Explainable AI Meets Persuasiveness: Translating Reasoning Results Into Behavioral Change Advice

Mauro Dragoni

Fondazione Bruno Kessler, Trento, Italy

Ivan Donadello

Fondazione Bruno Kessler, Trento, Italy

Claudio Eccher

Fondazione Bruno Kessler, Trento, Italy

Abstract

Explainable AI aims at building intelligent systems that are able to provide a clear, and human understandable, justification of their decisions. This holds for both rule-based and data-driven methods. In management of chronic diseases, the users of such systems are patients that follow strict dietary rules to manage such diseases. After receiving the input of the intake food, the system performs reasoning to understand whether the users follow an unhealthy behavior. Successively, the system has to communicate the results in a clear and effective way, that is, the output message has to persuade users to follow the right dietary rules. In this paper, we address the main challenges to build such systems: i) the natural language generation of messages that explain the reasoner inconsistency; and, ii) the effectiveness of such messages at persuading the users. Results prove that the persuasive explanations are able to reduce the unhealthy users' behaviors.

Keywords: Explainable AI, Explainable Reasoning, Natural Language Generation, mHealth, Ontologies

---

Email addresses: dragoni@fbk.eu (Mauro Dragoni), donadello@fbk.eu (Ivan Donadello), cleccher@fbk.eu (Claudio Eccher)

1. Introduction

Explainable Artificial Intelligence (XAI) aims at explaining the algorithmic decisions of AI solutions with non-technical terms in order to make these decisions trusted and easily understandable by humans [1]. This is of great interest for both Machine Learning (ML) methods and symbolic reasoning in rule engines. The explanation of a reasoning process can be very difficult, especially when a system is based on a set of complex logical axioms whose logical inferences are performed with, for example, tableau algorithms [2]. Indeed, inconsistencies in logical axioms may be not well understood by users if the system limits to just report the violated axioms. Indeed, users are generally skilled to understand neither formal languages nor the behavior of a whole system. This is crucial for some applications, such as a power plant system where a warning message to the user must be clear and concise to avoid catastrophic consequences.

An interesting domain for XAI is healthcare, in particular the management of chronic diseases such as heart disease, cancer and diabetes. These are responsible for approximately 70% of deaths in Europe and U.S. each year and they account for about 75% of the health spending[1]. Such chronic diseases can be largely preventable by eating healthy, exercising regularly, avoiding smoking, and receiving preventive services. Prevention would help people stay healthy, avoid or delay the onset of diseases, and keep diseases they already have far from becoming worse or debilitating; it would also help people lead productive lives and reduce the costs of public health. The challenges of an explainable system that supports users in following an healthy behavior are: (i) the ability of providing a clear and comprehensible message regarding user's behavior, and (ii) the effectiveness of the message to *persuade* the user at adopting a healthy lifestyle. This is fundamental as often people do not know the importance of following diet rules, hence they may not be sufficiently motivated to adopt healthy behaviors. Differently from the case of the power system, here the message must be persuasive and personalized in order to keep people engaged in using the system.

[1] http://www.who.int/nmh/publications/ncd_report_full_en.pdf

In this paper we present a XAI system based on logical reasoning that supports the monitoring of users' behaviors and persuades them to follow healthy lifestyles [2]. The concepts and rules of healthy behaviors are formalized as a TBox of the HeLiS ontology [3]. This ontology is one of the most updated conceptual models formalizing dietary and physical activity domains. The axioms in HeLiS encode the Mediterranean diet rules that can be associated with user profiles. The user data about her/his dietary behavior are acquired through a user's dietary diary with the help of a smartphone application. This information populates the HeLiS ABox with logical individuals. A reasoner module (Section 4) combines knowledge and user's data (TBox and ABox) to infer the user behavior and generates inconsistencies if the user does not follow the rules of a healthy lifestyle. Once an inconsistency, i.e., an unhealthy user behavior, is detected the system shows the user a natural language message explaining the wrong behavior and its consequences. This translation from a logic language to plain text comprehensible by humans leverages a computational persuasion framework [4] and Natural Language Generation (NLG) techniques [5]. The latter exploit dynamic and smart templates that can be adapted to every persuasion strategy. The proposed system has been integrated into the HORUS.AI platform [6] and it has been validated with a mobile application within the pilot project *Key To Health* run by our institution. Results compare the persuasive explanations with simple notifications of inconsistencies and show that the former are able to support users in improving their adherence to dietary rules. To the best of our knowledge this is the first work that joins reasoning explanations with persuasive messages.

The rest of the paper follows with Section 2 that provides a state-of-the-art of techniques for generating explanations from reasoning inferences and of AI methods for supporting behavior change. Section 3 describes the main concepts of HeLiS used in our explainable and persuasive system. Section 4 shows the reasoning process that checks if a user follows a healthy dietary behavior. Section 5 describes the developed

[2]This work is compliant with good research practice standards. More details at:
http://ec.europa.eu/research/participants/data/ref/fp7/89888/ethics-for-researchers_en.pdf
http://www.who.int/medicines/areas/quality_safety/safety_efficacy/gcp1.pdf

template system for the automatic generation of natural language persuasive explanations. Section 6 presents the *Key To Health* project in which we deployed the system, whereas Section 7 shows its evaluation. Section 8 concludes the paper.

2. Related Work

XAI generally relates to strategies able to provide human-understandable descriptions of learning algorithms usually perceived as black boxes by users [1] in order to make them transparent, interpretable, and comprehensible. This research direction has been widely explored in the last years [7], but most of the contributions focused only on the analysis of how learning models (a.k.a. black boxes) work. This is a limited view of the topic since there is a school of thought arguing that an effective explainability of learning models cannot be achieved without the use of domain knowledge since data analysis alone is not enough for achieving a full-fledged explainable system [8]. This statement has been further discussed recently by asserting that the key for designing a completely explainable AI system is the integration of Semantic Web technologies [9, 10]. Semantic Web technologies enabling the the design of strategies for providing explanations in natural language [11, 12] where explanations are provided through textual rule-like notation. NLG strategies have been designed also for generating natural language text from triples [13] and for translating SPARQL queries into a natural language form understandable by non-experts [14]. Here, we focused on applying XAI to explain the results of inference processes. Our aim is to generate natural language explanations of logic inferences for supporting end-users in understanding the behavioral change recommendations provided by intelligent systems. For this reason, in the following, we compare our work with XAI methods for reasoning results and AI systems for behavior change. We do not present works of NLG as we use a standard technique based on templates [15].

2.1. XAI Methods for Reasoning Systems

The explanation of the logical reasoning in an ontology is implemented with two two orthogonal approaches: *justifications* and *proofs*. The former computes the minimal subset of the ontology axioms that logically entails an axiom. The latter computes also all the inference steps [16].

One of the first user studies dealing with explanations for entailments of OWL ontologies was performed by [17]. The study investigated the effectiveness of different types of explanation for explaining unsatisfiable classes in OWL ontologies. The authors found that the subjects receiving full debugging support performed best (i.e., fastest) on the task, and that users approved of the debugging facilities. Similarly, [18] performed a user study to evaluate an explanation tool, but did not carry out any detailed analysis of the difficulty users had with understanding these explanations. While, [19] presents a user study evaluating a model-exploration based approach to explanation in OWL ontologies. The study revealed that the majority of participants could solve specific tasks with the help of the developed model-exploration tool, however, there was no detailed analysis of which aspects of the ontology the subjects struggled with and how they used the tool. The work [20] presents several algorithms for computing all the justifications of an entailment in a OWL-DL knowledge base. However, nor study or user evaluation is performed to assess the capability of the computed justifications of the logical entailments. The work in [21] focuses on the explanation, through justifications, of the disclosure of personal data to users (patients and staff) of hospitals. This is performed by translating SWRL rules inconsistencies into natural language utterances. This is similar to our proposal, however, this is a preliminary work as no strategy is addressed for selecting a proper inconsistency is discussed. Moreover, the SWRL rules translation is performed axiom by axiom, thus generating a quite long sentence. This could require too much time for reading and understanding. Whereas, our method returns only a single utterance summarizing the whole justification.

Formal proofs are the other form of explanation for logical reasoning. In [22] the authors present an approach to provide proof-based explanations for entailments of the CLASSIC system. The system omits intermediate steps and provides further filtering strategies in order to generate short and simple explanations. The work proposed in [23] first introduced a proof-based explanation system for knowledge bases in the Description Logic ALC [2]. The system generates sequent calculus style proofs using an extension of a tableaux reasoning algorithm, which are then enriched to create natural language explanations. However, there exists no user studies to explore the effectiveness of these proofs. In [24] the authors proposed several (tree, graphical, logical and hybrid) visualizations of defeasible logic proofs and present a user study in order to evaluate the impact of the different approaches. These representations are hard to understand for non-expert users. Indeed, the study is based on participants from a postgraduate course (who have attended a Semantic Web course) and from the research staff. In general, proof algorithms for Description Logic are based on Tableau techniques [2] whereas proof algorithms for other logics are studied in the field of Automated Reasoning [25].

This wide range of approaches to explanation of logical entailments is more focused on the development of efficient algorithms than on effective algorithms for common users. Indeed, all the computed explanations are sets of logical axioms understandable only by expert users. The aim of our work is to provide and effective representation to explanation for all users. This representation is based on the verbalization of the explanation in natural language. This verbalization can be performed by using methods that translate axioms of an OWL ontology in Attempto Controlled English [26, 27] or in standard English [28] with the use of templates. This last work also presents some users' studies on the quality of the generated sentences. However, these works do not handle with the reasoning results (justifications or proofs), indeed, no strategy for selecting and rendering an explanation is studied. Differently, our work addresses some effective (and persuasive) strategies based on behavior change theories, see Section 5.

2.2. AI Methods for Behavior Change Systems

Behavior Change Systems (or Persuasive Systems) are designed to change users' behavior or attitude towards a given argument or goal in healthcare [29]. The literature is huge and can be divided into two approaches: *horizontal* and *vertical* ones. The former are general and study effective methods of generation of persuasive content without any grounding to a specific domain. The vertical approaches present systems that are tailored and effective on specific domains.

The theoretical works in [30, 31, 32] define in details the fundamental concepts and methodologies for building and evaluating Behavior Change Systems across different domains. The main outcome of these works is a persuasive systems design model. If we shift towards the implementation side, the works in [33, 34, 35, 36] define important features that behavior change systems should have such as, objective outcome measurements, self-monitoring, personalized feedback, behavioral goal setting and social support. Focusing on the generation of persuasive content, some seminal works are based on argumentation theories for generating motivational sentences [37, 38]. However, the focus is more on the validity of the generated messages instead of their effectiveness. The work in [39] proposes a persuasive framework that combines NLG strategies with users' information harvested from social-media. In [4] the authors propose a theoretical framework for generating tailored motivational messages for behavior change applications. Basic important properties of these kind of messages (see Section 5) are defined, such as timing, intention, content and representation. Other works [40, 41] combine the affective computing with NLG for generating the motivational messages. A thorough review and classification of available horizontal systems for persuasive content generation through text can be found in [42]. These works are easily adaptable to new domains, however, they usually are theoretical works with no in depth evaluation. Our work includes some of the mentioned key features for behavior change systems with a users' evaluation on the persuasiveness of the generated messages.

Regarding the vertical approaches, the main AI examples of behavior change applications can be found in the fields theory, statistical learning, and recommendation systems. Concerning argumentation theory, in [43] the authors proposed an assistive system for encouraging people at performing physical activity. Argumentation theory enables common-sense reasoning for building arguments. These are defeasible, that is, their validity can be disputed by other arguments. The idea here is to monitor users and to choose the best arguments to propose in order to motivate users' at doing physical exercise. Other works join argumentation with decision theory [44, 45, 46] in order to propose the best set of persuasive arguments based on the users' beliefs. Several strategies are developed to both propagate the beliefs and select the arguments on the basis of an utility function. Differently from our work, here the motivational messages are canned texts with limited possibility of personalized feedback. Argumentation theory is combined with logical reasoning in the PORTIA system [47] in order to select the most appropriate persuasive strategy and combine rational and emotional modes of persuasion. The system is able to simulate the persuasion process of a human through natural language dialogues. However, differently from our work, the PORTIA persuasiveness power is not tested with real users in a living lab.

