# Persuasive Explanation of Reasoning Inferences on Dietary Data

Ivan Donadello[1] [0000−0002−0701−5729], Mauro Dragoni[1] [0000−0003−0380−6571], and Claudio Eccher[1] [0000−0001−9643−0088]

Fondazione Bruno Kessler, Via Sommarive 18, I-38123, Trento, Italy
{donadello,dragoni,clecher}@fbk.eu

**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 behaviour. 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; ii) the effectiveness of such messages at persuading the users. Results prove that the persuasive explanations are able to reduce the unhealthy users' behaviours.

**Keywords:** Explainable AI · Explainable Reasoning · Natural Language Generation · mHealth · Ontologies

## 1 Introduction

Explainable Artificial Intelligence (XAI) aims at explaining the algorithmic decisions of AI solutions with non-technical terms in order to make these decision 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 [3]. 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 behaviour 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 behaviour are: i) the ability of providing a clear and comprehensible message regarding user's behaviour, and ii) the effectiveness of the message to *persuade* the user at adopting an 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.

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 [7]. 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 3) 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 behaviour, is detected the system shows the user a natural language message explaining the wrong behaviour and its consequences. This translation from a logic language to plain text comprehensible by humans leverages a computational persuasion framework [2] and Natural Language Generation (NLG) techniques [10]. The latter exploit dynamic and smart templates able to adapt to every persuasion strategy. The proposed system has been integrated into the HORUS.AI platform [8] and it has been validated with a mobile application within the pilot project *Key To Health* run into 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. Section 3 shows the reasoning process that checks if a user has a healthy dietary behaviour. Section 4 describes the developed template system for the automatic generation of natural language persuasive explanations. Section 5 presents the *Key To Health* project in which we deployed the system, whereas Section 6 shows its evaluation. Section 7 concludes the paper.

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

[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

## 2 Related Work

Explainable Artificial Intelligence (XAI) generally relates to strategies able to provide human-understandable descriptions of learning algorithms usually perceived as black boxes by users [1]. Here, we focused on applying XAI to the results of inference processes. Within the whole XAI research area, our aim is to generate natural language explanations of logic inferences for supporting end-users in understanding the recommendations provided by intelligent systems.

One of the first user studies dealing with explanations for entailments of OWL ontologies was performed by [13]. 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, [15] 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, [4] 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.

In order to gain an understanding of how OWL users interact with ontology axioms and constructors, the work proposed in [18] compiled a set of OWL "antipatterns". These logical and non-logical "antipatterns" correspond to the errors users frequently make in the use of OWL constructors, for example, by mis-interpreting the meaning of constructors, leading to unwanted effects (or non-effects) in the ontology. Our study of justification patterns is based on a similar idea of naturally occurring patterns in OWL ontologies, but rather than finding common errors, our aim is to identify potential aids in the ontology development process.

Besides justifications, formal proofs are considered to be the most prevalent alternative form of explanation for logic-based knowledge bases. In [17] the authors present an approach to providing 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 [5] first introduced a proof-based explanation system for knowledge bases in the Description Logic ALC. 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 exist no user studies to explore the effectiveness of these proofs. In [14] the authors proposed several graph-based visualizations of defeasible logic proofs and present a user study in order to evaluate the impact of the different approaches. The study, testing 17 participants from a postgraduate course and research staff, is based on similar task-oriented principles as the Experiments 2 to 4 presented in this paper.

Finally, as ontologies are often considered to be technical artifacts akin to software, we may regard ontology and justification comprehension as analogous to software comprehension. There has been a significant amount of work on predicting the complexity of understanding and the ease of maintaining software. In particular, seminal work described in [16], which devised a complexity metric known as cyclomatic complexity was based on the control flow paths through software. In [11] the author uses various syntactic measures such as program vocabulary and program length to calculate volume and difficulty of understanding of a program. The concept of a complexity model for OWL justifications builds upon the general idea of measuring software complexity; however, due to the difference in syntax and semantics, software complexity metrics are not directly applicable to OWL justifications.

