IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 26, 2020, accepted June 2, 2020, date of publication June 8, 2020, date of current version June 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3000815

HealthAssistantBot: A Personal Health Assistant for the Italian Language

MARCO POLIGNANO[1], FEDELUCIO NARDUCCI[2], ANDREA IOVINE[1], (Graduate Student Member, IEEE), CATALDO MUSTO[1], MARCO DE GEMMIS[1], AND GIOVANNI SEMERARO[1]

[1]Department of Computer Science, University of Bari Aldo Moro, 70125 Bari, Italy
[2]Department of Electrical and Information Engineering, Politecnico di Bari, 70125 Bari, Italy

Corresponding author: Marco Polignano (marco.polignano@uniba.it)

This work was supported by the project DECiSION under grant BQS5153 of the Apulian INNONETWORK programme, Italy, and in part by the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska-Curie grant number 691071.

ABSTRACT In this article, we present HealthAssistantBot, an intelligent virtual assistant able to talk with patients in order to understand their symptomatology, suggest doctors, and monitor treatments and health parameters. In a simple way, by exploiting a natural language-based interaction, the system allows the user to create her health profile, to describe her symptoms, to search for doctors or to simply remember a treatment to follow. Specifically, our methodology exploits machine learning techniques to process users symptoms and to automatically infer her diseases. Next, the information obtained is used by our recommendation algorithm to identify the nearest doctor who can best treat the user's condition, considering the community data. In the experimental session we evaluated our HealthAssistantBot with both an offline and online evaluation. In the first case, we assessed the performance of our internal components, while in the second one we carried out a study involving 102 subjects who interacted with the conversational agent in a daily use scenario. Results are encouraging and showed the effectiveness of the strategy in supporting the patients in taking care of their health.

INDEX TERMS Intelligent virtual assistant, eHealth, conversational systems, healthcare, recommender systems, health, dialog, chatbot, machine learning.

I. INTRODUCTION

eHealth is defined in [1] as the practice of healthcare supported by electronic processes and communications. The report generated by McKinsey[1] well describes the phenomenon of the significant growth of interest in technology-driven innovation in the healthcare area. It shows that emerging technologies are revolutionizing the way of thinking about healthcare. People rely more and more frequently on health tracking devices, connected health devices, and personalized and proximity medicine. The key to the development of new technologies will be the use of continuous interaction models with the patient. Through innovative technologies, the patients' needs will be met quickly and effectively, allowing for timely diagnosis and continuous monitoring of the clinical status, which will reduce the risk of critical complications. Healthcare advances in this direction are going to deliver great benefits to society, bringing material improvements in average life spans and quality of life [2]. This will bring on a digital evolution of the health sector, characterized by the ability to quickly make a diagnosis, identify the hospitals, and the practitioners that are best equipped to treat the condition. From 2010 onwards, the annual investment into the area of "Health tech and Digital health" in the USA has been increased by 138%, reaching a peak of more than 30 billion US dollars. This effort is expected to increase until 2025, generating an annual revenue between 350 and 410 billion dollars. Technological progress in this domain will lead to a reduction of the costs of the healthcare system, while leaving the possibility for medical technologies to make significant improvements. It is possible to imagine different strategies to be part of this innovative process, for example, by providing tools for telemedicine or for the digitization of health data. In our case, we want to be part of this advancement by proposing an e-health approach that integrates an *intelligent virtual assistant*. Our aim is to provide patients with a tool to support their daily life tasks, making the monitoring of health parameters more convenient, and treatment more effective. We also want to focus on a tool that can be used in the Italian language. It is, in fact, essential

The associate editor coordinating the review of this manuscript and approving it for publication was Rashid Mehmood.

[1]https://www.mckinsey.com/industries/healthcare-systems-and-services/our-insights/the-era-of-exponential-improvement-in-healthcare

to consider that many older patients have difficulties with the English language, and only few eHealth support systems are available in that language. Our project goal is to overcome this limitation, in order to support Italian patients in their native language.

We considered to base our approach on an Intelligent Virtual Assistant (IVA), i.e. a system able to support the end user in performing a task faster and more efficiently than usual [3]. In recent years, IVAs have become more and more important in everyday life, thanks to the significant investment of resources by large international companies and the possibility to enrich them with an intelligent behavior[2] [4]. In particular, thanks to machine learning approaches, natural language interaction with these systems is now simple and effective. Nowadays it is possible to ask IVAs like Apple Siri, Amazon Alexa, Google Now/Home, Microsoft Cortana, to perform simple tasks for us, such as playing music, asking simple questions or requests for information [5], [6]. Moreover, such systems are able to provide a response through a voice interface similar to that of a real human, making the user feel comfortable during the dialogue [7]. These capabilities are the strength of modern IVAs that we often find already installed on many of our digital devices. In terms of system architecture, IVAs represent a sort of meta-layer of *intelligence* that acts as a hub for apps and external services. Indeed, a generic IVA typically works in two steps: first, it understands the *intent* expressed by the user, then it invokes specific services that can fulfill the request [8]. The most popular IVAs are general purpose and *multi-domain*. They are constantly updated with new features. They can tell the weather, suggest games, movies and songs, translate pieces of text, or remind planned activities [9]. Skill discovery is a challenge for them, mostly for two primary reasons: the *affordances*, or capabilities, of virtual assistants are often unclear and the number of functionalities available in IVAs is increasing rapidly [10]. Moreover, despite such a great variety of features, current IVAs have the limit of being just partially personalized or non-personalized at all [11]. As an example, if a user asks Alexa to recommend a movie, the answer is often the same for all the users, because it cannot exploit the data in the user's profile. Similarly, by asking an IVA to suggest a hospital, it will answer by identifying the closest one. Our approach would overcome these limits providing a system with vertical functionalities immediately discoverable from our user interface, completely tailored for the user profile.

We argue that very little has been done towards integrating and exploiting health data in IVAs. In fact, they offer little to no health-related functionalities, despite personal healthcare being one of the aspects that affects the quality of life the most. In addition, the operations of analysis, diagnosis and treatment performed by a specialized doctor are often complex to plan and expensive. For this reason, we think that an IVA should be provided with some functionalities to support patients, especially with chronic illnesses, by collecting information regarding her health status and by supporting her with personalized suggestions and reminders.

In this paper, we present *HealthAssistantBot* (HAB), an intelligent virtual assistant that supports patients. In particular, HAB allows: *(i)* to identify the user's condition through a Symptom Checker (SC); *(ii)* to find the best doctor for her by using a Recommender System (RS); *(iii)* to support monitoring of treatments and health parameters; *(iv)* to increase the user's awareness about related symptoms and diseases.

However, we are aware of the fact that an automatic system cannot reliably replace an experienced doctor. In case the conditions are severe or uncertain, HealthAssistantBot will help the user make an appointment with a doctor.

To sum up, through this paper we provide the following contributions:

- We design a modular IVA that allows to provide users with *personalized* services such as recommendation of doctors and the possibility to monitor treatments and health parameters.
- We release a knowledge base in Italian for mapping symptoms with diseases;
- We introduce a strategy to detect the user's condition and clinical area by relying on the symptoms described through the platform;
- We evaluate the effectiveness of our design choices in an *in-vivo* study, in which we asked users to complete tasks by using both our conversational interface, and a typical web interface.

The remainder of the paper is organized as follows: Section II presents related works in the area. Next, Section III focuses on the description of the platform. First, we will introduce the profiling strategy we exploited in this work, next we will provide details of our strategy to detect clinical area from symptoms and we will detail the module which manages the dialog with the end user. Finally, in Section IV we discuss the findings of the experiments and Section V identifies future research directions and concludes the work.