Statistical learning can be used to learn users' behavioral patterns for self reflection [48, 49, 50] or personal experiments [51] systems. These systems have been proved to be effective in behavioral change. They start by monitoring users' habits (e.g., walking at least 15 minutes per day) and their effects (e.g., level of glycemia in the blood). Then statistical analysis is applied to find correlations between habits and effects. Finally, these correlations are used to generate (with NLG templates) personalized messages for users, such as, "walking every day will decrease your glycemia". This induces the self reflection with a change in the users' behavior. Statistical learning is used in reinforcement learning for computing the goal setting feature [52]. The algorithm learns the best strategy for setting a daily goal (here the number of proposed daily steps) based on the user's behavior given the assigned goals in the previous days. These approaches are orthogonal to our work as they use statistical reasoning instead of logical one. However, our approach relies on a solid reference ontology that can provide a better personalization with recommendation messages for achieving a goal, such as, "Try some fresh fish instead of meat".

Recommendation systems propose personalized food suggestions based on collaborative filtering algorithms [53, 54, 55, 56]. That is, the recommendations are based on a ranking computed on the other users' preferences. This can be an interesting approach as similar users could receive similar motivational messages. However, this knowledge has to be combined with a medical ontology (as our work does) containing guidelines developed by the scientific community.

To conclude, the use of a state-of-the-art reference ontology allows us to generate a fine-grained personalization of the persuasive messages. Moreover, to date there have been no studies dealing directly with the impact on users' behaviors of explanations from OWL ontologies such as the one presented in this paper.

3. The Supporting Knowledge Base: The HeLiS Ontology

The presented explainable reasoning system focuses on the Mediterranean diet, that is effective for the prevention of chronic diseases related to an unhealthy nutrition, such as obesity, diabetes and cardiovascular diseases [57, 58, 59]. Therefore, we adopt HeLiS[3] [3] as knowledge base used in our reasoning system. HeLiS is a state-of-the-art ontology that formalizes the food and recipes composition, the rules of the Mediterranean diet, the physical activities domain and user preferences and habits in order to support the promotion of healthy lifestyles. The relevance of this ontology with respect to the state-of-the-art pivots around the integrated model representing (i) a fine-grained description of food at a level that is not present in the state-of-the-art; (ii) physical activities at the metabolic level enabling the definition of relationships with food entities; (iii) user profiles described through their physical status and possible allergies or diseases. HeLiS has been developed with the support of domain experts starting from the analysis of: (i) the standard literature about the Mediterranean diet pyramid [60] (for example, for formalizing the diet rules); (ii) documents, such as, the archives of the Italian Minister of Agriculture[4], of Healthcare[5] and the archives of the Italian Epidemiological department[6]; (iii) the food Turconi atlas [61]; (iv) the Compendium of Physical Activities[7] and the U.S. Department of Health and Human Services[8] for the physical activity rules. Besides the conceptual model per se, the HeLiS ontology represents a valuable resource for the healthcare domain thanks to the knowledge included into the provided resource. Here we do not present the full modeling process and the content of HeLiS. The reader can refer to [3] for a complete presentation of the ontology engineering process and of the concepts involved in the conceptualization of user's profile and of the monitoring tasks.

The HeLiS ontology is expressed with the OWL 2 RL knowledge representation

[3]http://w3id.org/helis

[4]See the guidelines for a healthy diet at https://www.crea.gov.it/en/home

[5]http://www.salute.gov.it/portale/home.html

[6]http://www.bda-ieo.it/wordpress/en/

[7]https://sites.google.com/site/compendiumofphysicalactivities/home

[8]http://www.hhs.gov/

language: one of the most expressive knowledge representation language that is still decidable. This language allows users to model very expressive rules in a flexible way that can be used to reason on the data provided by users. Formally, HeLiS is the union of two sets of axioms: the TBox (terminological axioms) and the ABox (assertional axioms). The former contains statements about relations between concepts. For example, the composition of a pasta with Carbonara sauce is expressed in the TBox as: $\forall x(PastaCarbonara(x) \wedge hasIngredient(x,y) \rightarrow Pasta(y) \vee ColdCuts(y) \vee Eggs(y) \vee EVO(y) \vee AgedCheese(y))$[9]. The ABox contains factual information, that is, facts about individuals in the ontology. For example, the fact that a user has consumed a meal containing red meat is expressed with $hasConsumed(user1, meal1)$, $contains(meal1, redMeat)$. The axioms in HeLiS are organized in three main groups:

Domain knowledge defines in the TBox the concepts modeling the domain of interest. In particular, the HeLiS ontology contains knowledge about the dietary (i.e., taxonomy of food categories and food compositions) and physical activities (i.e., effort needed for accomplishing a specific activity) domains. Examples are the axioms stating all the nutrients of aged cheese or the consumed calories for a bicycling activity.

Monitoring knowledge defines in the TBox the set of rules enabling the monitoring tasks and the detection of undesired behaviors (hereafter called *violations*). Examples are the rules of the Mediterranean diet. Currently, our system integrates a set of 220 rules describing the Mediterranean diet and a set of 27 rules concerning the physical activity domain for maintaining a healthy lifestyle. The former have been provided by the Italian Ministry of Healthcare, while the latter have been extracted from the guidelines defined by the U.S. Department of Health and Human Services.

User knowledge defines in the ABox the concepts describing user profiles and the data populating the knowledge base, i.e., food consumed and activities performed by users.

[9] This example is in First-Order Logic for the readers not familiar with the OWL 2 RL syntax.

An undesired behavior given by the union of TBox and populated ABox will trigger a logical inconsistency of the monitoring knowledge that has to be explained. For each food category, the HeLiS ontology defines both its associated positive and negative aspects. Such aspects are exploited by the NLG module as described in Section 5.

The ontology contains six root concepts: `Food`, `Nutrient`, `Activity`, `TemporalEvent`, `UserEvent`, and `MonitoringEntity`. In addition, the `User` concept is fundamental for associating users with specific events. Here below there is a brief description of the mentioned concepts. The reader can refer to [3] and [62] for further details.

Food and Nutrient. The `Food` root concept subsumes two macro-groups of entities descending from `BasicFood` and `Recipe` concepts. Instances of the `BasicFood` concept describe food for which micro-information concerning nutrients (carbohydrates, lipids, proteins, minerals, and vitamins) is available, while instances of the `Recipe` concept describe the composition of complex dishes by expressing them as a list of instances of the `RecipeFood` concepts. Beside the food-related concepts, the classification of nutrients is also defined. The `Nutrient` concept subsumes 81 different type of nutrients properly categorized.

Activity. The second groups of entities relates to physical activities. The `PhysicalActivity` concepts subsumes 21 subclasses representing likewise physical activity categories and a total of 856 individuals each one referring to a different kind of activity.

TemporalEvent. The `TemporalEvent` concept defines entities used for representing specific moments or delimited timespans which the data to analyze refer to. These concepts are used in two ways. First, when users provide data packages, these data have to be associated with a specific temporal event. Second, the other descendant of the `TemporalEvent` concept is `Timespan`. Instances of the children of `Timespan` are used for driving the data selection and reasoning operations to a specific portion of data, see Section 4.2.

User. The `User` concept is responsible for managing the instantiation of every single user and works as glue for linking the static knowledge represented by the `MonitoringEntity` individuals and the dynamic knowledge represented by the UserEvent individuals. Each User has two important object properties linking him/her with the data he/she provides: consumed and performed. The consumed object property relates a User with Meal individuals that in turn collect ConsumedFood individuals representing the actual food eaten by a User. Instead, the performed object property links a User with PerformedActivity individuals representing a specific session of an Activity performed by a User.

UserEvent. This concept subsumes the conceptualization of information that a user can provide, e.g., food consumption or performed activities, and also links them with the possible violation that can be generated after their analysis. Concerning the representation of users' activities and personalized information, we modeled the ConsumedFood and the PerformedActivity concepts. Both concepts are used as reification of the fact that a user has consumed a specific quantity of a food or has performed an activity for a specific amount of time. In the first case, every Meal is associated with a list of ConsumedFood through the hasConsumedFood object property. While, in the second case, instances of the PerformedActivity concept associate a user with the amount of time he/she spent in performing a specific activity.

MonitoringEntity. Concepts subsumed by MonitoringEntity are responsible for modeling the knowledge enabling the monitoring of users' behaviors. Here, we can appreciate five concepts: MonitoringRule, Violation, Profile, Goal, and Interval. The MonitoringRule concept provides a structured representation of the parameters inserted by the domain experts for defining how users should behave. Two examples of MonitoringRule instances are shown in Figures 2b and 2a. Violation instances describe the results of the reasoning activities and they can be used by third-party applications. The content of each Violation instance is computed according to the user data that triggered the violation. An example of Violation instance is presented in Figure 4. Each MonitoringRule is linked to at least one Profile concept. A Profile represents a set of rules a User should follow for being compliant with the guidelines provide by the physician. An example of Profile is the Mediterranean diet that contains 220 dietary rules. The Goal concept represents a specific objective, in the context of a `Profile`, that a user is expected to achieve within a given timestamp. In practice, a `Goal` is composed by a subset of the `MonitoringRule` instances linked to a `Profile`. Finally, the `Interval` concept subsumes concepts used for describing interval of values, i.e., `ValueInterval` and `ViolationInterval`. The `ValueInterval` allows the system to specify the validity boundaries (e.g., minimum and maximum value) of a rule. While, the instances of the `ViolationInterval` concept allow the system to transform the percentage representing the difference between expected and observed values into discrete levels representing how much a `MonitoringRule` has been violated.

4. The KB-based Explainable Model

The ontology described in Section 3 is exploited by a SPARQL-based reasoner for detecting undesired situations within users' behaviors, i.e., verifying if user's dietary and activities actions are consistent with the monitoring rules defined by domain experts. When inconsistencies are detected, the knowledge base is populated with individuals of type `Violation` that, in turn, are used by the NLG component for providing feedback to users. Reasoning can be triggered in two ways. First, each time a user adds new data packages, or modifies existing ones, in the knowledge base, the reasoner is invoked for processing the new, or updated, information. Second, at the end of specific timespan, such as the end of a day or of a week, with the aim of checking the overall user's behavior in such timespan. In the latter case, the reasoner works on a collection of data labeled with a timestamp valid within the considered timespan. The integrated reasoner relies on the architecture implemented in RDFpro [63], that is a reasoner providing out-of-the-box OWL 2 RL reasoning and supporting the fixed point evaluation of `INSERT... WHERE...` SPARQL-like entailment rules that leverage the full expressivity of SPARQL (e.g., `GROUP BY` aggregation, negation via `FILTER NOT EXISTS`, derivation of RDF nodes via `BIND`). RDFpro has been chosen for two main reasons. Firstly, the architecture of RDFpro allows the integration of custom methods into reasoning operations (i) for performing mathematical calculations on users' data and (ii) for exploiting real-time information acquired from external sources without materializing them within the knowledge repository. Secondly, as reported in [63], efficient analysis performed on RDFpro demonstrated the suitability of this reasoner with respect to other state-of-the-art reasoners into a real-time scenario. In this work, RDFpro has been adapted and extended in order to better fit with the needs of the proposed solution. The extension consisted in the integration of new methods supporting the aggregation of nutritional information associated with the consumed food provided by the users and the enrichment of the set of SPARQL-based entailment rules in order to enable real-time stream reasoning from sensors data. This way, we were able to support the real-time processing of users' data in a more efficient manner.

We organize the reasoning in two phases: *offline* and *online*. The *offline* phase consists in an one-time processing of the *static* part of the ontology (monitoring rules, food, nutrients, activities) when the system starts. This is performed to materialize the ontology deductive closure, based on OWL 2 RL and some additional pre-processing rules that identify the most specific types of each individual defined in the static part of the HeLiS ontology ABox. Furthermore, this kind of information greatly helps in performing the aggregation operations during the online reasoning phase.

Whereas, during the *online* phase, each time the reasoning is triggered by a user event (e.g., a new data package is entered by a user) or by a time event (e.g., a specific timespan ended), the user data is merged with the closed ontology and the deductive closure of the rules is computed. The resulting `Violation` individuals and their RDF descriptions are then stored back in the knowledge base. The generation of each `Violation` individual is performed in two steps. First, information inferred by aggregating the domain, monitoring, and user knowledge is used for generating the `Violation` individuals. Second, accessory information are integrated into the `Violation` individuals for supporting the NLG component when the explanation concerning the detected violation is generated. Accessory information includes, for example, references to other individuals of the ontology enabling the access to the positive and negative aspects associated with the food category, or the number of times that the specific rule has been violated. This kind of information can be used for deciding the enforcement level of the persuasion contained within the generated feedback.

