Article

IoT Service Clustering for Dynamic Service Matchmaking

Shuai Zhao [1],* [iD], Le Yu [2], Bo Cheng [1] and Junliang Chen [1]

[1]   State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China; chengbo@bupt.edu.cn (B.C.); chjl@bupt.edu.cn (J.C.)

[2]   China Mobile Information Security Center, Beijing 100033, China; yule@chinamobile.com

*   Correspondence: zhaoshuaiby@bupt.edu.cn; Tel.: +86-010-6119-8022

Abstract: As the adoption of service-oriented paradigms in the IoT (Internet of Things) environment, real-world devices will open their capabilities through service interfaces, which enable other functional entities to interact with them. In an IoT application, it is indispensable to find suitable services for satisfying users' requirements or replacing the unavailable services. However, from the perspective of performance, it is inappropriate to find desired services from the service repository online directly. Instead, clustering services offline according to their similarity and matchmaking or discovering service online in limited clusters is necessary. This paper proposes a multidimensional model-based approach to measure the similarity between IoT services. Then, density-peaks-based clustering is employed to gather similar services together according to the result of similarity measurement. Based on the service clustering, the algorithms of dynamic service matchmaking, discovery, and replacement will be performed efficiently. Evaluating experiments are conducted to validate the performance of proposed approaches, and the results are promising.

Keywords: Internet of things; semantic similarity measurement; multidimensional model; service clustering

1. Introduction

1.1. Background

The Internet of things (IoT) integrates user requirement, cyberspace and physical space, which enables the seamless cooperation of human-machine-thing. SOC (Service-Oriented Computing) proposes techniques for provision, selection, discovery, and composition of Web services, and integrates heterogeneous and complicated software entities together organically [1,2]. As the adoption of service-oriented paradigms in the IoT environment [3], real-world devices will open their capabilities through service interfaces, which enable other functional entities to interact with them. In an IoT application, it is indispensable to find suitable services for satisfying users' requirements or replacing the unavailable services.

With the rapidly growing number of IoT services, discovery and selection for numerous services under the dynamic and large-scale environment of IoT is becoming a crucial task. Several middleware solutions have been proposed for the integration of the physical world with the Web, such as OpenIoT [4], GSN (Global Sensor Networks) [5], and Xively [6]. These solutions act as service platforms that manage millions of services around the world, which enable people to share and monitor environmental data from objects that are connected to the Web. However, most leading middleware solutions provide only limited service discovery and selection functions. It is effective to discover services through service matchmaking techniques [7,8]. However, from the perspective of performance, it is unreasonable to discover services from online repositories directly in the context

of the IoT-scale environment [9,10]. Instead, if services are classified offline or clustered according to their similarity, and then the online examining of services will be controlled within several limited clusters. Then, the performance of finding desired services online will be optimistic [11,12]. Besides, because of the dynamic of IoT, the selected services may become unavailable or unfit for the current context, so re-selection and replacing services by similar services from the same cluster is necessary. Therefore, techniques that cluster services offline according to their similarity are critical for dynamic service matchmaking, discovery, and replacement [13].

1.2. Motivation

As the research [14] discusses, nearly 12,000 Web services are active on the Web. Even in such conditions, the similarity measurement and clustering of Web service has become a challenging problem. The same issue of IoT service will become a much more complex challenge due to the scale and complexity of IoT. As the IoT service acts on ternary space (i.e., user, cyber, and physical space) rather than Web service that only exists in cyber space, the context of IoT service is more complex than Web service. IoT services imply multidimensional semantic, for instance, the physical quantity observed by the service, the observation capabilities of service, the observation area of the service, and so on. When clustering or measuring similarity between services, these information should be taken into consideration.

A bundle of approaches about measuring similarity between Web services has been proposed in recent years. Basically, it can be divided into: information content-based approaches [15,16] and semantic-network-structure-based approaches [17,18]. However, it is inappropriate that directly using existing approaches on IoT services. Existing similarity measurements focus on the hierarchy and inheritance relations between services in the semantic model. They ignore the relation types, relation contexts, and relation restrictions that imply meaningful semantic information for distinguishing service. Besides, service nodes in a semantic model are defined with data-type and object-type properties. The properties should also not be ignored when computing the similarity between services.

Moreover, existing service models mix multiple feature dimensions of IoT service to construct complicated models. The dimension-mixed model cannot obtain a well-defined taxonomy structure. Thus, using semantic structure-based algorithms to measure the dimension-mixed model will not achieve a satisfactory accuracy. Besides, the property and restriction descriptions of multiple dimensions interfere with each other when measuring similarity based on service description. Therefore, using the existing algorithms to measure IoT services will not conform to the equivalence soundness and disjointedness incompatibility principles of similarity measurement, and the measurement results cannot reflect the real similarity between services [19]. Without accurate similarity measurement, it is impossible to obtain satisfactory clustering results, which will influences the effect of follow-up matchmaking and discovery of services.

This paper proposes the multidimensional semantic model for describing IoT services. Each dimension constructs a semantic model including well-defined service classification, service properties, and property constraints. Based on this multidimensional service model, we propose an MDM (Multiple Dimensional Measuring) algorithm to calculate the similarity between services on each dimension by taking both model structure and model description into consideration. The similarity between services on each dimension is measured concurrently. If the context of service changes, MDM just needs to re-measure the similarity of changed dimensions, rather than existing approaches which require re-measuring the whole similarity. Thus, compared with dimension-mixed approaches, MDM is more accurate and efficient. After that, based on the result of similarity measurement, we employs density-peaks-based clustering [20] to divide services into clusters according to the distribution of their similarity. The similar service clusters are generated automatically without the artificial estimating of parameter (e.g., cluster size or number of cluster). Different services have personalized cluster sizes, which take the heterogeneity of service context into consideration. After clustering, the agile service matchmaking and discovery are possible. In particular, this paper has the following contributions:

1. This paper proposes the MDM algorithm for measuring the similarity between IoT services based on multidimensional service model. The accuracy and efficiency of MDM outperform the dimension-mixed approaches.
2. MDM algorithm employs a density-peaks-based clustering approach to gather similar services together according to the actual distribution of services. It avoids the complicated process of estimating or optimizing parameters.
3. To evaluate the applicability of proposed approaches, we use a combined data set including real and synthetic data. The experiment results indicate that the performance of proposed approaches are applicable to real-life scenarios.

2. Preliminaries: Multidimensional Service Model and Model Vectorization

A series of works have been proposed to formally describe IoT services in ontology models, such as references [21–28]. However, existing models mix multiple feature dimensions of IoT service to construct complicated models. Therefore, in a model hierarchy multiple classifying criterions are referenced. The dimension-mixed model cannot obtain a well-defined taxonomy structure, and the distance and positional relationships between nodes are meaningless to reflect the similarity between services. Therefore, using semantic structure-based algorithms to measure the similarity between services will not achieve a satisfactory accuracy. Besides, the restrictions and property descriptions of different dimensions interfere with each other when measuring similarity based on service descriptions.

