



## Article

# A Personalized Ontology Recommendation System to Effectively Support Ontology Development by Reuse

Marwa Abdelreheim <sup>1,\*</sup>, Taysir Hassan A. Soliman <sup>1</sup> and Friederike Klan <sup>2,\*</sup>

<sup>1</sup> Department of Information Systems, Faculty of Computers and Information, Assiut University, Assiut 71515, Egypt; taysirhs@aun.edu.eg

<sup>2</sup> German Aerospace Center, Institute of Data Science, Maelzerstraße 3-5, 07743 Jena, Germany

\* Correspondence: marwa.abdelrehem@aun.edu.eg (M.A.); friederike.klan@dlr.de (F.K.)

**Abstract:** The profusion of existing ontologies in different domains has made reusing ontologies a best practice when developing new ontologies. The ontology reuse process reduces the expensive cost of developing a new ontology, in terms of time and effort, and supports semantic interoperability. Existing ontology development tools do not assist in the recommendation of ontologies or their concepts to be reused. Also, existing ontology recommendation tools could suggest whole ontologies covering a set of input keywords without referring to which parts of them (e.g., concepts) can be reused. In this paper, we propose an effective ontology recommendation system that helps the user in the iterative development and reuse of ontologies. The system allows the user to provide explicit preferences about the new ontology, and iteratively guides the user to parts from existing ontologies which match his preferences for reuse. Finally, we developed a prototype of our ontology recommendation system and conducted a user-based evaluation to assess the effectiveness of our approach.

**Keywords:** ontology recommendation; ontology reuse; biomedical ontologies; recommender systems



**Citation:** Abdelreheim, M.; Soliman, T.H.A.; Klan, F. A Personalized Ontology Recommendation System to Effectively Support Ontology Development by Reuse. *Future Internet* **2023**, *15*, 331. <https://doi.org/10.3390/fi15100331>

Academic Editor: Petros Patias

Received: 1 September 2023

Revised: 22 September 2023

Accepted: 3 October 2023

Published: 7 October 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Ontologies are formal representations of concepts and their relations, used to represent, share, and reuse the knowledge of a specific domain. Ontology development is usually a complex and time-consuming process that needs extensive and collaborative efforts between ontology engineers and domain experts. It is usually a long and iterative process that needs repeated revisions until a final accepted version is reached [1]. Ontology reuse is defined as the process of assembling, extending, and integrating other ontologies as parts of a new resulting ontology which represents a given domain [2]. As reusability is a main characteristic of ontologies, it is a common practice to reuse existing ontologies when developing new ontologies to save time and effort and develop high-quality ontologies in a reasonable amount of time.

Existing ontology development tools (e.g., Protégé [3]) can assist the user in developing a new ontology and enable him to add new concepts, relations, and individuals. However, they might not be helpful when the user needs to develop a new ontology by reusing parts from existing ontologies to extend an already existing one, or part of it. This ontology reuse process typically involves two main steps: (1) identifying ontologies that potentially contain concepts for reuse among a large set of existing ontologies, and (2) identifying appropriate concepts to reuse within these ontologies (i.e., often comprising thousands of concepts). Existing ontology development tools do not support these mentally demanding tasks.

According to [4,5], there are currently no tools to support the ontology development process through the reuse of parts from existing ontologies. However, some tools can do each step separately, including ontology selection and ontology matching tools. Ontology selection tools (e.g., BioSS [6] and NCBO Ontology Recommender [7]) accept a set of

keywords from the user and select appropriate ontologies using pre-defined evaluation criteria. These tools have limited functionality which hinders their use in the ontology reuse process. First, they are not helpful if the user has preferences/requirements about the ontologies he wants to reuse, other than the pre-defined evaluation criteria, such as the ontology domain or ontology type. For example, if the user wants to develop a new ontology representing a certain domain (i.e., Botany), then he wants the system to recommend ontologies related to the “Botany” domain, likewise if he wants to select a specific type of ontology (e.g., taxonomies). Second, they give one-shot recommendations and do not support the iterative nature of the ontology development process. Finally, they recommend whole ontologies but cannot guide the users to the ontology components (i.e., concepts) to be reused. Also, ontology matching tools (e.g., LYAM++ [8] and AML [9]) use lexical, structural, and semantic matching approaches to find correspondences between concepts in a source and a target ontology. They successfully determine alignments between concepts that are richly described (i.e., have annotations such as synonyms, definitions, comments, etc.), but they have limited performance if one of the input ontologies is a basic one with few concepts and no rich annotations describing these concepts. This is the case for most ontologies (e.g., bio-ontologies), where scientists are interested in defining a hierarchy of concepts describing a domain without adding many annotations describing each concept in detail. This causes ontology matching tools, in some cases, to identify corresponding concepts that have identical labels, but this does not necessarily mean that those concepts are semantically equivalent (i.e., sharing the same word sense). In this case, we need to learn more about the users’ interests as an additional source of knowledge to conduct proper ontology matching.

In this paper, we propose a novel personalized ontology recommendation and reuse system which can effectively support an ontology engineer in the task of developing a new ontology by suggesting parts from existing ontologies to extend an input ontology. We developed a utility-based recommender system that explicitly collects user preferences and recommends appropriate ontologies, along with their matching concepts. Then, the user can select appropriate components to extend his input ontology. The proposed recommender system uses a technique based on reinforcement learning to learn the user’s explicit preferences and implicit feedback to adaptively offer personalized relevant recommendations according to the user’s preferences.

We present three main contributions. First, we propose a utility-based ontology recommender system that (a) helps the user to iteratively develop a new ontology using an input ontology, (b) recommends ontologies along with their appropriate concepts to the user so that the user can reuse them and extend the input ontology, (c) creates a user profile for each user to recommend and rank ontologies and their appropriate classes based on the user’s preferences, and (d) uses the user’s feedback to update his user profile to cope with any change in his preferences throughout the interaction session. Second, we develop a prototype and use the biomedical domain as a case study, as it has hundreds of existing ontologies in different repositories. Third, we evaluate the system using real users to obtain a realistic evaluation of the system.

The rest of the paper is structured as follows. Section 2 reviews and compares the features of existing ontology recommendation tools. Next, Section 3 illustrates the proposed interactive ontology recommender framework, and Section 4 describes the utility-based ontology recommendation system. Section 5 provides the implementation details of the proposed framework. Section 6 describes the user-based evaluation of the proposed framework and discusses the outcoming results. Finally, Section 7 concludes the paper with possible suggestions for future work.

## 2. Related Work

Ontology reuse is one of the best practices when developing a new ontology, as it promotes using content that has been modeled before and reusing it to represent a new domain. These contents could be whole ontologies (hard ontology reuse) or a selected subset (soft

ontology reuse) [10]. MIREOT [11] specifies the minimal information needed to specify the term for reuse, i.e., source ontology URI, source term URI, and target direct superclass URI. Ontofox [12] follows the full MIREOT guidelines with the advantage of allowing the user to extract and reuse specific terms from an ontology, which promotes modularization. There have also been attempts that made use of natural language processing techniques to extract terms and relations from text documents representing a target domain, and to reuse them to develop a new ontology modeling the target domain [13]. The problem with these approaches is that they require the user to have good knowledge of all of the existing ontologies and their concepts, which is very difficult, especially for a person who is not an expert in the target domain. Ontology development tools (e.g., Protégé [14] and NeOn toolkit [15]) support hard ontology reuse (owl: imports), allowing the user to import and reuse whole ontologies [10], which is not preferable for ontology experts as they import concepts that may be irrelevant and incompatible with the requirements of the target ontology [10]. However, there are some plugins that support soft ontology reuse. MIREOT Protégé plugin [16] facilitates MIREOT in practice and implements crucial variables of specification. ProtégéLOV [17] allows the user to access Linked Open Vocabularies (LOV) and reuse terms from it during the development of an ontology. Watson [18] is a NeOn plugin and is mainly a semantic web search engine which searches and locates the content of ontologies on the Web. XOD [19] is an “eXtensible Ontology Development” strategy which recommends principles to support the active development and usage of extensible ontologies and to suggest different tools, “Ontoanimal”, that fulfill the requirements of the principles. It emphasizes that there is currently no tool to support the whole extensible ontology development process.

Based on these studies, there were several attempts to support ontology reuse by adding concepts from existing ontologies to the target ontology. However, determining and selecting concepts for reuse was done manually by an expert or using a semantic search engine. Selecting such concepts in more complex domains that have hundreds of ontologies representing different interdisciplinary domains (e.g., the BioPortal repository contains more than one thousand ontologies [20]) is a challenging task which requires searching, evaluating, and ranking those concepts according to their relevance to the context of the target domain. Thus, ontology recommendation, i.e., the process that includes searching, evaluating, and ranking ontologies, becomes an essential part of the ontology reuse process.