Figure 1 summarizes the *online* phase of reasoning process which main components and steps are detailed in the following sections. The green path, drawn with a continu-

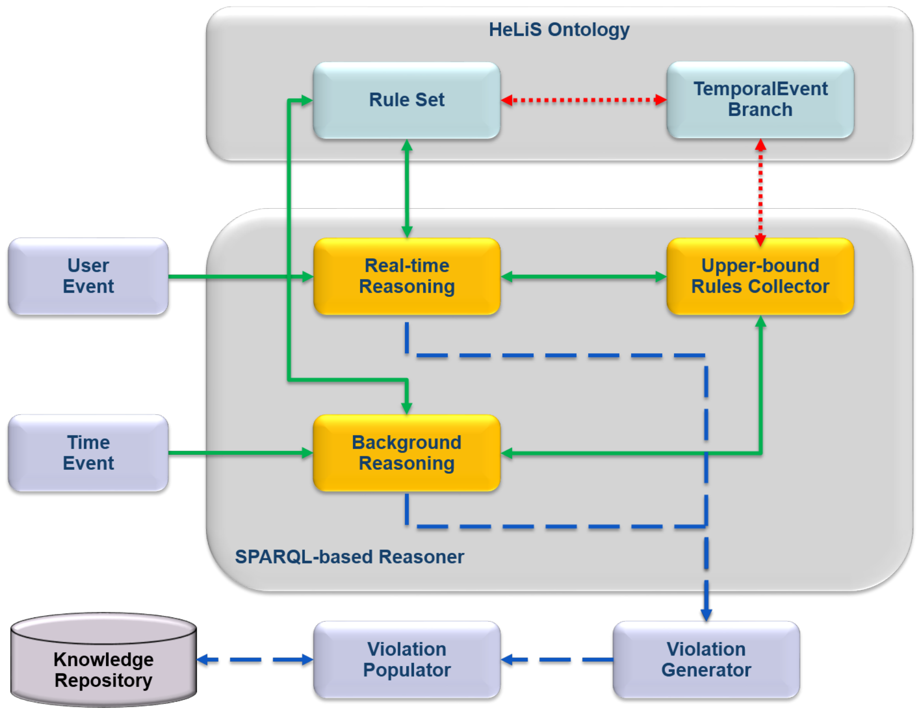

Figure 1: The overall picture of the online reasoning process. Continuous lines represent the first reasoning path followed by the path in dotted lines. The last reasoning step is in dashed lines.

ous line, executed as first step, is in charge of collecting the rules to validate depending on the trigger received by the reasoner (Section 4.1). The red path, drawn with a dotted line, executed as second step, is invoked for collecting rules that can be validated as semantically associated with the ones collected during the green path (Section 4.2). The black path, drawn with a dashed line, executed as third step, generates and populates violations before storing them into the knowledge repository (Section 4.3).

4.1. Rules Collection

As mentioned above, the *online* reasoning phase can be triggered by two kind of events: when a user provides, or updates, a data package or when a specific timespan (modeled as a HeLiS concept) ends. The former triggers the *real-time reasoning* task that is responsible, mainly, of analyzing the single data package provided by the user.

The latter triggers the *background reasoning* that is in charge of processing the whole data packages provided within the timespan which end triggers the reasoning process. Besides efficiency reasons, the rationale of having two different triggers is due to the different nature of the rules defined within the system. These monitoring rules are divided in two sets:

- Event-Based Rules (EB-Rules): these rules define, for the specified entity to monitor, the interval of values allowable for a specific event. Examples of these rules are the maximum amount of a specific food category that can be consumed during a single meal, or the minimum number of minutes of a specific activity that a user has to perform.

- Timespan-Based Rules (TB-Rules): these rules define, for the specified entity to monitor, the interval of values allowable for a specific timespan. Examples of these rules are the minimum number of portions of a specific food category that have to be consumed during a day, or the minimum number of sport activity sessions that a user has to perform during a week.

Figures 2a and 2b show two examples of TB-Rule and EB-Rule, respectively. Briefly, the meanings of each property of a rule are the following. Row 3 specifies the profile which the rule is associated with. Our system supports the possibility of monitoring at the same time different profiles associated with specific user groups (e.g., patients affected by diabetes, patients with cardiovascular diseases, etc.) Row 4 defines the priority of the rule. This value is mainly exploited by the NLG component. When multiple violations are generated, a violation selection strategy is applied as described in Section 5.

Row 5 provides the kind of validation that the reasoner has to perform. The `command` property can assume four values: *contains*, *notcontains*, *occurrence*, *property*. The first two values are used when a rule monitors the detailed amount of a specific entity (e.g., grams of a specific food category, minutes of a specific activity). The third value is used when a rule monitors the number of times that a specific entity occurs in the data (e.g., the number of portion of fruits, or the number of training sessions). Finally, the fourth value is used when a specific property of an entity has to

```
1.   vc:MR-MEDDIET-010-NDAY  rdf:type vc:MonitoringRule ;
2.                           vc:hasRuleId "MR-MEDDIET-010-NDAY" ;
3.                           vc:appliesTo vc:MEDDIET ;
4.                           vc:hasPriority 1 ;
5.                           vc:command "occurrence" ;
6.                           vc:hasOperator "greater" ;
7.                           vc:timing vc:Day ;
8.                           vc:monitoredEntity vc:Fruits ;
9.                           vc:monitoredEntityType vc:Food ;
10.                          vc:hasMonitoredValue 3 .
```

(a) A Timespan-Based Rule that defines the minimum number of portions of fruit that a user should consume during a day.

```
1.   vc:MR-MEDDIET-13-QB  rdf:type vc:MonitoringRule ;
2.                        vc:hasRuleId "MR-MEDDIET-13-QB" ;
3.                        vc:appliesTo vc:MEDDIET ;
4.                        vc:hasPriority 1 ;
5.                        vc:command "contains" ;
6.                        vc:hasOperator "less" ;
7.                        vc:timing vc:Meal ;
8.                        vc:monitoredEntity vc:WhiteMeat ;
9.                        vc:monitoredEntityType vc:Food ;
10.                       vc:hasMonitoredValue 100 .
```

(b) An Event-Based Rule that defines the maximum amount of white meat that a user should consume within a single meal.

Figure 2: Examples of Timespan and event-based Mediterranean diet rules. The amounts are reported according with the specifics of the Mediterranean Diet. `vc` is the namespace prefix used for the concepts of the HeLiS ontology. The rules are described by using the RDF language.

be monitored (e.g., the amount of calories). In the latter case, the Row 8 contains the property `monitoredProperty` instead of `monitoredEntity`. Row 6 specifies the mathematical operator used for validating the rule. Allowable values are *less*, *lessequal*, *equal*, *greater*, *greaterequal*, and *percentage*. For both the `command` and `hasOperator` properties, the string literals do not represent any ontological entity. These properties are exploited at reasoning time for invoking specific methods during the execution of data aggregation and comparison operations. These reasoning capabilities are allowed by the high flexibility of RDFpro.

Row 7 provides the timing of the rule. This property enables the classification of the rule as EB-Rule or TB-Rule according to the ontological class of the specified concept. Finally, Rows 8, 9, and 10 describe the entity (as a HeLiS class) that is monitored by the rule, the type of the entity (this information facilitates the reasoner in performing some steps when it is not possible to automatically infer the type of the monitored entity), and the values making the rule satisfied.

4.2. Inference Execution

Depending on the trigger received, the SPARQL-based reasoner starts the process from the *real-time reasoning* or from the *background reasoning* task. Once the reasoning process started, as first step, the reasoner collects the rules to validate among the ones available within the rule-set provided by the domain expert. The collection of the monitoring rules to validate is performed in two steps. A first group of rules is collected depending on the event triggering the reasoner, e.g., if the *real-time reasoning* is invoked, only the rules classified as EB-Rule are extracted. Differently, in case of the *background reasoning* execution, the TB-Rule classified with the same timing information provided by the *background reasoning* are extracted.

After this operation, the *upper-bound rules collector* task is performed. This task, that is executed for both real-time and background reasoning, is in charge of collecting rules that can be validated even if they are not directly linked with the activated reasoning. Examples are: a TB-Rule during a real-time reasoning or a TB-Rule having a timing different from the one with which the background reasoning has been invoked. This operation exploits the structure of the `TemporalEvent` branch of the HeLiS ontology. Concepts of this branch are associated through the `isSubTimespanOf` object property if a specific timespan is temporally included in another one or not. An instance of this property is the following: `Day isSubTimespanOf Week`. This means that even if the reasoner has been invoked for validating the monitoring rules labeled with the timing `Day`, a subset of the rules labeled with the `Week` timing information can be validated as well. Such a subset is selected according to the value of the `hasOperator` property. In particular, only rules having the values `less` or `lessequal` for the property `hasOperator` are extracted. This strategy enables the generation of some negative feedback before the end of specific timespans, if necessary. For example, let us consider to monitor a rule having `timing Week` saying that the user must eat `WhiteMeat` no more than two times during the week (i.e., `lessequal 2`). If after the fourth day of the week, a user eats the `WhiteMeat` for the third time, thanks to the *upper-bound rules collector*, the system is able to generate a `Violation` triggering the generation of a feedback about this specific rule.

Figure 3 provides one of the inference SPARQL queries integrated into our reasoner. In particular, this query is used to detect if the portions of a specific food category consumed by a user exceeded the daily quota or not.

Rows from 3 to 7 contain the definition of the `Violation` individual. From row 12 to row 16 the reasoner selects which rules to validate, in this case the ones focusing on the consumption of the number of *occurrences* of food that should not exceed (`less` operator) the daily (timing `Day`) limit. Rows from 17 to 19 allows the reasoner to verify if the current user is associated to a profile linked to the selected `MonitoringRule`. Rows from 23 to 25 check if the food consumed by the current user is of the same type of the food monitored by the rule. Rows 28 and 29 perform trivial checks about the coherence between the rule's operator and the computed quantity. Finally, at rows 30 and 31 we included two functions we implemented, and dynamically called at reasoning time, to compute the violation level based on the difference between the detected number of portions and the expected one, and for generating the `Violation` identifier.

4.3. Violation Generation and Population

The result of the inference activity described above is a set of structured data packages containing information about the detected undesired behaviors, i.e., violations. Each data package is an instance of the `Violation` concept, it is stored within the knowledge repository and it is made available for the NLG component. As introduced in the beginning of this section, the generation and the population of each instance of type `Violation` is performed in two separated steps. First, the `Violation` is generated as result of the reasoning activity and all information inferred by the SPARQL-based reasoner are stored into it. Second, further information that are exploited by

```
:check_contains_food_occurrence_less_day a rr:Rule, rr:NonFixpointRule;
  rr:phase  10;
  rr:insert """ ?v :hasViolationRule ?rule; :hasViolationGoal ?goal; :hasViolationUser ?user;
                   :hasViolationQuantity ?quantity; :hasViolationConstraint ?operator;
                   :hasViolationEntityType ?et; :hasViolationLevel ?level;
                   :hasViolationStartTime ?minTimestamp; :hasViolationEndTime ?timestamp;
                   :hasTimestamp ?timestamp. """;
  rr:where  """ {
                  SELECT ?rule ?goal ?user ?et ?mv (MAX(?mealTs) AS ?timestamp)
                         (MIN(?mealTs) AS ?minTimestamp) (COUNT(DISTINCT ?cf) AS ?quantity)
                  WHERE {
                    ?rule a :MonitoringRule; :timing ?timing; :command "occurrence";
                            :monitoredEntityType :Food; :hasMonitoredValue ?mv;
                            :monitoredEntity ?class.
                    FILTER EXISTS {?rule :hasOperator "less"}
                    FILTER EXISTS {?rule :timing :Day}
                    {SELECT DISTINCT ?rule ?goal ?user WHERE {
                      {?rule :appliesTo ?user} UNION
                      {?rule :appliesTo ?goal. ?goal ^:belongsProfile ?user.}}}
                    ?meal :hasUser|^:consumed ?user; :hasTimestamp ?mealTs.
                    {?timing rdfs:subClassOf :Timespan} UNION {?meal a ?timing}
        BIND (:mintEntityType(?class) AS ?et)
                    ?cf   ^:hasConsumedFood ?meal; :hasFood ?food; :amountFood ?amount .
                    FILTER(?amount > 0.0)
                    ?food a ?class. }
                  GROUP BY ?rule ?goal ?user ?et ?mv }
                ?user :hasUserId ?userId.
                ?rule :hasOperator ?operator; :hasMonitoredValue ?value; :hasRuleId ?ruleId.
                FILTER (?operator = "less" && ?quantity >= ?value)
                BIND (:computeViolationLevel(?mv, ?quantity) AS ?level)
                BIND (:mintViolation(?ruleId, ?userId, ?timestamp) AS ?v) """.
```

Figure 3: Example of inference SPARQL query that detects if a user exceeds with the daily amount of a food.

the NLG component are retrieved from the knowledge repository and stored into the `Violation` instance. This way, each `Violation` instance is a self-contained object including all information needed (i) by the NLG component for generating the feedback, and (ii) by the system for statistics purposes.