Based on the multidimensional service model proposed in our previous work [29], the service classification, service properties, and property constraints of each dimension are well defined. Then, the MDM algorithm discussed in Section 3 can calculate the similarity between services on each feature dimension accurately and concurrently. To reflect the similarity meeting the perspectives of different users, the whole similarity values are aggregated by users' personalized weight values. In this section, four representative dimensions are described to demonstrate the idea of multidimensional model, shown in Figure 1. The other parts of service model and the detailed discussion about the problem of existing model are presented in [29].

Figure 1a shows the dimension of observation principle, which is based on the standard definition of observation physical principle described in [30]. As a sensor is a converter of transforming nonelectrical effects into electrical signals, several steps are needed before outputting the electric signal. For example, the measurement principle of capacitive water-level sensor is dielectric-constant. This sensor is fabricated in a form of a coaxial capacitor where the surface of each conductor is coated with an isolating layer. If the water level increases, water occupies more and more space between the coaxial conductors, then transforming the capacitance. The model of this dimension is helpful to discover suitable services according to users' application scenarios. For instance, magnetic sensors are unfit for the environment with magnetic interference.

Figure 1b depicts the dimension of observation quantity type. It defines the physical quantities that be measured by the IoT services. The quantity type model is a key criterion for service matchmaking; it avoids the ambiguous representation of physical quantity. For instance, the services of body temperature and environment temperature have the similar type of output. Without an exact definition of quantity type, a body temperature service may be offered to user when he requires observing the ambient temperature. The model is constructed based on the Climate and Forecast standard of W3C CF (Climate and Forecast)-feature ontology [31], which makes a standard definitions for common observed physical quantity. For instance, it has more than 50 quantity types to express temperature, such as surface air temperature, canopy temperature, and dry-bulb temperature, etc.

Figure 1c indicates the dimension of application domain deriving from reference [30]. This dimension will help users to choose the services that are fit for their application domain. For instance, if we select a service to measure the gas concentration in coalmine domain, the service must be coalmine dedicated and "intrinsically-safe".

Figure 1d shows the measurement capability dimension. This model is derived from the capability model of W3C SSN (Semantic Sensor Network) ontology [32]. Due to the performance of IoT service may be influenced by operation environment, this model expresses the measurement capabilities of services under certain conditions, consisting the concepts of sensitivity, frequency, drift, and accuracy, etc. It can be used to check whether the service has been properly used or to determine how a service will perform in a particular environment. It is also an important criterion for service matchmaking. For instance, the capability of a temperature observation service is: with temperature -200 to $500\ ^{\circ}\text{C}$ the accuracy is $\pm 1.0\ ^{\circ}\text{C}$, while from 500 to $800\ ^{\circ}\text{C}$ it is $\pm 0.5\%$.
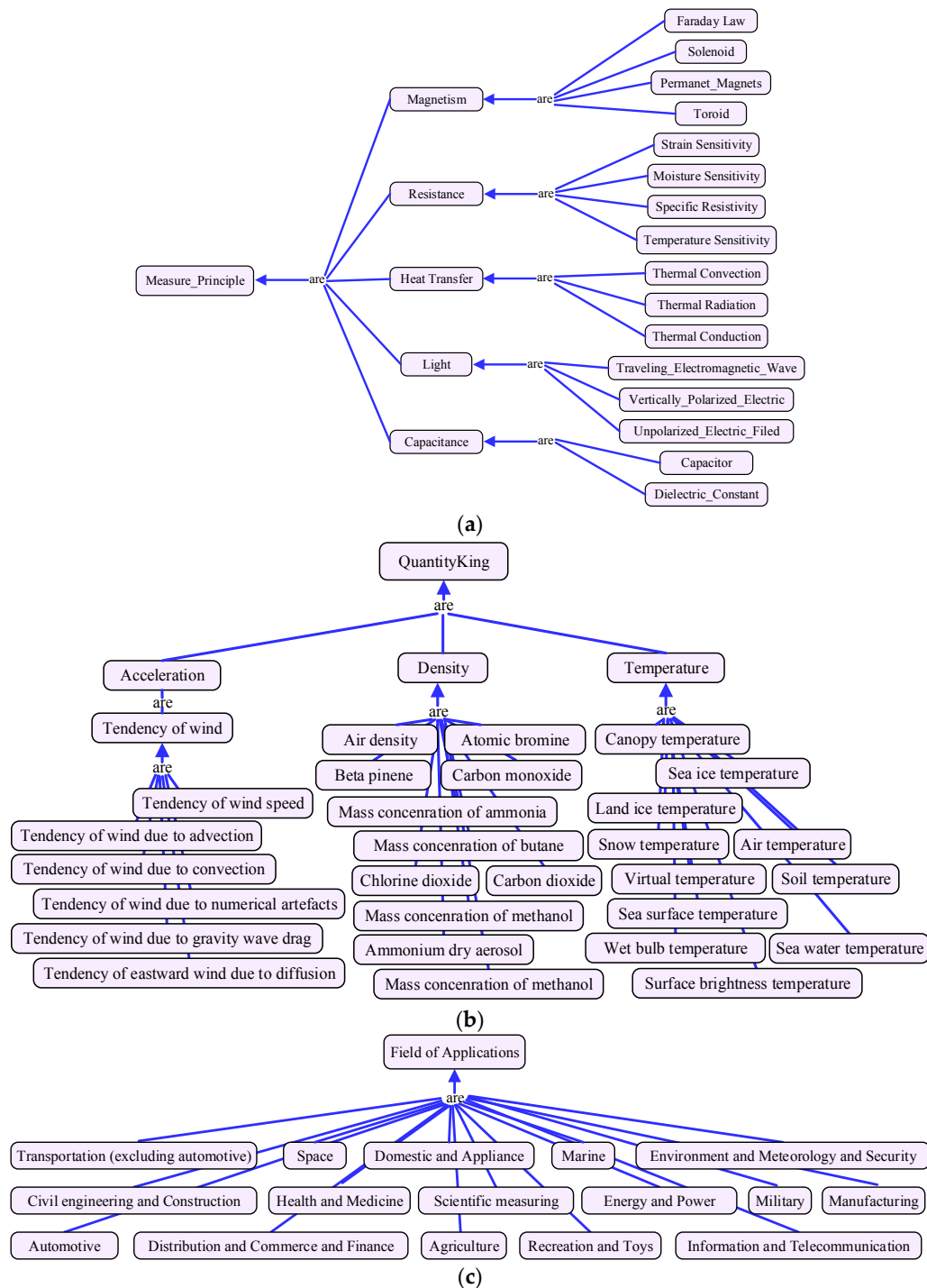

(a)

(b)

(c)