II. RELATED WORK

The area of Computer Science that focuses on developing technologies to improve health, well-being, and healthcare is commonly known as eHealth [12]. In particular, Oh *et al.* [13] described eHealth as a way to communicate with patients through technology. This definition well described the situation of the research area at its beginning. Indeed, with the diffusion of the Internet, many applications were developed, which provided health-related information to patients quickly and without the need for a phone. eHealth has become a vast area, and consequently, many categorizations have been proposed. Van Gemert-Pjnen *et al.* [14] proposed to categorize

[2]https://www.invest.mywallst.com/post/2-reasons-why-big-tech-is-so-invested-in-voice-assistants

eHealth applications based on three aspects: the purpose of the system, the device used, or the influence of the service or platform on the public healthcare system. The first aspect focuses on the functionality of eHealth systems, i.e., their capability to *support* and *manage* the cure, or to *promote prevention*. Following this definition, our HealthAssitantBot can be primarily classified as a strategy for *supporting the cure*, providing elements for simplifying the diagnosis, therapy, treatment, and monitoring of the illness. The second aspect focuses on the technologies employed by the system. In particular, we chose to distribute HealthAssistantBot through mobile devices, which are available for a large portion of the worldwide population. This allows us to reach a very wide range of users, also including people that do not have access to more complex and specialized devices. As for the third aspect, HealthAssistantBot has not the purpose to directly support the public healthcare system. Instead, we aim to assist patients by reducing the complexity of their most common tasks in health domain, in order to contribute to the public healthcare system indirectly. Examples of patient-driven health care services and platforms with characteristics similar to our HealthAssitantBot are Eliza [15], Florence,[3] HealthTap Dr.A.I.,[4] Babylon Health,[5] Melody.[6] Each of them has its peculiarities, but none of them integrates self-diagnosis, treatment management, monitoring of health parameters, and doctor recommendation functionalities into the same tool. Moreover, they are all only available for the English language.

Eliza is one of the first conversational systems able to emulate the style of a psychotherapist. It is a straightforward system based on simple, non-personalized pattern-matching answers. Currently, some implementations of Eliza are still available online [16], but the lack of up-to-date functionalities leave it as a simple research prototype.

Florence is a health chatbot whose primary purpose is to generate reminders for medications and treatments. An additional medical dictionary function has been recently added. Through this function, the user can ask for information about diseases and symptoms. The interaction model makes use of buttons to make the dialogue straightforward. Although it may seem very similar to our HealthAssistant bot, the main differences concern personalized services. Specifically, our system adds the ability to identify the user's condition based on the mentioned symptoms. In addition, the functionality that suggests doctors with a personalized content-based strategy adds further value to our HealthAssistantBot.

Dr.A.I. is an application developed by HealthTap, and has telemedicine consultations as its strong point. Specifically, users make requests for information, medical advice, and diagnosis of illnesses to medical staff. The system works thanks to the vast availability of doctors on the platform. Thus, it does not rely on automatic diagnosis.

Babylon Health is an application that offers a chat-like interface. Once the user has started a chat with the digital assistant, she can provide her health status, and answer additional questions that are asked by the system. At the end of the consultation, Babylon proposes to locate a nearby hospital, or book a consultation in video-conference with a doctor. Therefore, the user is not provided with any kind of self-diagnosis or care support service.

Melody was developed and distributed in October 2016 by the Baidu research laboratories. The chatbot is part of the Baidu Doctor application, which was created as a platform for patients to meet their doctors. The objective of Melody is not to replace a professional doctor, but to provide a first basic diagnosis, so that the user can quickly understand if the symptoms require a visit from a doctor. It works only in the English language, and does not provide the user with any additional care management services.

As mentioned earlier, HealthAssistantBot integrates most of the functionalities offered by the aforementioned systems into a single tool. Some of its added values are the ability to recommend doctors, and the interaction through a conversation in natural language. In recent years, many works have been proposed in the literature on these two topics [17]–[20]. Narducci *et al.* proposed in [17], [21], [22] a health social network for connecting patients. The platform, named HealthNet, was designed to record users' information such as symptoms, diseases, and treatments for recommendation purposes. In particular, a list of doctors and a list of similar patients with whom to connect was proposed. Moreover, it proposed a preliminary auto-diagnosis tool, which can detect the clinical area of interest from the user's symptoms. This system is very similar to the one presented in this work. However, it was based on a classical user interface, accessible through a browser. In our work, instead, we focus on a strategy based on the interaction with an intelligent virtual agent, that more closely mimics the way people usually communicate with others.

A collaborative recommender system of primary care doctors has been proposed by Han *et al.* in [23]. In that work, the authors focused on matching patients with doctors that they are willing to consult with a high sense of trust. A similar scenario is also investigated by [24] that proposed a strategy based on machine learning for recommending the best doctor to the final user, considering her health parameters.

Cordero *et al.* [25] proposed a recommender system based on fuzzy rules for the diagnosis of schizophrenia, schizoaffective, and bipolar disorders. The user can dialog with the system describing her symptoms. At every step of the conversation, the system applies a fuzzy closure operation to refine its decision. When the user has provided enough symptoms, the system generates a recommendation, i.e. the diagnosis. This is an interesting example of auto-diagnosis tool. However, it is limited to the schizophrenia-related disorders. Our

[3]http://www.florence.chat
[4]https://www.healthtap.com/for-members
[5]https://www.babylonhealth.com/
[6]https://www.topbots.com/project/chinese-baidu-bot-ai-doctor/

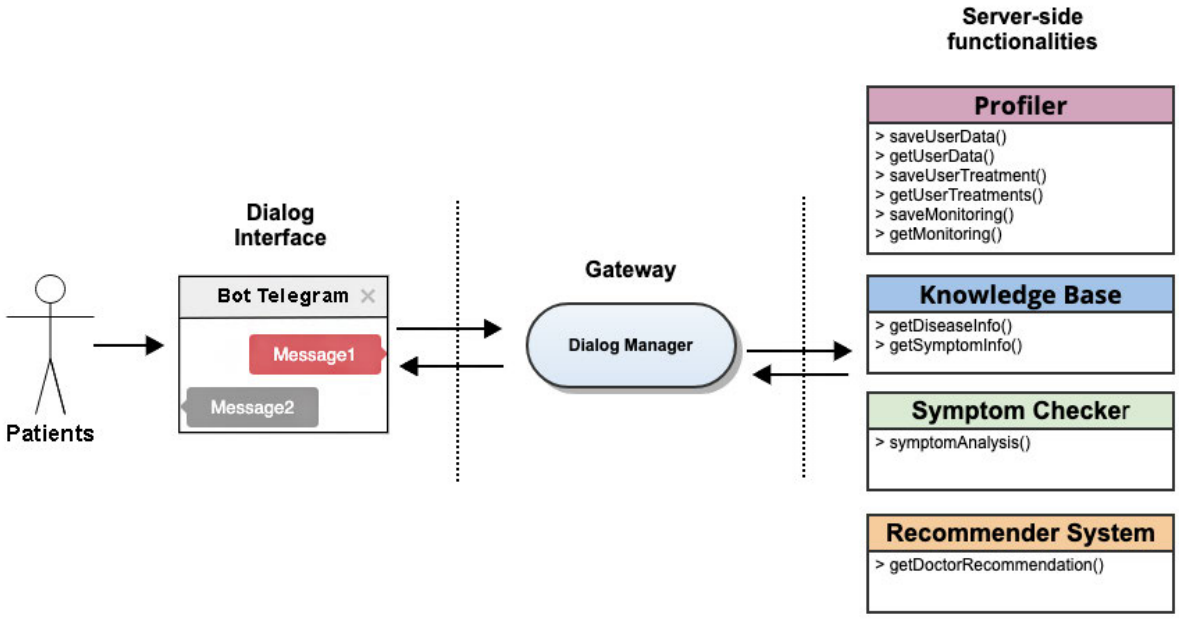

FIGURE 1. The architecture of the HealthAssistantBot here proposed.

work extends on this by including a much wider range of medical conditions.