As example, we report in Figure 4 the complete `Violation` instance generated by the system in case a user violated the monitoring rule described in Figure 2a.

```
 1. vc:VIOLATION-FB267-1491063927420   rdf:type vc:Violation ;
 2.                                    vc:hasUser vc:FB267 ;
 3.                                    vc:hasViolationId "Violation-FB267-1491063927420" ;
 4.                                    vc:hasRule vc:MR-MEDDIET-010-NDAY ;
 5.                                    vc:hasRuleId "MR-MEDDIET-010-NDAY" ;
 6.                                    vc:hasPriority 1 ;
 7.                                    vc:command "occurrence" ;
 8.                                    vc:hasOperator "greater" ;
 9.                                    vc:timing vc:Day ;
10.                                    vc:monitoredEntity vc:Fruits ;
11.                                    vc:monitoredEntityType vc:Food ;
12.                                    vc:expectedQuantity 3 ;
13.                                    vc:actualQuantity 2 ;
14                                     vc:unit "portion" ;
15.                                    vc:hasLevel: 1 ;
16.                                    vc:hasTimestamp: 1491063927420 ;
17.                                    vc:hasStartTimestamp: 1491043927420 ;
18.                                    vc:hasEndTimestamp: 1491063927420 ;
19.                                    vc:hasHistory: 1 ;
20.                                    vc:hasMeal: MEAL-58ccf3cbfd110f24e59eeced ;
21.                                    vc:hasMeal: MEAL-58ccf3cbfd110f24e59dffab ;
22.                                    vc:hasMeal: MEAL-58ccf3cbfd110f24e59cbeaf .
```

Figure 4: Example of the violation bean produced by the reasoner as consequence of the violation of the rule shown in Figure 2a. `vc` is the namespace prefix used for the concepts of the HeLiS ontology. The violation is described by using the RDF language.

Rows 1, 2, and 3 contain information about both the violation and the user ids. Information provided between rows 4 and 12 are inherited by the rule definition that has been violated. This information is preparatory for the feedback generation task since it avoids the NLG component to perform further queries on the knowledge repository. Rows from 13 to 16 contain information directly provided by the reasoner, i.e., the quantity observed and related to the entity described at row 10, the unit of measure of the specified quantity, the violation level, and the timestamp in which the `Violation` has been detected. The violation level gives a dimension of the violation, the higher the gap between the actual and the expected values is, the higher the value of the violation level parameter will be. Finally, information between rows 17 and 22 are computed during the *Violation Population* task that is executed after the inference step. Here, the knowledge repository is queried for retrieving more specific information about the generated violation. The start and end timestamps shown at rows 17 and 18 are extracted from the collection of the events (in this case the list of meals shown in rows 20, 21, and 22) that caused the generation of the `Violation`. Finally, the violation history value provided at row 19 is computed. This value provides a recidivism index about how a user is inclined to violate specific rules and it is exploited by the NLG component for choosing the proper terminology at feedback generation time.

5. Explaining Logical Inconsistencies with Natural Language

Here we present a method that performs a linguistic realization of the violation beans of Figure 4 that is useful as motivational message. This realization has to be human understandable and convince users to avoid undesired behaviors that trigger such inconsistencies. Therefore, we need (i) a persuasive framework that helps users in conduct a good healthy behavior (Section 5.1); and (ii) an effective natural language generator method that translates the logical language of the reasoning results (Section 5.2). Both components need the HeLiS ontology to retrieve the necessary data. Figure 5 shows the architecture of our method. The core part relies on templates (a grammar) that encode the several parts (feedback, arguments and suggestion) of a persuasion message. The terminal symbols of these templates are organized according to a hierarchy where the most specific terms are related to specific persuasion strategies. A filler layer manages the filling of the terminal symbols into the templates. Once the templates are filled, a sentence realizer generates natural language sentences that respect the grammatical rules of a target language (here Italian).

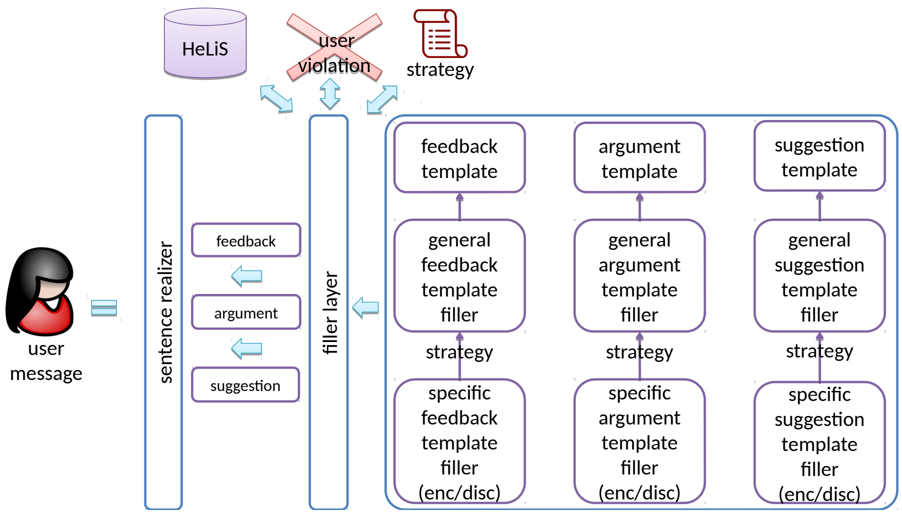

Figure 5: Architecture of our method: the templates are a grammar that translates a logical language into a natural one. They are organized according to persuasion strategies.

5.1. The Persuasive Framework

We inspired our work from the theoretical framework discussed in [4] for encoding real-time tailored messages in behavior change applications that can be adapted to different generation strategies ranging from canned text to deep generation. The framework proposes a principled model for defining motivational messages based on four basic concepts of the literature about motivational messages: *timing*, *intention*, *content* and *representation*. Timing and intention are related to the *persuasion strategy* whereas the others involve the *persuasive content* of the message. We choose this framework as it is a good balance between a vertical approach, deeply focused on the domain but with poor generalization properties, and a horizontal one that is not bounded to a specific domain but it is limited to be only at a theoretical/conceptual level.

5.1.1. Persuasion Strategy

The violation bean of Figure 4 contains all the information explaining the inconsistency of the user's dietary behavior with respect to the HeLiS ontology. In addition, at the end of a day/week many of this beans can be generated. However, a long list of these beans is understandable mainly by the domain experts and, most of all, it does not prevent the user to avoid such an erroneous behavior. A persuasion strategy addresses this challenge by considering the right *timing* for sending the bean, the *choice of the violation* bean to send to the user (not covered in [4]) and the *intention* the system wants to communicate to the user.

The **timing** represents the event prompting the creation of a new message. Message generation can be triggered by specific events (e.g., the generation of a new violation bean) or by temporal events. In particular, our system works with three kinds of events:

- events related to user's habits and behavior (i.e., the generated violations). This is a *post* timing strategy that occurs after a user's action and is used as positive/negative feedback for future similar actions [64].
- time scheduling: the need to send particular information to the user at specific time of the day or of the week. This is a *pre* timing strategy that occurs before a foreseen user's action and is used to divert her/him from a wrong behavior [65].
- localization: the third event triggering the generation of a message after recognizing that the user is in a specific place (e.g., near a vending machine). This is a *during* timing strategy that recognizes the happening of a user's action and is used to support or divert him/her in real-time [66].

The first kind of events is directly triggered by the detection (through the logical reasoning process of Section 4) of a violation; hence, those information are used for generating the persuasive explanation. The second and third kinds of events, instead, generate persuasive explanations by starting from a pool of past violations.

Once a list of violation beans has been generated, a **choice of the violation** is performed to avoid annoying the user with too many and repetitive messages. This negative behavior would risk to decrease the attention level of users when they receive the feedback. If the list of violations is empty, the system infers that the user adopted a healthy behavior so it sends messages with *positive* reinforcing feedback. If such list is not empty, the system sends a message regarding only one violation to provide the user with varied content about different aspects of a correct behavior. The violation is chosen according to (i) its priority, (ii) the number of times it was committed (see the history property in Figure 4), and (iii) the number of times the same violation was the object of a message. For example, if a message discouraging to drink sweet beverages has already been sent in the last 4 days, the next highest priority violation bean not sent recently is chosen.

Once a violation bean is selected, a persuasion strategy computes the **intention** (or aim) the persuasive message should convey. According to [4], the intention is composed by a *feedback* on user's activity, an *argument* about the consequences of user's behavior and a *suggestion* to follow a healthy behavior. We consider two kinds of intentions: to *encourage* or *discourage* the user to follow a healthy or unhealthy behavior. In the example of Figure 4, the user drank too much sweet beverages, thus the intention is to discourage this behavior.

5.1.2. Persuasion Content

The **content** of the message is the information the message has to convey to the user. The content generation is the filling of the feedback, argument, suggestion components:

Feedback is the part of the message that informs the user about the unhealthy behavior. Feedback is generated considering data included in the selected violation: the entity of the violation represents the object of the feedback, whereas the level of violation (the deviation between the expected food quantity and the actual one) is used to represent the severity of the incorrect behavior. Feedback contains also information about timing to report the moment in which violation was committed.

Argument is the part of the message that informs the user about the possible consequences of a behavior. For example, in the case of diet recommendations, the argument consists of two parts: (i) information about nutrients contained in the food intake that caused the violation and (ii) information about consequences that nutrients have on health. Consequences imply the positive or negative aspects of nutrients according to the encourage or discourage intention, respectively.

Suggestion this part is the solution proposed to the user in order to motivate him/her to change his behavior. This suggestion informs the user about the alternative and healthy behavior that he/she can adopt.

The **representation** regards the format of the content to present to the users. We focus on a natural language representation, however, the persuasive framework deals also with audio or visual formats, for example hGraphs[10] (health Graphs) can be adopted. These are standardized visual representations of a patient's health status to display a complete overview of an individual's health. A hGraph is composed by a thick green ring, and a series of dots that represent the health/well being data (e.g., the weekly meat consumption) according to definite metrics. Dots inside the green zone indicate that the subject is behaving well respect to those parameters. Dots outside the green ring indicate unhealthy conditions. In this way, users can easily identify which parameters are in the normal range and those that may be too high or too low. An aggregated score (1-100) summarizes the person's health status regarding the defined metrics.

5.2. Linguistic Realization of the Persuasive Content

We describe the process of generating the persuasive explanation starting from the received violation bean, the chosen strategy (here encourage or discourage) and HeLiS. As shown in Figure 5, the natural language generation of the content is performed with templates. This is due to the fact that it is very difficult to build a big and tailored dataset of persuasion sentences to perform the linguistic realization with deep learning techniques. In addition, by considering that we are within the healthcare domain, we need the total control on the generated output as wrong indications could lead to serious effectiveness problems of the proposed solution. Moreover, our template system is devised to allow the dynamic construction of tailored sentences thus avoiding standard canned texts. Here, we encode the feedback, argument and suggestion components with some templates, i.e., a grammar with non-terminal/terminal symbols and production rules. The terminal symbols are selected in the filler layer module to fill the non-terminal ones according to the violation, the strategy and HeLiS. Once the

[10] http://hgraph.org/

1) Structure of the feedback template:

```
feedback := temporal_adv + feed_verb + adj + quantity + food_entity
```

2) Structure of the argument template:

```
argument := intro + food_ent_category + verb_adj + food_property + conseq_verb +
            consequence
```

3) Structure of the suggestion template:

```
suggestion := intro + food_entity + alternative
```

Table 1: First layer of the template system regarding the structure of the templates.

templates are filled, they are sent to a sentence realizer that adjusts the raw sentence according to the syntax rules of the selected natural language.