Figure 1. *Cont.*
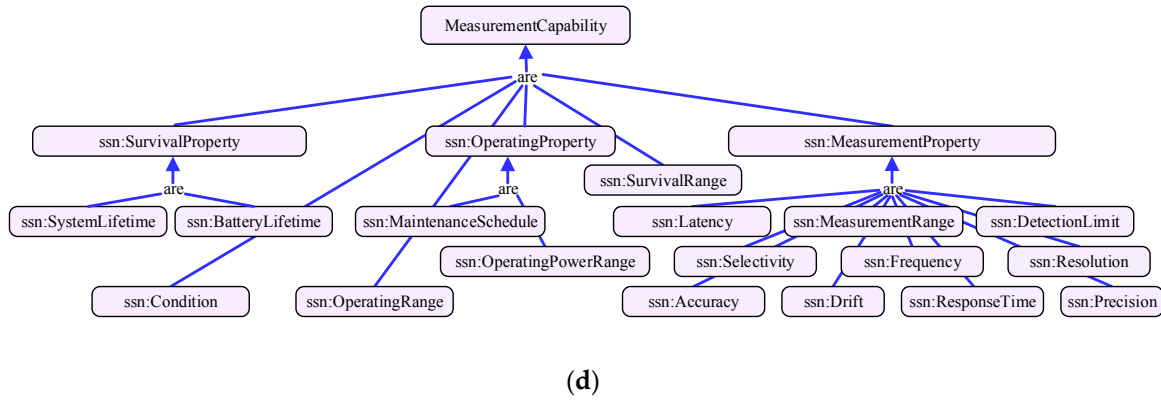
(**d**)

Figure 1. Parts of multidimensional service model. (**a**) Measurement principle dimension; (**b**) Measurement quantity type; (**c**) Application domain dimension; (**d**) Measurement capability dimension

Based on the formally multidimensional model, the semantic similarity between IoT services can be measured. Before measuring, the model of services should be vectorized, that is, transforming the model description of a service to a tuple of terms. The model conversion approach of [33] is adopted. After vectorization, the semantic concept *C* will be denoted as a tuple, as Equation (1) defines:

$$tuple = \left\{ C, \left[\delta_i, \gamma_{\delta_i} \right], \left[o_j, \gamma_{o_j} \right], \left[o_j, C_{o_j}^x \right], C_{o_j}^x, \lambda_{o_j}^y \right\} \tag{1}$$

where in OWL-annotated semantic documents, *C* is the name (or URI) of the concept *C*, each [] is a property term including a property and its restriction, δ_i ($i = 1 \ldots n$) is a datatype property of the concept *C*, γ_{δ_i} is a restriction for the datatype property δ_i, o_j ($j = 1 \ldots m$) is an object property of the concept *C*, γ_{o_j} is a restriction for the object property o_j, $C_{o_j}^x$ ($x = 1 \ldots k$) is a concept related by the object property o_j, and $\lambda_{o_j}^y$ is a Boolean operation between concepts $C_{o_j}^x$.

After the model vectorization, the semantic description of an IoT service *i* can be represented as a tuple:

$$tuple_i = \{term_1, term_2, \ldots, term_i, \ldots, term_n\}$$

Example 1. *We use a simplified model structure (shown in Figure 2) to demonstrate the process of model vectorization. In this model,* C1 *to* C5 *are classes that form the inheritance structure,* s1 *to* s4 *are service instances (i.e., objects that belong to different classes).* P1 *and* P2 *are object properties that denote the relationships between service instances. Assuming that we want to measure the similarity between* s1, *s2, and s3. Before similarity measurement, we should vectorize the model of services into tuples as following according to above discussion:*

$$tuple_{s1} = \{s1, s3, C4, P2, typeOf, [typeOf, C4], [P2, s3]\};$$

$$tuple_{s2} = \{s2, s3, C3, P2, typeOf, [typeOf, C3], [P2, s3]\};$$

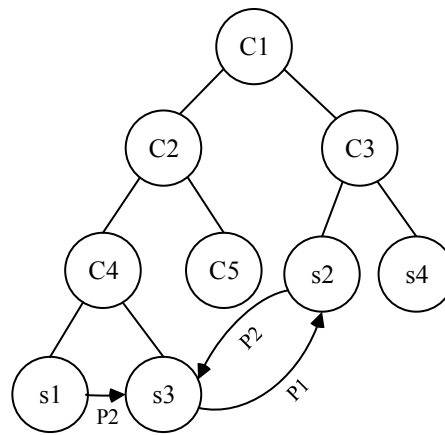$$tuple_{s3} = \{s2, s3, C4, P1, typeOf, [typeOf, C4], [P1, s2]\};$$

Figure 2. An example of service vectorization.

3. MDM Similarity Measurement

Before clustering IoT services, the similarities (or distances) between services should be measured based on MDM. MDM matches both the structure information of the model hierarchy and the description of service properties, relations and restrictions. It employs Li's approach [34] as the similarity computing method of structure information, which proposed a hybrid semantic similarity model by adopting a nonlinear model. For measuring the similarity of service description, based on the model vectorization algorithm discussed in Section 2, it adapts the *TF-IDF* (Term Frequency and Inverse Document Frequency) [35] and Cosine Similarity to calculate the similarity of service tuples. By combining the similarity of structure and description, MDM can measure the similarity of every dimension accurately and concurrently. Then, the overall similarity will be generated by aggregating the similarity of multiple dimensions according to users' preferences, for instance, allocating different weights for different dimensions.

3.1. Structure Similarity

A series of algorithms for measuring structure similarity have been proposed, considering the aspect of information content [15,16], depth in the hierarchy [36,37], semantic density [34,38], and shortest path length [39,40], etc. In order to achieve a good similarity measure, Li [34] investigated the effectiveness of a variety of strategies considering possible structure information. Its research results demonstrate that comparing the performance against human common sense is the only way to evaluate the quality of a method for calculating concept similarity. Therefore, the closer the result compares with human judgment, the better it will be. The work of [34] has confirmed the hypothesis that the human judgment of similarity is a nonlinear process. Its measurement algorithm, which models the length and depth of shortest path into a nonlinear function and combines them by multiplication, can obtain a dramatic improvement compared to previous methods. We employ their approach to calculate the structure similarity of services. Given the service a and b, the structure similarity on dimension *i* between their class *C* can be measured by Equation (2).

$$Sim_{C_i}(a,b) = \begin{cases} e^{-\alpha l} \cdot \frac{e^{\beta h}-e^{-\beta h}}{e^{\beta h}+e^{-\beta h}} \ if \ C_{ai} \neq C_{bi} \\ 1 \ if \ C_{ai} = C_{bi} \end{cases} \tag{2}$$

where h denotes the depth of the subsume Class of C_{ai} and C_{bi}, and l is the shortest path length between C_{ai} and C_{bi}. α and β are the impacts of l and h. Li [34] configures the optimal parameters that $\alpha = 0.2$ and $\beta = 0.6$. Under these parameters, the correlation coefficient between this measurement and human similarity judgments is 0.8914, while correlation between different people is 0.9015. It indicates that the measurement performs nearly at a level of human replication.