Next, we present benchmark ontology recommendation systems, summarize and compare their features, see Table 1, and discuss their limitations which motivated our proposed system.

**Table 1.** Main features of existing ontology recommendation tools.

Reference	Main Contribution	Input	Output	Support Iterative Ontology Recommendations	Support User Explicit/Implicit Preferences	Ontology Ranking	Ontology Reuse
WebCore [21]	Ontology recommendation tool	Keywords	List of selected ontologies	No	Support explicit preferences	Yes	No
BioSS [6]	Ontology selection and ranking system	Keywords	List of selected ontologies	No	Support explicit preferences	Yes	No
JOYCE [22]	Ontology selection tool	Keywords / A paragraph of text	List of ontology modules	No	Support explicit preferences	Yes	No
RecOn [23]	Ontology Recommendation	Unstructured Query (i.e., keywords)	List of selected ontologies	No	No	Yes	No
NCBO 2.0 recommender [7]	Ontology selection and ranking tool	Keywords/Paragraph of text	List of selected ontologies	No	Support Explicit preferences	Yes	No
Ontology Recommendation Framework [24]	Ontology recommendation framework	Ontologies + user requirements	An Ontology	No	Support Explicit preferences	No	No

WebCORE [21] is a semi-automatic Collaborative Ontology Reuse and Evaluation system which provides an NLP model that lexically evaluates ontologies. CORE is one of the first attempts to apply recommender system techniques for ontology recommendations, as it uses previous ontology ratings and evaluations to apply a collaborative filtering technique to recommend the appropriate ontologies which represent a given domain. The drawback of this approach is that it is hard to collect ratings in real situations as the number of ontology development users (for one ontology) is not large enough to satisfy the collaborative filtering technique, in contrast to other product-based recommender systems.

BioSS [6] is an ontology selection system that accepts user input as a set of keywords which describe a domain of interest. Lexical pre-processing tasks are applied to the input keywords, such as spell-checking, query expansion, and resolving ambiguities. It recommends ontologies based on four criteria: input coverage, popularity, knowledge richness, and knowledge formality. Their ontology ranking model uses a sum of fixed weights of those criteria.

JOYCE [22] is an ontology recommendation tool that accepts from some keywords or a paragraph from a scientific publication, given by the user, and identifies relevant ontology modules from a repository of existing ontologies. The criteria coverage, overhead, and overlap are used to evaluate ontologies, and user preferences regarding the importance of these criteria are taken into account. Finally, the output given by JOYCE is a list of ontology modules.

RecOn [23] is an ontology recommendation and ranking system that accepts unstructured queries and converts them to SPARQL queries to retrieve matching ontologies. Ontologies are ranked by defining a knapsack optimization problem using three ranking features: matching cost, informativeness, and popularity.

The NCBO 2.0 ontology recommender [7] is a tool that recommends and ranks ontologies using a lexical matcher that parses some input keywords and assigns matching terms from the ontology repository. It evaluates the candidate ontologies and ranks them using a function that combines weights with each evaluation criterion score. The weights' values can be adjusted by the user. Finally, their evaluation is based on a qualitative comparison between the features of their new version (2.0) with the previous version (1.0). The authors mention that user-based evaluation would better help to understand the effectiveness of their system's utility, but they did not conduct it as it is a challenging task.

Reference [24] is an ontology recommendation framework that employs text categorization and unsupervised learning techniques. Explicit user requirements along with the ontology repository are the input to the framework. Ontologies in the repository are organized into clusters based on their domains, then whenever a user gives a requirement related to a specific domain, the system recommends the most appropriate ontology. The authors built an evaluation model that compares the results of a set of algorithms which are used to determine the correct ontology for given user requirements.

The previously discussed ontology recommendation and ranking systems recommend whole ontologies (or ontology modules) based on some pre-defined evaluation criteria. However, they are not adequate to fulfill our system requirements and to support the best practice of the ontology development process by reusing parts from existing ontologies for the following reasons: (1) they are not designed to extend an input ontology that has some basic concepts representing a specific domain, as most of them accept input keywords and output a list of recommended ontologies without referring the user to the parts or the concepts that can be reused. (2) They recommend whole ontologies and allow importing them into the newly developed ontology but are not useful when we need to reuse parts of the recommended ontologies. Nevertheless, JOYCE [22] has recommended ontology modules, which is an important step toward reuse, since the user will not reuse the entire ontology as it is, but instead, will reuse a module of it. However, this process is still a challenge, as ontology modules may consist of hundreds of concepts that may not all be needed in the reuse process. (3) Some of them consider explicit user preferences in terms of adjustable weights of the evaluation criteria, but this is not enough to personalize the

recommendations for each user as they do not collect information about the ontologies that the user wants to reuse (i.e., the domain, type, format, etc.). Also, manual changing of the weights' values would be a difficult task for inexperienced users. (4) They do not consider user feedback as an important aspect to improve the ontology development process. (5) Finally, they do not support the iterative nature of ontology development, but instead give a recommendation once.

### 3. The Proposed Personalized Ontology Recommendation and Reuse Framework

To overcome the above limitations, we propose a new ontology recommendation and reuse framework to support the ontology development process. We designed a utility-based recommender system that collects explicit and implicit user preferences and recommends ontologies along with their matching concepts based on their utilities. As the ontology development process is an iterative one, the input ontology is extended at each iteration with concepts selected by the user. Our proposed framework is inspired by the reinforcement learning technique, which learns from interactions with the environment. Figure 1 illustrates the architecture of our proposed framework. It has two main components:

- The environment: contains the user who interacts with the system and the ontology repository.
- The agent: interacts with the environment and observes explicit and implicit user preferences to support the user in his decision by recommending appropriate ontologies along with their matching concepts that best fit his preferences. It consists of three main components: the User Preferences Model, the Recommendation Manager, and the Feedback Manager.

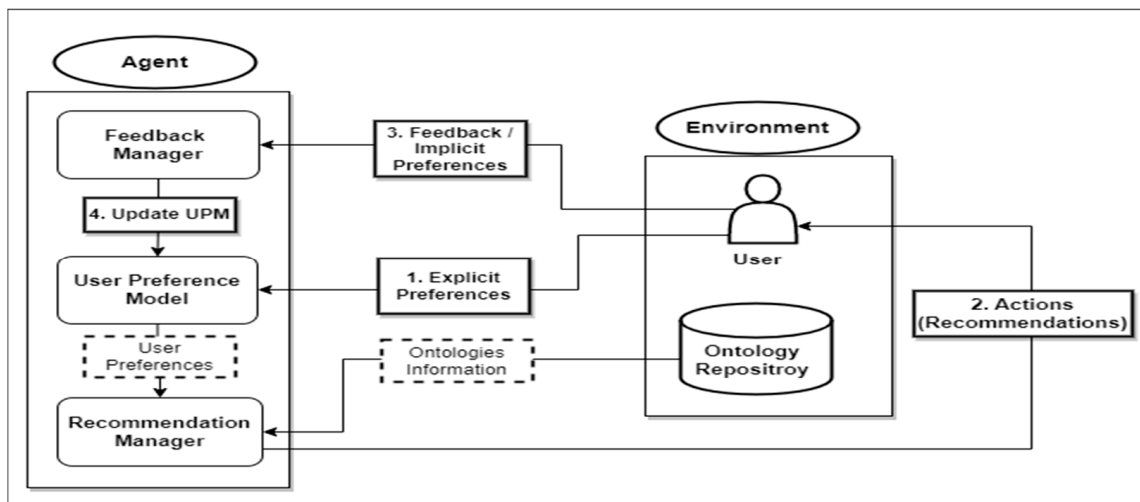


Figure 1. The architecture of the proposed personalized ontology recommendation and reuse framework.

The proposed system works as follows:

1. The interactive session begins when the agent initiates the User Preferences Model, which is responsible for managing the user model of our recommendation system. The user model represents the information needed to describe the user's preferences and personalize his experience in the ontology reuse process. This information is collected explicitly by asking the user to input his preferences regarding the ontologies he wants to reuse, then implicitly by observing his interactions and selections. The explicit information, in the user model, includes the following information related to ontologies:
  - The preferred ontology domain: the ontology domain(s) the user is interested in (sometimes many related domains).