In summary, there has been a wide range of approaches to explanation in the areas of ontologies, logics, and software comprehension, with some user studies that aim at evaluating the effects of supporting techniques. However, 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 KB-based Explainable Model

The explainable model implemented within the HORUS.AI platform relies on two main components: the HeLiS ontology [7] and the RDFpro [6] reasoner. The HeLiS ontology provides three main kinds of information:

**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.

**Monitoring knowledge** defines in the Tbox the set of rules enabling the monitoring task and the detection of undesired behaviors (hereafter called "violations").

**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.

An undesired behaviour given by the union of Tbox and populated Abox will trigger a logical inconsistency of the monitoring knowledge that has to be explained. In this paper, we do not present the full modeling process and the content of HeLiS. The reader can refer to [7] 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. For each food category, the HeLiS ontology defines both its associated positive and negative aspects. Such aspects are exploited by the Natural Language Generator module as described in Section 4.

The second component is the reasoner. Reasoning in HORUS.AI has the goal of verifying if user's dietary actions are consistent with the monitoring rules defined by domain experts, detecting and possibly materializing violations in the knowledge base, upon which further actions may be taken. Reasoning is triggered each time a user's profile or associated data are added or modified in the system, and also at specific points in time such as the end of a day or week, to check a user's behavior in such timespans. We implement reasoning in HORUS.AI using RDFpro [6], a tool that allows us to provide out-of-the-box OWL 2 RL reasoning, supporting the fixed point evaluation of `INSERT... WHERE...` SPARQL-like entailment rules that leverage the full

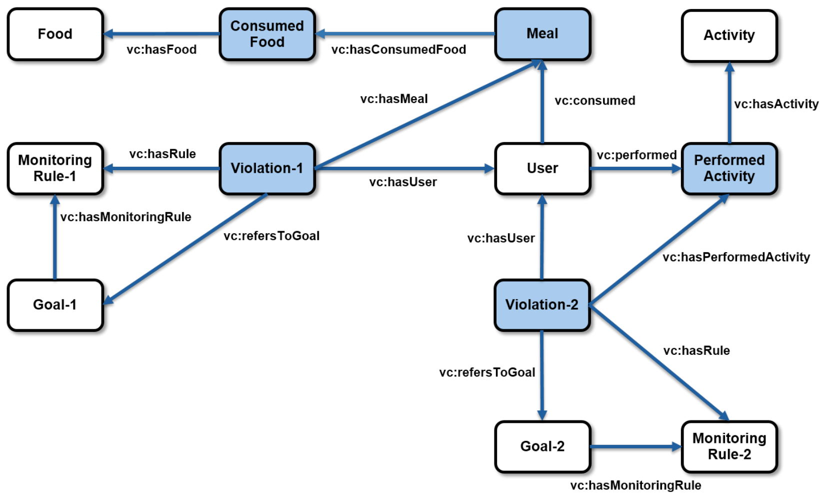

**Fig. 1.** Example describing how a violation individual is linked with the other HeLiS entities.

expressivity of SPARQL (e.g., `GROUP BY` aggregation, negation via `FILTER NOT EXISTS`, derivation of RDF nodes via `BIND`).

Figure 1 summarizes the knowledge graph generated by the reasoner. In order to understand the remaining of the section, individuals of type `Violation` contain all information about the unhealthy behaviors detected by the reasoner from user's data. While, individuals of type `MonitoringRule` contain information about the recommendations that users should follow. The descriptions of the other concepts are in [7].

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, and nutrients). 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 `Nutrient` individual (this information greatly helps in aggregating their amounts). Whereas, during the *online* phase, each time the reasoning is triggered (e.g., a new meal is entered), the user data is merged with the closed ontology and the deductive closure of the rules is computed. This process can be performed both on a per-user basis or globally on the whole knowledge base. 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 instantiating the `Violation` individual. Second, accessory information is integrated into the `Violation` individual for supporting the Natural Language Generation module in the generation of the explanation concerning the detected violation. 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 messages.