3.2. Service Description Similarity

Assuming that S is the candidate service set $S = \{s_1, s_2, \ldots, s_i, \ldots, s_m\}$, then according to Equation (1), S can be represented as:

$$S = \{tuple_1, tuple_2, \ldots, tuple_i, \ldots, tuple_m\}$$

Then we construct the feature vector of each service using the *TF-IDF*. *TF-IDF* is the product of two statistics: term frequency (*TF*) and inverse document frequency (*IDF*). The former is the frequency of a term in a document, while the latter represents the occurrence frequency of the term across all documents. It is obtained by dividing the total number of documents by the number of documents containing the term and then taking the logarithm of that quotient. The higher *TF-IDF* of a term, the more important it is for a document. In our study, corpus is the service set, document and term are tuple and description term respectively. We adopt *TF-IDF* to calculate the frequency of terms in the service tuple. The *TF* of a term in a service tuple is:

$$TF_{term} = \frac{f}{|tuple|}$$

$|tuple|$ is the size of terms of the tuple, and f is the occurrence frequency of term in this tuple. The *IDF* of the term can be measured by:

$$IDF_{term} = log\frac{|S|}{|\{tuple \in S : term \in tuple\}|}$$

The cardinality of service set S is denoted as $|S|$, and $|\{tuple \in S : term \in tuple\}|$ represents the amount of tuples that includes the *term*. Thus, the $TF - IDF$ can be calculated by:

$$TF - IDF_{term} = TF_{term} \cdot IDF_{term}$$

Then, a vector of a service by calculating the *TF-IDF* of terms in its tuple is obtained. For a service s, its tuple $tuple_s = \{term_1, term_2, \ldots, term_i, \ldots, term_k\}$ and its vector is:

$$vector_s = \{TFIDF_1, TFIDF_2, \ldots, TFIDF_i, \ldots, TFIDF_k\}$$

The similarity between two vectors can be measured by the cosine-similarity. The *IDF* not only strengthens the effect of terms whose frequencies are very low in a tuple, but also weakens the effect frequent terms. For instance, the property subClassof: Thing occurs in most ontology concepts, then the *IDF* of it is close to zero. Therefore, the terms with low *IDF* value will have weak impact on the cosine similarity measurement. The description similarity on the dimension d between two services i and j can be measured by:

$$Sim_{P_d}(i, j) = \frac{vector_i \cdot vector_j}{||vector_i|| \cdot ||vector_j||} \tag{3}$$

3.3. Multidimensional Aggregation

The similarity in the i dimension between two services a and b can be calculated by combining sim_C (Equation (2)) and sim_P (Equation (3)). δ is the impact parameter which indicates the effect of structure information on the similarity measurement.

$$Sim_i(a, b) = \delta \cdot sim_{C_i}(a, b) + (1 - \delta) \cdot sim_{P_i}(a, b) \tag{4}$$

The similarity values of each dimension can be aggregated by weights according to the users' preferences:

$$Sim(a,b) = \sum_{i=1}^{n} w_i \cdot sim_i(a,b), \quad \sum_{i=1}^{n} w_i = 1 \tag{5}$$

where n is the dimension number of semantic service model.

4. IoT Service Clustering

This paper employs density-peaks-based clustering [20] to divide services into clusters according to the potential density distribution of similarity between services. Density-peaks-based clustering is a fast and accurate clustering approach for large-scale data. After clustering, the similar services are generated automatically without the artificial determining of parameter. The distance between two services can be calculated by Equation (6):

$$Dist(a,b) = 1 - Sim(a,\text{b}) \tag{6}$$

4.1. Local Density and Distance Calculating

The density-peaks algorithm is based on the assumptions that cluster centers are surrounded by neighbors with lower local density, and they are keep a large distance from other points with higher density. Assuming that $S = \{s_i\}_{i=1}^{N}$ is the service set that will be clustered, s_i is a service of S, $I_S = \{1, 2, \ldots, N\}$ is the set of index. For each service s_i in S, two quantities are defined: its local density ρ_s and its distance θ_s from services of higher density. The local density ρ of service i is defined as:

$$\rho_i = \sum_j \chi(d_{ij} - d_c) \tag{7}$$

where d_c is a cutoff distance. If $x < 0$, $\chi(x)$ equals to 1, otherwise $\chi(x) = 0$. θ is calculated by measuring the closest distance between the service i and other services with higher density than i:

$$\theta_i = \min_{j:\rho_j > \rho_i} d_{ij} \tag{8}$$

For the service with highest density, its density is defined as: $\theta_i = max_j(d_{ij})$. Note that θ_i is much larger than the typical nearest neighbor distance only for services that are local or global maxima in the density. Algorithm 1 describes the procedure of calculating clustering distance. Firstly, the data density are sorted in descending order, set $\{q_i\}_{i=1}^{N}$ is the index generated from the descending order, i is the index of descending order and q_i is the original index. Then, the clustering distance θ_i of service s_i is calculated by $\theta_i = Dist(i, j)$, s_j is the service that has larger density (than s_i) and closest to s_i. In S, we use n_i to denote the index of s_j, namely, $n_i = j$. $\{n_i\}_{i=1}^{N}$ is defined as:

$$n_{q_i} = \begin{cases} \arg\min_{\substack{q_j \\ j < i}} \left\{ d_{q_i q_j} \right\}, & i \geq 2; \\ 0, & i = 1. \end{cases} \tag{9}$$

The clustering distance of the point with largest density is defined as $max_j(d_{ij})$ that is the maximum value of all data points, and the index is $n_{q_1} = 0$.

Algorithm 1. Calculating Clustering Distance

Input:	F: the matrix of distances between services;
	ρ: local density of each service;
Output:	θ: the clustering distance of each service;
	n_{q_i}: the index of service that has larger density and closest to s_{q_i};

Sort in descending order by density ρ

1: $\{\rho_i\}_{i=1}^N$

2: $q_i \leftarrow$ descending order index of density;

Distance assignment of θ

3: $\{n_i\}_{i=1}^N \leftarrow 0$;

4: **for** i:=1 to N **do**

5: 　$\theta_{qi} \leftarrow d_{max}$;

6: 　　**for** j:=1 to i−1 **do**

7: 　　　**if** $dist(s_{q_i}, s_{q_j}) < \theta_{qi}$ **then**

8: 　　　　$\theta_{qi} = dist(s_{q_i}, s_{q_j})$;

9: 　　　　$n_{q_i} = qj$;

10: 　　　**end if**

11: 　　**end for**

12: 　**end for**

13: $\theta_{q_1} = \max\limits_{j \geq 2} \theta_j$;