- The ontology popularity: a measure of the popularity of the ontology within the community that uses it.
  - The ontology coverage: the percentage of candidate ontology classes with labels that exactly or partially match the input ontology classes.
  - The ontology type: whether the ontology is a taxonomy (i.e., a hierarchy with is-a relations), or a full semantic ontology (i.e., it makes use of other OWL language constructs beyond is-a relations).
  - The preferred ontology: an ontology (ontologies) that the user knows (i.e., commonly used in his domain) and wants to reuse.
2. Next, the Recommendation Manager receives the user preferences along with information about the existing ontologies (from the ontology repository), matches them with the input ontology and the selected concept, and generates a list of recommendations. Each recommended item in the list consists of a recommended ontology and a class that semantically matches the input class.
  3. Then, the user provides his feedback (i.e., the selection of an ontology and a class to reuse from the recommendation list) to the Feedback Manager.
  4. The Feedback Manager performs the main learning task. It collects and records the user's feedback (i.e., ontologies and classes selected in previous iterations) to update the User Preferences Model, if necessary.
  5. Finally, the system suggests subclasses to the user, which he might reuse to extend the input ontology. Then, the user chooses a new class from the input ontology for the next iteration and repeats steps 2, 3, and 4 until he reaches the desired goal (a newly developed ontology) and ends the session.

#### 4. The Utility-Based Ontology Recommendation System

The utility-based recommender system is a type of knowledge-based recommendation system. It computes the utility of each item using a utility function and applies this function to generate a ranked list of recommended items [25]. Its advantage is that it avoids many of the problems of other recommender systems such as new users, new items, and data sparsity problems. However, it needs a significant number of user interactions to obtain a complete user preference function, which allow is to give accurate recommendations [26]. The Multi-Attribute Utility Theory (MAUT) is used to apply objective measures to help the decision-maker choose the alternative (item) that yields the greatest utility [27]. Attribute values and their weights are combined and aggregated into a multi-attribute utility function to assign a utility to each alternative, and the importance of each attribute is determined by a weight value. Finally, the alternative with the highest score is expected to be preferred by the user.

Our utility-based ontology recommender system needs to calculate the utility of each candidate ontology and the utilities of their candidate classes to generate recommendations. It consists of four main components: (1) the ontology evaluation component, which receives information about the user's preferences and evaluates each candidate ontology from the ontology repository according to four evaluation criteria, i.e., ontology domain, popularity, coverage, and type; (2) the concept evaluation component, which checks each ontology for a class(es) that has a corresponding match with the user's input class and evaluates this class(es) based on its context semantic matching and the concept semantic richness; (3) the feedback management component which receives user feedback (i.e., selected ontology to reuse), interprets it as the implicit user preferences underlying his selections and interactions, and accordingly updates the user model; and (4) the ontology recommendation and ranking component, which receives the scores from the former components and determines the total ontology utility ( $TOU(O,c)$ ) to calculate the final score for each ontology  $O$  along with its matching concept  $c$  and generate a ranked list of ontology recommendations. Next, we provide a detailed description of each component in our recommendation system.

#### 4.1. The Ontology Evaluation Component

The ontology evaluation component receives information about the user’s preferences from the User Preferences Model. Then, it evaluates each candidate ontology from the ontology repository according to four evaluation criteria: the preferred ontology domain, input coverage, ontology popularity, and ontology type. For each ontology  $O$ , the ontology utility function ( $OU(O)$ ) is an aggregated sum function that aggregates the scores of the evaluation criteria to provide an ontology utility score for each candidate ontology. Now, we will explain how each evaluation criteria score is calculated.

##### 4.1.1. The Ontology Evaluation Criteria

(a) *The Preferred ontology domain ( $Domain(O)$ ):* a measure to what extent the user’s preferred domain(s) match the domain(s) of a candidate ontology. We use Equation (1) to calculate the  $Domain(O)$  for any candidate ontology  $O$ .

$$Domain(O) = \frac{\sum_{i=1}^n Domain(O, d_i)}{n} \tag{1}$$

Since the user can select one or many preferred domains, we use  $n$  as the number of domains preferred by the user.  $Domain(O, d_i)$  is the ontology  $O$ ’s domain score for the user’s preferred domain  $d_i$ . Given the fact that an ontology  $O$  could represent more than one domain, we calculate the ontology domain score  $Domain(O, d_i)$  for each user’s preferred domain  $d_i$  by aggregating the  $domain\_match(d_i, d_j)$  score (which is a score measuring the relative similarity between an Ontology  $O$ ’s domain  $d_j$ , and the user’s preferred domain  $d_i$ ):

$$Domain(O, d_i) = \frac{\sum_{j=1}^m domain\_match(d_i, d_j)}{m}, \tag{2}$$

$$domain\_match(d_i, d_j) = 1/2^{L(d_i, d_j)}$$

where  $m$  is the number of domains represented by ontology  $O$ , and  $L(d_i, d_j)$  is the difference in levels in the hierarchy of domains between the ontology  $O$ ’s domain  $d_j$  and the user’s preferred domain  $d_i$ . Figure 2 shows an example of a hierarchy of domains in the biomedical and biological fields.

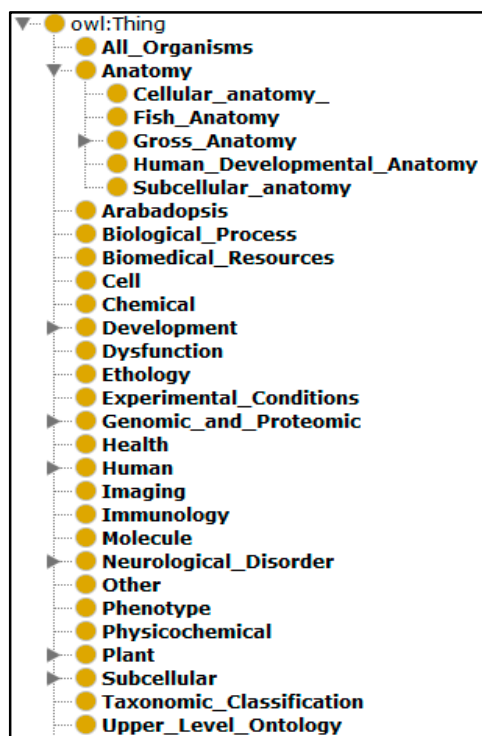


Figure 2. BioPortal\_Categories Ontology.

For example, if ontology  $O$  represents the domains “Anatomy, Mouse Anatomy, and Animal Development”, and the user’s preferred domain is “Anatomy”, then the system calculates the preferred ontology domain score as follows: it first calculates the  $domain-match(d_i, d_j)$  between the user’s preferred domain “Anatomy”, and every domain represented by ontology  $O$  (i.e., “Anatomy, Mouse Anatomy, and Animal Development”). Then, the system divides it by the number of ontology  $O$  domains using Equation (2).

- When ontology  $O$ ’s domain is “Anatomy”, and the user’s preferred domain is “Anatomy”, the two domains match exactly, and  $L(d_i, d_j) = 0$  and  $domain-match(Anatomy, Anatomy) = 1$ .
- When ontology  $O$ ’s domain is “Mouse Anatomy”, and the user’s preferred domain is “Anatomy”, where the “Mouse Anatomy” domain is a direct sub-domain of the “Anatomy” domain (see Figure 2), then,  $L(d_i, d_j) = 1$  and  $domain-match(Anatomy, Mouse Anatomy) = 0.5$ .
- When ontology  $O$ ’s domain is “Animal Development”, and the user’s preferred domain is “Anatomy”, the two domains do not have a match. Then,  $domain-match(Anatomy, Animal Development) = 0$ .
- Since the number of the user’s preferred domains is 1 then,  $Domain(O) = \frac{\sum_{i=1}^3 Domain(o, d_i)}{3} = (1 + 0.5 + 0)/3 = 1.5/3 = 1/2$ .

(b) *The Ontology popularity (Pop(O))*: a measure of how popular an ontology  $O$  is within the community that uses it. To quantify the popularity of ontology  $O$ , we consider both the number of visits and the number of projects that use ontology  $O$ . The popularity score  $Pop(O)$  is the aggregation of both the normalized ontology visits and ontology project scores. It is calculated using Equation (3).

$$Pop(O) = \left( \frac{No. of visits for ontology (O)}{Max no. of visits for an ontology} + \frac{No. of Projects for ontology (O)}{Max no. of projects for an ontology} \right) / 2 \tag{3}$$

(c) *The Input coverage (Cover( $O_{in}, O$ ))*: a measure to assess how well the input keywords (i.e., the labels of the input ontology concepts) are covered in a candidate ontology  $O$ . We use the label of each concept in the input ontology  $O_{in}$  as an input keyword  $i$  and examine the lexical match between this label and the labels of the concepts in candidate ontology  $O$ . The lexical match function ( $lexical\_match(i, O)$ ) gives “1” or “0.5” if the input keyword  $i$  has an exact or partial match with the concept’s label from the ontology  $O$ , respectively. To make it easier to read, we use  $Cover(O)$  to indicate the coverage score for ontology  $O$  in Equation (4):

$$Cover(O) = \sum_{i=1}^n \frac{lexical\_match(i, O)}{n} \tag{4}$$

where  $n$  is the number of input concepts in the input ontology.