A contribution to Conversational Recommender Systems is proposed by Iovine *et al.* [20] and Narducci *et al.* [26]. The authors discuss the possibility of integrating Conversational Recommender Systems with Digital Assistants. In particular, they implement ConveRSE, a framework for Conversational Recommender Systems. During the conversation, the user provides her preferences to the system via natural language messages. Based on these preferences, the system will generate a set of recommendations, that will be evaluated by the user. At the surface level, there seem to be many similarities with the solution proposed in this paper. However, the focus of this paper is mainly on using conversation for recommendation purposes. In our work, recommendation is only one of the functionalities offered by HAB. Moreover, the dialog structure is completely different in order to fit the medical domain.

For a comprehensive review of Recommender Systems in healthcare domain, it is possible to consult the works of Wiesner and Pfeifer [27], Hors-Fraile *et al.* [28] and Afolabi and Toivanen [29], that describes the area in detail analyzing challenges and research opportunities. Jannach *et al.* [30], provide, instead, a complete survey about the topic of Conversational Recommender Systems.

The literature about Symptom checkers is also extensive, and limits and opportunities of its use are often discussed. An example is provided by Bisson *et al.* [31]. They observed that patients show difficulties in identifying the exact cause of their pain in a list of diseases, even if a symptom checker can provide them the correct one in the list of first five results. This situation can suggest us some issues in the use of this tool in a real scenario. On the other hand, Morita *et al.* [32] discusses their importance, especially in situations in which it is not immediately possible to consult a doctor, such as in rural areas. Taking care of these considerations, we decided to include a symptom checker functionality in our system, which can be useful for users that would otherwise not have access to fast medical diagnosis. In particular, we decided to base the tool on machine learning approaches due to its efficacy in the domain, as shown by the encouraging results obtained in [33], [34].

III. HEALTH ASSISTANT BOT

The main goal of *HealthAssistantBot* (HAB) is to support the patient in many of the most common tasks she faces daily to support her health-related goals or tasks. Fig. 1 shows the architecture of HealthAssistantBot. Specifically, by observing the interaction from left to right, the user starts the conversation by selecting one of the commands presented in the IVA interface. Then, the request is captured by the Intent Recognizer module included in the client runtime, and sent as a push message to the Dialog Manager implemented as a server-side service. This module is responsible for receiving the user intent and redirecting the request to the server-side functionalities. We organized the functionalities into four modules: *Profiler*, *Symptom Checker*, *Knowledge Base* and *Recommender System*. The Profiler is an ecosystem that manages the patient's profile and stores clinical data, including generic user information such as gender, age, name, treatments, and health parameters. The Symptom Checker identifies the disease and its clinical area from the symptoms described by the user. The Knowledge Base contains information about diseases and symptoms, and allows users to

consult details about them in more detail. The Recommender System will use the Symptom Checker and Knowledge Base to identify a list of doctors capable of treating the patient. The different modules that compose HAB will be detailed more in the following sections.

To make this process intuitive and straightforward, we designed HAB as a Conversational Agent (CA) that interacts with the end-user in natural language. In order to correctly interact with the user, a CA has to address two tasks: Intent Recognition (IR) and Entity Recognition (ER). The first concerns the identification of the intentions and needs of the user from the message. The second aims to identify any mention to entities or keywords that are necessary in order to understand the details of the user needs. As demonstrated by the comprehensive analysis carried out by Braun *et al.* [35], IR and ER tasks can be challenging when the user is free to write everything she thinks. In that work, the authors compared numerous NLU tools available for performing IR and ER tasks, and showed their real performance in several domains related to customer support. The experiment concluded that the recognition accuracy of the tools ranged from not optimal to poor.

In domains such as health, where accuracy is essential, it is preferable not to leave the dialogue totally free, but to constrain it through alternative methods of interaction in order to reduce ambiguity on the input. Consequently, our intention is not to develop a system akin to Amazon Alexa, Apple's Siri, Microsoft Cortana, or Google Assistant, in which free text is predominantly used. On the contrary, we want to develop a system that can deal with ambiguities effectively, and that can respond to the end user's need in as few dialogue steps as possible.

Therefore, we decided to build our CA by exploiting the famous instant messaging platform Telegram.[7] The HealthAssistantBot uses Telegram as Dialog User Interface. That means HealthAssistantBot is accessible via the Telegram App. The use of Telegram is a non-binding designing choice and it does not prevent the migration of the chatbot to other platforms (e.g. Facebook messenger). The use of Telegram as a platform for releasing the bot has enabled us to not implement the user interface from scratch and to reach a large number of people for the experimental evaluation in an easy way.

The interaction with the agent (also known as chatbot) is guided by "commands". By using such commands, the user can activate specific functionalities of the CA. Thanks to the adoption of guided commands, the Intent Recognition task is more robust to interpretation errors. Indeed, each intent the user can express is uniquely mapped to a descriptive label which is identified as a "command," i.e., preceded by the "/"" character. "/start" (Fig. 2) is a typical example of initial command sent to an agent for beginning the interaction. It is used to create a dialog session between the user and the chatbot, and to initialize all the functionalities. Each entity

[7]https://telegram.org/

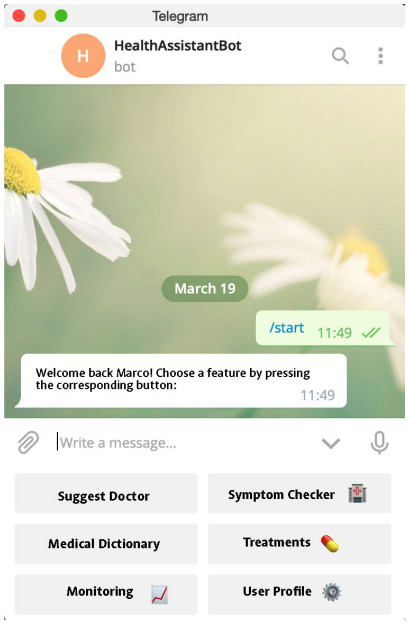

FIGURE 2. Landing page of the HealthAssistantBot.

which is necessary for the execution of the conversational agent is explicitly requested to the user, and its acquisition is performed by going through the answer provided by the user. An exhaustive list of the commands supported by the chatbot is reported in the Table 1.

TABLE 1. List of the commands supported by the HealthAssistantBot.