4.2. Cluster Center Selecting

For services $\{x_i\}_{i=1}^N$ in S, their local density and clustering distance can be calculated: $\{(\rho_i, \theta_i)\}_{i=1}^N$. Cluster centers are the services that have both large ρ and large θ. In order to eliminate the difference of magnitude, the ρ and θ of each service are normalized to [0,1]. Then, the values that are comprehensive consideration of ρ and θ are calculated:

$$\gamma_i = \rho_i \theta_i, \; i \in I_S \tag{10}$$

Obviously, the higher value of γ, the more likely it becomes a cluster center. $\{\gamma_i\}_{i=1}^N$ are sorted in descending order. The sorted γs are drawn on the coordinate plane, the horizontal axis is the index of γ, the vertical axis is the value of γ, as shown in Figure 3. This coordinate plane is defined as decision graph. In addition, then a number of service points are intercepted from front to back as the cluster centers. The decision graph shows that the γ values of cluster centers are larger and discrete, while non-center services are continuous and smooth. The transition of γ value from the cluster centers to the non-center services has a significant "jump", this "jump" can be detected by numerical detection method [41]. Therefore, the cluster center of the dataset S will be determined according to decision graph and numerical detection method.
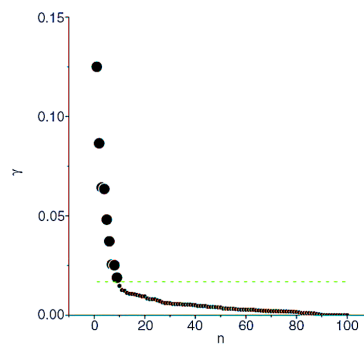

Figure 3. Decision graph for assuming cluster centers. $\gamma_i = \rho_i \theta_i$ is the combination of local density ρ_i and clustering distance θ_i of service i. n is the index of services after they are sorted in descending order by γ.

4.3. Cluster Assignment

After the center of every cluster is assumed, the next step is to assign non-center services to clusters. Algorithm 2 describes the procedure of cluster assignment. Each service are assigned in the order of density descending, which is from the cluster center services to the cluster core services to the cluster halo services in the way of layer by layer.

Suppose that n_c is the total number of cluster centers, naturally, the number of clusters is also n_c. $\{m_j\}_{j=1}^{n_c}$ is the index of corresponding service for each cluster center, i.e., service s_{m_j} is the center of the jth cluster. $\{c_i\}_{i=1}^N$ is the cluster of each service belongs to, i.e., service s_i belongs to cluster c_i. According to the definition of $\{n_i\}_{i=1}^N$ in Equation (9), n_i is the index of service which has larger density than ith service (s_i) and closest to s_i.

Algorithm 2. Cluster Assignment

Input:	q_i: the descending order of index according to density ρ;
	$\left\{m_j\right\}_{j=1}^{n_c}$: the index of cluster center of cluster j;
	n_c: total number of clusters (centers);
	n_{q_i}: the index of service which has larger density than s_{q_i} and closest to s_{q_i};
Output:	$\{c_i\}_{i=1}^N$: the cluster of each service belongs to, i.e., s_i belongs to c_i;

1: $\{c_i\}_{i=1}^N \leftarrow -1$; //initialization of c_i
2: **for** j:=1 to n_c **do**
3: c_{m_j}=j;   //cluster centers
4: **end for**
Non-center services assignment
5: **for** i:=1 to N **do** //descending order of index
6:       **if** $c_{q_i} = -1$ **then**
7:             $c_{q_i} = c_{n_{q_i}}$;
8:       **end if**
9: **end for**

If the dataset has more than one cluster, each cluster can be furthermore divided into two parts: cluster core and cluster halo. The cluster core with higher density is the core part of a cluster. The cluster halo with lower density is the edge part of a cluster. The procedure of determining cluster core and cluster halo is described in Algorithm 3. We define the border region of a cluster as: the border region of cluster c_1 is consisted by the services s_i that belongs to c_1, and the distance between s_i and s_j (which belongs to another cluster c_2) is less than d_c. An average density bound is defined as $\left\{\rho_{c_i}^b\right\}_{i=1}^{n_c}$, $\rho_{c_i}^b$ is the average density bound of cluster c_i. If the density ρ of service s is larger than $\rho_{c_i}^b$, then service s belongs to the core part of cluster c_i; otherwise, it belongs to the halo part of cluster c_i.

After clustering, the similar service neighbors are generated automatically without the estimation of parameters. Moreover, different services have personalized neighbor sizes according to the actual density distribution, which may avoid the inaccurate matchmaking caused by constant neighbor size.

Algorithm 3. Determining Cluster Core and Cluster Halo

Input:	F: the matrix of distances between services;
	d_c: cut-off distance;
	$\{c_i\}_{i=1}^{n_c}$: the cluster of each service belongs;
Output:	$\{h_i\}_{i=1}^{N}$: the signal of core or halo that service s_i belongs to;

Initialization

1: $\{h_i\}_{i=1}^{N} \leftarrow 0$;

2: $\left\{\rho_{c_i}^{b}\right\}_{i=1}^{n_c} \leftarrow 0$;

3: **for** i:=1 to N−1 **do**

4: **for** j:=I + 1 to N **do**

5:　　**if**　$c_i \neq c_j$ and $dist\left(s_i, s_j\right) < d_c$ **then**

6:　　　　$\bar{\rho}_{ij} = \frac{1}{2}\left(\rho_i + \rho_j\right)$;

7:　　　　**if** $\bar{\rho}_{ij} > \rho_{c_i}^{b}$ **then**

8:　　　　$\rho_{c_i}^{b} = \bar{\rho}_{ij}$;

9:　　**end if**

10:　　**if** $\bar{\rho}_{ij} > \rho_{c_j}^{b}$ **then**

11:　　　　$\rho_{c_j}^{b} = \bar{\rho}_{ij}$;

12:　　**end if**

13:　**end if**

14:　**end for**

15: **end for**

16: **for** i:=1 to N

17: **if** $\rho_i < \rho_{c_i}^{b}$ **then**

18:　　　$h_i = 1$;　　　//belongs to halo part of cluster c_i

19:　　**end if**

20: **end for**

---

5. Experimental Evaluation

　　In this section, we evaluate the performance of proposed MDM measurement and service clustering. We use a combined data set including real and synthetic data, which collects service from multiple sources and adds essential service instances and descriptions. The data sources of combined service set are shown in Table 1. In this paper, 510 real sensor services are collected from 6 sensor sets, including indoor and outdoor sensors. Then, the amount of service is expanded to 1000, and essential semantic service descriptions are supplemented for similarity measuring. The experimental evaluation is performed under the environment of 64-bit Windows 7 Professional, Java 7, Intel Xeon Processor E5-2650 2.3GHz processor, and 32 GB RAM. Section 5.1 discusses about the performance of MDM, and Section 5.2 discusses about the performance of service clustering.

Table 1. Service set.

	Source	Number of Services	Number after Expansion
	ABM	105	200
	LSM	100	200
Outdoor	CCMWS	93	200
	DHCIS	76	200
Indoor	IntelLab	54	100
	MavHome	82	100
Total		510	1000

5.1. Performance of Similarity Measurement

To evaluate the performance of similarity measurement, we employ the most widely used performance metrics from the information retrieval field. The performance metrics in this experiment are defined as follows:

Precision. Precision is used to measure the preciseness of a search system. Precision for a single service refers to the proportion of matched and logically similar services in all services matched to this service, which can be represented by the following equation:

$$Precision = \frac{|A \cap B|}{|B|}$$

where A is the number of logically similar service and B is the number of matched services calculated by MDM.