5.2.1. The Template System

The template system is the organization of the templates according to the presence of non-terminal/terminal symbols and the persuasion strategy. They are organized in layers. The first is the structure of the feedback, argument and suggestion components. It is encoded as a set of production rules between generic non-terminal symbols (Table 1). The second layer consists of production rules between non-terminal and terminal symbols about the domain. This regards the content of the templates (Table 2). The third layer contains rules between non-terminal and more specific terminal symbols related to the chosen persuasion strategy (Table 3). This decoupling of the templates structure from their content allows the portability of the templates. Indeed, the first layer could be adapted in other domains with other languages with very low effort. On the other hand, if a different persuasion strategy needs to be adapted this reflects only the last layer.

Table 1 shows the structure of the feedback, argument and suggestion components. This is the concatenation (symbol +) of some non-terminal symbols that are filled with the terminal ones of tables 2 and 3. The filling can be direct (see `intro` symbol of Table 2) or dependent from other data such as the violation or HeLiS. This dependency needs to be computed by the filler layer module and it can be just a query to HeLiS or

1) Terminal symbols for the feedback template:

```
temporal_adv := ["today"|"in the last seven days"]violation
feed_verb := ["to eat"|"to consume"|"to intake"|"to drink"]violation, tense
food_entity := []violation, HeLiS
```

2) Terminal symbols for the argument template:

```
intro := "do you know that"
food_ent_category := []violation, HeLiS
```

Table 2: Second layer of the template system regarding the content of the templates.

could require more complex operations. For example, the symbols `food_entity` or `food_ent_category` are filled with the corresponding HeLiS labels retrieved by using the field `entity` of Figure 4. Some non-terminal symbols (e.g., the `feed_verb`) can be dependent from the verb and its tense: e.g., beverages imply the use of the verb *to drink* while for solid food we used *to eat*. To increase the variety of the message the verbs *to consume* and *to intake* are also used. Simple past tense is used when violation is related to specific moments (*Today you did not eat enough vegetables*), while simple present continuous is used when the violation is related to a period of time not yet ended (*This week you are drinking a lot of fruit juice*). The filling of other symbols can require more complex operations as long as we are processing the most specific layers of the template system. Indeed, the symbols of Table 3 needs the computation of the strategy. This is given by the field `constraint` in the violation bean: a *less* constraint (fruitjuice $<=$ 200ml) refers to an excess of this food and this behavior has to be discouraged. A *greater* constraints (vegetables $>=$ 200g) implies an insufficient amount of this food and this behavior has to be encouraged. Therefore, a *less* constraint will trigger a discourage strategy, whereas a *greater* constraint will trigger an encourage strategy with the consequent choice of the right terminal symbols in the third template layer. Other template filling could require meta-reasoning strategies to identify the appropriate content that can depend on qualitative properties of food, user profile, other specific violations, and the history of messages sent. This can be noticed in the choice of alternative food for the suggestion template. HeLiS provides food that are valid alternatives to the consumed food (e.g., similar-taste relation, list of nutrients,

Encourage	Discourage
1) Specific terminal symbols for the feedback template:	
`adj := ["not enough"| "too little"]`$_{\text{violation}}$	`adj := ["a lot of"| "too much"]`$_{\text{violation}}$
`quantity := ["({} of at least {})"]`$_{\text{violation}}$	`quantity := ["({} of maximum {})"]`$_{\text{violation}}$
2) Specific terminal symbols for the argument template:	
`verb_adj := ["to be rich of"]`	`verb_adj := ["to contain a lot"]`
`food_property := []`$_{\text{HeLiS, violation}}$	`food_property := []`$_{\text{HeLiS, violation}}$
`conseq_verb := ["that help to"]`	`conseq_verb := ["that can cause"| "that may contribute to"]`
`consequence := []`	`consequence := []`
3) Specific terminal symbols for the suggestion template:	
`intro := ["next time try to alternate"]`	`intro := ["next time try with"]`
`food_entity := []`$_{\text{violation}}$	
`alternative := "with" + []`$_{\text{HeLiS}}$	`alternative := []`$_{\text{HeLiS}}$

Table 3: Third layer of the template system regarding the strategy/content of the templates.

consequences on user health). Then, these alternatives are filtered according to the user profile: even if fish is an alternative to legumes it will not be proposed to vegetarians. Moreover, food that can cause a violation of *less* or *equal* constraints cannot be suggested, e.g., meat cannot be recommended as alternative to cheese if the user has already eaten its maximum quantity. Finally, control on messages history is performed to avoid the repetitiveness of the message content.

5.2.2. The Sentence Realizer

Our system creates the message directly in the desired language through the Sentence Realizer (SR). The SR takes in as input the filled templates for the feedback, argument and suggestion components and generates a complex and well-formed sentence according to the grammar rules of the target language, putting spaces, capital letters and choosing the correct inflected forms of the lemmas. In particular, the Italian language is morphologically richer than English and it entails additional linguistic resources management to harmonize the various parts of the sentences. To this end, the SR implements a morphological engine based on Morph-it!, a morphological resource for the Italian language [67] with a lexicon of inflected forms with their base lemmas and morphological features: gender and number for nouns and articles; gender, number and positive, comparative, superlative for adjectives; tense, person and number for verbs; number, gender, person for pronouns, etc. The Morph-it! version used in the system contains about 35,000 lemmas and 500,000 entries. The SR invokes the morphological engine to compose the basic lemmas and to agree verbs, articles, articulated propositions and adjectives with the nouns according to the different roles that the noun plays in a sentence (subject, object, possessive form, etc.) according to the Italian grammar rules. Regarding our example of Figure 4, the final persuasive message is: *Today you have drunk too much (300 ml of maximum 200 ml) fruit juice [feedback]. Do you know that sweet beverages contain a lot of sugars that can cause diabetes [argument]? Next time try with a fresh fruit [suggestion].* Figure 6 shows a screenshot of our system communicating the persuasive message to the user.

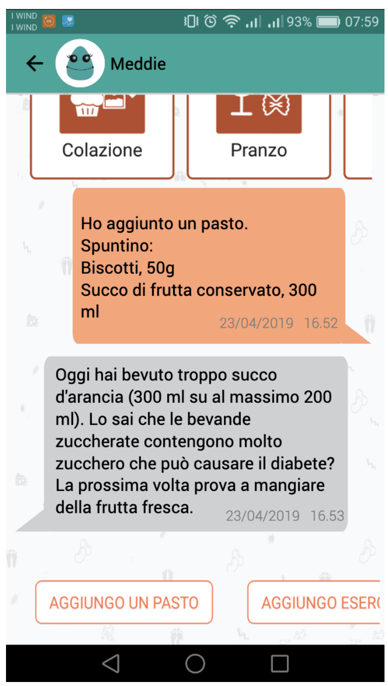

Figure 6: A screenshot of the system communicating the persuasive message of not exceeding with the intake of sweet beverages.

6. Use Case: The *Key to Health* Project

Systems for personalized healthy lifestyle recommendations fall in the broad area of decision support. The goal of these systems is to help and guide users in taking healthy-informed decisions about their lifestyle, on aspects such as food consumption. Such systems have to take a decision (e.g., suggesting conscious and healthy food consumption), similarly as a human expert would do, based on available data (e.g., nutrients ingested in the last meals, user health conditions), and to communicate these decisions to the users according to their preferred means and modalities.

As a specific case study, the presented system has been implemented into the HORUS.AI platform [6] and deployed and evaluated in the context of the project *Key to Health* on workplace health promotion (WHP) inside our institution (Fondazione Bruno Kessler, FBK). WHP, defined as *the combined efforts of employers, employees, and society to improve the mental and physical health and well-being of people at work*[11], aims at preventing the onset of chronic diseases related to an incorrect lifestyle through organizational interventions directed to workers. Actions concern the promotion of correct diet, physical activity, and social and individual well-being, as well as the discouragement of bad habits, such as smoking and alcohol consumption. Within the *Key to Health* project, a mobile application has been created based on the services included into HORUS.AI. This mobile application has been used by 120 FBK's workers (both researchers and employers) as a tool to persuade and motivate them to follow WHP dietary recommendations. All of the 120 participants voluntarily enrolled in the project and signed an informed consent before the beginning of the experiment. The only requirement was that users were in good health and not under medical supervision for nutritional-related diseases since the living lab has not to be considered a clinical trial. Table 4 shows main demographic information concerning the users involved in the performed evaluation campaign.

[11] Luxembourg Declaration on workplace health promotion in the European Union, 1997.

Dimension	Property	Value
Gender	Male	57%
	Female	43%
Age	25-35	12%
	36-45	58%
	46-55	30%
Education	Master Degree	42%
	Ph.D. Degree	58%
Occupation	Ph.D. Student	8%
	Administration	28%
	Researcher	64%

Table 4: Distribution of demographic information of the users involved in the evaluation.

6.1. Behavior Change Theory Features

The implemented mobile application presents some important features from the behavior change theory:

- *Goal setting* is the activity carried out by users to setting a specific goal. This activity has been proven to lead better performance than without setting any goal or setting a generic one, for example, "do one's best" [33]. Our mobile application has this feature enabling the user to select the goal from a menu. The goals are associated to the Mediterranean diet rules, for example, a user would like to increase the number of consumed daily portions of vegetables.

- *Self-monitoring* that is the activity of tracking a specified behavior in order to gain awareness and change it [34]. The mobile application has a food diary that allows users at inserting food categories constituting a meal and the corresponding quantities, see Figure 7 for a screenshot. The user can choose the food searching in the list of recipes and dishes in the food ontology. Using the nutritional information in the ontology, the system calculates the nutritional values of each food for the specific recorded quantity. The diary is associated to a tracker widget that graphically shows the adherence to the chosen goal. In addition, a doughnut chart allows users to measure the adherence to all the rules of the Mediterranean diet according to some single food categories. This enables an objective measurement of the whole outcome of using such a persuasion system for a healthy food consumption, see Figure 8 for a screenshot.

- *Personalized feedback* that is the activity of communicating to users any information regarding their personal behavior and its consequences [35]. This has been studied and implemented in the previous section, see Figure 6 for a screenshot.

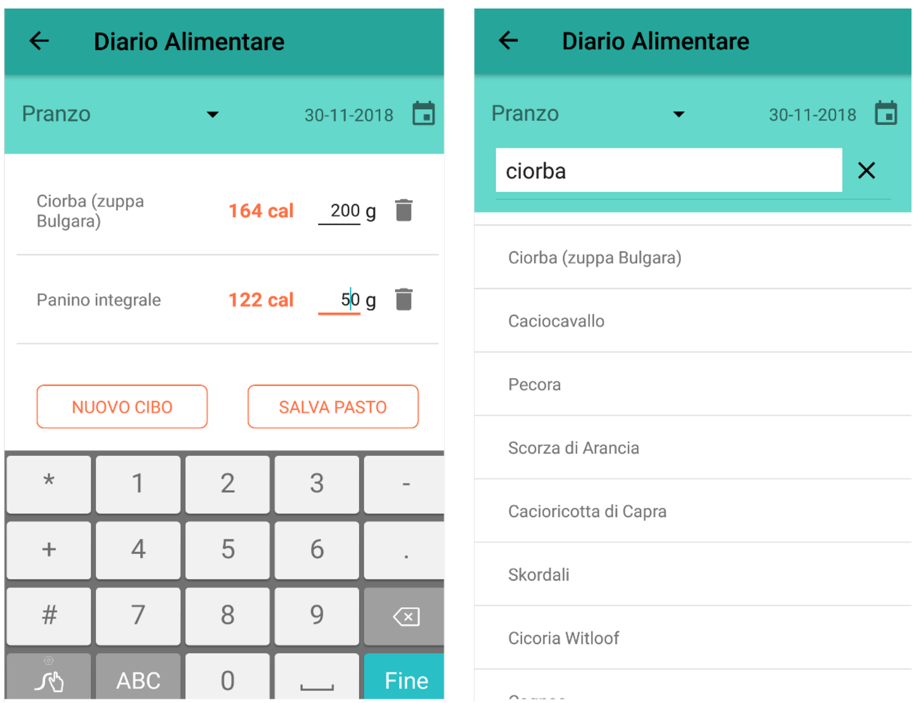

Figure 7: The diet diary of the application allows the users to record the consumed meal with the respective quantities (left). The user can search and choose the meal from a list generated from the ontology content (right).