The result of the reasoning activity is a set of structured packages containing information about the detected unhealthy behaviors. By considering as example the dietary domain, each package contains: (i) the list of meals that contributed to generate the violation; (ii) the actual quantity, for a specific food category, provided by the user; (iii) the expected quantity for the same food category; (iv) the violation level (this value 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); and, (v) the violation history: the reasoner computes this value in order to provide a recidivism index about how a user is inclined to violate specific rules. This information, together with the identifiers of the violated rule and user, the rule priority, and the reference of the food (or food category, or nutrient) violated by the user, is sent to the persuasive explanation component that elaborates these packages and decides which information to use for generating the feedback sent to the user. An example of violation instance represented by using the JSON format is shown in Figure 2.

```
violation: [ userId: fb267
             violationId: violation_fb267
             ruleId: MR-MEDITERRANEAN-028-QB
             meal: MEAL-58ccf3cbfd110f24e59eeced
             history: 1
             expectedQuantity: 200
             quantity: 300
             unit: ml
             level: 1
             timestamp: 1491063927420
             priority: 1
             rule: MR-MEDITERRANEAN-001-GWEEK
             entity: SweetBeveragesAndJuices
             entityType: FOODCATEGORY
             startTime: 1491043927420
             endTime: 1491063927420
             constraint: less
             goal: MEDITERRANEAN-GOAL-D-190
           ]
```

**Fig. 2.** Example of the violation bean produced by the reasoner in consequence of the violation of a rule that limits the consumption of fruit juice to 200 ml.

## 4 Explaining Logical Inconsistencies with Natural Language

Here we present a method that performs a linguistic realization of the violation beans of Figure 2 that is useful as motivational message. This realization has to be human understandable and convince users to avoid undesired behaviours that trigger such inconsistencies. Therefore, we need i) a persuasive framework that helps users in conduct a good dietary behaviour (Section 4.1); ii) an effective natural language generator method that translates the logical language of the reasoning results (Section 4.2). Both components need the HeLiS ontology to retrieve the necessary data. Figure 3 shows the architecture of our method. The core part relies on templates (a grammar) that encode

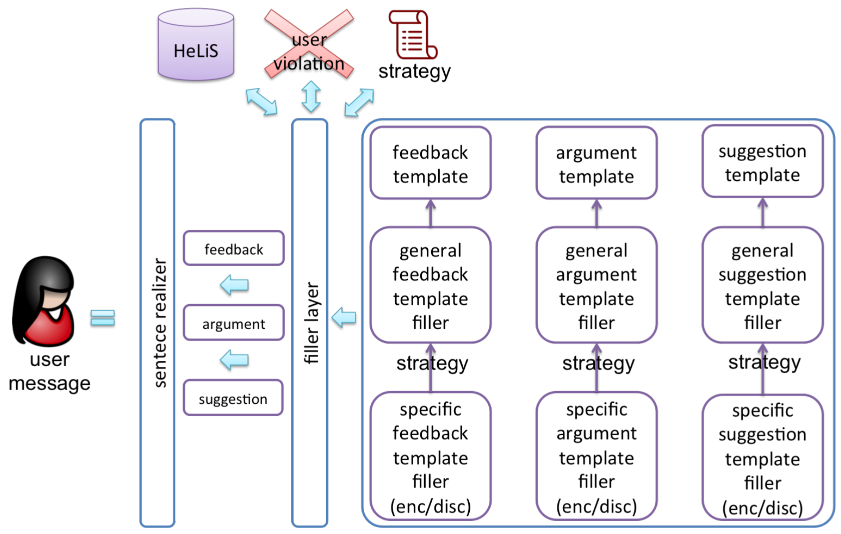
HeLiS
user violation
strategy
user message
sentece realizer
feedback
argument
suggestion
filler layer
feedback template
argument template
suggestion template
general feedback template filler
general argument template filler
general suggestion template filler
strategy
strategy
strategy
specific feedback template filler (enc/disc)
specific argument template filler (enc/disc)
specific suggestion template filler (enc/disc)

**Fig. 3.** Architecture our method: the templates are a grammar that translates a logical language into a natural one. They are organized according to persuasion strategies.