(d) *The Ontology type (Type(O))*: measures whether the ontology is a taxonomy which is a hierarchy of concepts with “is-a” relations only or a full semantic ontology that contains other OWL language elements other than “is-a” relations. We determine the ontology type score which ranges from 0 to 1 using Equation (5). A larger ontology type score means that ontology  $O$  contains a large number of object properties, which indicates more semantics.

$$Type(O) = No. of object properties / No. of classes \tag{5}$$

Equation (5) considers the general case where, in any ontology, the number of classes is greater than the number of object properties. For the special case, when the number of classes is less than the number of object properties in any ontology, then  $Type(O) = 1$ .

(e) *The Preferred ontology*: a score that indicates if the user prefers a particular ontology. This happens when the user knows a particular domain ontology, or used one before and wants to reuse it again. The preferred ontology score is set in the first iteration of the session and then aggregated in the subsequent iterations by the Feedback Manager (illustrated in the feedback management component section).



#### 4.1.2. The Ontology Utility Function $OU(O)$

As the user provides his preferences towards the ontologies he is interested in, the system provides a normalized numerical score for each of the previous evaluation criteria. It aggregates these scores, multiplied by pre-determined weights, to calculate the ontology utility score  $OU(O)$  for each candidate ontology in the ontology repository, using Equation (6):

$$OU(O) = w_1 * Domain(O) + w_2 * Pop(O) + w_3 * Cover(O) + w_4 * Type(O) \quad (6)$$

where the total sum of the weights equals 1. It is important to note that the weights here reflect the relative importance of each evaluation criterion. We assigned the weight based on careful study and our experience in the ontological fields. For example, the weights of the  $Domain(O)$  and  $Cover(O)$  criteria are higher than the weights of  $Pop(O)$  and  $Type(O)$  criteria.

Finally, the output of this component is a ranked list of recommended ontologies based on their ontology's utility scores.

#### 4.2. The Concept Evaluation Component

The concept evaluation component receives a ranked list of candidate ontologies from the ontology evaluation component and the user's input class (i.e., the class from the input ontology that the user needs to extend). Then, it checks each ontology in the list for a class(es) that has a corresponding match with the user's input class. For each matching concept, it determines the concept utility  $CU(O,c)$ , which aggregates the scores provided by two measures (the concept context semantic matching and the concept semantic richness) to generate the concept utility score for each candidate concept  $c$  in the candidate ontology  $O$ . Next, we will explain the two measures in detail.

##### 4.2.1. The Concept Evaluation Criteria

(a) *The concept context semantic matching*  $Sem\_Match(O_{in}, c_{in}, O, c)$ : measures the contextual-semantic similarity between two corresponding concepts in two different ontologies. In [28], the contextual-semantic similarity is defined as the similarity in the intentional meanings of these concepts in the context of their ontologies. The authors propose a context-based semantic matching algorithm which extends the AgreementMakerLight (AML) [9], an ontology matching system that generates a set of ontology alignments, to exclude the alignments that do not have evidence for contextual semantic similarity. The basic idea of the context-based semantic matching algorithm [28] is to examine the corresponding semantic relations for the two corresponding concepts such as super-classes, sub-classes, siblings, and equivalent classes. Then, the concept context matching score  $Sem\_Match(O_{in}, c_{in}, O, c)$  is a numerical value between "0 and 1", and is used to measure if the user's input class  $c_{in}$  from the input ontology  $O_{in}$  has a contextual semantic match with its corresponding concept  $c$  in the candidate ontology  $O$ .

(b) *Concept semantic richness* ( $Sem\_Rich(O,c)$ ): an evaluation criterion that assesses the level of detail in the representation of knowledge for a specific concept in a given ontology [6,7,29]. In our case, we measure the semantic richness of concept  $c$  in ontology  $O$  concerning the following criteria: existence of subclasses, existence of superclass, existence of object properties, existence of definition, and existence of synonyms. The concept semantic richness  $Sem\_Rich(O,c)$  score is calculated using Equation (7):

$$Sem\_Rich(O,c) = \sum_{i=1}^n w_i * r_i \quad (7)$$

where  $n$  is the number of a concept's evaluation criteria,  $r_i = 1$  if the criterion is fulfilled by the concept  $c$ , and  $w_i$  is the weight of each criterion and  $\sum_{i=1}^n w_i = 1$ .

#### 4.2.2. The Concept Utility Function $CU(O,c)$

The concept utility function is used to measure the relevance of a concept  $c$  in a candidate ontology  $O$  to be reused to extend the input class  $c_{in}$  from the input ontology  $O_{in}$ . We omit the  $O_{in}$ , and  $c_{in}$  from Equation (8) to make it more readable:

$$CU(O,c) = w_1 * Sem\_Match(O,c) + w_2 * Sem\_Rich(O,c) \quad (8)$$

where  $w_1 = w_2 = 0.5$ , and  $w_1 + w_2 = 1$ .

#### 4.3. The Feedback Management Component

The feedback management component is part of the ontology evaluation component, which concerns the user's implicit preferences and feedback. It receives user feedback (e.g., the selected ontology) to update the User Preferences Model. To develop a new ontology by reusing existing ontologies, we need our recommender system to understand the implicit user preferences underlying his selections and interactions, and after a few iterations, the system updates the user's preferences model to recommend the ontologies that are most likely to be reused by the user (i.e., they should always have a high rank in the recommendation list). Also, the system should consider the case that the user is interested in an ontology and selects concepts for reuse from it, but after some iterations, is no longer interested in this ontology and shifts his selections to other ontologies. In this case, the system should realize this situation and this ontology should gradually vanish from the higher ranks of the list, leaving its place to other ontologies that are more important at that time for the user.

We model the user's feedback management component as a sliding window of size  $k$  to reflect the user's selection history, where  $k$  is the number of iterations that the system saves in his selection history. We assign an ontology score for each ontology in the repository, which is initially 0. The ontology score is updated and increased by 3 if the user selects this ontology, while it is increased by 1 if this ontology appeared in the recommendation list but was not selected. The ontology aggregated score  $OA(O)$  aggregates the ontology scores after each iteration to reflect the relevance of an ontology  $O$  according to the explicit user preferences as well as considering the importance of the ontology  $O$  to the user according to his feedback/previous selections. This would provide the user with the most relevant and preferred ontologies according to his preferences and place them first in the recommendation lists, while irrelevant ontologies should disappear from the first-ranking positions in the recommendation lists. Figure 1 illustrates an example of this scenario.

To illustrate this point, we provide an illustrative example of how the system learns from the user's feedback and previous choices. The user begins his ontology development session by explicitly providing his preferences to the system, then the system generates a list of recommendations (i.e., Ontology A, B, and C) using the ontology and concept utility functions discussed above. Figure 3a shows the initial state of the aggregation scores for each ontology in the first recommendation list (i.e., here the aggregation score for each ontology in the list is zero as the user did not provide his feedback yet). Then, the user selects ontology A for two successive iterations and reuses classes from it, then we can see in Figure 3b that the ontology's A aggregation score becomes 6 while the aggregation scores for ontology B and ontology C are 2 as they appeared in the recommendation list but were not selected for reuse. Using the aggregation score, the system now learns that ontology A is preferable to the user, and, hence, the system suggests ontology A to be in the first ranks in the recommendation lists. Next, if the user shifts his interest from ontology A to ontology B (which may represent different aspects from the domain of interest to the user), as he selects it for three successive iterations (see Figure 3c), then ontology B's aggregated score will increase gradually and could outperform ontology A's aggregated score. Now, the system learns that the user has a shift in interest in the ontologies he wants to reuse and this is reflected in the rankings of the ontologies in the recommendation list.

Iteration 0	
Recommendation List	Ontology Aggregated score OA(O)
1. Ontology A	0
2. Ontology B	0
3. Ontology C	0

(a)

Iteration 1		Iteration 2	
Recommendation List	Ontology Aggregated score OA(O)	Recommendation List	Ontology Aggregated score OA(O)
1. Ontology A	3	1. Ontology A	6
2. Ontology B	1	2. Ontology B	2
3. Ontology C	1	3. Ontology C	2

(b)

Iteration 3		Iteration 4		Iteration 5	
Recommendation List	Ontology Aggregated score OA(O)	Recommendation List	Ontology Aggregated score OA(O)	Recommendation List	Ontology Aggregated score OA(O)
1. Ontology A	7	1. Ontology A	8	1. Ontology B	11
2. Ontology B	5	2. Ontology B	8	2. Ontology A	9
3. Ontology C	3	3. Ontology C	4	3. Ontology C	5

(c)

**Figure 3.** The ontology aggregated scores for ontologies A, B, and C during an ontology development session. (a) In the beginning of the session, the ontology aggregated scores for ontologies A, B, and C are 0. (b) At Iterations 1 and 2 ontology A is selected and reused by the user: then the OA(ontology A) is updated and increased by 3, the OA(ontology B) and the OA(ontology C) is updated and increased by 1. (c) At Iteration 3, 4 and 5, ontology B is selected and reused by the user: then the OA(ontology B) is updated and increased by 3, the OA(ontology A) and the OA(ontology C) is updated and increased by 1.