Command	Functionality
/start	It is used for starting the interaction with the chatbot
/menu	It is used for coming back to the first dialog screen
/suggestdoctor (Suggest Doctor)	It is used for opening the functionality of doctors recommendation
/symptomchecker (Symptoms Checker)	It is used for opening the functionality of disease detection from user symptoms
/medicaldictionary (Medical Dictionary)	It is used for opening the functionality of medical term description
/therapy (Treatments)	It is used for opening the functionality for scheduling treatments
/tracking (Monitoring)	It is used for opening the functionality for monitoring user biological health parameters
/user (User Profile)	It is used for opening the functionality for managing the personal used data

A. THE DIALOG INTERFACE

HealthAssistantBot has been designed as a client-server architecture, that keeps the logical components of the bot separated from those related to the dialogue interaction. The graphical user interface has been designed to make the interaction through the dialog intuitive for the final user. To achieve this goal, we use a combination of textual elements

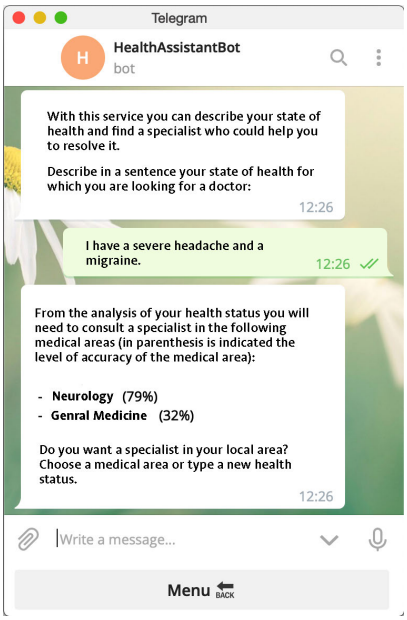

FIGURE 3. The interaction between user and chatbot about the recommendation of doctors - step 1.

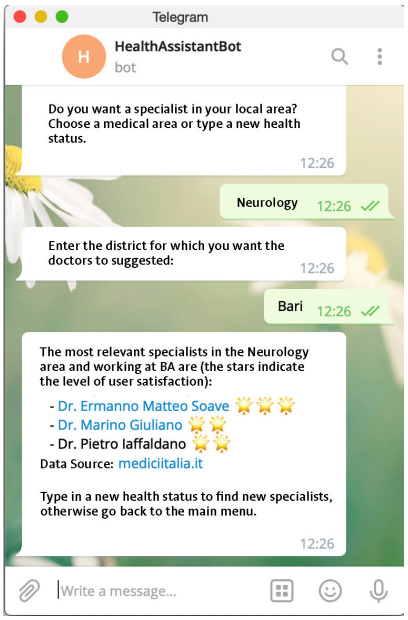

FIGURE 4. The interaction between user and chatbot about the recommendation of doctors - step 2.

(such as commands and free text), and graphical components (such as buttons and multiple-choice menus).

An example of the interface proposed to the user is shown in Fig. 1. Specifically, we decided to divide the interaction into six main functionalities: doctor recommendation, symptom analysis, medical glossary, treatments, monitoring, and user profile. In order to obtain a *recommendation*, i.e. a list of doctors relevant for the user, it is possible to click on the first button on the top left of the screen (''Suggest Doctor''). In this case, the system will ask the user to describe the symptoms by writing a text message, and to select one of the clinical areas automatically identified by the system. Next, the user can provide her province of residence (if it is not already stored in her user profile). Finally, she will obtain a list of nearby doctors that can solve her problems. An example of the output of this process is available in Fig. 3-4. The second button on the top (''Symptom Checker'') accesses the *symptom analysis* feature, that allows the user to use HAB as a self-diagnosis tool. Like in the previous case, the patient describes her symptoms in a text message, and the system will reply by providing a disease that is compatible with the symptoms. The results presented to the user include a description of the disease, supported by the confidence level of the prediction, and a link to Wikipedia with more details about the disease (Fig. 5). The primary purpose of the *medical dictionary* functionality (the middle left button in Fig. 2 - ''Medical Dictionary'') is to provide the user with a medical encyclopedia. The user can input a medical term on which she wants to obtain more information to the platform. The chatbot responds with a short description and a Wikipedia link. The response is shown in Fig. 6.

The *Treatment Management* functionality regards the storage and the management of the various medical treatments of the user (''Treatments'' in Fig. 2). For this use case,

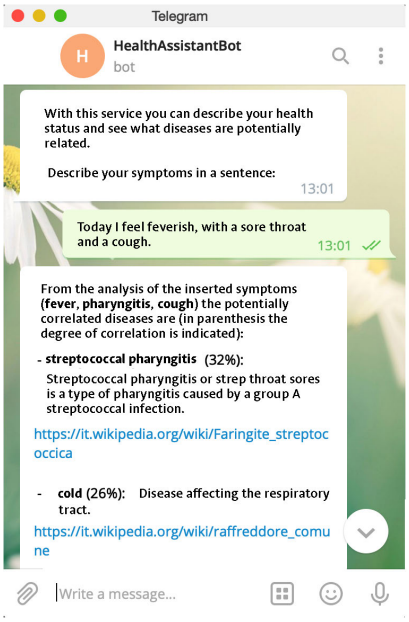

FIGURE 5. The interaction between user and chatbot about the symptom checker functionality.

the system will ask the user to provide the necessary information: the name of the treatment; the dosage of the medicines; the times at which the system should generate reminders; the frequency of the treatment (i.e. daily, alternate days, only certain days a week); and the last day of the treatment (Fig. 7-8). The last is an optional field. After entering a treatment, a summary of the therapies is shown. The user can choose one of them to see more details, or she can change/delete them (Fig. 9. The system will use the information provided

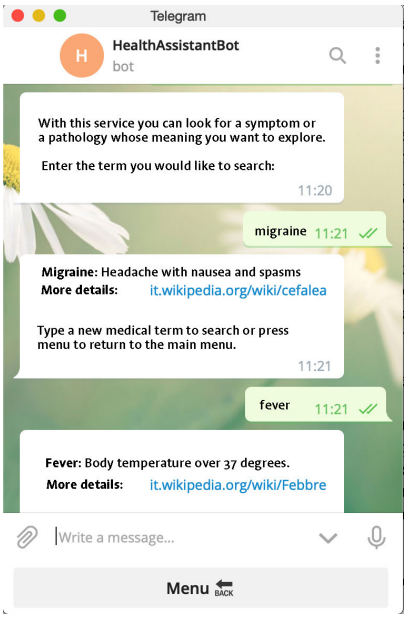

FIGURE 6. The interaction between user and chatbot about the consultation of the medical dictionary.

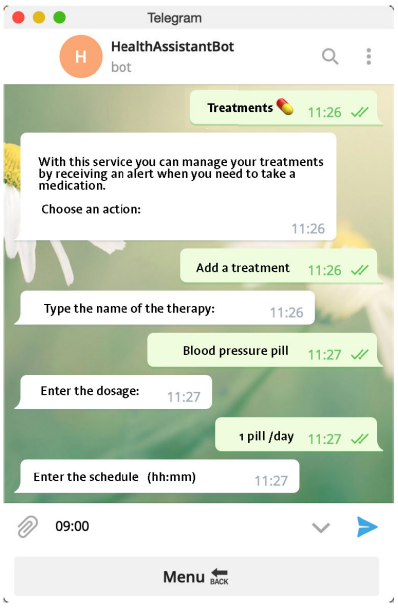

FIGURE 7. The interaction between user and chatbot about the management of treatments - step 1.

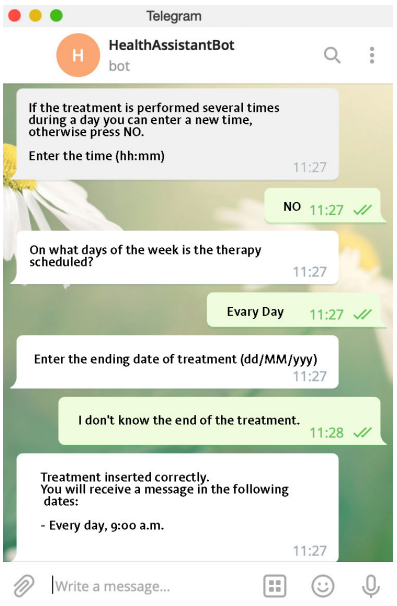

FIGURE 8. The interaction between user and chatbot about the management of treatments - step 2.

