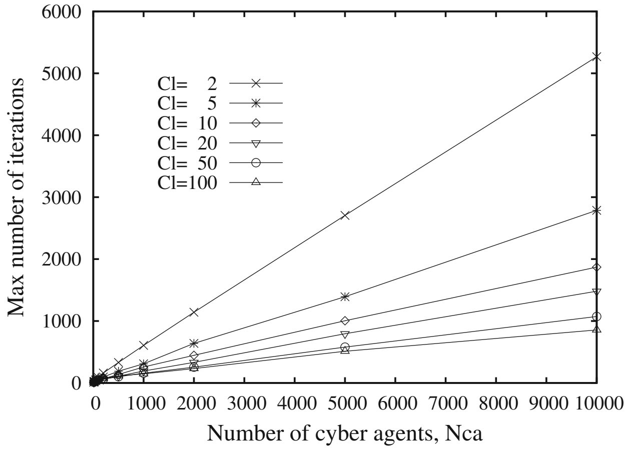

Fig. 5 Max number of iterations needed to obtain the ordered structure for different values of C_l, when the number of cyber agents ranges from 0 to 10.000 and *MeanNgh* is set to 6

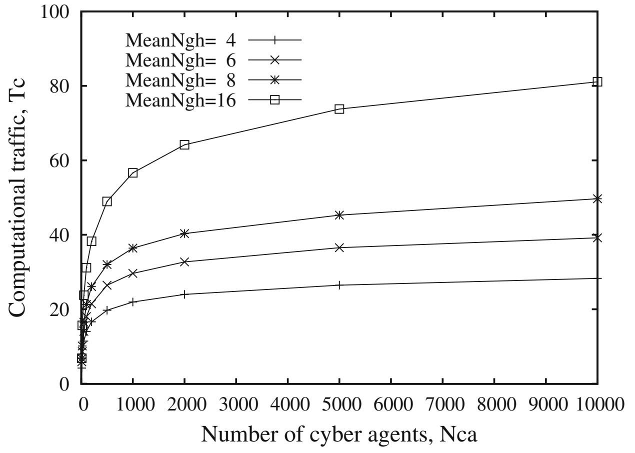

Fig. 8 The mean number of messages exchanged by each cyber agent to obtain the ordering for an increasing values of *MeanNgh*. C_l is set to 20

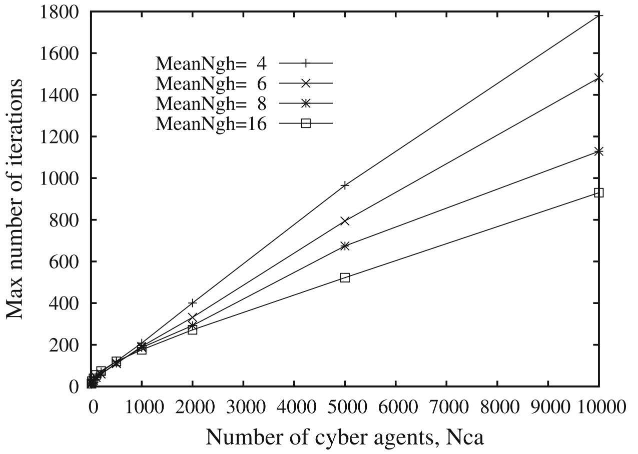

Fig. 6 Max number of iterations needed to obtain the ordered structure for different values of *MeanNgh*, when the number of cyber agents ranges from 0 to 10.000 and C_l is equals to 20

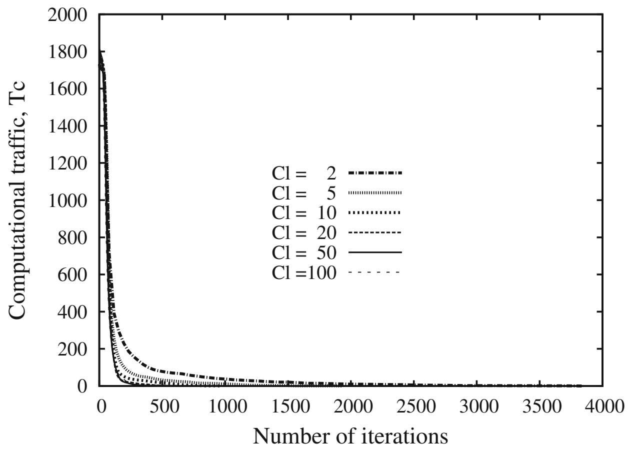

Fig. 9 The mean number of messages exchanged by each cyber agent to obtain the ordering for an increasing number of C_l. *MeanNgh* is fixed to 6