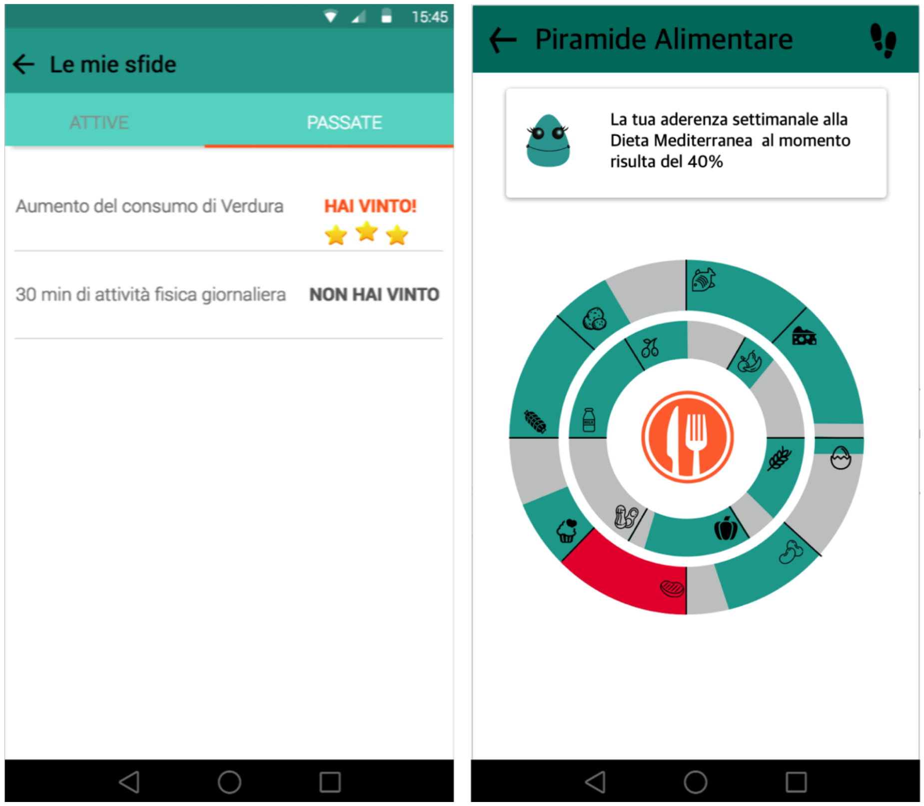

Figure 8: The tracker widget provides the users a visual description of his/her adherence to a goal, e.g., eating at least two portions of vegetables every day (left). The doughnut chart provides a visual description about the user's adherence to the prescriptions of the Mediterranean diet in a certain day of a week. Satisfied rules are in green (e.g., a maximum amount of sweets), violated rules are in red (e.g., too much meat).

7. Evaluation

In this Section, we report the evaluation activities we performed within our use case by adopting the HORUS.AI platform. The evaluation we propose is threefold.

First, we assess the overall usability of the mobile application (Section 7.1) with the use of questionnaires filled by the users. The aim of this validation is to check the presence of negative features of the application that could affect the persuasiveness of the system. Second, we present the validation performed by the domain experts with respect to the correctness and appropriateness of the generated messages (Section 7.2). This validation aims to verify that the explanations provided by the system are coherent with respect to the detected unhealthy behaviors. At the end, we discuss the effectiveness of generated explanations on users' behaviors (Section 7.3) by showing how the use of explanations leads to a lower number of dietary violations with respect to a control group of users receiving punctual feedback without any detail. The evaluation of reasoning performance is out of scope of this paper, see [68] for further details.

7.1. Usability Evaluation.

The usability of the mobile application provided to users has been evaluated through the System Usability Scale (SUS), analyzing the intuitiveness and simplicity of the system. The evaluation protocol consisted in multiple use sessions and followed the five steps below:

1. Training meetings with the users involved in the evaluation for an introductory explanation of the functionalities available in the mobile application.
2. Four days of usage of the mobile application by the users.
3. Meetings with the users for collecting questions about functionalities. Release of a new version of the mobile application integrating bug fixes reported by the users during the first four days of usage.
4. Four days of usage of the mobile application by the users.
5. Final meetings with the users and distribution of the evaluation questionnaires.

The usability test of the mobile application involved all the 120 employees participated to the campaign. According to the usability test requirements provided by [69], the number of users involved in the test granted the discovery of 100% of the usability problems. The average score obtained from the SUS was 81.5, that, according to the adjective rating scale proposed by [70], corresponds to *excellent*. Further interviews were conducted to evaluate the impact of the mobile application in the routine at work setting and extra-work setting, at the end of the seven weeks of pilot study. In general users appreciated the system and considered the mobile application a useful tool, especially for increasing the awareness about their eating habits. Indeed, users provided mainly dietary data because few of them had pedometer bracelets.

Finally, we report users' feedback about their actual perception on the personalization capabilities of the proposed solution. During the focus group we organized at the end of the *Key to Health* project, we asked to the users the strong and weak points of the system concerning personalized interactions. Overall, the users appreciated the system responsiveness and message tailoring capabilities when new data were provided. However, a common request was related to the possibility of better exploiting the geographical information that can be acquired through the smartphone sensors. This information was considered relevant for motivating people in changing habits within some real-life situations. Suggested examples include the possibility of sending alerts, based on the current user location, about close healthy nutrition shops, restaurants cooking recipes that are compliant with users goals, sport events related with preferred users habits or promotions of gyms close to users' usual places. These suggestions will lead the next version of the personalization component of HORUS.AI in order to improve the perception that the system is providing a real-time support to users.

7.2. Domain Experts Evaluation

The second validation of our approach concerns the correctness and appropriateness of the explanations generated by the system for supporting the interactions with users. Thus, we present below the procedure for defining and validating: (i) the structure of explanation templates and (ii) the appropriateness of the generated explanations with respect to the detected violations.

Explanation Templates Validation. Three experts [12] have been involved for modeling the templates adopted for generating the explanations. As it has been explained in Section 4, explanations are generated by starting from a finite set of templates that are combined together according to the information contained in the violation packages created by the reasoner. For example, given the category contained in the violation and the violation level, templates concerning the positive or negative properties of the specific food category are connected with verbs and adjectives for shaping the final message. The set of message templates has been validated by the experts that verified

[12] All experts are dietitians and well-being coaches of our local healthcare department.

the grammatical and content correctness of each template.

Appropriateness of Explanations. The second validation task, where experts were involved, concerned the appropriateness of the messages generated with respect to the violations detected by the reasoner. In order to perform this validation, we performed the following steps:

1. we built data packages representing combinations of meals that should trigger, for each rule contained in the system, the detection of the corresponding violation;

2. we verified that the reasoner correctly detected the violation associated with a given data package;

3. we checked, together with the experts, the appropriateness of the explanation generated with respect to each detected violation.

The analysis of the pairs violation-explanation triggered slight revisions of the linguistic fragments. In particular, some verbs and adjectives used in the fragments were changed to better contextualize the messages.

7.3. Effectiveness of Explanation

The third evaluation concerned the effectiveness analysis of generated explanations on the user study. We analyzed the usage of the mobile application connected with our platform for seven weeks by monitoring the information provided by the users and the associated violations. Our goal was to measure the effectiveness of the explanations generated by our platform by observing the evolution of the number of detected violations. The 120 users involved in the *Key to Health* project have been split in two groups. A first group of 92 users (hereafter called the intervention group) received the whole persuasive messages generated by using the template system. Whereas a second group of 28 users (hereafter called the control group) did not receive any composition of feedback, argument and suggestion, but only canned text messages notifying when a rule was violated. An example of canned text is "Today you have drunk too much (300 ml of maximum 200 ml) fruit juice" notified as soon as the related violation is detected.

Our hypothesis was to find a higher decrease in the number of violations through the time by the users receiving persuasive messages.

Results concerning the evolution of the violation numbers are presented in Figure 9. We considered three different kinds of dietary rules instantiating the EB-Rules or TB-Rules described in Section 4:

- QB-Rules (instances of EB-Rules): these rules define the right amount of a specific food category that should be consumed in a meal.
- DAY-Rules (instances of TB-Rules): these rules define the maximum (or minimum) amount (or portion) of a specific food category that can be consumed during a single day.
- WEEK-Rules (instances of TB-Rules): these rules define the maximum (or minimum) amount (or portion) of a specific food category that can be consumed during a week.

The three graphs show the average number of violations per user related to the QB-Rules, DAY-Rules, and WEEK-Rules sets respectively. The black and green lines represent the average number of violations for the intervention and control group respectively. Whereas, the red and orange lines represent the relative standard deviations. As mentioned earlier, QB-Rules are verified every time a user stores a meal within the platform; DAY-Rules are verified at the end of the day; while WEEK-Rules are verified at the end of each week. The increasing trend of the gap between the black and green lines demonstrates the positive impact of the persuasive messages sent to users. We can observe how for the QB-Rules the average number of violations is below 1.0 after the 7 weeks of the project. This means that some users started to follow all the guidelines about what to consume during a single meal. A positive result has been obtained also for the DAY-Rules and the WEEK-Rules. By considering the standard deviation lines, we can appreciate how both lines remain contained within low bounds without the presence of outliers. Moreover, if we examine the drop of violations after the 7 weeks of the project (Table 5) we notice that both QB and DAY rules obtained good drops. For the WEEK-Rules, however, the drop remained limited. This result is due to

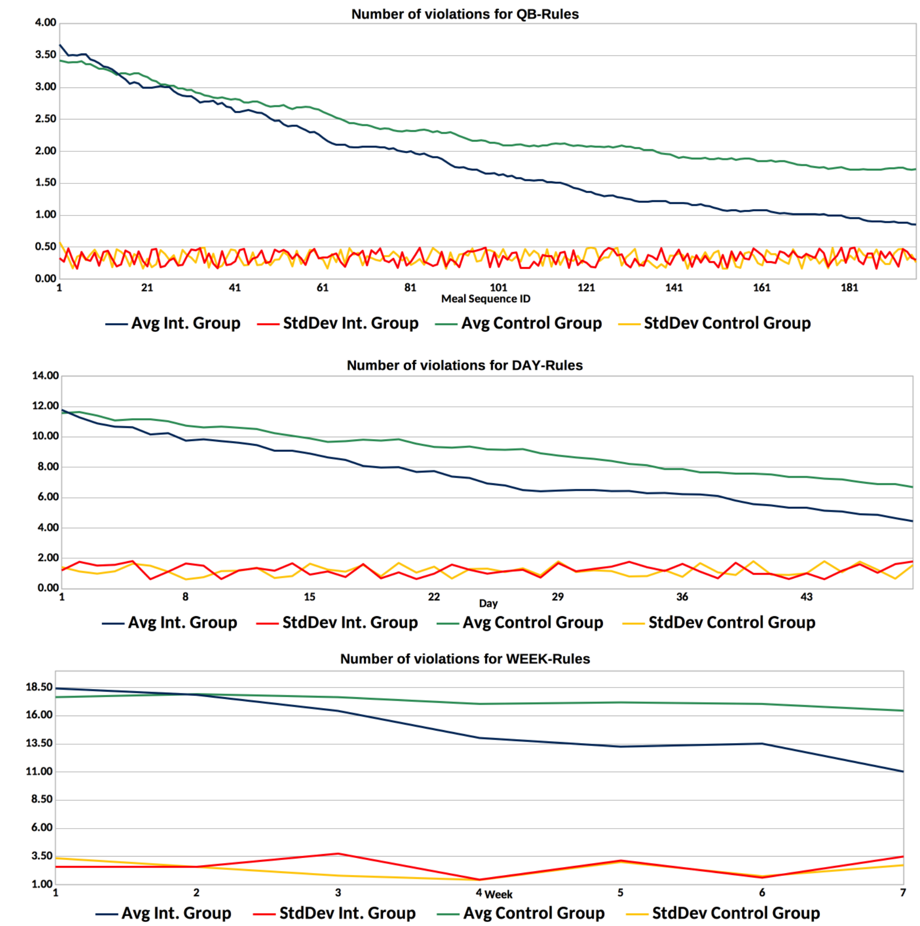

Figure 9: Variation of the (average) number of detected violations within the *Key To Health* time span. The intervention group present a more effective decay of the violations.

the fact that the first two rules are more frequently notified when violated: after every meal violations for the QB-Rules and after every day for the DAY-Rules, whereas the WEEK-Rules are notified once a week. Therefore, the users pay more attention to the more frequent kind of notifications. For all the three kinds of rules the intervention group has a bigger drop with respect the control group.