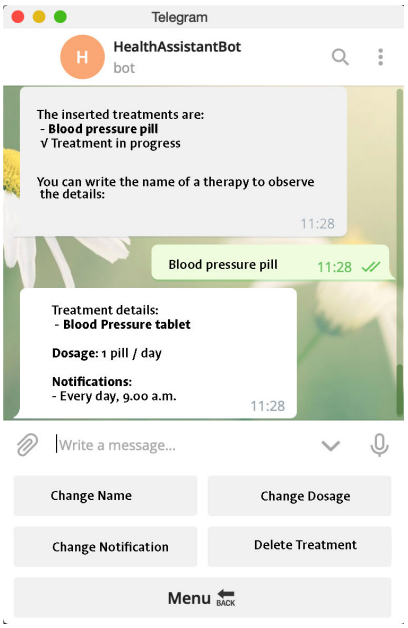

FIGURE 9. The interaction between user and chatbot about the management of treatments - step 3.

to send periodic notifications to the patient, reminding her of any medications to take, or activities to be performed in order to correctly follow her medical treatment.

The *monitoring* functionality (bottom left button - "Monitoring") allows the user to store health parameters that are obtained by scanning a report containing medical analysis. The acquisition is performed through a standard OCR tool (Tesseract library[8]). The content of the document is indexed using Apache Lucene,[9] so that it can be consulted at the end of the acquisition phase simply by writing the name of a health parameter contained in that document though a search function. It is also possible to modify and delete the indexed data if the user wishes so. The last functionality available to the user is the consultation of the *user profile* stored in the platform (last button on right - "User Profile"). In this section, it is possible to consult the personal data collected by the CA

[8]https://github.com/tesseract-ocr/tesseract

[9]https://lucene.apache.org/

and, in case, to modify or delete them in line with the most recent European rules regarding privacy [36].

B. GATEWAY

The gateway module of our platform routes user requests to the corresponding server-side functionality. This module is implemented as a RESTFul web service that provides high-level APIs to allow the client to interact with the various features of the chatbot. The connection between chatbot and gateway is made possible through the web-hook technique, that allows the client to receive requests in push mode. The gateway is distributed on the Heroku cloud platform [10] that grants free HTTPS access with SSL certificates, which are essential to communicate with the interface managed by Telegram.

C. SERVER-SIDE FUNCTIONALITIES

1) PROFILER

The Profiler module takes care of the persistence of all user information by using a PostgreSQL database.[11] Specifically, the module stores user demographic data, treatments and monitoring. This is a fundamental module to guarantee creation, reading, update and deletion of user profile data.

2) ITALIAN KNOWLEDGE BASE OF DISEASES AND SYMPTOMS

An IVA that supports the user in monitoring treatments, symptoms and diseases cannot work properly without a detailed knowledge base regarding these aspects. One of the most widely used resources in medicine regarding medical diagnosis is the ICD (International Classification of Diseases) catalogue. The tenth Revision (ICD-10) [37] is a classification system that has been established and maintained by the World Health Organization (WHO) with ten other International centers. ICD-10 provides guidelines to make the collection, treatment, categorization, and presentation of diseases universally comparable. By using ICD-10, it is easy to compare morbidity data (i.e. the percentage ratio between the number of cases of disease and the number of population) and mortality data. ICD-10 provides the most elaborate data for the degree, safety, and effectiveness of medical service measurement compared to other classifications. It provides more accurate diagnostics of patient health status (at a very fine level of granularity) and advanced data for epidemiological studies. This classification is available in a multilingual format, which includes Italian.

The main limitation of this classification is the absence of descriptive details of symptoms and diseases. In addition, the probability that a disease can cause one or more specific symptoms (symptom distribution) is not reported. This absence makes the task of automatically predicting a disease given the symptoms very complex and error-prone. In such a delicate application domain, it is necessary to be as accurate as possible, thus we decided to extend the information available in the ICD-10 with the information on Wikipedia [12] and Symcat Symptom Checker.[13] First of all, the ICD-10 diseases and symptoms were mapped to those in Symcat. This task allowed us to obtain the symptoms' distribution. Consequently, we extracted the corresponding descriptions from Wikipedia by matching the names of the diseases and symptoms with the title of the Wikipedia pages. Levenshtein's distance was used as a metric to rank the pages, and a manual check was performed to ensure of the correctness of the match. In total, 801 diseases and 474 symptoms were scraped out in English. For each disease we obtained: a description, a Wikipedia link and a list of symptoms with their likelihood of occurrence. Each symptom is, instead, characterized by a description, a Wikipedia link and a list of symptoms related to the current one.

The mapping to the Italian language has been carried out by querying BabelNet API [38] by providing the name of the symptom or disease as input. We use Babelnet instead of a classic translation provider such as Google Translate[14] because we performed the translation of a single word without its context of use. This scenario is challenging for an automatic translation system such as Google Translate. On the contrary, it is more simple if we search the term in a dictionary resource. Babelnet is a multilingual encyclopedic dictionary with lexicographic and encyclopedic coverage of terms. Specifically, for each keyword it returns one or more synsets, i.e. a set of synonyms that can be described by a single meaning. The same word, therefore, can be found in different synsets if it has different senses (meanings). If BabelNet returned only one synset, we simply extract the name, synonyms, descriptions and Wikipedia links. On the contrary, when BabelNet returned more than one synset, it is necessary to perform a manual disambiguation between them. At the end of this translation phase, only a portion of the symptoms and diseases have been translated. Consequently, we decided to focus only on common diseases, which were manually translated. This process allows the generation of an Italian dataset composed of 217 diseases and their associated 322 symptoms. The complete knowledge base of symptoms and diseases obtained by this process has been distributed through a public repository.[15] The absence of similar resources in the Italian language makes it an interesting contribution for future works in this domain of application.

In order to make the *medical dictionary* easier to consult, the knowledge base has been further extended by mapping the above mentioned diseases and symptoms with those of the 'Italian Consumer-oriented Medical Vocabulary' (ICMV).[16] The ICMV vocabulary acts as a link between medical language and the language used by patients, and

[10] https://www.heroku.com/
[11] https://www.postgresql.org/
[12] https://www.wikipedia.org/
[13] http://www.symcat.com/
[14] https://translate.google.com/
[15] http://www.di.uniba.it/~swap/repo/symptoms_diseases_KB_ITA.zip
[16] http://ehealthwiki.fbk.eu/index.php/Pagina_principale

aims to provide commonly used descriptions of terms related to medical terminology. 169 of the most common diseases and 236 symptoms have been manually mapped. Remaining elements have been left in their original form. For each symptom and disease, we have synchronized the following fields: ID, name, and description. The information obtained has been indexed on Apache Lucene, allowing quick keyword searches (using the TF-IDF technique [39]) by the user of the HealthAssistantBot (Fig. 6).