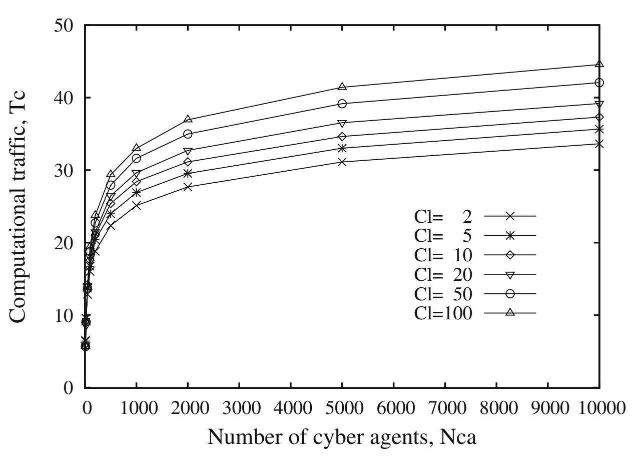

Fig. 7 The mean number of messages exchanged by each cyber agent to obtain the ordering for an increasing number of cyber agents. *MeanNgh* is fixed to 6

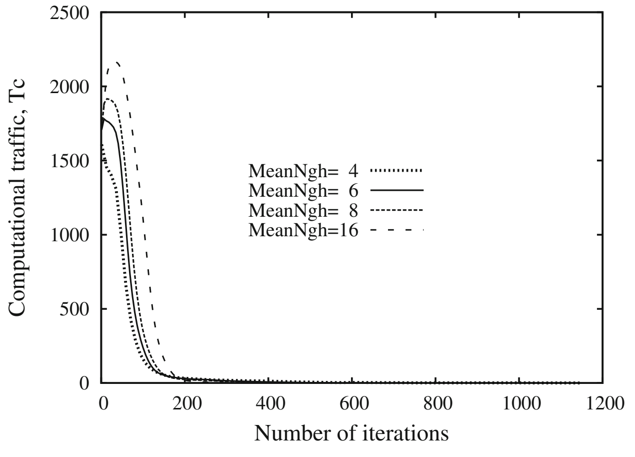

Fig. 10 The mean number of messages exchanged by each cyber agent to obtain the ordering for an increasing values of *MeanNgh*. C_l is set to 20

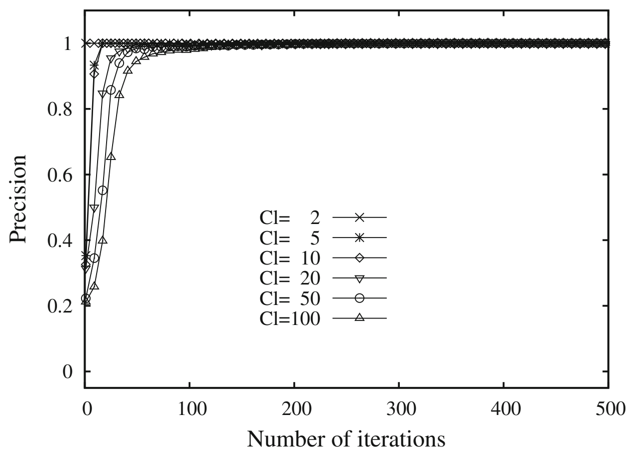

Fig. 11 Precision for different number of clusters. The number of cyber agents is fixed to 10.000, while *MeanNgh* is set to 6

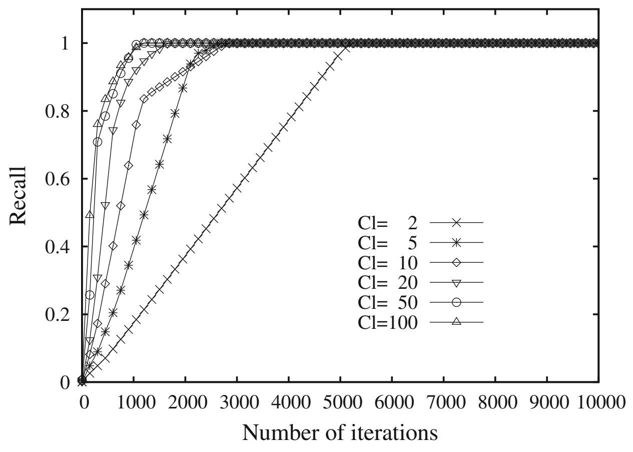

Fig. 12 Recallfor different number of clusters. The number of cyber agents is fixed to 10.000, while *MeanNgh* is set to 6

