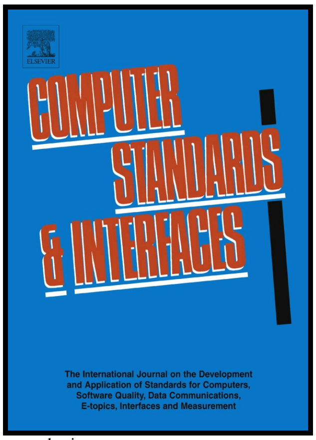
Author’s Accepted Manuscript



www.elsevier.com

PREFer: a Prescription-based Food recommender system

Devis Bianchini, Valeria De Antonellis, Nicola De Franceschi, Michele Melchiori

PII: S0920-5489(16)30130-1
DOI: http://dx.doi.org/10.1016/j.csi.2016.10.010
Reference: CSI3156

To appear in: *Computer Standards & Interfaces*

Received date: 6 April 2016
Revised date: 11 October 2016
Accepted date: 13 October 2016

Cite this article as: Devis Bianchini, Valeria De Antonellis, Nicola De Francesch and Michele Melchiori, PREFer: a Prescription-based Food recommende system, *Computer Standards & Interfaces* http://dx.doi.org/10.1016/j.csi.2016.10.010

PREFer: a Prescription-based Food recommender system

Devis Bianchini[1], Valeria De Antonellis, Nicola De Franceschi, Michele Melchiori*

University of Brescia, Department of Information Engineering, via Branze 38, 25123 Brescia (Italy)

Abstract

In literature, several researches on food recommendation and automatic menu generation have been proposed, taking into account different aspects, such as personal and cultural preferences, health and religion constraints, menu composition and recipe co-occurrence. However, recommending recipes and menus, which not only meet the user's preferences, but are also compliant with best food habits, is still an open issue. This paper presents the PREFer food recommender system, apt to provide users with personalized and healthy menus, taking into account both user's short/long-term preferences and medical prescriptions. Prescriptions classify the *ideal user's nutrition behaviour* from the health point of view, with constraints imposed by the specific user's phenotype. In fact, major novel contribution of the proposed system is the capability of associating users' profiles with prescription types to educate users to improve their behaviour in selecting food. The recommended menus are generated through three steps. First, according to user's request, recipes are selected by content-based filtering, based on comparisons among features used to annotate both users' profiles and recipes. Second, candidate menus are generated using the selected recipes. Third, menus are refined and ranked taking into account also prescriptions. The PREFer system has been developed within a regional

*Email: {devis.bianchini,valeria.deantonellis,nicola.defranceschi,michele.melchiori}@unibs.it.
[1]Corresponding author: Devis Bianchini.

project, related to the main topics "Feeding the Planet, Energy for Life" of the 2015 World Exposition (EXPO2015, Milan, Italy), where the University of Brescia aimed at promoting healthy behavioural habits in nutrition.

Keywords: Food recommender system, semantic annotation, recipe recommendation, menu generation, food prescriptions

1. Introduction

Recommender systems find information of interests, properly customized according to the users' own preferences [1]. This is valid also for specific application domains, such as health and nutrition, where any choice made upon automatically provided recommendations might have an impact on users' health and wellness. Several researches on food recommendation and automatic menu generation have been proposed (e.g., [2, 3, 4]), taking into account different aspects, such as personal and cultural preferences, health and religion constraints, menu composition and recipe co-occurrence. However, recommending recipes and menus, which not only meet the user's preferences, but are also compliant with best food habits, is still an open issue. Let's consider, for example, Jasmine, who is looking for recipe suggestions to have lunch during her working hours. Jasmine is registered to a food recommender system and has an associated profile. She prefers to have pasta and meat during meals. She suffers from long-term diseases, such as diabetes and high-blood-pressure, therefore white meat should be more advisable. She belongs to the Islamic religion, so recommendations about any food containing pork are not acceptable, since this food is prohibited to Muslims. A food recommender system would be very useful, not only for the high number of available recipes to be suggested[1], but also because it is really difficult to manually check all the constraints (e.g., religion constraints) and preferences to generate proper menus. In literature, several food

[1] The `http://allrecipes.com` web site lists thousands of recipes; for example, just considering appetizers, we can found more than 7,700 choices (`http://allrecipes.com/recipes/appetizers-and-snacks/`)

recommender systems are built around the concept of *feature* [2, 5, 6]. Features are used: (i) to annotate recipes (e.g., through the food category, such as meat or seafood, or through the cooking style, e.g., Chinese cuisine or Italian cuisine); (ii) to represent *short-term preferences*, explicitly specified by Jasmine in a request for suggestions issued to the system; (iii) to model *long-term preferences*, extracted from the history of past choices made by Jasmine [7]. Feature-based matching between profiles and recipes is the basis of content-based filtering for food recommendation. However, some Jasmine's preferences (e.g., having pasta and meat during meals, all the days throughout the week) may contrast with best habits, according to up-to-date nutrition guidelines [8]. To address this issue, in this paper we propose *PREFer* (**P**rescriptions for **RE**commending **F**ood), a food recommender system apt to provide users with personalized and healthy menus, taking into account both user's short/long-term preferences and medical prescriptions. Prescriptions are automatically generated to be compliant with prescription types, that classify the *ideal user's nutrition behaviour* from the health point of view with constraints imposed by the specific user's phenotype [9]. In particular, a prescription type is generated for each class of users' profiles and is composed of a set of features, that should be searched within available recipes in order to meet desiderable nutrition habits, and a set of constraints, that are used to exclude some recipes from suggestions (e.g., specific recipes that have to be excluded due to Jasmine's religion or pathologies). Therefore, the recommended menus are generated through three steps: (i) starting from the user's request, relevant recipes are selected by content-based filtering, based on comparisons among features used to annotate both user's profile and recipes according to constraints contained in the prescription types associated with the user's profile; (ii) candidate menus are generated using the selected recipes; (iii) menus are refined and ranked taking into account in a balanced way past menu choices and prescriptions compliant with prescription types. The major novel contributions of the proposed system are the following: (i) a recommendation method that is *education-oriented*, that is, aims at improving user's nutritional habits (by considering prescription types), balanced with user's preferences (by considering short-term and long-term preferences); (ii) an ontology-based method able to associate prescription types with users' profiles; (iii) an algorithm for menu generation and ranking, to ensure better performances and a higher variety in recommended recipes (variety is one of the highly desiderable features in good nutrition habits [8]).

This paper extends the research presented in [10], where we presented the general idea of Food Recommender System. Specifically, in this paper: (i) we refined the menu generation algorithm, ensuring variety in recommended recipes; (ii) we extended the set of selection features, by including the food seasonality; (iii) we introduced some optimization aspects throughout the recipe recommendation steps, in order to improve approach performances. In line with these improvements, we performed and described an extended experimental evaluation and we expanded the comparison with related work.

The open issues concerning healthy nutrition habits have been investigated within the SMART BREAK project - an acronym for SMART adaptive Bialetti REestorAtion Kit - a regional project related to the main topics "Feeding the Planet, Energy for Life" of the 2015 World Exposition (EXPO2015, Milan, Italy), where the University of Brescia aimed at promoting healthy behavioural habits in nutrition[2]. In this paper, we present our approach to the development of systems for food recommendation, taking into account the educational perspective that this activity might assume.

The paper is organized as follows: in Section 2 we compare our approach with related efforts in literature; Section 3 introduces the SMART BREAK project and sketches the general idea of our approach; Section 4 presents the PREFer reference architecture; in Section 5 and in Section 6 we describe the recommendation model and the recommendation steps based on the model; in Section 7 implementation and evaluation issues are discussed; finally, Section 8 closes the paper.

[2] `http://www.smartbreakproject.it`.

2. Related work

Literature on recommender systems covers several domains and has been developed in parallel with the Web, to properly suggest movies, books, applications, e-learning materials, recipes, etc. (a survey on recommender systems can be found in [1]). Nevertheless, given the number of domains where systems have been applied and their specific features, a cross-domain comparison between recommender systems might be difficult and useless. Therefore, we will focus here on recent approaches on food recommendation domain, relying on the following aspects: (i) *filtering algorithm*, determining how the system uses the dataset of items to be suggested and the user's preferences to produce the recommendations; (ii) *recommendation algorithm*, that is, techniques used for generating the recommendations (e.g., probabilistic approaches, nearest neighbors techniques, fuzzy models, similarity metrics); (iii) the availability of an *implementation* of the system; (iv) the availability of experimental results; (v) if the user can improve his/her nutrition habits by using the system (*educational aspects*). In Table 1 and in the rest of this section we will provide a comparison between available food recommender systems based on these characteristics. Among the considered aspects, filtering algorithm is one of the most important features. Different food recommendation approaches apply different filtering algorithms:

- *collaborative filtering*, based on the discovery of patterns in users' choices; the basic idea is that two users who share the same patterns are more likely to have similar tastes, hence past choices of one user can be used to make recommendations for the others;
- *content-based filtering*, based on the choices made by the user in the past; the system will recommend items similar to the ones already in the user's history;
- *demographic filtering*, in which the user is identified based on demographic information such as age, gender, education, and a proper recommendation is given according to the user's categorization;

- *hybrid filtering*, that is, a combination between collaborative filtering and content-based filtering;
- *context-aware filtering*, that uses information about the user's context (e.g., position, time, other environmental conditions) in order to provide recommendations;
- *semantic-based filtering*, based on the domain knowledge, usually defined with an ontology;
- *user-generated content-based filtering*, based on the analysis of the content the user generates on the Web, for example by extrapolating keywords from user's reviews or comments, in order to make recommendations that take into account the user's interests;
- *knowledge-based filtering*, based on the knowledge about items to be recommended, explicitly represented; in this case, the recommendations to be provided are based on a given context.