3) SYMPTOM CHECKER

One of the most important features of our HAB is the identification of the disease starting from symptoms that describe the user's condition. This functionality has been developed to allow the recommendation of doctors who can support the user in treating her condition. Beyond this, the Symptom Checker can act as a preliminary self-diagnosis tool to support doctors in the diagnosis process. The process of disease identification is carried out by following three steps: *data collection*, *identification of symptoms from the text*, *classification*. First, when the user accesses the functionality through the HAB, it is requested to briefly describe her current state of health with a sentence written in natural language, pointing out any symptoms and abnormal situations. In this phase of *identification of entities*, we used the TagMe tool [40], an entity linking system that is able to annotate n-grams in a sentence with the corresponding entities contained on Wikipedia. Each entity identified by TagMe is then compared with the names of the symptoms present in our knowledge base, and in case it is identified, it is stored in the user profile. With the information collected through the *data collection* and *name entity extraction* steps, the Symptom Checking problem can be formalized as a *multiclass classification problem* where:

- Diseases represent the classes of classification. We are going to try to predict the 217 diseases contained in the dataset;
- Symptoms represent features. The training dataset contains 322 binary features, each of which represents the absence or presence of a symptom.

The classifier takes the set of symptoms recognized by name entity recognition tool as input, and uses it to return a ranking of possible related illnesses.

$$h_\theta^{(i)}(x) = P(y = i|x; \theta) \quad (i = 1, 2, 3 \ldots 217) \tag{1}$$

In particular, we train a classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$, with x input vector and θ model parameters. The model has been trained on a synthetic dataset of patients, better described in Section IV-A. Various classification algorithms have been evaluated during the experimental session.

4) RECOMMENDER SYSTEM

The recommender system of doctors implements a content-based paradigm, which is more robust than collaborative approaches in cold-start situations (i.e. few users) [41]–[43]. The classic collaborative-filtering recommendation systems based on the ratings left by users on the platform needs a sufficient number of active users in order to take advantage of the *wisdom of the crowd*. Therefore, we have decided to use a strategy based on *item descriptions* [44]. In our case, the description of the user is given by the set of her symptoms, whereas the description of the doctor is given by his clinical area and the location of workplace. The list of doctors available on HAB is extracted from the site *medicitalia.it*. For each doctor, we collect the following information:

- *General information:* name, surname, personal website;
- *Score of satisfaction:* a score indicating the degree of satisfaction for the doctor consultancy by users of the site;
- *Medical areas:* the list of medical areas where the doctor consults;
- *Medical facilities:* the list of medical facilities (public or private) where the doctor receives or performs consultations.

Given the symptoms described by the user in natural language, we used the Symptom Checker module to obtain the two most likely diseases associated with the user. Next, we exploited such symptoms to trace back to the clinical area of reference. This was made possible thanks to a manual clustering of the 217 diseases in 24 main medical areas as already proposed in [17].

For example, if a patient describes her health status with the following sentence *"I have shortness of breath and chest pain"*, the medical text classifier assigns as the most likely class the cardiology medical specialization, while the following description "I have heartburn and reflux" is assigned to Gastroenterology and digestive endoscopy. We used the clinical areas obtained by the classifier to select doctors that work in the corresponding medical field and that operate in the same province as the final user. Finally, the list was sorted according to the score of satisfaction assigned by users to each doctor on medicitalia.it.

D. PRIVACY AND CONTROL MECHANISMS

In our work, we have decided to investigate the line of research regarding privacy for our prototype only marginally. We have proposed an architecture that could support the construction of a transparent, conversational agent that follows the guidelines of the recent GDPR regulations [36]. In the future, before the public release, we will implement a more privacy-conscious user profiling strategy, in which the end-user has to explicitly decide which facets of her profile she wants to save and make available for use in the platform. This will increase the user's control and awareness of the information encoded in the user model.

Currently, the platform grants complete user control over the insertion, modification, and deletion of her personal information, including demographic information and information about treatments, symptoms, and biometric

measurements. For the evaluation of the platform with real users we acquired their agreements to collect, manipulate, and publish their data in an aggregated and anonymised form. However, a more sophisticated management of user privacy will be required for real-use implementations. Two promising approaches, demonstrated effectiveness for other applications, that could be exploited successfully in HealthAssistantBot are the federated learning models [45]. Federated learning models have the advantage of exploiting decentralized (virtual) databases without the need of sharing private data with centralized servers. This is a very promising solution successfully adopted in the healthcare domain [46] In this way, a user can enter her own symptoms to get personalized suggestions, and her data will remain on her own device. Another interesting aspect to be taken into account is related to the changes that the European legislation brings in the territory of European Member States (Regulation EU/679/2016 or GDPR, General Data Protection Regulation). eHealth processes are involved in this regularization, in particular in Article 4 n. 15 of the GDPR that refers to ''personal data relating to the physical or mental health of a natural person, including the provision of health care services''. In accordance with art. 9 GDPR, data relating to the health of the person benefit from prohibition of management by third parties, except in the case of different exceptions in favor of the processing of health data for research and public health preventive medicine. In our opinion, federated learning is the right solution for meeting the abovementioned requirements. Furthermore, when we will make our bot public, we will provide the user with privacy management tools as described in [47]:

- *Openness and Transparency.* We will inform the final user through Terms of Service. In it, we will explain what personal data is collected, who has access to this data, how it is manipulated, and where it is stored. We will provide tools for downloading personal data collected and ask us to completely delete them.
- *Collection Limitation and Data Minimisation.* We will use the data only for the purposes we request the authorization. In case of different purposes of use, we will ask the user authorization again for the new way of use or manipulation.
- *Use Limitation and Data Integrity.* We will inform the user about any possible use of them. Moreover, we will guarantee data integrity and proper security encoding while they are moved across the network.
- *Individual Participation and Control.* The users will have all the rights and functionalities to insert, modify or delete their personal data from our server without any violation of Terms of Service.
- *Security Safeguards and Controls.* All security precautions, in the management, manipulation and transmission of personal data will be properly taken.
- *Accountability and Oversight.* The entity/company that will release the product will be fully responsible under the current European rules for any violations concerning the platform users' privacy and security.

IV. EXPERIMENTAL EVALUATION

The goal of the experimental evaluation is twofold:

- to evaluate the effectiveness of the internal components of the system, in particular the pathology prediction methodology;
- to assess whether HAB can be exploited to support the patients in their health-related tasks.

More specifically, in experiment 1, we test the accuracy of different classification algorithms to automatically associate the most relevant disease to a list of health problems (symptoms); in experiment 2, we asked 102 subjects to evaluate the effectiveness of the platform functionalities, the satisfaction during use, and the simplicity of the interaction.

A. DATASET AND EVALUATION MEASURES

To the best of our knowledge, there is no dataset available with real or synthetic information that associates the patient's symptoms to her condition. Moreover, due to privacy laws, real clinical data may not be publicly available even if anonymized. Therefore, we follow the strategy already used in [34] for dealing with the same limitation. For each of the 217 diseases, we generated k synthetic patients whose symptoms follow the probability distribution α for that disease. In particular, each patient k_i is associated with a vector of symptoms s where each component s_i can be 1 with a probability α and 0 with a probability $1 - \alpha$.

We chose 100, 1000, 2500, 5000 as values for the k parameter. This led to the creation of datasets consisting of 21700, 217000, 542500 and 1085000 instances. The final dataset dimension is, consequently, equal to $l \times s$, where l is equal to the size of the dataset, and s is equal to 322 columns, one for each symptom in our knowledge base, and 1 additional column for representing the disease (our classification classes). In some cases, the output vector from this procedure can be a vector of zeros that produces no new knowledge in the dataset. In these cases, the vector is removed from the set and another non-empty one is generated.

In order to use the dataset for both training and testing, we performed a random stratified split, using 70% as training set, 10% as validation set and 20% as test set.