#### 4.4. Ontology Recommendation and Ranking Component

The ontology recommendation and ranking component receives the scores from the former components (i.e., the ontology utility component, the class utility component, and the feedback management component) and determines the total ontology utility ( $TOU(O,c)$ ) to calculate the final score for each ontology  $O$  along with its matching concept  $c$ . Based on the total ontology utility score, the system ranks the recommendations and provides the user with a list of top-k recommendations that best fit his explicit and implicit preferences. The total ontology utility score for each candidate ontology is calculated using Equation (9):

$$TOU(O,c) = w_1 * OU(O) + w_2 * CU(O,c) + w_3 * OA(O) \tag{9}$$

where  $w_1 = w_2 = w_3$  are three equal weights, and their summation equals 1. It is important to note that, as a preliminary design, in assigning weights for those measures we assume that the user is not technically an expert and they are equally important to him. In later versions, we could ask the user to assign them by himself according to his requirements or we could begin with equal weights and learn them during the user’s interactions.

### 5. Implementation Details

We implemented a web-based platform [30] for our proposed ontology recommendation and reuse framework. It enables the users to iteratively develop a new ontology representing some domain(s) by extending the classes of an input ontology with parts from

existing ontologies. It also enables them to visualize the input ontology extending in each iteration, until the development process is completed. We chose the biomedical field to apply our approach due to several reasons. First, the diversity and heterogeneity of the bio-ontologies which cover different and interlinked domains gives us a comfortable area to study and test different ontological cases. Second, biomedical ontologies are available in repositories (e.g., BioPortal [20] and OBO Foundry [31]) which offer tools and web services that enable accessing available ontologies and information about them. Third, these repositories ensure availability of their details, in contrast to other domains that may refer to broken URIs and may not be accessible [32]. Finally, in domains with a limited number of ontologies, our ontology reuse framework is not needed, since the user can trace and select the needed ontologies for reuse, but in domains with hundreds of ontologies and hundreds of thousands of concepts, i.e., bio-domain ontologies, a tool is required to support the user in finding appropriate ontologies and proper concepts to reuse. However, we need to clarify the applicability of our system to any ontological domain.

Next, we will describe the main functionalities of the platform with a practical example of how the user can use the system.

- Collecting explicit user preferences:** on the home page, the user is prompted to enter his preferences. Figure 4 presents a form where the user is asked to input his explicit preferences. For example, suppose the user wants to develop an ontology about human diseases. Basically, he should upload the input ontology he wants to develop the new ontology upon, which is, in our example, the Core Ontology for Biology and Biomedicine (COB) [33], as it is a core ontology in BioPortal. Then, he selects the “Human” domain as the preferred domain of the new ontology and provides his preferences on the recommended ontologies as he prefers popular ontologies, high input coverage ontologies, and full semantic ontologies, and selects the Infectious Disease Ontology (IDO) as a preferred ontology to reuse if it contains matched concepts. Finally, the user selects the class “disease diagnosis” from the input ontology to run the ontology recommendation algorithm and generates the first ontology recommendations list.


	Loaded input ontology name : *	COB.owl
	Please select your domain(s) of interest : *	Human
	Your selected domain(s) : *	Human
	Are you interested in popular ontologies : *	<input checked="" type="radio"/> YES <input type="radio"/> NO
	Are you interested in high coverage ontologies : *	<input checked="" type="radio"/> YES <input type="radio"/> NO
	Which ontology type do you prefer : *	<input type="radio"/> Taxonomy <input checked="" type="radio"/> Full Semantic
	Is/Are there a particular ontologies you want to reuse them :	Infectious Disease Ontology(IDO)
	You particular ontologies selection you want to reuse :	Infectious Disease Ontology
	You particular ontologies selection acronym :	IDO
	Please select a class to begin the reuse process : *	disease diagnosis
	Class ID to begin the reuse process : *	http://purl.obolibrary.org/obo/COB_0000062
	<ul style="list-style-type: none"> <li>▶ obsolete macromolecular entity</li> <li>▶ process</li> <li>▶ material entity</li> <li>▶ characteristic</li> <li>▶ obsolete_elementary charge</li> <li>▼ information               <ul style="list-style-type: none"> <li>▶ document</li> <li>▶ conclusion based on data</li> <li>▶ data item</li> <li>▶ directive information entity</li> <li>▶ <a href="#">disease diagnosis</a></li> <li>▶ phenotypic finding</li> <li>▶ immaterial entity</li> </ul> </li> </ul>	<p><b>" Your Selected Class Definition "</b></p> <p>An information content entity whose concretizations indicate to their bearer how to realize them in a process</p>

Figure 4. The user’s preferences form on the platform’s home page.

- **Ontology recommendation system:** recommends ontologies along with their matching concepts based on the user’s preferences. The top-k ontologies in the recommendation list are presented to the user and are ranked according to the total ontology utility score for each ontology. Figure 5 represents a recommendation list which includes information about each ontology (e.g., ontology name and description), and information about the matched class in this ontology (e.g., class name, synonyms, and subclasses). In our example, the user investigates each option in the recommendation list carefully and decides to select the OBI ontology “Ontology for Biomedical Investigations” and reuse its matched class “diagnosis of infectious disease”.

Rank	Ontology Name	Class ID	Class Label	Class Synonyms	Class Definitions	Sub-Classes
<input checked="" type="radio"/>	1 Ontology for Biomedical Investigations	<a href="http://purl.obolibrary.org/obo/OBI_0002385">http://purl.obolibrary.org/obo/OBI_0002385</a>	diagnosis of infectious disease	[ ],	[ "A diagnosis that is an assertion that a patient who is the subject of a diagnostic process has an infectious disease." ],	[ "diagnosis of hepatitis B", "diagnosis of hepatitis C", "diagnosis of HIV" ],
<input type="radio"/>	2 Kidney Tissue Atlas Ontology	<a href="http://purl.obolibrary.org/obo/OPMI_0004482">http://purl.obolibrary.org/obo/OPMI_0004482</a>	age at diagnosis of disease	[ ],	[ ],	[ "age at diagnosis of diabetes", "age at diagnosis of kidney disease" ],
<input type="radio"/>	3 Ontology for Biobanking	<a href="http://purl.obolibrary.org/obo/OBI_0002385">http://purl.obolibrary.org/obo/OBI_0002385</a>	diagnosis of infectious disease	[ ],	[ "A diagnosis that is an assertion that a patient who is the subject of a diagnostic process has an infectious disease." ],	[ "diagnosis of hepatitis B", "diagnosis of hepatitis C", "diagnosis of HIV" ],
<input type="radio"/>	4 KB_Bio_101	<a href="http://www.projecthalo.com/aura#Diagnosis-Of-Renal-Disease">http://www.projecthalo.com/aura#Diagnosis-Of-Renal-Disease</a>	Diagnosis-Of-Renal-Disease	[ ],	[ ],	[ ],
<input type="radio"/>	5 Ontology of Precision Medicine and Investigation	<a href="http://purl.obolibrary.org/obo/OPMI_0004482">http://purl.obolibrary.org/obo/OPMI_0004482</a>	age at diagnosis of disease	[ ],	[ ],	[ "age at diagnosis of diabetes", "age at diagnosis of kidney disease" ],

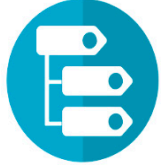
Figure 5. An ontology recommendations list.

- **Ontology reuse and iterative expansion of the input ontology:** at this stage, the system suggests the subclasses of the selected class to the user. Figure 6 presents the subclass of the selected class “diagnosis of infectious disease” to the user, and he can select all or some of them to extend the input class. At this step, the system checks for inconsistencies and, if there are inconsistencies, it does not add the new classes and displays a message to the user that the class cannot be added. If there is not any inconsistency, the selected subclasses are added as subclasses to the class selected by the user from the input ontology, see Figure 7. The process is repeated, and every time the user chooses a new class to extend from the input ontology until he is satisfied with the new ontology.



**Please select some or all of the concepts subclasses:**


The selected subclasses will be added as subclasses of the input class to extend the input ontology. If they cause inconsistency in the input ontology.



Sub-Classes :

- diagnosis of hepatitis B
- diagnosis of hepatitis C
- diagnosis of HIV

**Figure 6.** The user selects subclass(es) to extend the input class.



Please Select a new class to extend : Select.....▼

Your Selected Class Definition :

- ▶ obsolete macromolecular entity
- ▶ process
- ▶ material entity
- ▶ characteristic
- ▶ obsolete\_elementary charge
- ▼ information
  - ▶ document
  - ▶ conclusion based on data
  - ▶ directive information entity
  - ▶ data item
  - ▼ disease diagnosis
    - ▶ diagnosis\_of\_hepatitis\_C
    - ▶ diagnosis\_of\_HIV
    - ▶ diagnosis\_of\_hepatitis\_B
  - ▶ phenotypic finding
- ▶ immaterial entity

**Figure 7.** The input ontology has been extended after adding the selected sub-classes as new sub-classes of the input class.