Some existing approaches for recommending food and health-related information focus on content-based filtering (considering aspects like personal and cultural preferences, health and religion constraints). In [2] recipes are modeled as complex aggregations of different features, extracted from ingredients, categories, preparation directions, nutrition facts, and authors propose a content-driven matrix factorization approach (CTRMF engine) to face the latent dimensions of recipes, users and their features. Teng et al. [11] apply collaborative filtering for recipe recommendation: recipes taken from the `allrecipes.com` Web site are suggested on the basis of users' ratings and reviews and on the basis of co-occurrences of ingredients used to prepare them. In the paper, an interesting survey is provided on other approaches that consider ingredients, recipe ratings and cooking directions. The same information is used in [12, 13], where content-based, collaborative and hybrid filtering are compared for recipe recommendation purposes.

Recommender system	Filtering algorithm	Recommendation algorithm	Implementation	Experiments	Educational aspects
CTRMF engine [4]	Content-based	Content-driven matrix factorization	N/A	Yes	No
Teng et al. [11]	Collaborative filtering	SVM + Stochastic gradient boosting tree	N/A	Yes	No
Freyne et al. [12], Sobecki et al. [13]	Content-based + Collaborative + Hybrid	N Neighbors	N/A	Yes	No
OSGi [14]	Content-based + Demographic + Ontology + Knowledge-based	Compares habits to the reference table of nutrition	Mobile application	No	No
JADE [15]	Semantic reasoning	N/A	Web application	Yes	No
CarePlan [5]	Semantic representation framework	N/A	service-oriented Web application	No	Yes, but not personalized for user
Buon appetito RS [16]	User generated content + Sentiment Analysis	Fuzzy sets	N/A	Yes	No
King Mongkut's [17]	Semantic based with ontology	N/A	Web application	No	Thailand's ministry of health's guidelines
U-BabSang [18]	Context-aware	Physiological signals and environment data	Smart home simulator display	Yes	Yes
YemekSepti.com [19]	Content-based	Decision vector	Web application	Yes	No
iRid [20]	Collaborative filter (memory based)	K-Nearest Neighbors	Mobile application	No	No
Yelp [21]	Collaborative filter (memory based)	Pearson correlation coefficient	Mobile application	No	No
Personalized Recommendation for Weightlifting [22]	Semantic based with ontology	N/A	Java application	No	Athletes only
PREFer	Semantic based with ontology	Concept-based	Web application	Yes	Yes

Table 1: Comparison matrix of food recommender systems; approaches with similar features have been analyzed in the related work section.

Other approaches combine content-based and demographic filtering techniques with ontology-based and knowledge-based tools to enhance recommendation results [14, 15]. Ontologies are used to model personal and cultural preferences, health and religion constraints, but no educational issues are taken into account. CarePlan [3] is a semantic representation framework for healthcare plans, that mixes the patients' health conditions with personal preferences, but ignores other aspects, such as personalization coming from educational health information, user's culture and religion, that impact on the food choice. In [4] an ontology containing fuzzy sets is used to sort recommended recipes according to prices and users' ratings, in combination with attributes like sex, age, weight, physical activity, used to calculate Basal Metabolic Rate (BMR), Activity Factor (AF) and Body Mass Index (BMI). Authors implement a demographic filtering algorithm, thus providing common suggestions to people with common attributes.

The "Buon appetito" recommender system [16] generates menu suggestions using a user-generated content filter, that exploits the user's preferences from the analysis of the reviews associated with every recipe. It uses fuzzy sets based on the user's sentiment, estimated with the Sentiment Analysis of the sentences written in the reviews. The food recommender system of King Mongkut's Institute [17] provides daily nutritional guidance to users based on their diet, using an ontology containing nutritional values associated with every recipe, taking into account the guidelines of Thailand's ministry of health. u-BabSang [18] is a Korean food recommender system that uses a context-aware approach based on the users' profile, real time physiological signals and environmental information, with the main focus on the educational aspect. YemekSepti.com [19] uses a content-based filtering based on the rating that the users gave to the recipes in the past. The system recommends similar recipes and the user can rate the received suggestions, so that the system can learn and give better recommendations in the future. There is no educational purpose. iRIS [20] uses ratings by all the users to identify similarity in their tastes, and provides suggestions using a memory-based collaborative filtering on the whole set of users, implemented as a K-Nearest Neighbors approach. Similarly, Yelp [21] uses a collaborative filter based on the Peason correlation coefficient in order to find users with similar tastes. The system described in [22] is designed for weightlifting athletes, in order to provide them suggestions for a correct diet during their training. An ontology has been used, to model the domain of nutritional needs based on the kind of training the user is following.

This variety of approaches demonstrates that users' profiling, in particular for sectors and domains such as the food and health recommendation, is mainly addressed in an ad-hoc manner, not necessarily aiming at providing some educational effects on the users. The papers described in [23, 24] highlighted this open issue. In particular, [23] presents preliminary research on how to detect bad and correct food habits by analyzing users' ratings on `allrecipes.com`, while in [24] authors discovered that online food consumption and production are highly sensitive in time. Although these approaches do not provide a recommender system, their research could be fruitfully exploited for food recommendation purposes. Other works [25, 26, 27] explicitly address the issue of promoting healthful choices, by suggesting recipes to users based on their past food selections and nutrition intakes. We will propose a step forward compared to these approaches, promoting healthy behaviour through prescription types, that are based not only on nutrition intakes, but are specifically modeled considering phenotypes, that classify ideal users' nutrition behaviour. The PREFer food recommender system takes into account both user's short/long-term preferences and medical prescriptions, that classify the ideal user's nutrition behaviour from the health point of view. A proper domain ontology is used to model such knowledge and is used with content-based filtering for enhanced food recommendation.

3. Motivating project

Nutrition and its effects on health, on the metabolism and on performance at school, during sports or at work, are becoming more and more important.

Indeed, people are calling on nutritionists and food experts. Motivations are not only related to the treatment of eating disorders or to aesthetic reasons, but people also want to ensure their well-being and healthiness. Addressing these requirements also means facing the increasing amount of data, services and applications freely available on the Web. The promotion of healthy behavioural habits in nutrition is the main goal of the SMART BREAK project. In a more general way, the project is finalized to the improvement of the nutrition habits of the individuals, thanks to an increased awareness of food and drinks ingestion.

Within the project, a Web and mobile application for supporting food diary compilation has been developed, where expertise usually brought by medical doctors and nutritionists in traditional food diary compilation is properly embedded. The introduction of a food diary to be filled in a semi-automatic way is one of the most innovative aspects of the project. The diary can be filled through a smartphone APP, to track the ingestion of food and drinks in every moment and everywhere. In this way, the user will have at disposal information about nutrition facts on food and drinks assumed in every moment. In this context, we investigated the application of recommender systems for food recommendation, taking into account the educational perspective that this activity might assume. In particular, we implemented the PREFer recommender system, for which we present here the general idea. In the following sections we will provide technical details of the PREFer food recommender system and we will discuss its implementation and evaluation.

3.1. General idea

Let's consider the running example described in the introduction, where Jasmine is looking for a personalized menu for her meals. Some important aspects should be considered here. First, recipes can be combined into different menus, but not all aggregations are acceptable. Specific combinations of recipes might be due to particular *menu templates* (e.g., appetizer, first course, second course, dessert), according to user's preferences. Second, recommendations might be given according to *prescriptions*, that should be used as first-class citizens in

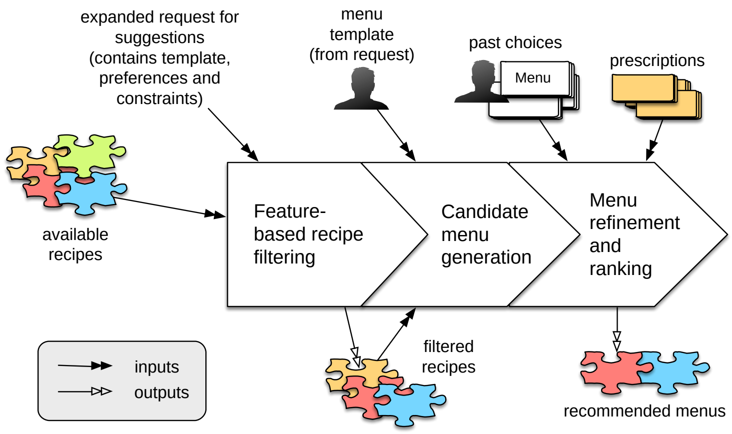


Figure 1: The three steps of food recommendation approach driven by user's preferences and prescriptions.

recommending recipes to users who present particular profiles. Third, although prescriptions can be used to improve the users' nutrition habits, they cannot be imposed to users, disregarding their own preferences. Prescriptions should *gradually* move users' choices towards more healthy recipes. We will meet this requirement by taking into account both user's short/long-term preferences and prescriptions to recommend recipes.