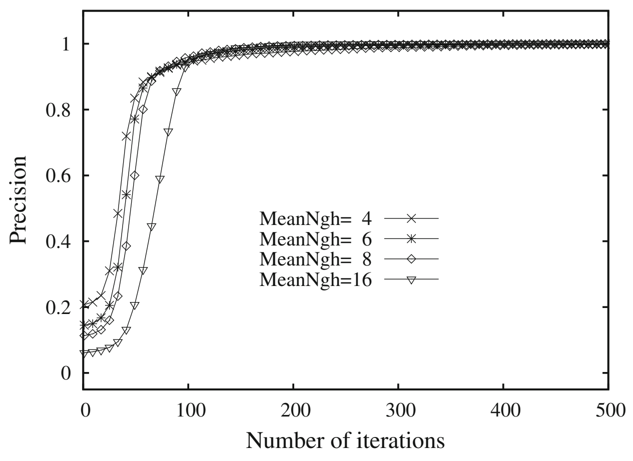

Fig. 13 Precision for different initial mean number of neighbors for each cyber agent. The number of cyber agents involved is fixed to 10.000, while $C_l = 100$

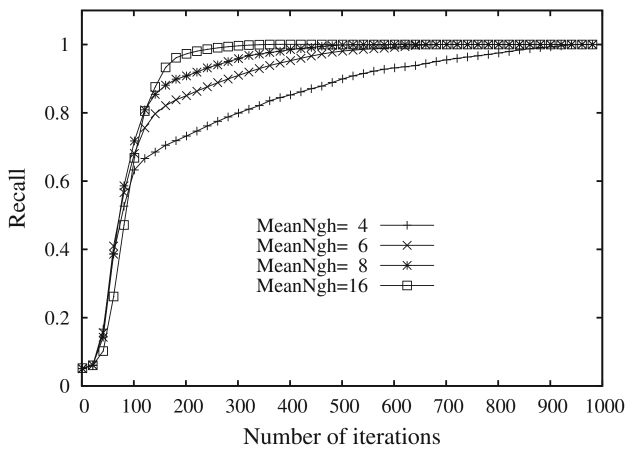

Fig. 14 Recall for different initial mean number of neighbors for each cyber agent. The number of cyber agents involved is fixed to 10.000, while $C_l = 100$

of cyber agents, when *MeanNgh* ranges from 4 to 16, the number of groups (clusters) in which the cyber agents are divided, C_l is set to 20. Also in this case, for a number of agents higher than about 2.000, T_c is almost constant.

Figures 9 and 10 show the values of T_c when the algorithm evolves, i.e., the number of iterations increases, for C_l ranging from 2 to 100 and $MeanNgh = 6$, in the first case, and for *MeanNgh* ranging from 4 to 16 and $C_l = 20$, in the second one. In both cases, we can note that, after a limited number of iterations in which a high number of messages are exchanged (transition phase), the traffic becomes almost zero, meaning that the algorithm converged and the ordered structure is emerged.

4.2 Relevance evaluation

We also exploited precision and recall measures, to evaluate the relevance of the behavior of the recommendation approach proposed in this paper [21]. precision and recall are reported in formulas (3), where TP (True Positive) is the number of items correctly assigned to the positive class, TN (True Negative) is the number of items correctly assigned to the negative class, FP (False Positive) is the number of items assigned to the positive class which instead belong to the negative class and FN (False Negative) is the number of observations assigned to the negative class, which instead belong to the positive class.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}; \qquad Recall = \frac{\text{TP}}{\text{TP} + \text{FN}}; \qquad (3)$$

In information retrieval, precision is the fraction of relevant items among the detected items, while recall is the fraction of all relevant items that were effectively detected. precision can be seen as a measure of exactness or quality, whereas recall is a measure of completeness or quantity.

For recommender systems, precision can be calculated as the ratio of relevant recommended items to all recommended items, while recall can be calculated as the ratio of relevant recommended items to the all relevant items. In our case, precision is calculated as the ratio of relevant linked agents to all liked agents, while recall is calculated as the ratio of relevant linked agents to the all relevant agents, for each cluster.

Figures 11 and 12 depict the performance in terms of precision and recall values for different number of clusters, when the number of cyber agents involved is fixed to 10.000 and the number of neighbors of each agent is set to 6. Notice that the approach shows very good performance.

Figures 13 and 14 report the values of precision and recall for different initial mean number of neighbors, *MeanNgh*. The cyber agents involved is fixed to 10.000, and the number groups in which agents are partitioned, C_l is set to 100. The performance is very positive also in this case, for each considered value. Indeed, independently of the number of linked neighbors for each objects and the fixed similarity threshold—implicitly set selecting the number of groups C_l—the algorithm achieves very good clustering performance, as proved by all experiments.