## 6. Evaluation

We conducted a user-based evaluation experiment, where users were asked to develop a new ontology and interact with the system to assess its effectiveness in generating relevant recommendation lists, according to the users' preferences, during the ontology development session (i.e., which consists of many iterations). However, this evaluation was challenging for the following reasons. First, each user has a specific understanding of what makes an ontology relevant and thus worth recommending. Our system considers this by recommending ontologies to each user according to his or her specific preferences and selections which might differ from those of other users. Second, the system generates a different recommendation list for each user in each interaction/iteration during the development session, which ranges from two to seven iterations for each user in the experiment we conducted. Consequently, there is no single reference list (ground truth recommendation) that is valid for every potential user. Moreover, for each user, we would have to consider all potential interactions with the system and create a reference list for each of those interactions. That is not feasible. For these reasons, we came up with a more viable approach. Each user/evaluator re-sorted the ranking of the ontologies in the recommendation lists according to his preferences. The re-ranked list expressed the user's personal and user-specific assessment of relevance. We assessed the rankings' quality using information retrieval evaluation metrics.

### 6.1. Experimental Setup

To test the utility and ranking quality of the output list of our ontology recommendation system, we conducted a user-based evaluation to compare each output list with those produced by the expert test users. During each recommendation cycle, the user was asked to check the recommendation list and to provide feedback on the ranking of ontologies by re-ranking the ontologies according to his opinion.

To make the evaluation tasks of the test users comparable, we chose one input “core” ontology with familiar, abstract concepts to begin the process. We chose the Core Ontology for Biology and Biomedicine (COB) as it is a small-sized, upper-level ontology that brings together main key terms that can be reused in both biological and biomedical domains, and thus promises to give a large number of reuse opportunities that are tied with existing biomedical ontologies. Thereby, the evaluator’s task was to extend the upper-level ontology “COB” to develop a new ontology for any selected domain (e.g., botany, human health, chemicals, anatomy, etc.).

We expected that the best responses would come from test users who are experts in ontologies with a biomedical/bioinformatics background. However, it was hard to reach a large number of such expert test users, so we tried to balance between test users who are familiar with ontologies (i.e., understand the ontology reuse concept) and those who are familiar with bio-terminologies and bio-domains but with a minor ontological background. We obtained responses from 16 candidates who wanted to help with the evaluation task. We asked them to test the system by choosing a domain they know and to extend some classes from the input ontology with classes from ontologies related to the chosen domain.

The evaluators were free to choose when to stop the iterations and the extensions of the input ontology classes whenever they were satisfied with the resulting output ontology. The number of iterations that each user went through ranged from two to five iterations (46 iterations in total). We also asked them to send us their feedback and notes about the strengths and weaknesses of our platform.

### 6.2. Evaluation Metrics

We used the test users’ responses and various evaluation metrics to evaluate the effectiveness of our recommendation system by directly evaluating the rankings’ quality. We considered the following evaluation metrics, which are discussed in [34,35]:

(1) Mean Absolute Error (MAE): used to measure the effectiveness of a recommendation algorithm in terms of accuracy. It computes the average of the absolute value differences between the rank predicted by the recommender and the actual rank given by the user over all of the recommendation items. The MAE is calculated using Equation (10), the lower the MAE the better the accuracy:

$$MAE = \frac{1}{n} \sum_{i=1}^n |d_i - d'_i| \quad (10)$$

where  $d_i$  is the actual rank,  $d'_i$  is the predicted rank, and  $n$  is the number of items in the recommendation list.

(2) Mean Reciprocal Rank (MRR): an information retrieval measure that evaluates the recommendation list based on the position of the first relevant item. It measures if the recommender system places the user’s most relevant item at the top of the list. The larger the MRR value, the better the performance of the recommender system. The MRR is defined using Equation (11):

$$MRR = \frac{1}{q} \sum_{i=1}^q \frac{1}{rank_i} \quad (11)$$

where  $q$  is the number of recommendation lists (i.e., iterations), and  $rank_i$  is the rank of the user’s most relevant item (ranked 1) in the list of recommendations generated by the system.

(3) Mean Average Precision (MAP): in the context of recommender systems, precision  $P$  measures the fraction of the recommended items which are relevant. It is defined as:

$$P = \frac{\text{number of relevant items}}{\text{total number of items in recommendation list}} \tag{12}$$

while  $P@k$  is the fraction of relevant items in the top- $k$ -recommended results, defined as:

$$P@k = \frac{\text{number of relevant items in top } k \text{ results}}{k} \tag{13}$$

The average  $P@k$  ( $AP@k$ ) calculates the precision for only the relevant items that are recommended. It is defined as:

$$AP@k = \sum_{k=1}^k P@k * rel(k) \tag{14}$$

where  $rel(k)$  is an indicator function that equals 1 if the  $k$ th item is relevant, and 0 otherwise. An item is relevant if the user is satisfied with its ranking in the list (i.e., does not change its position), and is considered to be non-relevant otherwise.

$$AP@k = \sum_{k=1}^k P@k * rel(k) \tag{15}$$

The Mean Average Precision (MAP) measures the  $AP@k$  averaged over the recommendation lists of all users. It is defined as:

$$MAP = \frac{1}{n} \sum_{i=1}^n AP_i \tag{16}$$

where  $n$  is the number of recommendation lists generated by the test users.

(4) Normalized Discounted Cumulative Gain (NDCG): a metric to measure the rankings' quality, where the gain  $G_i$  for an item  $i$  is the same as the ranking relevance score, and it was used to measure the effectiveness of a system's ranking of some items compared with the user's ranking for the same items. The Cumulative Gain  $CG(k)$  is defined as the sum of gains up to a position  $k$  in the recommendation list.

$$CG(k) = \sum_{i=1}^k G_i \tag{17}$$

The drawback of CG is that it neglects the importance of the ordering of items (i.e., by swapping the relative order of any two items, the CG would be unaffected). To overcome the drawback of CG, Discounted Cumulative Gain (DCG) has been introduced to penalize highly relevant items being placed at the bottom.

$$DCG(k) = \sum_i^k \frac{G_i}{\log_2(i+1)} \tag{18}$$

The DCG score still has a drawback, since it adds up with the length of the recommendation list. Therefore, if we compare the DCG score for a system recommending the top 5 and the top 10 items, the latter is always the higher score not because of its recommendation quality but because of its length. IDCG (ideal DCG) is the DCG score for the most ideal ranking, which is the ranking of the items top-down according to their relevance, up to position  $k$ :

$$IDCG(k) = \sum_i^{|I(k)|} \frac{G_i}{\log_2(i+1)} \tag{19}$$

where the  $I(k)$  represents the length of the ideal list of items up to position  $k$ ,  $|I(k)| = k$ .

Then, *NDCG* is the normalization of the *DCG* score by the *IDCG*, where its value is always between 0 and 1 regardless of the length of the recommendation list.

$$NDCG(k) = \frac{DCG(k)}{IDCG(k)} \quad (20)$$

### 6.3. Results and Discussion

In this subsection, we report the results of applying the evaluation metrics to measure the performance of our ontology recommender system. As mentioned earlier, 16 evaluators tried our system and made 46 iterations in total. In each iteration, the evaluator received a recommendation list of ontologies and their concepts that best match his query and preferences. The user could check the information about each ontology (e.g., name and description) and its matched class (e.g., label, synonyms, and sub-classes) to decide if the system ranks the recommendations properly or if they needed to be re-ranked. We collected the test users' responses in log files to record their choices in each iteration. The differences in the rankings between the recommended list and the user's ranked list (i.e., the absolute error) for each iteration done by each user, are shown in Table 2. An absolute error value equal to 0 means that the test user is satisfied with the algorithm's ranking, and he did not perform any re-rank. If the user made a minor change in the rankings by moving one or two ontologies a step downward or upward (usually in the first positions), we would obtain a reasonable absolute error value (e.g., 2 or 4). High absolute error values for some test users (e.g., 14, 16, and 20) come when they move an ontology from a bottom position to a top one, where this causes a shift for all the in-between ontologies and thus the absolute error value becomes high. We also notice that the evaluators conducted most of the re-ranking of the recommendation list at the first iterations, while conducting a minor re-ranking in later iterations. This is an indicator that our recommender investigates user preferences in the first iterations and then learns from the user's selections, whereas, in later iterations, the recommendations gradually become much more appropriate to user preferences.