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).

## 4.1 The Persuasive Framework

We inspired our work from the theoretical framework in [2] 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 is based on four basic properties: *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.

**Persuasion Strategy** The violation bean of Figure 2 contains all the information explaining the inconsistency of the user's dietary behaviour 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 behaviour. 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 [2]) 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);
– time scheduling: the need to send particular information to the user at specific time of the day or of the week;
– 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).

The first kind of events is directly triggered by the detection (through the logical reasoning process of Section 3) 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. 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 parameter in Figure 2), 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 [2], the intention is composed by a *feedback* on user's activity, an *argument* about the consequences of user's behaviour and a *suggestion* to follow a healthy behaviour. We consider two kinds of intentions: to *encourage* or *discourage* the user to follow a healthy or unhealthy behaviour. In the example of Figure 2, the user drank too much sweet beverages, thus the intention is to discourage this behaviour.

**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 actually 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 we can use hGraphs (`http://hgraph.org/`).

### 4.2 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 3, 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, we need the total control on the generated output as wrong indications could lead to serious problems in the healthcare domain. Morevoer, 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 nonterminal/terminal symbols and production rules. The terminal symbols are selected in the filler layer module to fill the nonterminal ones according to the violation, the strategy and HeLiS. Once the 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, here Italian.

**The Template System** The template system is the organization of the templates according to the presence of nonterminal/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 nonterminal symbols, Table 1. The second layer consists of production rules between nonterminal and terminal symbols about the domain. This regards the content of the templates, see Table 2. The third layer contains rules between nonterminal 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. Indeed, our target language is Italian but the templates are the same for English and we here just translate the terminal symbols. On the other hand, if a different persuasion strategy needs to be adapted this reflects only the last layer.

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.

Table 1 shows the structure of the feedback, argument and suggestion components. This is the concatenation (symbol +) of some nonterminal 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 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 2. Some nonterminal 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

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.

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 behaviour has to be discouraged. A “greater” constraints (vegetables $>=$ 200g) implies an insufficient amount of this food and this behaviour 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

| 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.

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 foods for the suggestion template. HeLiS provides foods that are valid alternatives to the consumed food (e.g., similar-taste relation, list of nutrients, 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, foods 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.

**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, capitol 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 [19] 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 2, 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]".

## 5 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 our HORUS.AI platform and deployed and evaluated in the context of the project *Key to Health* in 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*[3], 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, HORUS.AI 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. Table 4 shows main demographic information concerning the users involved in the performed evaluation campaign. All users were in good health. Indeed, in this first pilot we decided to not involve people affected by chronic diseases or other diseases.

## 6 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 twofold. First, we present the validation performed by the domain experts with respect to the correctness and appropriateness of the generated messages (Section 6.1). This validation aims to verify that the explanations provided by the system are coherent with respect to the detected unhealthy behaviors. Second, we discuss the effectiveness of generated explanations on users' behaviors (Section 6.2) by showing how the use of explanations resulted more helpful with respect to a control group of users received punctual feedback without any detail. The evaluation of reasoning performance is out of scope of this paper. The reader may find these details in [9].

[3] 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 Domain Experts Evaluation

The first 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 [4] have been involved for modeling the templates adopted for generating the explanations. As it has been explained in Section 3, 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 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.