5 Conclusions

In this paper, a recommendation platform in IoT environment leveraging on intelligent agents-based algorithms was proposed. Two agents-based algorithms, that allow building dynamic virtual structures enabling fast and effective recommendation operations, were designed, and their validity has been demonstrated. Thanks to the benefits offered by agents, as self-organizing, extensibility, flexibility and autonomy, the approaches are very suitable for dynamic and high dimensional environments like IoT, because they automatically adapt to changes. Moreover, the proposed approaches allow executing *informed* discovery operations, typical of structured systems, in unstructured and highly dynamic environments such as IoT, also allowing range queries. Indeed, unstructured systems do not manage range queries of data efficiently, which is a common useful access pattern in IoT applications. Experimental results have shown how the proposed algorithms achieve an effective resources reorganization allowing effective recommendation tasks.

As future work, we aim to incorporate further intelligence and advanced functions in the objects networking and similarity computations. A cognitive process that can capture current network conditions, choose, schedule, act on those conditions, learn from its actions, all while following given objectives.

The purpose is to provide the platform with the ability to be aware of its operational status and adjust its operational parameters to fulfill specific tasks and behaviors, such as detecting changes in the environment and user requirements. Content-addressable network paradigm and semantic web technologies will be investigated to enhance the proposed approach designing advanced functionality.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Adomavicius G, Mobasher B, Ricci F, Tuzhilin A (2011) Context-aware recommender systems. AI Mag 32:67–80
2. Altulyan M, Yao L, Wang X, Huang C, Kanhere SS, Sheng QZ (2021) A survey on recommender systems for internet of things: techniques. Comput J Appl Future Direct. https://doi.org/10.1093/comjnl/bxab049
3. Amato F, Mazzeo A, Moscato V, Picariello A (2013) A recommendation system for browsing of multimedia collections in the internet of things. Springer, Berlin, pp 391–411
4. Atzori L, Iera A, Morabito G (2010) The internet of things: a survey. Computer networks 54(15):2787–2805
5. Bahirat P, He Y, Menon A, Knijnenburg B (2018) A data-driven approach to developing iot privacy-setting interfaces. In: In Proceedings of 23rd international conference on intelligent user interfaces. ACM, pp 165–176. https://doi.org/10.1145/3172944.3172982
6. Balabanović M, Shoham Y (1997) Fab: content-based, collaborative recommendation. Commun ACM 40(3):66–72
7. Breese JS, Heckerman D, Kadie C (1998) Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the 14th conference on uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., pp 43–52
8. Cha S, Ruiz MP, Wachowicz M, Tran LH, Cao H, Maduako I (2016) The role of an iot platform in the design of real-time recommender systems. In: Proceedings of 2016 IEEE 3rd world forum on internet of things (WF-IoT), pp 448–453. https://doi.org/10.1109/WF-IoT.2016.7845469
9. Cook D, Schmitter-Edgecombe M, Crandall A, Sanders C, Thomas B (2009) Collecting and disseminating smart home sensor data in the casas project. In: Proceedings of CHI09 workshop on