Recall. Recall is used to measure the effectiveness of a search system. Recall for a single service is the proportion of matched and logically similar services in all services that are logically similar to this service, which can be represented by the following equation:

$$Recall = \frac{|A \cap B|}{|A|}$$

F-measure. F-measure is employed as an aggregated performance scale for a search system. In this experiment, F-measure is the mean of precision and recall, which can be represented as:

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

When the F-measure value reaches the highest level, it means that the aggregated value between precision and recall reaches the highest level at the same time.

In order to filter out the dissimilar services with lower similarity values, an optimal threshold value is needed to be estimated. In addition, the aggregative metric of F-measure is used as the primary benchmark for estimating the optimal threshold value. Besides, parameter δ is the impact of description and structure similarity for similarity measuring. To obtain the best performance, an optimal δ value should also be estimated. The initial values of two parameters are set to 0, and increasing incrementally by 0.1 until 1.0.

Figures 4 and 5 demonstrate the variation of F-measure values of dimension-mixed and multidimensional model as the changing of these two parameters. When the value of F-measure reaches the highest point, it achieves the best performance, and the optimal value of threshold and δ will be determined. As Figures 4 and 5 indicates, δ = 0.5 and threshold = 0.8 are the optimal values of dimension-mixed model, and the F-measure is 40 with these parameters; meanwhile δ = 0.7 and threshold = 0.7 are the optimal values of multidimensional model, and the F-measure is 63 with these parameters. Besides, the overall F-measure values of multidimensional model are higher than dimension-mixed model.

The performance comparison between multidimensional and dimension-mixed model is shown in Figure 6. As the results indicate, the performance of similarity measurement based on the multidimensional model outperforms to the dimension-mixed way. The reason is that, employing the multidimensional model, both description similarity and structure similarity can be measured accurately. For the structure similarity, each dimension has a well-defined semantic structure in which the distance and positional relationships between nodes are meaningful to reflect the similarity between services. For the description similarity, each dimension only focuses on the descriptions that are contributed to expressing the features of current dimension. Conversely, using the dimension-mixed way, which mixes the semantic structures and descriptions of all dimensions into a complicated model, the measurement can only obtain an overall similarity value.
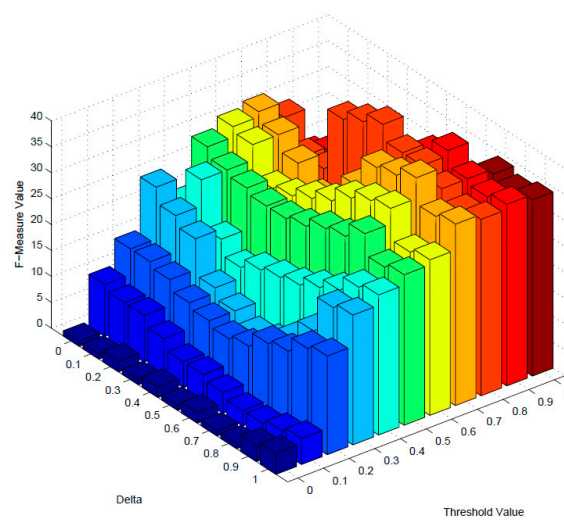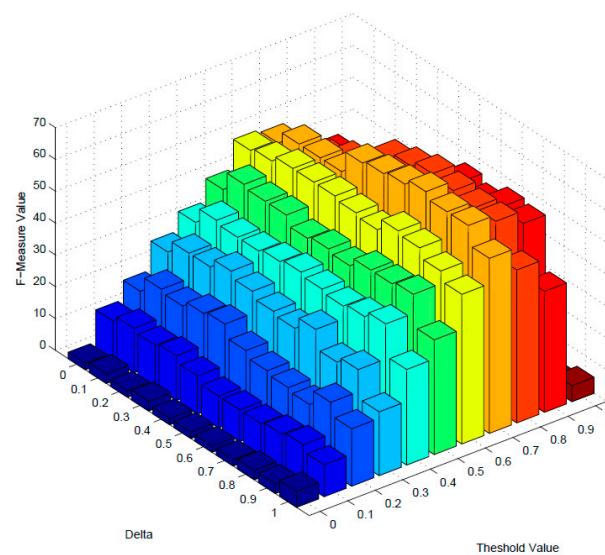
Figure 4. The F-measure of dimension-mixed.

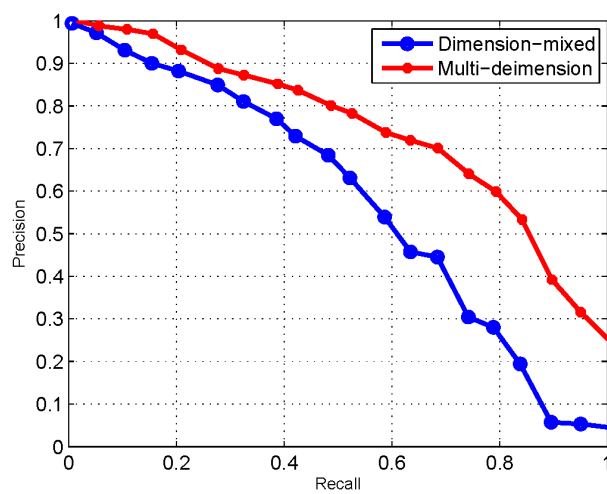**Figure 5.** The F-measure of multi-dimension.

Figure 6. The performance of dimension-mixed and multidimensional.

5.2. Performance of Service Clustering

In this section, we evaluate the performance of clustering. The number of service that will be clustered is 1000 with essential semantic description and structure, as Table 1 describes. The cut-off distance d_c for calculating local density of services is set to 0.03. As Figure 7 shows, although the service set is high overlap in data distribution, the proposed approach successfully detects the cluster structure. The services are clustered into 5 clusters, the borders of clusters are clear, and each cluster is dense and compact.
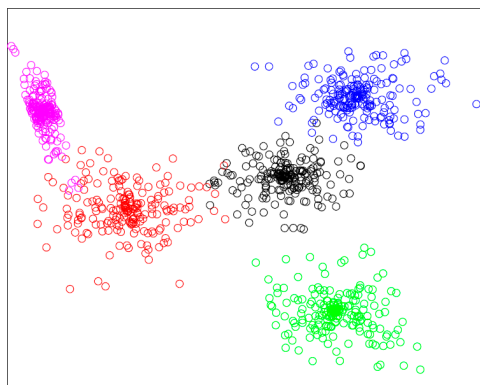
Figure 7. The result of clustering.