The approach followed here for food recommendation is articulated over a set of steps, that are summarized in Figure 1:

- *feature-based recipe filtering* - the overall set of recipes is properly pruned taking into account the menu template, a set of constraints (e.g., related to the religion or chronic diseases of the user) and the features used to model long-term and short-term preferences;
- *candidate menu generation* - candidate menus that are compliant with the template are generated, only considering the recipes that have not been filtered out in the previous step;
- *menu refinement and ranking* - candidate menus are properly ranked according to their average similarity with the past menu choices made by the

user, who is looking for suggestions, and with the prescriptions advertised for that user.

4. The PREFer Reference Architecture

The functional architecture of PREFer food recommender system, implemented as a Web application, is shown in Figure 2. The *PREFer Web Interface* guides the user through the registration process, the menu recommendation, the publication of new recipes. Registration of a user (*User registration* module) is performed by answering a food frequency questionnaire (FFQ), that is used to collect information about the frequency with which the user assumes well-defined food categories (e.g., meat or seafood) in order to identify his/her phenotype [9]. Phenotypes (and constraints associated with the user's profile, such as the ones related to religion or particular pathologies) are used to prepare suggested prescriptions. The description of this task, that in the SMART BREAK project is performed by medical doctors from the University of Brescia, is out of the scope of this paper. To give an idea, medical doctors can be supported in the identification of phenotypes and have a web interface at their disposal (*Prescription Manager*) to prepare and insert prescription types for a given class of users' profiles, to denote the ideal nutrition behaviour for that class from the health point of view. The way prescriptions and class of profiles are modeled within the PREFer recommender system will be detailed in the next section. Prescriptions that are compliant with prescription types are then automatically generated starting from the dataset of available recipes. *Menu recommendation* module implements the recommendation process described in Section 6. It relies on *similarity modules* (i.e., *Concept Similarity Evaluator* and *Aggregation Similarity Evaluator*), that we exploited for feature-based recipe filtering and menu ranking.

The menu recommendation module supports the user throughout the formulation of the request for suggestions through a proper wizard. The wizard guides the user to choose the features to be specified in the request. Fea-

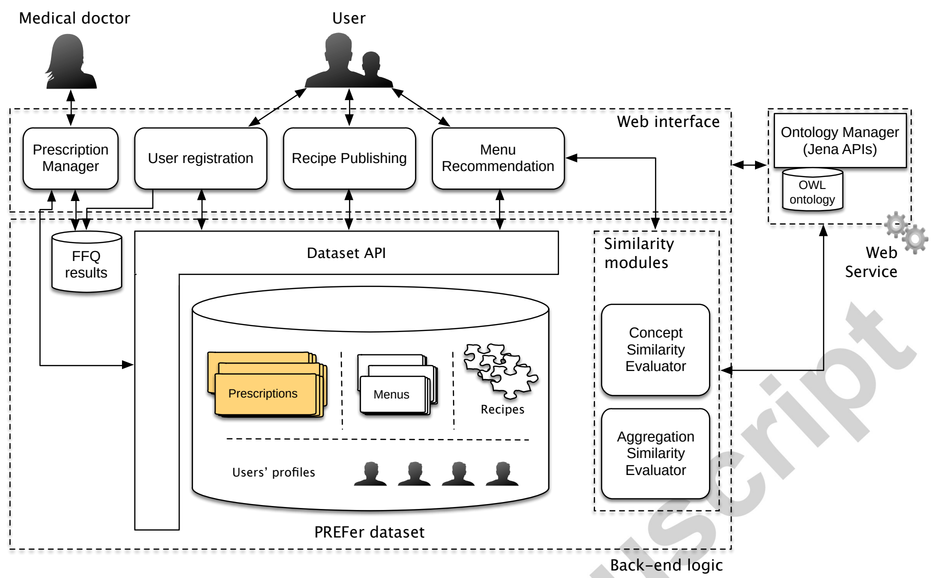


Figure 2: The functional architecture of the PREFer food recommender system.

tures are defined within a reference ontology, formalized using Web Ontology Language. The adopted ontology extends the `food.owl` ontology[3] with the concepts of `CookingStyle` (e.g., Asian cuisine), `Health&CulturalConstraint` such as `Religion` (e.g., Islamic) and `Pathology` (e.g., diabetes, high-blood pressure), `CourseType` (e.g., appetizer, first course, second course, fruits, dessert), `PrescriptionType` and `Phenotype`. In our approach, semantic disambiguation techniques we applied in other Semantic Web applications [28] have been included in the *Ontology Manager* module. When a new request for suggestions is issued by specifying desired features, like the cooking style or the course type, a text field is provided to enter such preferences. As the user inputs the characters of the feature name he/she wants to use for specifying the request, the Ontology Manager module provides an automatic completion mechanism based on the set of concepts contained within the ontology. Starting from the name specified by the user, the module queries the ontology, retrieves the concepts with the speci-

[3]`http://krono.act.uji.es/Links/ontologies/food.owl/view`.

fied names and/or other concepts related to the specified one through semantic relationships, in order to enable the user to explore the ontology and refine the annotation. The PREFer Web interface also provides a wizard to publish new recipes (*Recipe Publishing* module), according to the model described in the next section. Users can annotate recipes with features being supported by the same wizard described for the menu recommendation. In this case, other candidate concepts are also provided through semantic disambiguation techniques, according to the string distance between concept names and terms contained in the recipe name and description.

Following the rationale presented in [29], we distinguish between the ontology and the recipe and menu recipes, menus and prescriptions database. The ontology contains the concepts used to model the domain knowledge, while the database contains specific instances of recipes, menus and prescriptions compliant with prescription types associated with the users' profile, annotated with concepts taken from the ontology. The conceptual model of the database and ontological concepts and their use for food recommendation will be detailed in the next sections. The Ontology Manager implementation is based on the Jena reasoner.

5. Recommendation model

The PREFer food recommender system relies on a recommendation model that is organized over three levels, namely recipes, aggregations (either prescriptions or menus) and user's profile. The PREFer conceptual data model is shown in Figure 3 and will be detailed in the following.

Recipes. Recipes represent the most fine-grained items to be recommended. A recipe is stored in the database and is modeled as $r_i = \langle R_i, n_i, \mathcal{C}_i \rangle$ ($\forall i = 1, \ldots N_R$), where: R_i is the unique identifier of the recipe (we denote with $\mathcal{R}$ the overall set of $N_{\mathcal{R}}$ recipes available within the dataset); n_i is the name of the recipe; $\mathcal{C}_i$ is a set of concepts taken from the ontology, used to characterize the

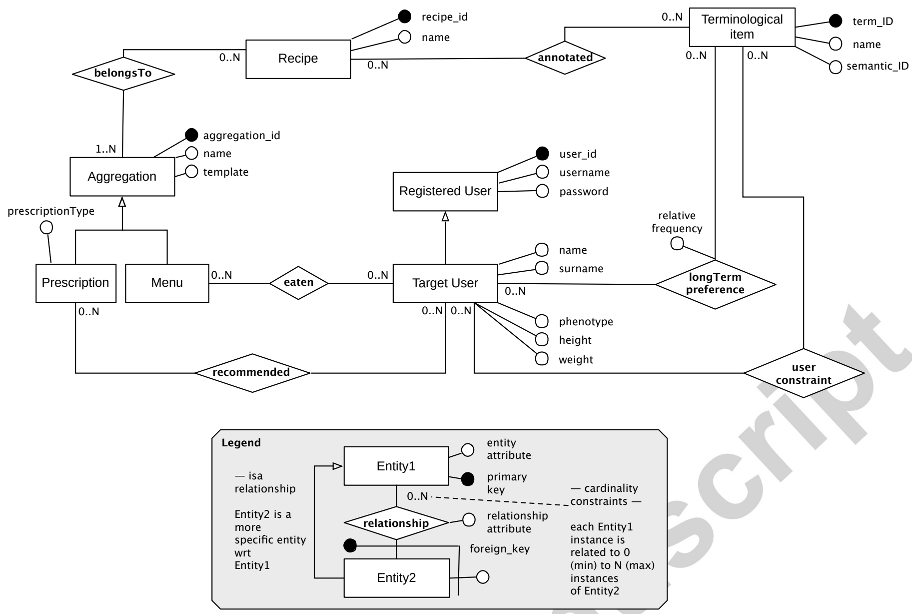


Figure 3: The PREFer conceptual data model (ER notation).

recipe. In our approach each recipe can be classified through the `CourseType`, the `CookingStyle`, the `FoodCategory`, and their sub-concepts shown in Figure 4, where only the TBox of the ontology is shown. Semantic annotation of recipes is saved within the PREFer data model through the `annotated` relationship between `Recipe` and `Terminological_item` entities. The latter refers to a specific ontological concept by means of the `Semantic_ID` attribute, that is an attribute that uniquely identifies the concept within the ontology through the concept URL (`ontology_URL#concept_name`). In Figure 5 eight different recipes are depicted, with concepts extracted from the ontology, where, for clarity purposes, only concept names are shown. Recipes are further featured by their seasonality. This is implemented through a seasonality table, where rows correspond to food categories (`FoodCategory` concept in the ontology) and columns correspond to months. Within each cell of the table, a food category is labeled as "coming in" or "at its best" according to the seasonality of that category. Table 2 reports a portion of seasonality table. Seasonality will be exploited to