developing shared home behavior datasets to advance HCI and ubiquitous computing research
10. Cook DJ, Crandall AS, Thomas BL, Krishnan NC (2013) Casas: a smart home in a box. Computer 46(7):62–69. https://doi.org/10.1109/MC.2012.328
11. Crespo A, Garcia-Molina H (2002) Routing indices for peer-to-peer systems. In: 22nd international conference on distributed computing systems, 2002. Proceedings. IEEE, pp 23–32
12. De S, Barnaghi P, Bauer M, Meissner S (2011) Service modelling for internet of things, pp 949–955
13. De Koninck P, vanden Broucke S, De Weerdt J (2018) act2vec, trace2vec, log2vec, and model2vec: representation learning for business processes. In: Weske M, Montali M, Weber I, vom Brocke J (eds) Business process management. Springer, Cham, pp 305–321
14. Dharaneeshwaran, Nithya S, Srinivasan A, Senthilkumar M (2017) Calculating the user-item similarity using pearson's and cosine correlation. In: 2017 International conference on trends in electronics and informatics (ICEI), pp 1000–1004. https://doi.org/10.1109/ICOEI.2017.8300858
15. Di Martino S, Rossi S (2016) An architecture for a mobility recommender system in smart cities. Procedia Comput Sci 98:425–430. https://doi.org/10.1016/j.procs.2016.09.066
16. Djamaa B, Senouci MR, Bessas H, Dahmane B, Mellouk A (2021) Efficient and stateless p2p routing mechanisms for the internet of things. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2021.3053339
17. Djellabi B, Younis M, Amad M (2020) Effective peer-to-peer design for supporting range query in internet of things applications. Comput Commun 150:506–518. https://doi.org/10.1016/j.comcom.2019.12.017
18. Forestiero A, Mastroianni C, Spezzano G (2007) Antares: an ant-inspired p2p information system for a self-structured grid. In: 2007 2nd bio-inspired models of network, information and computing systems, pp 151–158. https://doi.org/10.1109/BIMNICS.2007.4610103
19. Forestiero A, Mastroianni C, Spezzano G (2008) Building a peer-to-peer information system in grids via self-organizing agents. J Grid Comput 6(2):125–140. https://doi.org/10.1007/s10723-007-9062-z
20. Fortino G, Russo W, Savaglio C, Shen W, Zhou M (2018) Agent-oriented cooperative smart objects: From iot system design to implementation. IEEE Trans Syst Man Cybern Syst 48(11):1939–1956. https://doi.org/10.1109/TSMC.2017.2780618
21. Gunawardana A, Shani G (2009) A survey of accuracy evaluation metrics of recommendation tasks. J Mach Learn Res 10:2935–2962. https://doi.org/10.1145/1577069.1755883
22. Huang CY, Wu CH (2016) A web service protocol realizing interoperable internet of things tasking capability. Sensors 16:1395. https://doi.org/10.3390/s16091395
23. Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: International conference on machine learning, pp 1188–1196
24. Lee J, Su Y, Shen C (2007) A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In: Proceedings of 33rd annual conference of the IEEE industrial electronics society, pp 46–51. https://doi.org/10.1109/IECON.2007.4460126
25. Lorenzi F, Baldo G, Costa R, Abel M, Bazzan A, Ricci F (2010) A trust model for multiagent recommendations. J Emerg Technol Web Intell 2:2010. https://doi.org/10.4304/jetwi.2.4.310-318
26. Lorenzi F, Bazzan AL, Abel M, Ricci F (2011) Improving recommendations through an assumption-based multiagent approach: An application in the tourism domain. Expert Syst Appl 38(12):14703–14714. https://doi.org/10.1016/j.eswa.2011.05.010
27. Magerkurth C, Sperner K, Meyer S, Strohbach M (2011) Towards context-aware retail environments: an infrastructure perspective. In: Proceedings of the mobile interaction in retail environments (MIRE 2011), pp 1–4
28. Morais AJ, Oliveira E, Jorge A (2012) A multi-agent recommender system. Adv Intell Soft Comput. https://doi.org/10.1007/978-3-642-28765-7_33
29. Organero M, Ramirez-Gonzalez G, Merino P, Delgado-Kloos C (2010) A collaborative recommender system based on space-time similarities. IEEE Pervasive Comput 9:81–87. https://doi.org/10.1109/MPRV.2010.56
30. Piccialli F, Jeon G (2021) Context-aware computing for the internet of things. Internet Things 14:100154
31. Piccialli F, Jeon G (2021b) Toward the internet of things of year 2020: Applications and future trends. Concurr Comput Practice Exp 33::e5733, https://doi.org/10.1002/cpe.5733
32. Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) Grouplens: an open architecture for collaborative filtering of netnews. In: Proceedings of the ACM conference on Computer supported cooperative work. ACM, pp 175–186
33. Savaglio C, Ganzha M, Paprzycki M, Bădică C, Ivanović M, Fortino G (2020) Agent-based internet of things: State-of-the-art and research challenges. Future Gener Comput Syst 102:1038–1053. https://doi.org/10.1016/j.future.2019.09.016
34. Selmi A, Brahmi Z, Gammoudi M (2014) Multi-agent recommender system: State of the art. In: Proceedings of the 16th international conference on information and communications security
35. Skocir P, Marusic L, Marusic M, Petric A (2012) The mars: A multi-agent recommendation system for games on mobile phones. In: Proceedings of the 6th international conference on agent and multi-agent systems: technologies and applications, pp 104–113. https://doi.org/10.1007/978-3-642-30947-2_14
36. Yavari A, Jayaraman PP, Georgakopoulos D (2016) Contextualised service delivery in the internet of things: parking recommender for smart cities. In: Proceedings of 2016 IEEE 3rd world forum on internet of things (WF-IoT), pp 454–459. https://doi.org/10.1109/WF-IoT.2016.7845479