We are also interested in the time spent by our system to be effective. Figure 9 shows us that the two groups tend to diverge at a certain point during the *Key To Health* time span. We measure the day/week when the two groups start to diverge with a statis-

	QB-Rules	DAY-Rules	WEEK-Rules
Control Group	50.00%	42.33%	8.17%
Intervention Group	**76.60%**	**62.20%**	**40.12%**

Table 5: Drop of violations at the end of the project. The more frequent rules have the higher drop of violations.

tical significance. Table 6 reports these days/week along with the p-values and average number of violations in the starting day/week for both the intervention and control group. The QB-Rules are the slowest to be effective taking 30 days of system usage.

	Starting day/week	p-value	Violations Intervention Group	Violations Control Group
QB-Rules	30^{th} day	0.001	1.42 ± 1.25	2.11 ± 1.37
DAY-Rules	19^{th} day	0.011	8.09 ± 2.88	9.82 ± 2.85
WEEK-Rules	4^{th} week	0.030	14.03 ± 3.18	17.03 ± 3.90

Table 6: Starting point of the project time where the persuasion system takes effect with statistical significance.

This can be explained with the high frequency of these rules (and thus notifications when violated) that makes the users to respect them. Indeed, for both groups the average number of violations is quite small. The DAY-Rules present the quickest starting point as the two groups start to diverge from the 19^{th} day, that is, the persuasion system took less than of the 39% of the project time to be effective. For the WEEK-Rules the persuasion system took an intermediate time to be effective: around the 57% of the project time. This is due to the shorter frequency of notifications.

Further considerations can be done about the abandon rate of the system, a too pushy notification system could have a high abandon rate. In our case, all the users used the system until the end of the project and no complains about the notifications have been raised during the usability evaluation, Section 7.1.

Finally, in the described living lab, we evaluated the platform over the dietary domain. The extension of the proposed strategy to other domains (e.g., the physical activity or the mental health ones) requires the definition of the knowledge supporting the generation of the explanations (e.g., negative consequences associated with a specific rule) rather than the definition of new templates since the templates are the materialization of the adopted persuasive strategy. Hence, since the effectiveness of the proposed approach have been measured on the persuasive strategy (i.e., the structure of the templates), it is feasible to think that by bringing the same persuasive strategy to other domains, such an effectiveness should be comparable with respect to the results reported here. We are aware that the set of templates used in our living lab does not cover all the possible persuasive strategies defined in the literature. However, we remark that their effectiveness would be preserved independently by the domains in which they are applied.

8. Conclusions

We presented a XAI system supporting the users in following healthy lifestyles. The system checks the presence of unhealthy behaviors based on the food consumed and activity performed by users. We discussed in particular the role of the natural language generation component and how it exploits information inferred by the reasoner for generating contextual effective explanations. We evaluated our system in a real-world context by discussing the effectiveness of using persuasive explanations with respect to canned texts. Results demonstrated how persuasive explanations allows the user to follow a healthy dietary behavior. Moreover, the modular template systems allows the dynamic construction of natural language sentences and the templates portability in other domains.

This experience opened the possibility of extending and improving our solution from both the research and technological perspective. Concerning the former, we will focus on improving the user interface by adding persuasive elements according to the criteria defined in the standard guidelines [71]. These criteria are credibility, privacy, personalization, attractiveness, solicitation, priming, commitment and ascendency. Moreover, we will study the integration of this persuasive explanations of user' behavior with argumentation and decision theory as the one reported in [46]. Our long-term goal is to develop a conversational agent able to understand the user's needs and difficulties to better persuade him/her at following healthy lifestyles. Finally, the reasoner engine developed in our system will enable the adoption of stream reasoning approaches [72] into complex real-world scenarios. This possibility will advance the start-of-the-art of the stream reasoning research area by providing a feasible test-bed for working with real-world data streams. Concerning the latter, we will dedicate effort on improving the connectivity with sensor networks in order to improve the quantity and the quality of the data collected from users. Moreover, recent advances in the Internet of Things (IoT) domain will open the possibility of acquiring nutritional information not only by considering the calories intake but from a real-time analysis of physiological evacuation. As last point, we want to mention that the results obtained on the *Key to Health* project lead the opportunity to integrate the proposed system into two further projects. The first one is funded by the Trentino Healthcare Department and it consists in applying the guidelines validated in the *Key to Health* project to the citizens of the Trentino Region. The mobile application provided to the users will include also social features allowing users to interact with people that are trying to achieve the same goals. The second one is funded by the Italian Ministry of Healthcare and it consists in the adoption of AI-based solutions for educating teenagers about healthy lifestyles. Users will use a mobile application providing recommendations with also the involvement, in the persuasion process, of their parents. All these solutions aim to increase the overall awareness about healthy lifestyles and to diminish unhealthy habits.

References

[1] A. Adadi, M. Berrada, Peeking inside the black-box: A survey on explainable artificial intelligence (XAI), IEEE Access 6 (2018) 52138–52160.

[2] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, 2003.

[3] M. Dragoni, T. Bailoni, R. Maimone, C. Eccher, Helis: An ontology for supporting healthy lifestyles, in: International Semantic Web Conference (2), Vol. 11137 of Lecture Notes in Computer Science, Springer, 2018, pp. 53–69.

[4] H. op den Akker, M. Cabrita, R. op den Akker, V. M. Jones, H. Hermens, Tailored motivational message generation: A model and practical framework for real-time physical activity coaching, Journal of Biomedical Informatics 55 (2015) 104–115.

[5] A. Gatt, E. Krahmer, Survey of the state of the art in natural language generation: Core tasks, applications and evaluation, J. Artif. Intell. Res. 61 (2018) 65–170.

[6] M. Dragoni, T. Bailoni, R. Maimone, M. Marchesoni, C. Eccher, HORUS.AI - A knowledge-based solution supporting health persuasive self-monitoring, in: International Semantic Web Conference (P&D/Industry/BlueSky), Vol. 2180 of CEUR Workshop Proceedings, CEUR-WS.org, 2018.

[7] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, L. Kagal, Explaining explanations: An overview of interpretability of machine learning, in: F. Bonchi, F. J. Provost, T. Eliassi-Rad, W. Wang, C. Cattuto, R. Ghani (Eds.), 5th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2018, Turin, Italy, October 1-3, 2018, IEEE, 2018, pp. 80–89. doi:10.1109/DSAA.2018.00018.
URL https://doi.org/10.1109/DSAA.2018.00018

[8] V. Cherkassky, S. Dhar, Interpretation of Black-Box Predictive Models, Springer International Publishing, Cham, 2015, pp. 267–286. doi:10.1007/978-3-319-21852-6_19.
URL https://doi.org/10.1007/978-3-319-21852-6_19

[9] A. Holzinger, C. Biemann, C. S. Pattichis, D. B. Kell, What do we need to build explainable AI systems for the medical domain?, CoRR abs/1712.09923. arXiv:1712.09923.
URL http://arxiv.org/abs/1712.09923

[10] A. Holzinger, P. Kieseberg, E. R. Weippl, A. M. Tjoa, Current advances, trends and challenges of machine learning and knowledge extraction: From machine learning to explainable AI, in: A. Holzinger, P. Kieseberg, A. M. Tjoa, E. R. Weippl (Eds.), Machine Learning and Knowledge Extraction - Second IFIP TC 5, TC 8/WG 8.4, 8.9, TC 12/WG 12.9 International Cross-Domain Conference, CD-MAKE 2018, Hamburg, Germany, August 27-30, 2018, Proceedings, Vol. 11015 of Lecture Notes in Computer Science, Springer, 2018, pp. 1–8. doi:10.1007/978-3-319-99740-7_1.
URL https://doi.org/10.1007/978-3-319-99740-7_1

[11] Q. Ai, V. Azizi, X. Chen, Y. Zhang, Learning heterogeneous knowledge base embeddings for explainable recommendation, Algorithms 11 (9) (2018) 137. doi:10.3390/a11090137.
URL https://doi.org/10.3390/a11090137

[12] O. Z. Khan, P. Poupart, J. P. Black, Explaining recommendations generated by mdps, in: T. Roth-Berghofer, S. Schulz, D. B. Leake, D. Bahls (Eds.), Explanation-aware Computing, Papers from the 2008 ECAI Workshop, Patras, Greece, July 21-22, 2008. University of Patras, 2008, pp. 13–24.

[13] P. Vougiouklis, H. ElSahar, L. Kaffee, C. Gravier, F. Laforest, J. S. Hare, E. Simperl, Neural wikipedian: Generating textual summaries from knowledge base triples, J. Web Semant. 52-53 (2018) 1–15. doi:10.1016/j.websem.2018.07.002.
URL https://doi.org/10.1016/j.websem.2018.07.002

[14] B. Ell, A. Harth, E. Simperl, SPARQL query verbalization for explaining semantic search engine queries, in: V. Presutti, C. d'Amato, F. Gandon, M. d'Aquin, S. Staab, A. Tordai (Eds.), The Semantic Web: Trends and Challenges - 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings, Vol. 8465 of Lecture Notes in Computer Science, Springer, 2014, pp. 426–441. doi:10.1007/978-3-319-07443-6_29.
URL https://doi.org/10.1007/978-3-319-07443-6_29

[15] E. Reiter, R. Dale, Building applied natural language generation systems, Natural Language Engineering 3 (1) (1997) 57–87.

[16] Y. Kazakov, P. Klinov, A. Stupnikov, Towards reusable explanation services in protege, in: Description Logics, Vol. 1879 of CEUR Workshop Proceedings, CEUR-WS.org, 2017.

[17] A. Kalyanpur, B. Parsia, E. Sirin, J. A. Hendler, Debugging unsatisfiable classes in OWL ontologies, J. Web Semant. 3 (4) (2005) 268–293.

[18] J. S. C. Lam, Methods for resolving inconsistencies in ontologies, Ph.D. thesis, University of Aberdeen, UK (2007).

[19] J. Bauer, U. Sattler, B. Parsia, Explaining by example: Model exploration for ontology comprehension, in: Description Logics, Vol. 477 of CEUR Workshop Proceedings, CEUR-WS.org, 2009.

[20] A. Kalyanpur, B. Parsia, M. Horridge, E. Sirin, Finding all justifications of OWL DL entailments, in: ISWC/ASWC, Vol. 4825 of Lecture Notes in Computer Science, Springer, 2007, pp. 267–280.

[21] R. G. Hamed, H. J. Pandit, D. O'Sullivan, O. Conlan, Explaining disclosure decisions over personal data, in: ISWC Satellites, Vol. 2456 of CEUR Workshop Proceedings, CEUR-WS.org, 2019, pp. 41–44.

[22] D. L. McGuinness, A. Borgida, Explaining subsumption in description logics, in: IJCAI (1), Morgan Kaufmann, 1995, pp. 816–821.

[23] A. Borgida, E. Franconi, I. Horrocks, Explaining ALC subsumption, in: W. Horn (Ed.), ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, August 20-25, 2000, IOS Press, 2000, pp. 209–213.

[24] E. Kontopoulos, N. Bassiliades, G. Antoniou, Visualizing semantic web proofs of defeasible logic in the DR-DEVICE system, Knowl.-Based Syst. 24 (3) (2011) 406–419.

[25] J. A. Robinson, A. Voronkov (Eds.), Handbook of Automated Reasoning (in 2 volumes), Elsevier and MIT Press, 2001.

[26] K. Kaljurand, N. E. Fuchs, Verbalizing OWL in attempto controlled english, in: OWLED, Vol. 258 of CEUR Workshop Proceedings, CEUR-WS.org, 2007.

[27] K. Kaljurand, ACE view — an ontology and rule editor based on attempto controlled english, in: OWLED, Vol. 432 of CEUR Workshop Proceedings, CEUR-WS.org, 2008.

[28] I. Androutsopoulos, G. Lampouras, D. Galanis, Generating natural language descriptions from OWL ontologies: the naturalowl system, J. Artif. Intell. Res. 48 (2013) 671–715.