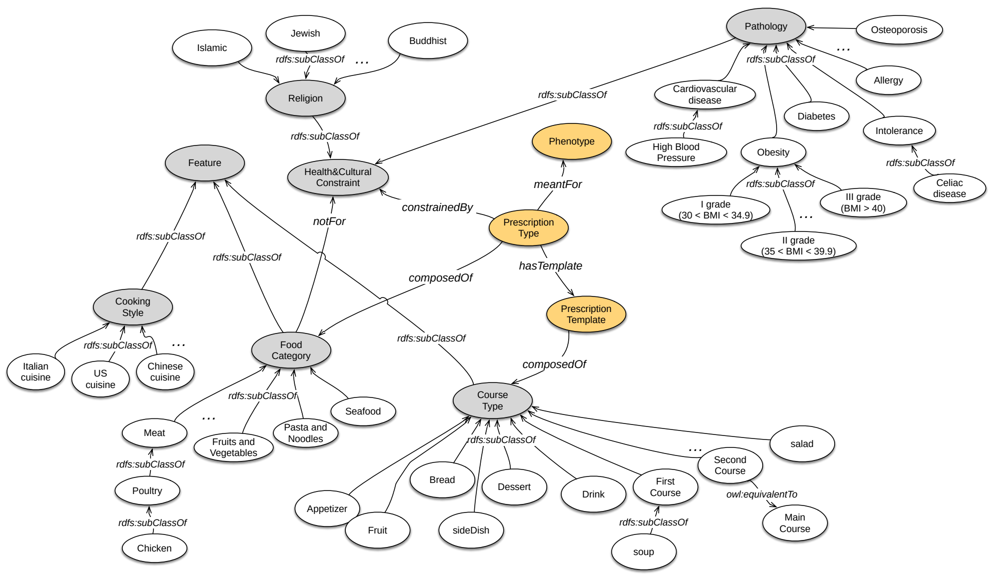


Figure 4: Main concepts of the ontology adopted for food recommendation (TBox).

refine the recommendation process as shown in the next sections.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Chicken	●	●	●	●	●	●	●	●	●	●	●	●
Pork	●	●	●	●	●	●	●	●	●	●	●	●
Eggplant						●	●	●	●	●		
Basil						◑	●	●				
Potatoes	◑	◑	◑	◑	◑	●	●	●	◑	◑	◑	◑

Table 2: A sample portion of the seasonality table for some food categories taken from the running example (◑ stays for ”coming in”, ● stays for ”at its best”).

Menus and Prescriptions. Recipes are aggregated to be proposed in a combined way. In the context of our food recommendation approach, we distinguish two kinds of aggregations: (a) available *menus*, that is, combinations of recipes chosen in the past by the users (these menus are used to extract long-term preferences of the users, exploiting them during the recommendation phase, see below for details); (b) *prescriptions*, that is, proper combinations of recipes that

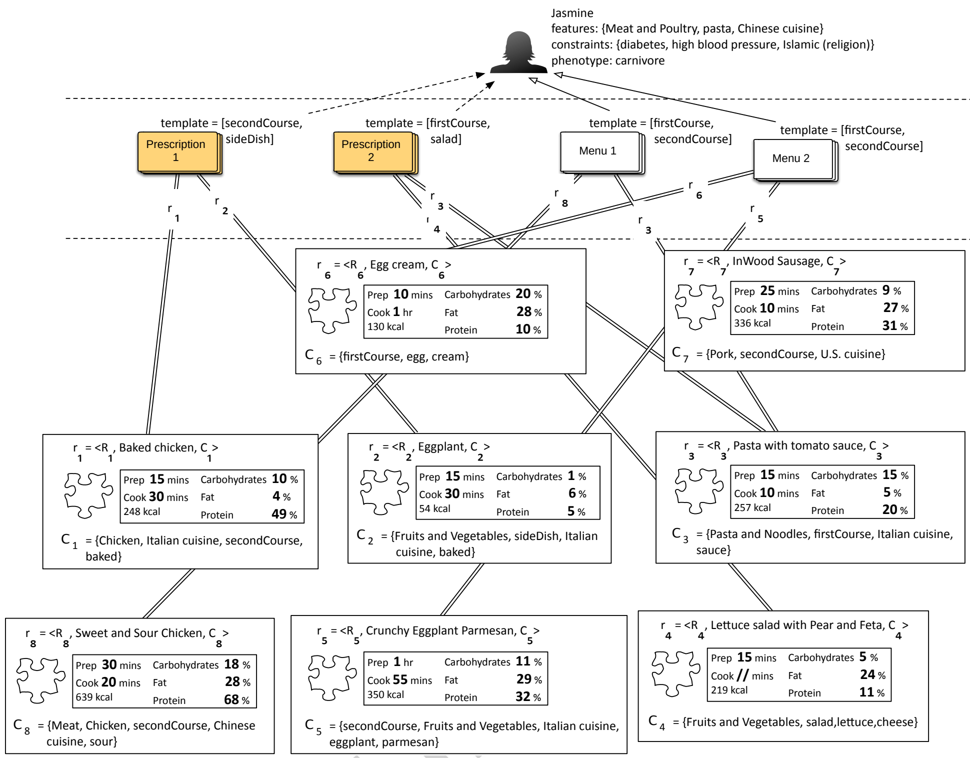


Figure 5: Recipes, menus and prescriptions of the running example.

are advisable for specific kinds of users, compliant with prescription types, in order to take into account health and cultural constraints.

Formally, we define a *menu* $m_j \in \mathcal{M}$ as $m_j = \langle n_{m_j}, \mathcal{R}[m_j], \tau_{m_j} \rangle$, where: $\mathcal{M}$ denotes the overall set of menus; n_{m_j} is the name of the menu; $\mathcal{R}[m_j] \subseteq \mathcal{R}$ is the set of recipes aggregated in m_j; τ_{m_j} is the template of the menu, expressed in terms of `CourseType` concept and corresponding sub-concepts (e.g., `Appetizer`, `Fruit`, `sideDish`, etc., see Figure 4). Examples of templates may be [`Appetizer`, `FirstCourse`, `SecondCourse`, `Dessert`] or [`FirstCourse`, `Fruit`]. Templates play an important role for the formulation of the request for suggestions (Section 6.1) and to speed up the generation of the recommendation output (Section 6.3).

We define a *prescription* $p_h \in \mathcal{P}$ as $p_h = \langle n_{p_h}, \mathcal{R}[p_h], \tau_{p_h}, t_{p_h} \rangle$, where: $\mathcal{P}$ denotes the overall set of prescriptions in the database; n_{p_h} is the prescription name; $\mathcal{R}[p_h]\subseteq\mathcal{R}$ is the set of recipes aggregated in p_h; τ_{p_h} is the template associated with the prescription (described like τ_{m_j} above); t_{p_h} is a reference to the `PrescriptionType` defined within the ontology.

The way prescriptions are associated with users depends on the features used to describe users' profiles. In our food recommendation approach, given a phenotype, one or more instances of `PrescriptionTypes` are advisable for it, as specified in the ABox of the ontology. In the ontology (TBox), a `PrescriptionType` is described by: (i) a `PrescriptionTemplate`, that is, a set of `CourseTypes`; (ii) a set of `FoodCategories`, that are used to identify candidate recipes for instances of prescriptions that are compliant with the particular `PrescriptionType`; (iii) a set of `HealthAndCulturalConstraints`, used to filter out not advisable recipes for that prescription type. Phenotype and health and cultural constraints are used to model a class of users' profiles for which the `PrescriptionType` is meant. For example, Jasmine's features identify her phenotype as *carnivore* (Figure 5). Within the ontology, one of the prescriptions advisable for this phenotype can be modelled through a template containing a second course based on chicken, fruits and vegetables. Therefore, `prescription1` in Figure 5, composed of recipes r_1 and r_2, is compliant with these constraints. Prescriptions compliant with a given `PrescriptionType` are automatically generated within the database, given the available recipes.

Users' Profiles. Users are profiled according to their preferences and past menu choices, that are collected to represent the history of menu selections made by the user in the past. Formally, we define the profile $p(u)$ of a user $u\in\mathcal{U}$ as $p(u) = \langle ID_u, \mathcal{N}[u], \mathcal{C}[u], \mathcal{M}[u], \mathcal{P}[u]\rangle$, where: $\mathcal{U}$ denotes the overall set of users; ID_u is used to identify the user u; $\mathcal{N}[u]$ is the set of constraints (e.g., based o religion or specific pathologies) for the user u (`user-constraint` relationship in Figure 3 towards the `terminologicalItem` entity, that refers to constraints defined in the ontology through the `semantic_ID` attribute as well); $\mathcal{C}[u]$ is the set of ontological concepts used to denote long-term preferences of u;

$\mathcal{M}[u]$ is the set of menus chosen by the user in the past, from where long-term preferences are extracted; $\mathcal{P}[u]$ is the set of prescriptions assigned to the user in the system, given his/her phenotype and corresponding prescription types. For what concerns long-term preference extraction, they are collected and updated using traditional techniques from the literature [7]. Each concept $c \in \mathcal{C}[u]$ is weighted with the *relative frequency* $freq(c) \in [0, 1]$, that is computed on a menu basis, i.e., is proportional to the number of times c has been used in one of the recipes of menus in $\mathcal{M}[u]$ with respect to the total number of menus in $\mathcal{M}[u]$ associated with the profile $p(u)$. Less frequent concepts will be considered as less important for identifying candidate recipes.