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

## 6.2 Effectiveness of Explanation

The second evaluation concerned the effectiveness analysis of generated explanations on the user study designed within the *Key to Health* project. The user study consisted in providing to a group of users a mobile application we created based on the services included into the HORUS.AI platform. We analyzed the usage of a 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 received the whole persuasive messages generated by using the template system. Whereas a second group of 28 users, that was our control group, did not receive any composition of feedback, argument and suggestion, but only canned text messages notifying when a rule was violated. The expectation 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 4. We considered three different kinds of dietary rules:

– QB-Rules: these rules define the right amount of a specific food category that should be consumed in a meal.
– DAY-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: 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 blue line represents the number of violations, while the red line the average standard deviation observed for each single event. Then, the green line represents the average number of violations generated by the control group and the orange one the associated standard deviation. 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 blue 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 first 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. In particular, for what concerns DAY-Rules the average number of violations per user at the end of the observed period is acceptable by considering that it drops of about $67\%$. For the WEEK-Rules, however, the drop remained limited. By considering the standard deviation lines, we can appreciate how both lines remain contained within low bounds and after a more in depth analysis of the data, we did not observe the presence of outliers.

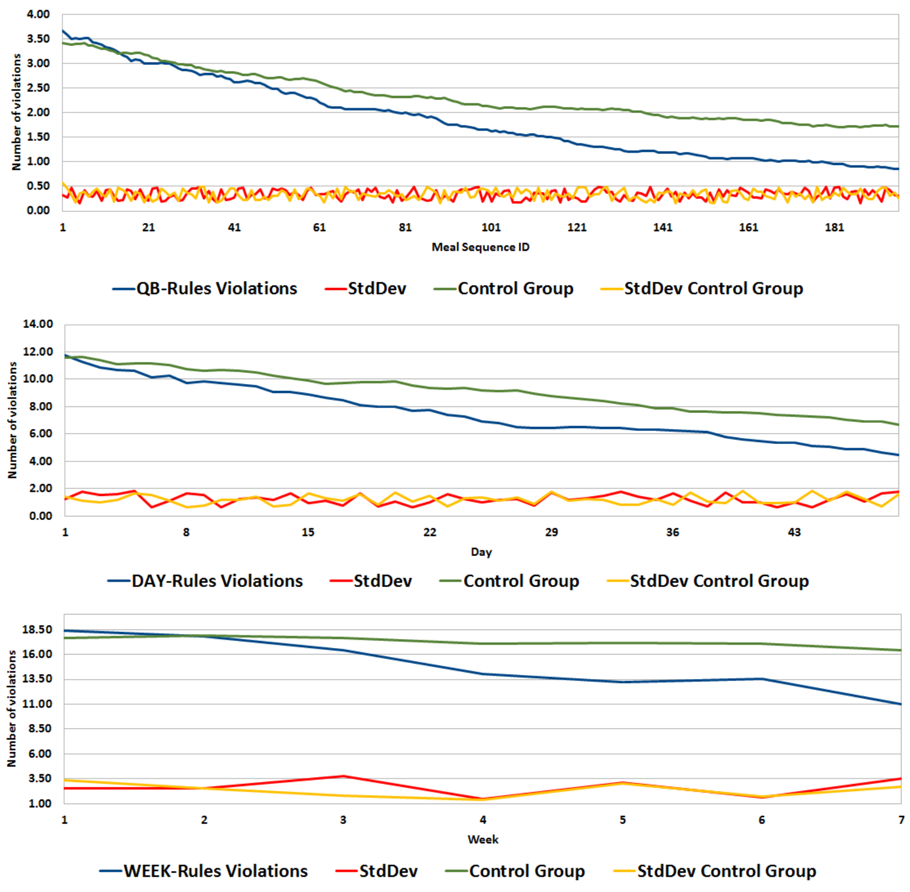

**Fig. 4.** Variation of the number of detected violations within the *Key To Health* time span.

## 7 Conclusions

We presented an explainable AI system supporting the users in following an healthy diet. The system checks the presence of unhealthy behaviours based on the food consumed 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 an healthy dietary behaviour. Moreover, the modular template systems allows the dynamic construction of natural language sentences and the templates portability in other domains. As future work, the persuasive explanations of user' behavior will be used in a Computational Persuasion framework [12] to develop a chatbot that understands the user's needs and difficulties to better persuade him/her at following healthy lifestyles.