**Table 2.** Evaluators' responses and calculating Mean Absolute Error (*MAE*).

	No. of Iterations	Absolute Error of Every Iteration					Total Absolute Error
		Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5	
Evaluator 1	2	0	0	-	-	-	0
Evaluator 2	2	0	0	-	-	-	0
Evaluator 3	3	6	2	0	-	-	8
Evaluator 4	2	0	0	-	-	-	0
Evaluator 5	3	0	0	0	-	-	0
Evaluator 6	4	0	14	0	4	-	18
Evaluator 7	4	12	0	14	0	-	26
Evaluator 8	2	4	0	-	-	-	4
Evaluator 9	3	20	10	0	-	-	30
Evaluator 10	2	0	0	-	-	-	0
Evaluator 11	5	6	10	0	0	0	16
Evaluator 12	2	0	0	-	-	-	0
Evaluator 13	2	8	0	-	-	-	8
Evaluator 14	3	12	4	2	-	-	18
Evaluator 15	4	0	0	2	0	-	2
Evaluator 16	3	16	6	0	-	-	22
<b>Total = 46</b>							<b>Total = 152</b>

We measured the accuracy of our recommendations by applying the *MAE* metric, and we obtained a score of 3.3. We expect a lower *MAE* score if the evaluators try the system for other iterations. In statistics, the reciprocal rank of a query response is the multiplicative

inverse of the rank of the first correct answer. For example, 1 for first place, 1/2 for second place, 1/3 for third place, and so on. Table 3 shows the reciprocal rank for each iteration, where the mean reciprocal rank (*MRR*) is the average of the reciprocal ranks across all 46 iterations. We obtained an *MRR* score value equal to 0.88, which means, in about 88% of the users' iterations, the system successfully managed to place the user's most relevant item at the top of the list.

**Table 3.** The reciprocal rank for each iteration and the *MRR* score value.

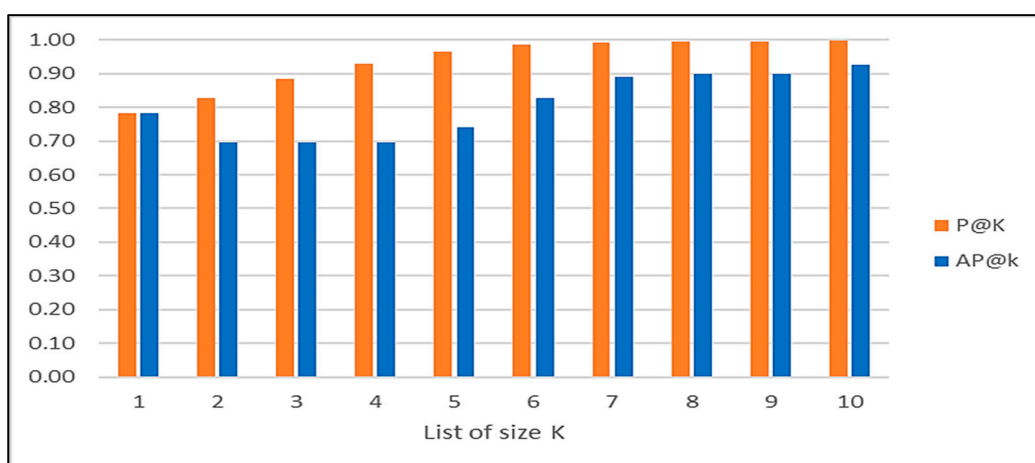
<b>Iteration No.</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>	<b>16</b>
<b>1/rank1</b>	1	1	1	1	1/4	1/2	1	1	1	1	1	1	1	1/3	1	1/2
<b>Iteration No.</b>	<b>17</b>	<b>18</b>	<b>19</b>	<b>20</b>	<b>21</b>	<b>22</b>	<b>23</b>	<b>24</b>	<b>25</b>	<b>26</b>	<b>27</b>	<b>28</b>	<b>29</b>	<b>30</b>	<b>31</b>	<b>32</b>
<b>1/rank1</b>	1/3	1	1/2	1	1	1	1/2	1	1	1	1	1	1	1	1	1
<b>Iteration No.</b>	<b>33</b>	<b>34</b>	<b>35</b>	<b>36</b>	<b>37</b>	<b>38</b>	<b>39</b>	<b>40</b>	<b>41</b>	<b>42</b>	<b>43</b>	<b>44</b>	<b>45</b>	<b>46</b>	<b>MRR</b>	
<b>1/rank1</b>	1	1	1/2	1	1/3	1	1	1	1	1	1	1/2	1	1	0.88	

The overall results of the precision evaluation are shown in Table 4. To calculate the precision up to any position *k* for any recommendation list, we considered the fraction of items in the system's recommendation list of size *k* that also exists in the user's recommendation list of the same size, regardless of their positions. For example, if the system's recommendation list of size 3 for a particular user query contains ontologies *A*, *B*, and *C*, and the users re-ranked ontology *A* to be in the 4th position (so it is not in the list of size 3), then the user's recommendation list of size 3 contains the ontologies *B*, *C*, and *D*, and the *P@3* is 0.66. Table 4 shows the *P@k* scores up to the 10th position for all recommendation lists of the overall users' iterations. Our recommender exhibits a lower average precision of about 0.7 in the up-to-5-items recommendation lists, while the average precision value increases up to 0.93 in the up-to-10-items recommendation lists. This is an indicator that users re-ranked items in the first positions in the recommendation lists and rarely re-ranked items in the lower positions in the list. Figure 8 shows the *P@k* and *AP@k* scores for the size *k* recommendation lists up to 10 items, and the overall *MAP* score is 0.81.

**Table 4.** Overall result of the precision metric.

<b>Iteration No.</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>
<i>P@K</i>	0.78	0.83	0.88	0.93	0.97	0.99	0.99	0.99	0.99	1
<i>AP@K</i>	0.78	0.70	0.70	0.70	0.74	0.83	0.89	0.90	0.90	0.93

**MAP = 0.81**



**Figure 8.** *P@k* and *AP@k* scores for size *k* recommendation lists.



To calculate the Cumulative Gain, we define the Gain for an item as an indicator of the relevancy of this item. It is a binary value indicating whether the user agreed with the item's rank ( $G_i = 1$ ) or re-ranked it and put it in a different position ( $G_i = 0$ ). We calculated the value of the *NDCG* for the top seven recommendations (as the smallest number of items in the recommendation lists is seven), which equals 0.96. A high *NDCG* value indicates that our system's recommendations were, in most cases, relevant to the user and need not be re-ranked by the user, thus indicating a good ranking quality.

## 7. Conclusions and Future Work

In this paper, we propose a personalized ontology recommendation framework that aims to support a user in developing a new ontology by reusing parts from existing ontologies. We use a utility-based ontology recommendation system to recommend and rank ontologies along with their contextually matching concepts that best match the user's preferences to help him reuse parts of existing ontologies to extend an input ontology. Our recommendation system uses a User Preferences Model that represents the user's explicit and implicit preferences to recommend appropriate ontologies to each user and detect the changes in the user's interests to update the user model. We develop a prototype of our approach using ontologies from the biomedical domain. A user-based evaluation has been conducted to evaluate the system's recommendations and rankings. We evaluate the performance of our proposed recommender system using different evaluation metrics, which prove its effectiveness in generating and ranking ontology recommendations to the user.

In the future, we plan to extend our recommendation system to include recommending other ontology components, e.g., object properties, and add the ability to recommend and reuse ontology modules. To comprehensively improve the ontology development framework, we aim to add a natural language processing component that can extract key concepts of a given domain from textual documents, e.g., scientific papers, as the user could use this to enrich the input ontology. We need to test our framework using other domains, e.g., travel and tourism, to prove its ability to recommend ontologies from any domain.