6. Prescription-based model for menu recommendation

6.1. Formulating a request for suggestions

When Jasmine is looking for menu suggestions, she generates a request $r_r(u)$ formulated as $r_r(u) = \langle \mathcal{C}_r, \tau_r \rangle$, where: $\mathcal{C}_r$ is a set of concepts that represent immediate, *short-term preferences* of Jasmine; τ_r is the menu template Jasmine is searching for. The recommender system takes into account the profile $p(u)$ of the user u (Jasmine), whom the request comes from. To this aim, the request $r_r(u)$ is expanded with the long-term preferences $\mathcal{C}[u]$ and constraints $\mathcal{N}[u]$ that are associated with the Jasmine's profile $p(u)$. We denote with $\widehat{r}_r(u)$ the expanded version of the request, where $\widehat{r}_r(u) = \langle \widehat{\mathcal{C}}_r, \mathcal{N}[u], \tau_r \rangle$. The set $\widehat{\mathcal{C}}_r$ contains both the concepts specified in $\mathcal{C}_r$ (short-term preferences) and the long-term preferences $\mathcal{C}[u]$ within $p(u)$. The set $\mathcal{C}_r$ might also be empty, thus denoting that the system should exclusively rely on the long-term preferences contained within $p(u)$. Each concept $c_r \in \widehat{\mathcal{C}}_r$ is weighted by means of a coefficient $\omega_r \in [0, 1]$ such that:

$$\omega_r = \begin{cases} 1 & if\ c_r \in \mathcal{C}_r \\ freq(c_r) \in [0, 1] & otherwise \end{cases} \tag{1}$$

The value of ω_r means that a concept explicitly specified in the request $r_r(u)$ will be considered the most for identifying candidate recipes. The term $freq(c_r)$ is the relative frequency of concept c_r among all the concepts that annotate the recipes contained in the profile $p(u)$ as explained in the previous section. If a concept c_r is present in $\mathcal{C}_r$, then $\omega_r = 1$. Relative frequencies are computed on a menu basis, since recipes are recommended only within aggregations, represented as menus. The set $\mathcal{N}[u]$ of constraints is used to filter out forbidden recipes according to religion and health constraints, as explained in Section 6.2. In future versions of the $PREFer$ system we aim at integrating here further collaborative filtering and demographic filtering recommendation techniques [1].

Example. Let's consider the recipes and Jasmine's profile of the running example (Figure 5). Let's imagine that Jasmine submits the following request: $r_r(u) = \langle\{\texttt{poultry}, \texttt{baked}\}, [\texttt{FirstCourse}, \texttt{SecondCourse}]\rangle$. The request is issued to search for menus and recipes containing `baked poultry`, according to [`FirstCourse`, `SecondCourse`] template τ_r. The following expanded version of the request is generated (relative frequency values are specified among parenthesis):

$$\widehat{\mathcal{C}_r} = \{\texttt{poultry}(1.0), \texttt{meat}(0.5), \texttt{chicken}(0.5), \texttt{SecondCourse}(1.0), \texttt{Chinesecuisine}(0.5),$$
$$\texttt{PastaandNoodles}(0.5), \texttt{FirstCourse}(0.5), \texttt{Italiancuisine}(1.0), \texttt{FruitsandVegetables}(0.5),$$
$$\texttt{baked}(1.0), \texttt{sour}(0.5), \texttt{cream}(0.5), \texttt{egg}(0.5), \texttt{eggplant}(0.5), \texttt{parmesan}(0.5), \texttt{sauce}(0.5)\}$$
$$\mathcal{N}[u] = \{\texttt{diabetes}, \texttt{highBloodPressure}, \texttt{Islamic}\}$$
$$\tau_r = [\texttt{FirstCourse},\ \texttt{SecondCourse}]$$

In particular, for what concerns the formulation of $\widehat{\mathcal{C}_r}$:

- `poultry` and `baked` are associated with a frequency value equal to 1.0 since these two terms are contained in $\widehat{\mathcal{C}_r}$ within the request $r_r(u)$ (short-term preferences);
- `meat` is associated with a frequency value equal to 0.5 because it is present in 50% of menus associated with the Jasmine's profile (see Figure 5, where

`meat` is associated with recipe r_8, that in turn composes `Menu#1`); the same consideration holds for `chicken`, `Chinesecuisine` and `sour` (recipe r_8), `PastaandNoodles` and `sauce` (recipe r_3), `FruitsandVegetables`, `eggplant` and `parmesan` (recipe r_5), `cream` and `egg` (recipe r_6);

- `SecondCourse` is associated with a frequency value equal to 1.0 because it is included in all menus associated with the Jasmine's profile (see Figure 5, where `SecondCourse` is associated with recipe r_6, that in turn composes `Menu#1`, and with recipe r_3, that in turn composes `Menu#2`); the same consideration holds for `FirstCourse` (recipes r_3 and r_6), `Italiancuisine` (recipe r_3 and r_5).

6.2. Feature-based recipe filtering

The input of this step is the set $\mathcal{R}$ of all the available recipes and the expanded request $\widehat{r}_r(u)$. First, τ_r element specified in the request is considered. Recipes such that their `CourseType` is not included within τ_r will not pass the feature-based filtering step. To speed up the pre-selection based on `CourseType`, recipes are stored in the underlying dataset indexed with respect to the `CourseType` feature. With reference to the running example, only the r_1, r_3, r_5, r_6, r_7 and r_8 recipes will be further considered, according to the menu template $\tau_r = [$`FirstCourse`, `SecondCourse`$]$. In fact, r_2 and r_4 are not considered since they are not a first or a second course. Therefore, $\mathcal{N}[u]$ element specified in the expanded request is considered. Recipes that do not respect these constraints must be excluded before any other kind of comparison. These constraints are defined within the ABox of the domain ontology and are instances of the `notFor` semantic relationship between `FoodCategory` and `ReligionAndCulturalConstraints` concepts (Figure 4). For example, let's consider constraints imposed by Jasmine's religion, diabetes and high-blood pressure, in the running example. The Islamic religion within the Jasmine's profile excludes all recipes that are annotated with `pork`. Therefore, recipe r_7 will not be considered for the next recommendation steps.

After τ_r and constraints have been used to pre-select recipes, the filtering based on remaining features is applied, according to the *concept-based similarity*, and further refined using seasonality. This metric is computed as follows.

Concept-based similarity. The similarity of a recipe $r_i = \langle R_i, n_i, \mathcal{C}_i \rangle$ with respect to the request $\widehat{r}_r(u) = \langle \widehat{\mathcal{C}}_r, \mathcal{N}[u], \tau_r \rangle$ taking into account concepts in $\mathcal{C}_i$ and $\widehat{\mathcal{C}}_r$, denoted with $Sim(\widehat{r}_r, r_i) \in [0, 1]$, is computed as:

$$Sim(\widehat{r}_r, r_i) = \frac{2 \cdot \sum_{c_r, c_i} \omega_r \cdot ConceptSim(c_r, c_i)}{|\mathcal{C}_i|} \in [0, 1] \quad (2)$$

where c_r ranges over the set $\widehat{\mathcal{C}}_r$, c_i ranges over the set $\mathcal{C}_i$, $|\mathcal{C}_i|$ denotes the number of concepts in the set $\mathcal{C}_i$, ω_r denotes the weight of concept $c_r \in \widehat{\mathcal{C}}_r$, according to Equation (1), to take into account both short-term and long-term preferences. $ConceptSim(c_r, c_i)$ represents the *concept similarity* between c_r and c_i:

$$ConceptSim(c_r, c_i) = \frac{2 \cdot |c_r \cap c_i|}{|c_r| + |c_i|} \in [0, 1] \quad (3)$$

In Equation (3), we consider the two concepts c_r and c_i as more similar as the number of recipes that have been annotated with both the concepts, denoted with $|c_r \cap c_i|$, increases with respect to the overall number of recipes annotated with c_r (denoted with $|c_r|$) and with c_i (denoted with $|c_i|$). The domain ontology is considered in this case as well: in fact, given two concepts c_i and c_j such that $c_i \sqsubseteq c_j$ (c_i is `subclassOf` c_j), due to the semantics of the `subclassOf` relationship, all recipes annotated with c_i are considered as annotated with c_j as well. For example, $|\texttt{chicken}| = |\{r_1, r_8\}| = 2$, $|\texttt{poultry}| = |\{r_1, r_8\}| = 2$, since $\texttt{chicken} \sqsubseteq \texttt{poultry}$, $|\texttt{chicken} \cap \texttt{poultry}| = |\{r_1, r_8\}| = 2$, therefore $ConceptSim(\texttt{chicken}, \texttt{poultry}) = 1.0$.

Pairs of concepts to be considered in the $ConceptSim(c_r, c_i)$ computation are selected according to a maximization function, that relies on the assignment in bipartite graphs and ensures that each concept in $\mathcal{C}_i$ participates in at most one pair with one of the concepts in $\widehat{\mathcal{C}}_r$ and the pairs are selected in order to maximize the overall $Sim(\widehat{r}_r, r_i)$. The rationale behind Equation (2) is that the closer $Sim()$ to 1.0, the more concepts in $\mathcal{C}_i$ are similar to one of the concepts in $\widehat{\mathcal{C}}_r$.