In experiment 1, we evaluated the effectiveness of the classification model, i.e. the ability to assign a disease to an initial set of symptoms correctly. To measure the performance, we used *Accuracy@n* (Eq. 2), *Precision@n* (Eq. 3), *Recall@n* (Eq. 4), and *F1@n* (Eq. 5), on lists of top-n items. These metrics are computed in terms of the contingency table for each category c_i (disease) on the given test set.

$$acc = \frac{TP + TN}{total} \tag{2}$$

$$\pi = \frac{TP}{TP + FN} \tag{3}$$

$$r = \frac{TP}{TP + FN} \tag{4}$$

TABLE 2. Contingency table for each category c_i.

Contingency Table for the disease c_i			
Category c_i	Gold Label		
		YES	*NO*
Classifier result	*YES*	TPi	FPi
	NO	FNi	TNi

$$F1 = 2.\frac{\pi \,.\, r}{\pi + r} \tag{5}$$

We decided to perform a macro-average in order to aggregate the metrics for each class: scores are calculated independently for each class, and then they are averaged together. This is possible due to the fact that all classes are perfectly balanced in the dataset.

In experiment 2, we evaluated the accuracy of our Symptom Checker in a real scenario, and recorded the level of user satisfaction and simplicity of the interaction during the use of HAB. To assess the accuracy of the Symptom Checker, we asked users to provide the symptoms of a condition that was previously diagnosed by a doctor. The Symptom Checker module will work correctly only if it will be able to identify the same disease. The performance of the algorithm was obtained by calculating the percentage of diseases that were correctly identified by the Symptom Checker. This score is equivalent to the accuracy metric described in Eq. 2. In addition to this, users were asked to answer a questionnaire (Table 7, composed by 11 questions focused on the symptom checking task (QUEST_a).

To investigate the user satisfaction and the simplicity of use of the chatbot, we asked users to complete another questionnaire (Table 8) consisting of 15 questions. This questionnaire was submitted to the users after having used the chatbot for one week. To evaluate the answers, we used both 5-point Likert scale (1 is the lower score) and binary answers (through checkbox) (QUEST_b).

B. EXPERIMENT 1

The goal of Experiment 1 is to evaluate the validity of the Symptom Checker module by varying the classification algorithm and the amount of data used for the training. The classifier is trained on a synthetic patient dataset $P = p_1, p_2, \ldots, p_n$ where each p_i is described by a set of symptoms $S = s_1, s_2, \ldots, s_n$. Specifically, a one-hot representation was used to describe each user. Each symptom is represented by a cell of the patient vector p_i and the value 1 is inserted if this symptom is present in the patient, and 0 if it is not. P is used for training a classification model.

Among the different possible classification algorithms, we focused on *Naive Bayes, Logistic Regression, Random Forest, Multilayer Perceptron Network*.

Naive Bayes [48] is a supervised learning algorithm suitable for solving multi-class classification problems. It is based on the use of conditional probability to determine the probabilities of model elements. The peculiarity of the algorithm is the idea that all features are independent to one another. The presence or absence of one feature does not affect the presence or absence of others.

Logistic regression [49] can be considered as a classification method within the family of supervised learning algorithms that are robust to noise. Using the *logistic function*, logistic regression generates the probability that a given input value falls into a given class. It works particularly well for cases where the class is binary. For multi-class classification problems, it is possible to use the multinomial logistic regression, that generalizes the classification problem.

Random Forest [50] is an ensemble-type classifier, i.e. it is made up of a set of simpler classifiers. Specifically, it employs several decision trees, each capable of producing an output response when given an input example. The class of an item is determined by the majority voting of the classes returned by the individual trees. Unlike the Naive Bayes and Logistic Regression models, Random Forest is able to manage datasets with high dimensionality and categorical features effectively.

A Deep Multilayer Network [51] (Multilayer Perceptron Network, MLP) consists of a layer of input neurons, each of which corresponds to an explanatory variable, one or more hidden layers, each of which consists of a number of neurons, and an output layer, consisting of as many neurons as there are response variables. The neurons are connected to each other by appropriate weights, i.e. parameters, estimated through the training set. MLP networks have the property of being ''universal approximators'', that is, given a sufficiently large number of layers, they can approximate any continuous and defined function in a finite space $D \subset R^D$.

In this experiment, we want to test the following research hypotheses:

- **RH1.1**: the performances of the symptom checker are affected by the classification strategy applied;
- **RH1.2**: a larger dataset makes the classification model for the Symptom Checking task more accurate;
- **RH1.3**: ensemble strategies can improve the performance of the classification model for the symptom checking task;

In order to test hypothesis RH1.1, we evaluated the use of the following algorithms: Naive Bayes, Logistic Regression, Random Forest, Multilayer Perceptron Network. In order to test hypothesis RH1.2, we changed the number of patients for each disease k among the values: 100, 1000, 2500, 5000 obtaining datasets of corresponding size l of 21700, 217000, 542500 and 1085000 instances. In order to test hypothesis RH1.3, we decided to define m models to simulate specialists into one specific clinical area. A similar evaluation strategy has been already adopted in [34]. The results of each model specialized in a set of diseases have been merged and then ordered using the confidence of the model. We decided to split the diseases into 5, 10, 20 main groups using the K-Means algorithm on the descriptions of the items, and into 24 different groups using the subdivision obtained by the manual association of each disease to the corresponding clinical area.

TABLE 3. Accuracy scores obtained by the different classification algorithms varying the number of patients for each disease *k* into the dataset.

Accuracy Scores	Naive Bayes			Logistic Regression			Multilayer Perceptron			Random Forest		
	Top1	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*
k = 100	0.598	0.851	0.927	0.588	0.843	0.922	0.546	0.819	0.907	0.467	0.761	0.873
k = 1000	0.614	*0.87*	*0.942*	0.601	0.86	0.935	0.599	0.856	0.931	0.56	0.836	0.921
k = 2500	*0.616*	*0.87*	0.941	0.603	0.86	0.933	0.604	0.86	0.934	0.583	0.849	0.928
k = 5000	0.615	0.869	0.941	0.603	0.858	0.933	0.605	0.862	0.936	0.597	0.856	0.932

TABLE 4. F1 scores obtained by the different classification algorithms varying the number of patients for each disease *k* into the dataset.

F1 scores	Naive Bayes			Logistic Regression			Multilayer Perceptron			Random Forest		
	Top1	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*
k = 100	0.595	0.85	0.926	0.58	0.84	0.921	0.539	0.817	0.906	0.446	0.751	0.869
k = 1000	0.611	***0.87***	***0.942***	0.595	0.859	0.934	0.592	0.856	0.931	0.548	0.832	0.92
k = 2500	***0.615***	***0.87***	0.941	0.599	0.858	0.933	0.6	0.858	0.933	0.576	0.846	0.927
k = 5000	0.614	0.869	0.941	0.599	0.858	0.933	0.601	0.861	0.935	0.591	0.854	0.931

1) RESULTS OF EXPERIMENT 1

Results of experiment 1 are shown in Table 3 - 4, which reports the accuracy of the different classifiers trained using different amounts of data, i.e. varying the k parameter. Considering the results obtained in terms of **Accuracy** and **F1**, it is possible to notice that the *Naive Bayes* algorithm performs better than the others most of the time. Logistic regression takes second place, obtaining results that are most comparable to those of Naive Bayes. Surprisingly, the MLP model did not reach the top positions, despite the extensive use of neural networks in many close domains of applications.