**Author Contributions:** Conceptualization, M.A.; methodology, M.A. and F.K.; software, M.A.; validation, M.A., F.K. and T.H.A.S.; formal analysis, M.A.; investigation, M.A. and F.K.; writing—original draft preparation, M.A.; writing—review and editing, M.A., F.K. and T.H.A.S.; visualization, M.A. and F.K.; supervision, F.K. and T.H.A.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** Source code and experimental data are available at: <https://github.com/marwa811/OntologyReuseProject> (accessed on 2 October 2023).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Li, M.; Wang, D.; Du, X.; Wang, S. Ontology Construction for Semantic Web: A Role-Based Collaborative Development Method. In Proceedings of the Web Technologies Research and Development—APWeb 2005, Shanghai, China, 29 March–1 April 2005; Zhang, Y., Tanaka, K., Yu, J.X., Wang, S., Li, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 609–619.
- Simperl, E. Reusing Ontologies on the Semantic Web: A Feasibility Study. *Data Knowl. Eng.* **2009**, *68*, 905–925. [\[CrossRef\]](#)
- Protégé. Available online: <https://protege.stanford.edu/> (accessed on 22 November 2022).
- Zulkarnain, N.Z.; Meziane, F.; Crofts, G. A Methodology for Biomedical Ontology Reuse. In Proceedings of the Natural Language Processing and Information Systems, Salford, UK, 22–24 June 2016; Métais, E., Meziane, F., Saraee, M., Sugumaran, V., Vadera, S., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 3–14.
- Katsumi, M.; Grüninger, M. The Metatheory of Ontology Reuse. *Appl. Ontol.* **2018**, *13*, 225–254. [\[CrossRef\]](#)
- Martínez-Romero, M.; Vázquez-Naya, J.M.; Pereira, J.; Pazos, A. BiOSS: A System for Biomedical Ontology Selection. *Comput. Methods Programs Biomed.* **2014**, *114*, 125–140. [\[CrossRef\]](#)
- Martínez-Romero, M.; Jonquet, C.; O'Connor, M.J.; Graybeal, J.; Pazos, A.; Musen, M.A. NCBO Ontology Recommender 2.0: An Enhanced Approach for Biomedical Ontology Recommendation. *J. Biomed. Semant.* **2017**, *8*, 21. [\[CrossRef\]](#)

8. Tigrine, A.N.; Bellahsene, Z.; Todorov, K. Light-Weight Cross-Lingual Ontology Matching with LYAM++. In Proceedings of the on the Move to Meaningful Internet Systems: OTM 2015 Conferences, Rhodes, Greece, 26–30 October 2015; Debruyne, C., Panetto, H., Meersman, R., Dillon, T., Weichhart, G., An, Y., Ardagna, C.A., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 527–544.
9. Faria, D.; Pesquita, C.; Santos, E.; Palmonari, M.; Cruz, I.F.; Couto, F.M. The AgreementMakerLight Ontology Matching System. In Proceedings of the on the Move to Meaningful Internet Systems: OTM 2013 Conferences, Graz, Austria, 9–13 September 2013; Meersman, R., Panetto, H., Dillon, T., Eder, J., Bellahsene, Z., Ritter, N., De Leenheer, P., Dou, D., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 527–541.
10. Alharbi, R.; Tamma, V.; Grasso, F. Characterising the Gap Between Theory and Practice of Ontology Reuse. In Proceedings of the 11th on Knowledge Capture Conference, Virtual, 2–3 December 2021; Association for Computing Machinery (ACM): New York, NY, USA, 2021; pp. 217–224.
11. Courtot, M.; Gibson, F.; Lister, A.L.; Malone, J.; Schober, D.; Brinkman, R.R.; Ruttenberg, A. MIREOT: The Minimum Information to Reference an External Ontology Term. *Appl. Ontol.* **2011**, *6*, 23–33. [[CrossRef](#)]
12. Xiang, Z.; Courtot, M.; Brinkman, R.R.; Ruttenberg, A.; He, Y. OntoFox: Web-Based Support for Ontology Reuse. *BMC Res. Notes* **2010**, *3*, 175. [[CrossRef](#)] [[PubMed](#)]
13. Lonsdale, D.; Embley, D.W.; Ding, Y.; Xu, L.; Hepp, M. Reusing Ontologies and Language Components for Ontology Generation. *Data Knowl. Eng.* **2010**, *69*, 318–330. [[CrossRef](#)]
14. Noy, N.F.; Sintek, M.; Decker, S.; Crubézy, M.; Fergerson, R.W.; Musen, M.A. Creating Semantic Web Contents with Protege-2000. *IEEE Intell. Syst.* **2001**, *16*, 60–71. [[CrossRef](#)]
15. d’Aquin, M.; Gangemi, A.; Motta, E.; Dzbor, M.; Haase, P.; Erdmann, M. Neon Tool Support for Building Ontologies by Reuse. In Proceedings of the Demo International Conference on Biomedical Ontologies, Buffalo, NY, USA, 24–26 July 2009.
16. Hanna, J.; Chen, C.; Crow, W.A.; Hall, R.; Liu, J.; Pendurthi, T.; Schmidt, T.; Jennings, S.F.; Brochhausen, M.; Hogan, W. Simplifying MIREOT: A MIREOT Protégé. In Proceedings of the 2012th International Conference on Posters & Demonstrations Track, Boston, MA, USA, 11–15 November 2012; CEUR-WS.org: Aachen, Germany, 2012; Volume 914, pp. 25–28.
17. García-Santa, N.; Ateazing, G.A.; Villazón-Terrazas, B. The ProtégéLOV Plugin: Ontology Access and Reuse for Everyone. In Proceedings of the BT—The Semantic Web: ESWC 2015 Satellite Events, Portorož, Slovenia, 31 May–4 June 2015; Gandon, F., Guéret, C., Villata, S., Breslin, J., Faron-Zucker, C., Zimmermann, A., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 41–45.
18. d’Aquin, M.; Motta, E. Watson, More than a Semantic Web Search Engine. *Semant. Web* **2011**, *2*, 55–63. [[CrossRef](#)]
19. He, Y.; Xiang, Z.; Zheng, J.; Lin, Y.; Overton, J.A.; Ong, E. The EXTensible Ontology Development (XOD) Principles and Tool Implementation to Support Ontology Interoperability. *J. Biomed. Semant.* **2018**, *9*, 3. [[CrossRef](#)] [[PubMed](#)]
20. NCBO. BioPortal. Available online: <https://biportal.bioontology.org/> (accessed on 10 August 2022).
21. Cantador, I.; Fernández, M.; Castells, P. Improving Ontology Recommendation and Reuse in WebCORE by Collaborative Assessments. In Proceedings of the 16th International World Wide Web Conference (WWW 2007), Banff, AB, Canada, 8–11 May 2007.
22. Faessler, E.; Klan, F.; Algergawy, A.; König-Ries, B.; Hahn, U. Selecting and Tailoring Ontologies with JOYCE. In Proceedings of the European Knowledge Acquisition Workshop, Bologna, Italy, 19–23 November 2016; pp. 114–118.
23. Butt, A.S.; Haller, A.; Xie, L. RecOn: Ontology Recommendation for Structureless Queries. *Appl. Ontol.* **2016**, *11*, 301–324. [[CrossRef](#)]
24. Sarwar, M.A.; Ahmed, M.; Habib, A.; Khalid, M.; Ali, M.A.; Raza, M.; Hussain, S.; Ahmed, G. Exploiting Ontology Recommendation Using Text Categorization Approach. *IEEE Access* **2020**, *9*, 27304–27322. [[CrossRef](#)]
25. Zheng, Y. Utility-Based Multi-Criteria Recommender Systems. In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, Limassol, Cyprus, 8–12 April 2019; pp. 2529–2531.
26. Deng, F. Utility-Based Recommender Systems Using Implicit Utility and Genetic Algorithm. In Proceedings of the 2015 International Conference on Mechatronics, Electronic, Industrial and Control Engineering (MEIC-15), Shenyang, China, 1–3 April 2015; pp. 860–864.
27. Dyer, J.S. Maut—Multiattribute Utility Theory BT. In *Multiple Criteria Decision Analysis: State of the Art Surveys*; Figueira, J., Greco, S., Ehrogott, M., Eds.; Springer: New York, NY, USA, 2005; pp. 265–292, ISBN 978-0-387-23081-8.
28. Abdelreheim, M.; Klan, F.; Soliman, T.H.A. An Evidence-Based, Contextual Approach to the Validation of Concept Alignments to Support Ontology Reuse. In Proceedings of the Qurator 2020, Berlin, Germany, 20–21 January 2020.
29. Akremi, H.; Zghal, S. DOF: A Generic Approach of Domain Ontology Fuzzification. *Front. Comput. Sci.* **2021**, *15*, 153322. [[CrossRef](#)]
30. Abdelreheim, M. Interactive Ontology Recommendation and Reuse Platform. Available online: <http://195.246.49.66/index.aspx> (accessed on 1 August 2023).
31. OBO Foundry. Available online: <http://obofoundry.org/> (accessed on 20 August 2022).
32. Fernández-López, M.; Poveda-Villalón, M.; Suárez-Figueroa, M.C.; Gómez-Pérez, A. Why Are Ontologies Not Reused across the Same Domain? *J. Web Semant.* **2019**, *57*, 100492. [[CrossRef](#)]
33. The Core Ontology for Biology and Biomedicine (COB). Available online: <https://biportal.bioontology.org/ontologies/COB> (accessed on 30 August 2023).

34. Gunawardana, A.; Shani, G.; Yogev, S. Evaluating Recommender Systems. In *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 547–601.
35. Chen, M.; Liu, P. Performance Evaluation of Recommender Systems. *Int. J. Perform. Eng.* **2017**, *13*, 1246. [[CrossRef](#)]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.