6.3. Candidate menu generation

In this step, recipes are aggregated into menus that must be compliant with the template τ_r specified in the request $\widehat{r}_r(u)$. This significantly reduces the number of menus to be generated: in fact, a candidate menu can not contain two recipes r_i and r_j annotated with the same `CourseType`. If we consider, for example, m `CourseTypes`, with an average number of n candidate recipes for each `CourseType`, the number of candidate menus without considering the constraint imposed by the menu template would be equal to $f_1(n,m) = \frac{(n \cdot m!)}{m!(n \cdot m - m)!}$ (since we have $n{\cdot}m$ elements and among them m elements must be selected to be composed, without repetitions). In our approach, the number of possible menus is equal to $f_2(n,m) = n^m$ (cartesian product). Nevertheless, to obtain the optional ranking of menus according to the relevance $Sim(\widehat{r}_r, r_i)$ of component recipes, a brute force procedure should be applied, where all candidate menus are generated before being sorted. Formally, using the brute force procedure: (a) all possible n^m menus are generated; (b) for each menu $m_j = \langle n_{m_j}, \mathcal{R}[m_j], \tau_{m_j} \rangle$, that is compliant with menu template τ_r (i.e., $\tau_{m_j} = \tau_r$), the overall similarity of its recipes with respect to the request $\widehat{r}_r(u)$ is computed, that is, $\Gamma^{m_j} = \sum_{i=1}^{k} Sim(\widehat{r}_r, r_i)$, where k is the number of recipes r_i in m_j ($|\mathcal{R}[m_j]| = k$); (c) all n^m menus are ranked according to the decreasing Γ^{m_j} value. Although the reduced number of possible configurations $f_2(n,m) << f_1(n,m)$, the adoption of a brute force procedure presents a time complexity that is exponential with respect to the number of recipes, as demonstrated by experimental results (see Section 7). On the other hand, a random selection of recipes would be faster, but less effective in terms of relevance, since more similar recipes are not necessarily proposed first. To this aim, we propose a new algorithm, illustrated in Figure 6 with the help of the running example, denoted in the following as the *PREFer menu generation algorithm*.

Firstly, recipes are grouped with respect to their `CourseType` (considering

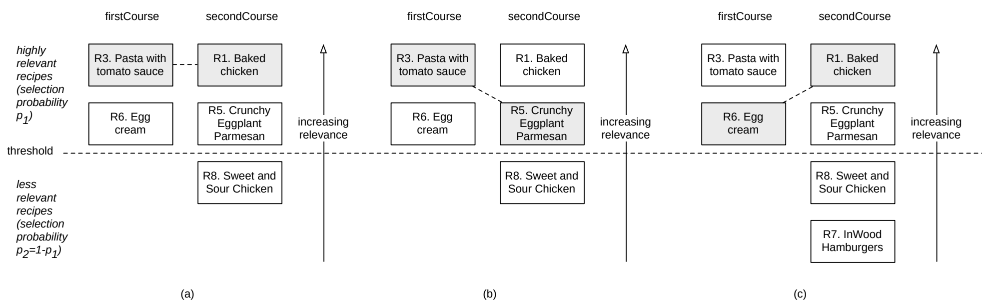


Figure 6: The candidate menu generation step.

the template τ_r) according to a decreasing similarity $Sim(\widehat{r}_r, r_i)$. Seasonality is exploited here to further refine ranking. A recipe r_i is ranked better than r_j (denoted with $r_i \prec r_j$) if: (a) $Sim(\widehat{r}_r, r_i) > Sim(\widehat{r}_r, r_j)$ or (b) $Sim(\widehat{r}_r, r_i) = Sim(\widehat{r}_r, r_j)$ and $\sigma(r_i) > \sigma(r_j)$, where $\sigma(r_i)$ is a weight assigned to r_i according to the seasonality of its food category. In particular: (i) $\sigma(r_i) = 1.0$ if each food category that annotates r_i is "at its best" for the current month in the seasonality table; (ii) $\sigma(r_i) = 0.5$ if at least one food category that annotates r_i is "coming in" for the current month in the seasonality table, while all the remaining food categories are "at their best"; (iii) $\sigma(r_i) = 0.0$ otherwise.

In Figure 6, the considered menu template is τ_r=[`FirstCourse`, `SecondCourse`]. A similarity threshold $\gamma \in [0, 1]$ is used to divide the recipes in each group between *highly relevant* and *less relevant* recipes. The threshold is set by the user, nevertheless we performed experiments (see Section 7) to provide an optimal setup of γ. At each selection step, a candidate menu $m_j = \langle n_{m_j}, \mathcal{R}[m_j], \tau_{m_j} \rangle$ is generated, where: (i) m_j is compliant with menu template τ_r; for each `CourseType`, r_i is chosen from highly relevant recipes with a probability $p_1 \in [0, 1]$ and from less relevant recipes with a probability $p_2 = 1 - p_1$ (with $p_2 << p_1$). The choice of p_1 and p_2 values has been experimentally determined (and will be described in Section 7).

This algorithm is sub-optimal with respect to the brute force solution. Nevertheless, experiments described in Section 7 show that the execution time is reduced by randomly selecting the recipes according to the PREFer menu generation algorithm. Furthermore, experiments demonstrate how the right setup of p_1 and p_2 probability values and of the threshold γ might also contribute to reduce the difference between the menus obtained using the brute force algorithm and the PREFer approach in terms of overall similarity $\Gamma^{m_j} = \sum_{i=1}^{k} Sim(\widehat{r}_r, r_i)$, for each candidate menu $m_j = \langle n_{m_j}, \mathcal{R}[m_j], \tau_{m_j} \rangle$ and $|\mathcal{R}[m_j]| = k$. Additionally, in the performed experiments, using the PREFer menu generation algorithm, we observed a better diversification of recipes in the generated menus. This leads to a broader set of choices for the user, compared to the brute force algorithm. In fact, in the latter case, the first menus proposed to the user contain highly overlapping recipes. As underlined in nutrition research [8], variety is one of the highly desiderable features in good nutrition habits. The final ranking of menus is adjusted using prescriptions as explained in the next step.

6.4. Menu refinement and ranking

Menus that have been generated in the previous step are ranked according to their similarity with respect to: (i) past menu choices made by the user u who is issuing the request for suggestions, represented by the set $\mathcal{M}[u]$; (ii) prescriptions prepared for the user u according to his/her profile, represented by the set $\mathcal{P}[u]$. Since both menus and prescriptions are characterized by their sets of recipes, the building block in this step is the similarity measure between items aggregations (*item aggregation similarity*), that is computed as follows:

$$Sim_{agg}(\mathcal{R}[a_i], \mathcal{R}[a_j]) = \frac{2 \cdot \sum_{r_i, r_j} Sim(r_i, r_j)}{|\mathcal{R}[a_i]| + |\mathcal{R}[a_j]|} \in [0, 1] \tag{4}$$

where a_i and a_j represent the two compared aggregations (menus or prescriptions), r_i (resp., r_j) is a recipe included within a_i (resp., within a_j), $|\mathcal{R}[a_i]|$ (resp., $|\mathcal{R}[a_j]|$) denotes the number of recipes included within a_i (resp., within a_j). $Sim(r_i, r_j) \in [0, 1]$ is a variant of concept-based similarity defined in Equation (2), where ω_r is not considered because we are not comparing a request for suggestions ($\omega_r = 1$). The rationale behind $Sim_{agg}()$ computation is the same as the one of the concept-based similarity: we consider two aggregations as more similar as the number of similar recipes in the two aggregations increases. Pairs of recipes to be considered in the $Sim(r_i, r_j)$ computation are selected according to a maximization function, that relies on the assignment in bipartite graphs and ensures that each recipe in $\mathcal{R}[a_i]$ participates in at most one pair with one of the recipes in $\mathcal{R}[a_j]$ and the pairs are selected in order to maximize the overall $Sim_{agg}()$.

The final ranking of a generated menu $m_k = \langle n_{m_k}, \mathcal{R}[m_k], \tau_{m_k}\rangle \in \mathcal{M}^*$, recommended to the user u who issued a request for suggestions, is performed through a ranking function $\rho : \mathcal{M}^* \mapsto [0, 1]$, computed as follows:

$$\rho(m_k) = \omega_m \cdot \frac{\sum_{m_j \in \mathcal{M}[u]} Sim_{agg}(\mathcal{R}[m_k], \mathcal{R}[m_j])}{|\mathcal{M}[u]|} + \omega_s \cdot \frac{\sum_{p_h \in \mathcal{P}[u]} Sim_{agg}(\mathcal{R}[m_k], \mathcal{R}[p_h])}{|\mathcal{P}[u]|} \tag{5}$$

where $m_j = \langle n_{m_j}, \mathcal{R}[m_j], \tau_{m_j}\rangle$ is a menu associated with user's profile; $p_h = \langle n_{p_h}, \mathcal{R}[p_h], \tau_{p_h}, t_{p_h}\rangle$ is a prescription generated for the user; $\omega_m, \omega_p \in [0, 1]$, $\omega_m + \omega_p = 1.0$, are weights used to balance the impact of past menu choices and prescriptions on the ranking of recommended menus. We have chosen $\omega_m < \omega_p$ (i.e., $\omega_m \cong 0.4$ and $\omega_p \cong 0.6$) in order to stimulate users on improving their food and nutrition habits, without recommending menus and recipes that are too much distant from users' preferences. This is one of the most innovative aspects of our approach compared with recent food recommendation literature (see Section 2). Setup of ω_m and ω_p has been performed on a training set of users (see Section 7).