The size of service set and the number of feature dimensions are two important factors to evaluate the efficiency of proposed clustering approach. Figure 8 shows the time of clustering as the size of services is increased from 100 to 1000. The time of clustering 1000 services is 3.2 s. The results show that the clustering time is linear with respect to the number of IoT services to be clustered, and the clustering time of hundreds services is controlled within a few seconds. Figure 9 shows the time of clustering as the dimensions of service model increasing from 1 to 10. The number of services that will be clustered is set to 1000. The minimum clustering time is 2.8 s, when there are four feature dimensions of the model; and the maximum clustering time is 3.3 s, when the number of feature dimensions is eight. The results show that the clustering time will not increase as the increase of dimension number. It is because that MDM measures each dimension's similarity concurrently. Thus, the whole time of measuring similarity of all dimensions is equal to the time of single dimension that takes longer time than other dimensions. Besides, the clustering is based on the measurement result of MDM (distances between services), it will not be influenced by the dimension number. Therefore, the proposed approaches improve the accuracy of similarity measurement and service clustering in the condition of not increasing the computation time.
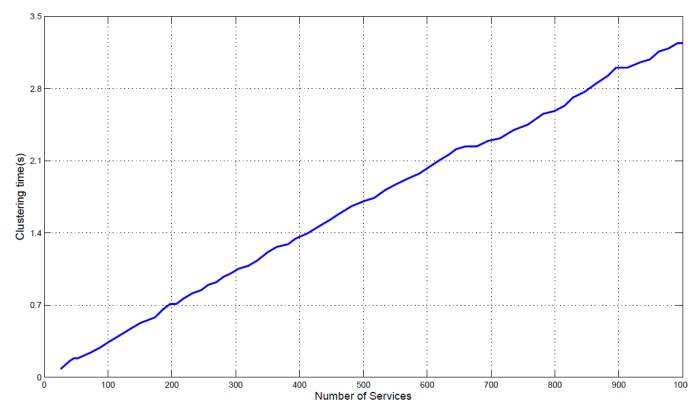

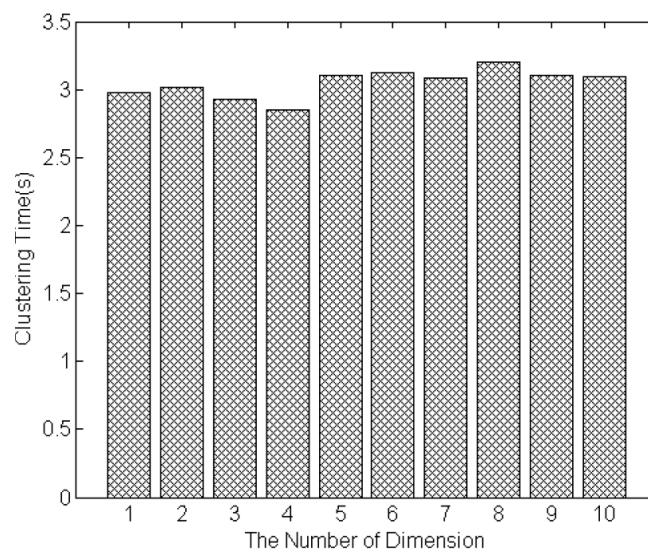**Figure 8.** The time of clustering influenced by services number.

Figure 9. The time of clustering influenced by dimension number.

The experimental results demonstrate that, the proposed clustering approach is able to cluster hundreds of IoT services in a reasonable amount of time. In the application domains of IoT SOC paradigm, the number of services usually does not exceed several thousands. Besides, if the scale of services is very large, the service clustering can be performed offline. Thus, the performance of proposed clustering approach is competent for applying in real application scenarios.

6. Conclusions

This paper proposes a multidimensional model-based approach to measure the similarity between IoT services. Then, density-peaks-based clustering is employed to gather similar services together according to the result of similarity measurement. A combined data set is used to evaluate the proposed approaches, which collects service from multiple sources and adds essential service instances and descriptions. The experiment results demonstrate that the performance of proposed approaches are promising and applicable to real-life scenarios.

Currently, the experiments are conducted using a centralized single dataset, and the size of test set is limited. Our future works include extending the experiments using distributed datasets and expanding the number of service set. Moreover, we plan to propose a quantitative model to diagnose the quality of service clustering, then to determine when the clustering structure becomes unacceptable and require re-clustering as the evolution of services.

Author Contributions: Shuai Zhao and Le Yu conceived and designed the experiments; Shuai Zhao performed the experiments; Le Yu and Bo Cheng analyzed the data; Bo Cheng and Junliang Chen contributed analysis tools; Shuai Zhao, Le Yu, Bo Cheng, and Junliang Chen wrote the paper.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Papazoglou, M.P.; Heuvel, W.-J. Service oriented architectures: Approaches, technologies and research issues. *VLDB J. Int. J. Very Larg. Data Bases* **2007**, *16*, 389–415. [CrossRef]
2. Du, Y.; Li, X.; Xiong, P. A petri net approach to mediation-aided composition of web services. *IEEE Trans. Autom. Sci. Eng.* **2012**, *9*, 429–435. [CrossRef]

3. Kyusakov, R.; Eliasson, J.; Delsing, J.; van Deventer, J.; G ustafsson, J. Integration of wireless sensor and actuator nodes with IT infrastructure using service-oriented architecture. *IEEE Trans. Ind. Inform.* **2013**, *9*, 43–51. [CrossRef]

4. Soldatos, J.; Kefalakis, N.; Hauswirth, M.; Serrano, M.; Calbimonte, J.-P.; Riahi, M.; Aberer, K.; Jayaraman, P.P.; Zaslavsky, A.; Žarko, I.P. Openiot: Open source internet-of-things in the cloud. In *Interoperability and Open-Source Solutions for the Internet of Things*; Springer: Berlin, Germany, 2015; pp. 13–25.

5. Butun, I.; Morgera, S.D.; Sankar, R. A survey of intrusion detection systems in wireless sensor networks. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 266–282. [CrossRef]

6. Sinha, N.; Pujitha, K.E.; Alex, J.S.R. Xively based sensing and monitoring system for IoT. In Proceedings of the 2015 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 8–10 January 2015; pp. 1–6.

7. Dong, X.; Halevy, A.; Madhavan, J.; Nemes, E.; Zhang, J. Similarity search for web services. In Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, ON, Canada, 31 August–3 September 2004; pp. 372–383.