[29] B. J. Fogg, A behavior model for persuasive design, in: Proceedings of the 4th international Conference on Persuasive Technology, ACM, 2009, p. 40.

[30] H. Oinas-Kukkonen, A foundation for the study of behavior change support systems, Personal and Ubiquitous Computing 17 (6) (2013) 1223–1235. doi:10.1007/s00779-012-0591-5.
URL https://doi.org/10.1007/s00779-012-0591-5

[31] S. Kelders, H. Oinas-Kukkonen, A. Orni, J. E.W.C.van Gemert, Health behavior change support systems as a research discipline; a viewpoint, International Journal of Medical Informatics 96.

[32] H. Oinas-Kukkonen, M. Harjumaa, Persuasive systems design: Key issues, process model, and system features, CAIS 24 (2009) 28.

[33] V. J. Strecher, G. H. Seijts, G. J. Kok, G. P. Latham, R. Glasgow, B. DeVellis, R. M. Meertens, D. W. Bulger, Goal setting as a strategy for health behavior change, Health Education Quarterly 22 (2) (1995) 190–200, pMID: 7622387. arXiv:https://doi.org/10.1177/109019819502200207, doi:10.1177/109019819502200207.
URL https://doi.org/10.1177/109019819502200207

[34] S. Michie, S. Ashford, F. F. Sniehotta, S. U. Dombrowski, A. Bishop, D. P. French, A refined taxonomy of behaviour change techniques to help people change their physical activity and healthy eating behaviours: the calo-re taxonomy, Psychology & health 26 (11) (2011) 1479–1498.

[35] M. B. Miller, E. Meier, N. Lombardi, T. R. Leffingwell, Theories of behaviour change and personalised feedback interventions for college student drinking, Addiction Research & Theory 23 (4) (2015) 322–335. arXiv: https://doi.org/10.3109/16066359.2014.1001840, doi:10.3109/16066359.2014.1001840.
URL https://doi.org/10.3109/16066359.2014.1001840

[36] K. Mercer, M. Li, L. Giangregorio, C. Burns, K. Grindrod, Behavior change techniques present in wearable activity trackers: a critical analysis, JMIR mHealth and uHealth 4 (2) (2016) e40.

[37] I. Zukerman, R. McConachy, K. Korb, Using argumentation strategies in automated argument generation, in: Proceedings of the 1st International Natural Language Generation Conference, 2000, pp. 55–62.

[38] C. Reed, D. Long, M. Fox, An architecture for argumentative dialogue planning, in: Practical Reasoning: Proceedings of the First International Conference on Formal and Applied Practical Reasoning (FAPR96), Springer Verlag, Berlin, 1996, pp. 555–566.

[39] S. Pan, M. Zhou, Pplum: A framework for large-scale personal persuasion, in: Proceedings of the 3rd Workshop on Data-Driven User Behavioral Modeling and Mining from Social Media, ACM, 2014, pp. 5–6.

[40] M. Miceli, F. deRosis, I. Poggi, Emotional and non-emotional persuasion, Applied Artificial Intelligence 20.

[41] L. Gatti, M. Guerini, O. Stock, C. Strapparava, Sentiment variations in text for persuasion technology, in: International Conference on Persuasive Technology, Springer, 2014, pp. 106–117.

[42] M. Guerini, O. Stock, M. Zancanaro, D. J. OKeefe, I. Mazzotta, F. de Rosis, I. Poggi, M. Y. Lim, R. Aylett, Approaches to verbal persuasion in intelligent user interfaces, in: Emotion-Oriented Systems, Springer, 2011, pp. 559–584.

[43] E. Guerrero, J. C. Nieves, H. Lindgren, An activity-centric argumentation framework for assistive technology aimed at improving health, Argument & Computation 7 (1) (2016) 5–33.

[44] E. Hadoux, A. Hunter, Strategic sequences of arguments for persuasion using decision trees, in: AAAI, AAAI Press, 2017, pp. 1128–1134.

[45] E. Hadoux, A. Hunter, S. Polberg, Biparty decision theory for dialogical argumentation, in: COMMA, Vol. 305 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2018, pp. 233–240.

[46] A. Hunter, Towards a framework for computational persuasion with applications in behaviour change, Argument & Computation 9 (1) (2018) 15–40.

[47] B. D. Carolis, I. Mazzotta, A user-adaptive persuasive system based on 'a-rational' theory, Int. J. Hum.-Comput. Stud. 108 (2017) 70–88.

[48] K. Tollmar, F. Bentley, C. Viedma, Mobile health mashups: Making sense of multiple streams of wellbeing and contextual data for presentation on a mobile device, in: PervasiveHealth, IEEE, 2012, pp. 65–72.

[49] F. Bentley, K. Tollmar, P. Stephenson, L. M. Levy, B. Jones, S. L. Robertson, E. Price, R. Catrambone, J. Wilson, Health mashups: Presenting statistical patterns between wellbeing data and context in natural language to promote behavior change, ACM Trans. Comput.-Hum. Interact. 20 (5) (2013) 30:1–30:27.

[50] R. Kocielnik, L. Xiao, D. Avrahami, G. Hsieh, Reflection companion: A conversational system for engaging users in reflection on physical activity, IMWUT 2 (2) (2018) 70:1–70:26.

[51] J. Schroeder, R. Karkar, J. Fogarty, J. A. Kientz, S. A. Munson, M. Kay, A patient-centered proposal for bayesian analysis of self-experiments for health, Journal of healthcare informatics research 3 (1) (2019) 124–155.

[52] M. Zhou, Y. Mintz, Y. Fukuoka, K. Goldberg, E. Flowers, P. Kaminsky, A. Castillejo, A. Aswani, Personalizing mobile fitness apps using reinforcement learning, in: IUI Workshops, Vol. 2068 of CEUR Workshop Proceedings, CEUR-WS.org, 2018.

[53] J. Freyne, S. Berkovsky, Intelligent food planning: personalized recipe recommendation, in: IUI, ACM, 2010, pp. 321–324.

[54] L. Yang, C. Hsieh, H. Yang, J. P. Pollak, N. Dell, S. J. Belongie, C. Cole, D. Estrin, Yum-me: A personalized nutrient-based meal recommender system, ACM Trans. Inf. Syst. 36 (1) (2017) 7:1–7:31.

[55] N. Leipold, M. Madenach, H. Schäfer, M. Lurz, N. Terzimehic, G. Groh, M. Böhm, K. Gedrich, H. Krcmar, Nutrilize a personalized nutrition recommender system: an enable study, in: HealthRecSys@RecSys, Vol. 2216 of CEUR Workshop Proceedings, CEUR-WS.org, 2018, pp. 24–29.

[56] T. N. T. Tran, M. Atas, A. Felfernig, M. Stettinger, An overview of recommender systems in the healthy food domain, J. Intell. Inf. Syst. 50 (3) (2018) 501–526.

[57] H. Schrder, Protective mechanisms of the mediterranean diet in obesity and type 2 diabetes, The Journal of Nutritional Biochemistry 18 (3) (2007) 149 – 160. doi:https://doi.org/10.1016/j.jnutbio.2006.05.006.
URL http://www.sciencedirect.com/science/article/pii/S0955286306001598

[58] J. Salas-Salvadó, M. Bulló, N. Babio, M. Á. Martínez-González, N. Ibarrola-Jurado, J. Basora, R. Estruch, M. I. Covas, D. Corella, F. Arós, V. Ruiz-Gutiérrez, E. Ros, , Reduction in the incidence of type 2 diabetes with the mediterranean diet, Diabetes Care 34 (1) (2011) 14–19. arXiv:https://care.diabetesjournals.org/content/34/1/14.full.pdf, doi:10.2337/dc10-1288.
URL https://care.diabetesjournals.org/content/34/1/14

[59] R. Estruch, E. Ros, J. Salas-Salvad, M.-I. Covas, D. Corella, F. Ars, E. Gmez-Gracia, V. Ruiz-Gutirrez, M. Fiol, J. Lapetra, R. M. Lamuela-Raventos, L. Serra-Majem, X. Pint, J. Basora, M. A. Muoz, J. V. Sorl, J. A. Martnez, M. A. Martnez-Gonzlez, Primary prevention of cardiovascular disease with a mediterranean diet, New England Journal of Medicine 368 (14) (2013) 1279–1290, pMID: 23432189. arXiv:https://doi.org/10.1056/NEJMoa1200303, doi:10.1056/NEJMoa1200303.
URL https://doi.org/10.1056/NEJMoa1200303

[60] A. Bach-Faig, E. M. Berry, D. Lairon, J. Reguant, A. Trichopoulou, S. Dernini, F. X. Medina, M. Battino, R. Belahsen, G. Miranda, et al., Mediterranean diet pyramid today. science and cultural updates, Public Health Nutrition 14 (12A) (2011) 22742284. doi:10.1017/S1368980011002515.

[61] G. Turconi, C. Roggi, Atlante fotografico alimentare. Uno strumento per le indagini nutrizionali.

[62] M. Dragoni, V. A. M. Tamma, Extending helis: From chronic diseases to behavior change, in: K. Stefanidis, P. Rao, D. Parry, H. Kondylakis, G. Troullinou (Eds.), Proceedings of the Second International Workshop on Semantic Web Meets Health Data Management (SWH 2019) co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019, Vol. 2515 of CEUR Workshop Proceedings, CEUR-WS.org, 2019, pp. 9–19.
URL http://ceur-ws.org/Vol-2515/paper1.pdf

[63] F. Corcoglioniti, M. Rospocher, M. Mostarda, M. Amadori, Processing billions of RDF triples on a single machine using streaming and sorting, in: ACM SAC, 2015, pp. 368–375.

[64] E. Erriquez, F. Grasso, Generation of personalised advisory messages: An ontology based approach, in: Proceedings of the Twenty-First IEEE International Symposium on Computer-Based Medical Systems, June 17-19, 2008, Jyväskylä, Finland, IEEE Computer Society, 2008, pp. 437–442. doi:10.1109/CBMS.

2008.103.
URL https://doi.org/10.1109/CBMS.2008.103

[65] K. O'Leary, L. S. Liu, J. B. McClure, J. D. Ralston, W. Pratt, Persuasive reminders for health self-management, in: AMIA 2016, American Medical Informatics Association Annual Symposium, Chicago, IL, USA, November 12-16, 2016, AMIA, 2016.
URL http://knowledge.amia.org/amia-63300-1.3360278/t004-1.3364525/f004-1.3364526/2500392-1.3364678/2499447-1.3364675

[66] A. E. Latimer, L. R. Brawley, R. L. Bassett, A systematic review of three approaches for constructing physical activity messages: What messages work and what improvements are needed?, International Journal of Behavioral Nutrition and Physical Activity 7 (1) (2010) 36. doi:10.1186/1479-5868-7-36.
URL https://doi.org/10.1186/1479-5868-7-36

[67] E. Zanchetta, M. Baroni, Morph-it! a free corpus-based morphological resource for the italian language, in: IN: PROCEEDINGS OF CORPUS LINGUISTICS, HTTP://DEV.SSLMIT.UNIBO.IT/LINGUISTICS/MORPH-IT.PHP, 2005.

[68] M. Dragoni, M. Rospocher, T. Bailoni, R. Maimone, C. Eccher, Semantic technologies for healthy lifestyle monitoring, in: International Semantic Web Conference (2), Vol. 11137 of Lecture Notes in Computer Science, Springer, 2018, pp. 307–324.

[69] J. Nielsen, T. K. L, A mathematical model of the finding of usability problems, in: Proceedings of ACM INTERCHI'93 Conference, 1993, pp. 24–29.

[70] A. Bangor, P. Kortum, J. Miller, Determining what individual sus scores mean: Adding an adjective rating scale, Journal of Usability Studies 4 (2009) 114–123.

[71] A. Némery, E. Brangier, Set of guidelines for persuasive interfaces: Organization and validation of the criteria, J. Usability Studies 9 (3) (2014) 105–128.
URL http://dl.acm.org/citation.cfm?id=2817713.2817716

[72] P. Bonte, R. Tommasini, E. D. Valle, F. D. Turck, F. Ongenae, Streaming MASSIF: cascading reasoning for efficient processing of iot data streams, Sensors 18 (11) (2018) 3832. doi:10.3390/s18113832.
URL https://doi.org/10.3390/s18113832