7. Implementation and experiments

The PREFer system has been implemented as a J2EE Web application. MySQL has been chosen as the DBMS, with the relational database abstracted by means of a Data Access Object using the framework Hibernate. The dataset of recipes has been obtained from `BigOven.com` website, from where we downloaded a set of around 350k recipes using the APIs provided by the website owner. Experiments have been run on an Intel laptop, with 2.53 GHz Core 2

	50 recipes	90 recipes	110 recipes
$Test_1$	684 ms	3306 ms	NA
$Test_2$	875 ms	4367 ms	NA
$Test_3$	1272 ms	4122 ms	NA
$Test_4$	1002 ms	4067 ms	NA

Table 3: Response times of the brute force algorithm for menu generation.

	50 recipes	90 recipes	110 recipes
$Test_1$	2.7 ms	13.22 ms	18.35 ms
$Test_2$	3.24 ms	17.89 ms	25.24 ms
$Test_3$	5 ms	15.66 ms	22.88 ms
$Test_4$	4 ms	15.46 ms	22.57 ms

Table 4: Response times of the brute force algorithm for menu generation (with optimization).

CPU, 2GB RAM and Linux OS, in order to evaluate: (a) the average response time of menu recommendation process, (b) its quality in terms of concept-based relevance and variety of suggestions.

Response time. We implemented the two algorithms for menu generation, the brute force one, which constitutes the optimal solution (since it generates all

	50 recipes	90 recipes	110 recipes
$Test_1$	2 ms	3 ms	9 ms
$Test_2$	1 ms	1 ms	8 ms
$Test_3$	1 ms	1 ms	8 ms
$Test_4$	1 ms	1 ms	8 ms

Table 5: Response times of the PREFer menu generation algorithm.

the menu before sorting them based on the relevance with the user's request), and the PREFer one. To test response times, we issued four kinds of requests, namely: (i) $\texttt{Test}_1$, that is, user's requests containing 6 short-term preferences, 5 prescriptions per user and 5 menus in the user's history; (ii) $\texttt{Test}_2$, that is, user's requests containing 6 short-term preferences, 10 prescriptions per user and 10 menus in the user's history; (iii) $\texttt{Test}_3$, that is, user's requests containing 6 short-term preferences, 20 prescriptions per user and 10 menus in the user's history; (iv) $\texttt{Test}_4$, that is, user's requests containing 6 short-term preferences, 10 prescriptions per user and 20 menus in the user's history. For each kind, we issued ten requests. Experimental results are obtained as average values over the set of requests. Experiments on response times have been carried on by setting up $p_1 = 0.8$, $p_2 = 0.2$ and $\gamma = 0.7$. Tables 3, 4 and 5 show the response times of the two approaches with an increasing number of recipes. As expected, even with a modest number of recipes, the brute force algorithm has a higher response time, and it isn't even able to complete the generation of the menus with a dataset of 110 recipes, that is a very small set if compared with the original set of 350k recipes (Table 3). To overtake these limitations, we introduced a set of optimizations throughout the recommendation steps, that is:

1. the relative frequency of concept c_r among all the concepts that annotate the recipes contained in the user's profile ($freq(c_r)$) can be computed offline for each user and updated every time the users' profile changes; nevertheless, during the formulation of a request for suggestions (Section 6.1), the value of ω_r cannot be computed offline, since it is based on $freq(c_r)$, but also on the set of concepts $\mathcal{C}_r$ specified by the user within the request (see Equation (1));
2. during the feature-based recipe filtering step (Section 6.2), the value of $ConceptSim(c_r, c_i)$ is computed offline for each pair of concepts $\langle c_r, c_i \rangle$, since all concepts are always defined in the reference ontology; therefore, a matrix is pre-computed, containing $ConceptSim()$ values for all pos-

sible pairs of ontological concepts; such a matrix is updated when new concepts or recipes are added; nevertheless, in the same recommendation step, $Sim(\widehat{r}_r, r_i)$ cannot be pre-computed, since it depends on the ω_r value, and therefore comments introduced in the previous point hold;

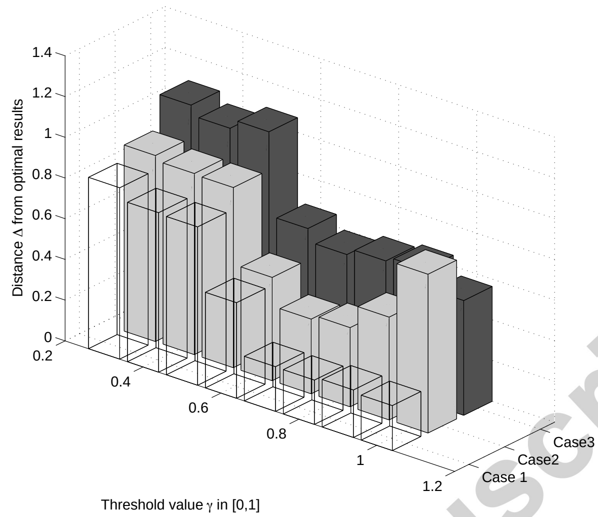
3. on the other hand, during the menu refinement and ranking (Section 6.4), $Sim(r_i, r_j)$ is computed offline for each pair $\langle r_i, r_j \rangle$, since it does not depends on ω_r.

Results are shown in Table 4, where increment in performances is evident. However, the PREFer menu generation algorithm further improves performances in terms of response time, due to the steps that are still (partially or totally) computed online. Furthermore, the PREFer menu generation algorithm also presents an additional advantage with respect to the brute force one, that is, it enables a better diversification of recipes in the generated menus, as explained in the following.

Quality of the PREFer menu generation algorithm. In order to evaluate the quality of the PREFer menu generation algorithm, we used the following metric Δ, that quantifies the distance of the PREFer algorithm from the brute force solution, that, as explained above, can be considered as the optimal solution:

$$\Delta = \frac{\sum_{j=0}^{N} |\Gamma^{m_j} - \widehat{\Gamma^{m_j}}|}{N} \in [0, T] \tag{6}$$

where: Γ^{m_j} (resp., $\widehat{\Gamma^{m_j}}$) is the sum of concept-based similarity values for the recipes contained in the menu m_j proposed in the j-th position by the PREFer menu generation algorithm (resp., by the brute force menu generation algorithm); T corresponds to the number of recipes in the considered menu template (in fact, since concept-based similarity belongs to the $[0, 1]$ range, if we consider a menu template of T elements, the sum of concept-based similarity belongs to the $[0, T]$ range). Since the brute force algorithm is the optimal case, then $\widehat{\Gamma^{m_j}}$ represents the upper bound. Therefore, the more Γ^{m_j} is close to $\widehat{\Gamma^{m_j}}$, the higher



Similarity threshold γ	Case 1	Case 2	Case 3
0.3	0.84166	0.91388	1.07506
0.4	0.78333	0.88965	1.02502
0.5	0.77777	0.88475	1.07223
0.6	0.46944	0.50750	0.66119
0.7	**0.21944**	0.36666	0.59722
0.8	**0.21758**	0.38888	0.63053
0.9	0.23333	0.50457	0.65277
1.0	0.22080	0.77942	0.56388

Figure 7: Quality of the menu generation algorithm: distance Δ from the brute force algorithm in terms of overall concept-based recipe similarity with respect to the request $\widehat{r}_r(u)$.

the quality of PREFer menu generation algorithm. The value of Δ is computed for the first N menus (in the experiments, we set $N = 20$). Using this metric we were able to obtain the best configuration of the parameters that allowed PREFer algorithm to be as closer as possible to the optimal one. We considered three different configurations of the PREFer algorithm:

- case 1: the algorithm has the 80% of probability of extracting a highly

PREFer algorithm		Brute force algorithm	
Γ^{m_j}	Generated menus	$\widehat{\Gamma^{m_j}}$	Generated menus
4.0	225, 440, 239, 16	4.0	225, 104, 215, 16
4.0	225, 251, 239, 419	4.0	225, 104, 215, 36
4.0	250, 364, 215, 362	4.0	225, 104, 215, 40
4.0	225, 296, 215, 275	4.0	225, 104, 215, 41
4.0	225, 218, 243, 275	4.0	225, 104, 215, 128
4.0	225, 135, 243, 170	4.0	225, 104, 215, 129
4.0	441, 218, 215, 129	4.0	225, 104, 215, 145
4.0	225, 364, 215, 227	4.0	225, 104, 215, 167
4.0	486, 135, 215, 316	4.0	225, 104, 215, 170
3.5454545	250, 364, 239, 17	4.0	225, 104, 215, 171
3.5454545	250, 218, 325, 362	4.0	225, 104, 215, 217
3.5454545	143, 218, 215, 145	4.0	225, 104, 215, 227
3.5454545	225, 104, 239, 1	4.0	225, 104, 215, 268
3.5454545	250, 104, 215, 1	4.0	225, 104, 215, 274
3.5	225, 369, 215, 419	4.0	225, 104, 215, 275
3.5	441, 179, 215, 36	4.0	225, 104, 215, 300
3.5	250, 371, 215, 362	4.0	225, 104, 215, 316
3.5	187, 440, 215, 421	4.0	225, 104, 215, 362
3.5	441, 218, 314, 420	4.0	225, 104, 215, 395
3.5	344, 251, 239, 40	4.0	225, 104, 215, 419

Table 6: Variety of recipes in the menu generation outputs for the PREFer and the brute force algorithms (request issued with a template τ_r=[`Appetizer`, `FirstCourse`, `SecondCourse`, `Dessert`]).

relevant recipe (p_1) and 20% of probability of extracting one of the low relevant recipes (p_2);

- case 2: the algorithm has the 70% of probability of extracting a highly relevant recipe (p_1) and 30% of probability of extracting one of the low relevant recipes (p_2);
- case 3: the algorithm has the 50% of probability of extracting a highly relevant recipe (p_1) and 50% of probability of extracting one of the low relevant recipes (p_2).