8. Plebani, P.; Pernici, B. URBE: Web service retrieval based on similarity evaluation. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1629–1642. [CrossRef]

9. Liu, X.; Huang, G.; Mei, H. Discovering homogeneous web service community in the user-centric web environment. *IEEE Trans. Serv. Comput.* **2009**, *2*, 167–181. [CrossRef]

10. Cheng, B.; Zhu, D.; Zhao, S.; Chen, J. Situation-aware IoT service coordination using the event-driven SOA paradigm. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 349–361. [CrossRef]

11. Zhang, L.-J.; Cheng, S.; Chang, C.K.; Zhou, Q. A pattern-recognition-based algorithm and case study for clustering and selecting business services. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2012**, *42*, 102–114. [CrossRef]

12. Yachir, A.; Amirat, Y.; Chibani, A.; Badache, N. Event-aware framework for dynamic services discovery and selection in the context of ambient intelligence and Internet of Things. *IEEE Trans. Autom. Sci. Eng.* **2016**, *13*, 85–102. [CrossRef]

13. Zhou, Z.; Sellami, M.; Gaaloul, W.; Barhamgi, M.; Defude, B. Data providing services clustering and management for facilitating service discovery and replacement. *IEEE Trans. Autom. Sci. Eng.* **2013**, *10*, 1131–1146. [CrossRef]

14. Pilioura, T.; Tsalgatidou, A. Unified publication and discovery of semantic web services. *ACM Trans. Web* **2009**, *3*, 11. [CrossRef]

15. Resnik, P. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *J. Artif. Intell. Res.* **1999**, *11*, 95–130.

16. Sánchez, D.; Batet, M.; Isern, D. Ontology-based information content computation. *Knowl. Based Syst.* **2011**, *24*, 297–303. [CrossRef]

17. Sussna, M. Word sense disambiguation for free-text indexing using a massive semantic network. In Proceedings of the 2nd International Conference on Information and Knowledge Management, Washington, DC, USA, 3–5 November 1993; pp. 67–74.

18. Pedersen, T.; Patwardhan, S.; Michelizzi, J. WordNet:: Similarity: Measuring the relatedness of concepts. In Proceedings of the HLT-NAACL 2004, Boston, MA, USA, 2–7 May 2004; pp. 38–41.

19. Klusch, M.; Fries, B.; Sycara, K. OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services. *Web Semant. Sci. Serv. Agents World Wide Web* **2009**, *7*, 121–133. [CrossRef]

20. Rodriguez, A.; Laio, A. Clustering by fast search and find of density peaks. *Science* **2014**, *344*, 1492–1496. [CrossRef] [PubMed]

21. Compton, M.; Neuhaus, H.; Taylor, K.; Tran, K.-N. Reasoning about sensors and compositions. In Proceedings of the 2nd International Conference on Semantic Sensor Networks, Washington, DC, USA, 26 October 2009; pp. 33–48.

22. Goodwin, J.C.; Russomanno, D.J.; Qualls, J. Survey of Semantic Extensions to UDDI: Implications for Sensor Services. In Proceedings of the International Conference on Sematic Web and Web Services, Las Vegas, NV, USA, 28 January 2007; pp. 16–22.

23. Perera, C.; Zaslavsky, A.; Christen, P.; Georgakopoulos, D. Sensing as a service model for smart cities supported by internet of things. *Trans. Emerg. Telecommun. Technol.* **2014**, *25*, 81–93. [CrossRef]

24. Rodríguez-García, M.Á.; Valencia-García, R.; García-Sánchez, F.; Samper-Zapater, J.J. Ontology-based annotation and retrieval of services in the cloud. *Knowl. Based Syst.* **2014**, *56*, 15–25. [CrossRef]

25. Feng, G.; Chen, H.; Liu, M. An ontology service model for flexible service customization. In Proceedings of the 2014 11th International Conference on Service Systems and Service Management (ICSSSM), Beijing, China, 25–27 June 2014; pp. 1–4.

26. Jin, X.; Chun, S.; Jung, J.; Lee, K.-H. IoT service selection based on physical service model and absolute dominance relationship. In Proceedings of the 2014 IEEE 7th International Conference on Service-Oriented Computing and Applications (SOCA), Matsue, Japan, 17–19 November 2014; pp. 65–72.

27. Kim, J.; Chung, K.-Y. Ontology-based healthcare context information model to implement ubiquitous environment. *Multimed. Tools Appl.* **2014**, *71*, 873–888. [CrossRef]

28. Klusch, M.; Kapahnke, P.; Schulte, S.; Lecue, F.; Bernstein, A. Semantic web service search: A brief survey. *KI-Künstliche Intell.* **2016**, *30*, 139–147. [CrossRef]

29. Zhao, S.; Zhang, Y.; Yu, L.; Cheng, B.; Ji, Y.; Chen, J. A multidimensional resource model for dynamic resource matching in internet of things. *Concurr. Comput. Pract. Exp.* **2015**, *27*, 1819–1843. [CrossRef]

30. Fraden, J. *Handbook of Modern Sensors: Physics, Designs, and Applications*; Springer Science & Business Media: Berlin, Germany, 2004.

31. Calbimonte, J.-P.; Yan, Z.; Jeung, H.; Corcho, O.; Aberer, K. Deriving semantic sensor metadata from raw measurements. In Proceedings of the 5th International Conference on Semantic Sensor Networks at ISWC, at ISWC, Boston, MA, USA, 12 November 2012; pp. 33–48.

32. Compton, M.; Barnaghi, P.; Bermudez, L.; GarcíA-Castro, R.; Corcho, O.; Cox, S.; Graybeal, J.; Hauswirth, M.; Henson, C.; Herzog, A. The SSN ontology of the W3C semantic sensor network incubator group. *Web Semant. Sci. Serv. Agents World Wide Web* **2012**, *17*, 25–32. [CrossRef]

33. Dong, H.; Hussain, F.K.; Chang, E. A context-aware semantic similarity model for ontology environments. *Concurr. Comput. Pract. Exp.* **2011**, *23*, 505–524. [CrossRef]

34. Li, Y.; Bandar, Z.A.; McLean, D. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Trans. Knowl. Data Eng.* **2003**, *15*, 871–882.

35. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent dirichlet allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.

36. Ikemura, S.; Fujiyoshi, H. Real-Time Human Detection Using Relational Depth Similarity Features. In Proceedings of the 10th Asian Conference on Computer Vision, Queenstown, New Zealand, 8–12 November 2010; pp. 25–38.

37. Rodríguez, M.A.; Egenhofer, M.J. Determining semantic similarity among entity classes from different ontologies. *IEEE Trans. Knowl. Data Eng.* **2003**, *15*, 442–456. [CrossRef]

38. Ahsaee, M.G.; Naghibzadeh, M.; Naeini, S.E.Y. Semantic similarity assessment of words using weighted WordNet. *Int. J. Mach. Learn. Cybern.* **2014**, *5*, 479–490. [CrossRef]

39. Varelas, G.; Voutsakis, E.; Raftopoulou, P.; Petrakis, E.G.; Milios, E.E. Semantic similarity methods in wordNet and their application to information retrieval on the web. In Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management, Bremen, Germany, 4 November 2005; pp. 10–16.

40. Bulskov, H.; Knappe, R.; Andreasen, T. *On Measuring Similarity for Conceptual Querying*; Springer: Beilin, Germany, 2002; pp. 100–111.

41. Stoer, J.; Bulirsch, R. *Introduction to Numerical Analysis*; Springer Science & Business Media: Berlin, Germany, 2013.