## References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). IEEE Access **6**, 52138–52160 (2018)
2. op den Akker, H., Cabrita, M., op den Akker, R., Jones, V.M., Hermens, H.: Tailored motivational message generation: A model and practical framework for real-time physical activity coaching. Journal of Biomedical Informatics **55**, 104–115 (2015)
3. Baader, F., Horrocks, I., Sattler, U.: Description logics. In: Handbook of Knowledge Representation, Foundations of Artificial Intelligence, vol. 3, pp. 135–179. Elsevier (2008)
4. Bauer, J., Sattler, U., Parsia, B.: Explaining by example: Model exploration for ontology comprehension. In: Description Logics. CEUR Workshop Proceedings, vol. 477. CEUR-WS.org (2009)
5. Borgida, A., Franconi, E., Horrocks, I.: Explaining ALC subsumption. In: Horn, W. (ed.) ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, August 20-25, 2000. pp. 209–213. IOS Press (2000)
6. Corcoglioniti, F., Rospocher, M., Mostarda, M., Amadori, M.: Processing billions of RDF triples on a single machine using streaming and sorting. In: ACM SAC. pp. 368–375 (2015)
7. Dragoni, M., Bailoni, T., Maimone, R., Eccher, C.: Helis: An ontology for supporting healthy lifestyles. In: International Semantic Web Conference (2). Lecture Notes in Computer Science, vol. 11137, pp. 53–69. Springer (2018)
8. Dragoni, M., Bailoni, T., Maimone, R., Marchesoni, M., Eccher, C.: HORUS.AI - A knowledge-based solution supporting health persuasive self-monitoring. In: International Semantic Web Conference (P&D/Industry/BlueSky). CEUR Workshop Proceedings, vol. 2180. CEUR-WS.org (2018)
9. Dragoni, M., Rospocher, M., Bailoni, T., Maimone, R., Eccher, C.: Semantic technologies for healthy lifestyle monitoring. In: International Semantic Web Conference (2). Lecture Notes in Computer Science, vol. 11137, pp. 307–324. Springer (2018)
10. Gatt, A., Krahmer, E.: Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. J. Artif. Intell. Res. **61**, 65–170 (2018)
11. Halstead, M.H.: Elements of Software Science (Operating and Programming Systems Series). Elsevier Science Inc., New York, NY, USA (1977)
12. Hunter, A.: Towards a framework for computational persuasion with applications in behaviour change. Argument & Computation **9**(1), 15–40 (2018)
13. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.A.: Debugging unsatisfiable classes in OWL ontologies. J. Web Semant. **3**(4), 268–293 (2005)
14. Kontopoulos, E., Bassiliades, N., Antoniou, G.: Visualizing semantic web proofs of defeasible logic in the DR-DEVICE system. Knowl.-Based Syst. **24**(3), 406–419 (2011)
15. Lam, J.S.C.: Methods for resolving inconsistencies in ontologies. Ph.D. thesis, University of Aberdeen, UK (2007)
16. McCabe, T.J.: A complexity measure. IEEE Trans. Software Eng. **2**(4), 308–320 (1976)
17. McGuinness, D.L., Borgida, A.: Explaining subsumption in description logics. In: IJCAI (1). pp. 816–821. Morgan Kaufmann (1995)
18. Roussey, C., Corcho, Ó., Blázquez, L.M.V.: A catalogue of OWL ontology antipatterns. In: K-CAP. pp. 205–206. ACM (2009)
19. Zanchetta, E., Baroni, M.: 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)