Moreover, we varied the relevance threshold γ within the range $[0.3 - 1.0]$. Results are shown in Figure 7.

The obtained results show best results with $p_1 = 0.8$, $p_2 = 0.2$ (case 1) and with a similarity threshold $\gamma \sim 0.7 - 0.8$. Moreover, we noticed how the application of the PREFer menu generation algorithm enables a highest variety, since with the brute force solution the first menus proposed to the user present highly overlapping component recipes. For instance, considering a template τ_r=[`Appetizer`, `FirstCourse`, `SecondCourse`, `Dessert`], the outputs of PREFer menu generation algorithm and of the brute force one are shown in Table 6 (where each number corresponds to the `recipe_id` of a recipe within the dataset). In the brute force algorithm output, menus with all the same recipes but the dessert are presented first, since using highly overlapping recipes the overall similarity with respect to the user's request increases, in contrast with the PREFer algorithm, where the random part of the selection process decreases the likelihood of having highly overlapping recipes in the top-ranked menus. On the other hand, quality of ranking as shown in Figure 7 demonstrates how the PREFer algorithm is comparable with the brute force procedure, that in terms of ranking quality can be considered as the optimal case.

8. Conclusions

In this paper, we presented *PREFer*, a menu generation system that uses a recipe dataset and annotations to recommend menus according to user's preferences. Compared to recent literature on food recommender systems, *PREFer* is apt to provide users with personalized and healthy menus, taking into account both user's short/long-term preferences and medical prescriptions, that classify the ideal user's nutrition behaviour from the health point of view. The recommendation method is based on an innovative algorithm for the generation and ranking of menus, that ensures better performances and a higher variety in recommended recipes (variety is one of the highly desiderable features in good nutrition habits [8]). In this paper we presented the PREFer food recommender system and we discussed its implementation and evaluation. In particular, we performed experiments to evaluate: (a) the average response time of menu recommendation process, (b) its quality in terms of concept-based relevance and variety of suggestions. In the current implementation of our system we can register the number of times an user chooses a recommended menu. A team of medical doctors from the University of Brescia, who collaborated with us within the Smart Break project, is defining a methodology for monitoring the state of PREFer users over time.

The application of Semantic Web technologies will enable further extensions of the PREFer food recommender system, namely: (a) in cases of violation of health and cultural constraints due to specific ingredients, similar recipes will be suggested, where only the prohibited ingredients are substituted; (b) a semi-automatic functionality for supporting medical doctors in the generation of prescription types will be provided as well. Moreover, we aim at further improving variety of recommended menus including novelty of recipes among the aspects that are considered for recipe ranking during the candidate menu generation. Finally, further experiments will be carried on in order to check the effectiveness of the proposed approach in improving nutritional habits and lifestyles.

Acknowledgements

This work has been partially funded by the SMART BREAK project (SMART adaptive Bialetti REestorAtion Kit), financed by Lombardy region, Italy, research area on Smart Cities and Communities.

References

[1] J. Bobadilla, F. Ortega, A. Hernando, A. Gutièrrez, Recommender systems survey, Knowledge-Based Systems 46 (2013) 109–132.

[2] C. Lin, T. Kuo, S. Lin, A Content-Based Matrix Factorization Model for Recipe Recommendation, in: Opinion Diffusion and Analysis on Social Networks (PAKDD 2014), 2014, pp. 560–571.

[3] S. Abidi, H. Chen, Adaptable personalized care planning via a semantic web framework, in: 20th Int. Conference of the European Federation for Medical Informatics, 2006.

[4] N. Maneerat, R. Varakulsiripunth, D. Fudholi, Ontology-based nutrition planning assistance system for health control, Asean Engineering Journal 1 (2) (2013) 28–41.

[5] O. Suominen, E. Hyvonen, K. Viljanen, E. Hukka, HealthFinland - A national semantic publishing network and portal for health information, in: Web Semantics: Science, Services and Agents on the World Wide Web, 2009, pp. 287–297.

[6] D. Dominguez, F. Grasso, T. Miller, R. Serafin, PIPS. An Integrated Environment for Health Care Delivery and Healthy Lifestyle Support, in: 4th Workshop on Agent applied in Healhcare ECAI2006, 2006.

[7] S. Gauch, M. Speretta, A. Chandramouli, A. Micarelli, User profiles for personalized information access, in: The Adaptive Web, Methods and Strategies of Web Personalization, 2007, pp. 54–89.

[8] S. Zeisel, H. Freake, D. Bauman, D. Bier, D. Burrin, J. German, S. Klein, G. Marquis, J. Milner, G. Pelto, K. Rasmussen, The nutritional phenotype in the age of metabolomics, Journal of Nutrition 135 (7) (2005) 1613–1616.

[9] T. Rankinen, C. Bouchard, Genetics of food intake and eating behavior phenotypes in humans, Annual Review of Nutrition 26 (2006) 413–434.

[10] D. Bianchini, V. D. Antonellis, M. Melchiori, A Food Recommendation System Based on Semantic Annotations and Reference Prescriptions, in: Advances in Conceptual Modeling, Vol. LNCS 9382, 2015, pp. 134–143.

[11] C.Teng, Y.Lin, L.A.Adamic, Recipe recommendation using ingredient networks, in: Proc. of the 4th Annual ACM Web Science Conference, 2011, pp. 298–307.

[12] J. Freyne, S. Berkovsky, Recommending Food: Reasoning on Recipes and Ingredients, in: User Modeling, Adaptation, and Personalization, Vol. LNCS 6095, 2010, pp. 381–386.

[13] J. Sobecki, E. Babiak, M. Slanina, Application of hybrid recommendation in web-based cooking assistant, in: Proc. of the Conference on Knowledge-Based Intelligent Information and Engineering Systems, 2006.

[14] J. Kim, K. Chung, Ontology-based healthcare context information model to implement ubiquitous environment, in: Multimedia Tools and Applications, LNCS, Vol. 71, 2014, pp. 873–888.

[15] A. Al Nazer, T. Helmy, M. Al Mulhem, User's Profile Ontology-Based Semantic Framework for Personalized Food and Nutrition Recommendation, Procedia Computer Science (32) (2014) 101–108.

[16] M. Trevisiol, L. Chiarandini, R. Baeza-Yates, Buon appetito: recommending personalized menus, in: Proc. of the 25th ACM conference on Hypertext and social media, 2014, pp. 327–329.

[17] N. Suksom, M. Buranarach, Y. Thein, T. Supnithi, P. Netisopakul, A knowledge-based framework for development of personalized food recommender system, in: Proc. of the 5th Int. Conf. on Knowledge, Information and Creativity Support Systems, 2010.

[18] Y. Oh, A. Choi, W. Woo, u-BabSang: a context-aware food recommendation system, Journal of Supercomputing 54 (1) (2010) 61–81.

[19] S. Bundasak, K. Chinnasarn, Dimensionality Reduction on Slope One Predictor in the Food Recommender System, in: 2013 International Computer Science and Engineering Conference (ICSEC), 2013, pp. 84–89.

[20] J. Kim, iRIS: A Large-Scale Food and Recipe Recommendation System Using Spark, in: Data Science and Engineering at scale, 2015.

[21] C. Ha, Yelp Recommendation System Using Advanced Collaborative Filtering, 2015.

[22] P. Tumnark, L. Oliveira, N. Santibutr, Ontology-Based Personalized Dietary Recommendation for Weightlifting, in: Int. Workshop on Computer Science in Sports (IWCSS 2013), 2013, pp. 44–49.

[23] A. Said, A. Bellogín, You are What You Eat! Tracking Health Through Recipe Interactions, in: Proc. of the 6th Workshop on Recommender Systems and the Social Web (RSWeb), 2014.

[24] T. Kusmierczyk, C. Trattner, K. Nørvåg, Temporality in Online Food Recipe Consumption and Production, in: Proc. of the ACM 2015 International Conference on World Wide Web Companion (WWW 2015), 2015.

[25] G. Geleijnse, P. Natchtigall, P. van Kaam, L. Wijgergangs, A personalized recipe advice system to promote healthful choices, in: In IUI, ACM, 2011, pp. 437–438.

[26] F. Kamieth, A. Braun, C. Schlehuber, Adaptive implicit interaction for healthy nutrition and food intake supervision, in: Human-Computer In-

teraction. Towards Mobile and Intelligent Interaction Environments, 2011, pp. 205–212.

[27] J. Hsiao, H. Chang, SmartDiet: A Personal Diet Consultant for Healthy Meal Planning, in: Proc. of IEEE 23rd International Symposium on Computer-Based Medical Systems (CBMS), 2010.

[28] D. Bianchini, V. De Antonellis, M. Melchiori, A Linked Data Perspective for Effective Exploration of Web APIs Repositories, in: Proc. of the 13th International Conference on Web Engineering (ICWE), Vol. LNCS 7977, 2013, pp. 506–509.

[29] P. Garcia, E. Mena, J. Bermùdez, Some Common Pitfalls in the Design of Ontology-driven Information Systems, in: Int. Conference on Knowledge Engineering and Ontology Development (KEOD09), 2009, pp. 468–471.