About the **RH1.1**, we can state that, despite the better performance of the Naive Bayes algorithm, the difference among the algorithms is not always statistically significant. To prove this claim, we applied the Wilcoxon-Mann-Whitney's non-parametric test in pairs. In particular, we considered the results obtained by the Naive Bayes algorithm, our best performing model, and we compared them with the results obtained from the other classification models. The only significant difference ($p < 0.05$) is the one between Naive Bayes and Random Forest, for any value of k and any size of the list of results. This suggests that, in a context where data is very sparse and only few elements provide a strong contribution to the classification, the algorithms that we tested are mostly interchangeable, in accordance with execution time and storage space constraints. We can conclude that hypothesis **RH1.1** is **partially rejected**.

Regarding hypothesis **RH1.2** (a larger dataset makes the classification model for the Symptom Checking task more accurate), it is clear that in machine learning, a large enough amount of data is required to allow the classifier to generalize well on the task. Independently of the algorithm applied, we can see that the model tends to converge correctly when we provide between 1,000 and 2,500 patient instances per disease. For example, we can take into consideration the performance of the Naive Bayes model that provides only one result in output (Top 1). The F1 score shows us an increase in performance from 0.595 when the model is trained with only 100 patients per disease, to 0.615 when trained with 2500 examples per disease. A similar trend can be observed for each classifier in Table 4. This allows us to state with certainty that the performance of the models considered here varies according to the amount of data provided in input. Therefore, we can **accept** the **RH1.2** hypothesis, and confirm that, in this specific domain, a number of between 1,000 and 2,500 examples per disease is sufficient to obtain classification models with almost optimal performance.

Finally, for hypothesis **RH1.3**, (ensembling strategies influence the performances of the classifier), we can observe in Table 5 that best scores have been obtained with an ensemble of only five models. In fact, there is a visible decrease in the performance of the ensemble, as the number of models employed increases. Specifically, the idea of realizing a model for each clinical area is not successful, probably due to the large number of diseases that share symptoms. Each model will search for the most likely disease based on its limited knowledge, and will assign it a high confidence value if some of the symptoms found are valid for the disease. The same will be done by each of the other models of the ensemble, who will then propose one of their candidates as the best one, consequently increasing the noise in the results list. If each model of the ensemble does not know the diseases on which the other sectorial models are working, it will be not able to consider them during its reasoning process. This

TABLE 5. F1 scores obtained by the ensemble of Naive Bayes classifiers varying the number of patients for each disease *k* into the dataset and the number of *m* models one for each clinical areas considered.

Ensable	5-Mediacal Areas			10-Mediacal Areas			20-Mediacal Areas			24-Mediacal Areas		
F1 scores	*Top1*	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*	*Top1*	*Top3*	*Top5*
k = 100	0.337	0.668	0.824	0.315	0.572	0.719	0.271	0.515	0.654	0.208	0.415	0.55
k = 1000	0.312	0.644	0.842	0.285	0.527	0.691	0.263	0.482	0.608	0.202	0.4	0.523
k = 2500	0.309	0.637	0.84	0.279	0.516	0.669	0.26	0.469	0.6	0.203	0.389	0.512
k = 5000	0.303	0.629	0.84	0.273	0.507	0.659	0.255	0.466	0.587	0.2	0.381	0.509

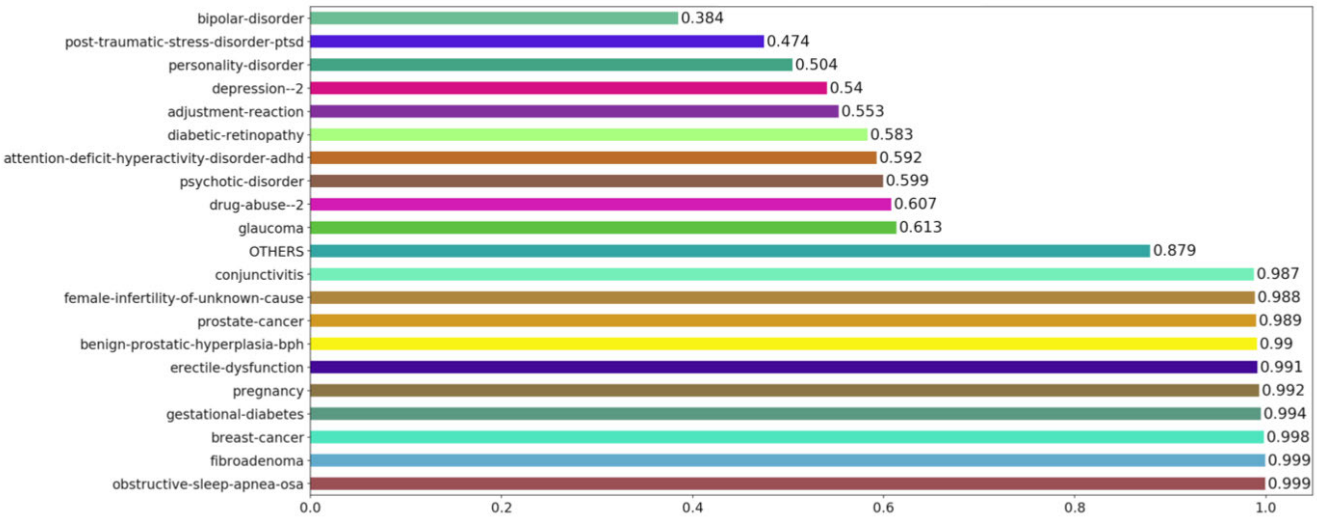

FIGURE 10. F1 of the best and worse 10 classification classes considering a list of 3 results.

also means that it will not be able to adjust its confidence value according to other diseases that could be more likely. Consequently, each candidate of the models of the ensemble will receive a high local confidence score. This consideration allows us to **reject** the **RH1.3** hypothesis.

It is possible to observe in Fig. 10 that there are classes for which the Naive Bayes classifier with k = 1,000 performs particularly well, and others for which it performs particularly bad. Specifically, the figure shows the ten classes in which the classifier performs best, such as sleep apnea, fibroadenoma, breast cancer, diabetes; and the ten in which it performs the worst, such as bipolarism, post-traumatic stress, personality disorders. These conditions are particularly difficult to diagnose even by specialist doctors, due to the vagueness and uncertainty of their symptoms. Speaking in general terms, we can say that the Symptom Checker module shows a qualitatively satisfactory behavior from our point of view, especially for the identification of common and well-defined diseases.

C. EXPERIMENT 2

Experiment 2 has two goals. The first is to evaluate the effectiveness of the Symptom Checker in a real scenario. The second is to analyze the level of user satisfaction and the simplicity of use of the chatbot. In this experiment we tested the following research hypotheses:

- **RH2.1**: The Symptom Checker module is able to detect a disease analyzing the symptoms provided by the user;
- **RH2.2**: The user feels satisfied by the services offered by the chatbot;
- **RH2.3**: The chatbot interface is efficient and simple to use.

To evaluate such performances with real users, we involved 102 subjects. Many of them ($\sim$ 82%) are men, with a high school diploma or university degree ($\sim$ 45% and $\sim$ 36% respectively) between 18 and 25 years old ($\sim$ 73%). The final configuration of the Symptom Checker used in the running example is the one based on a Bayesian classifier trained on a dataset with $k = 2,500$. In order to investigate the hypothesis RH2.1, we asked each subject to think about a disease diagnosed to them by a real doctor. We then asked them to input each symptom of the disease it into HAB, one at a time. The process will be iterated until the Symptom Checker will discover the disease, or all symptoms have been inserted. After that, we provided users with a questionnaire (Table 7) to collect their experiences (QUEST_a). The results

TABLE 6. Details about the subjects of the experimental sample.

	Possible Answers				
Gender	*Man*	*Woman*			
	84 (82.352%)	18 (17.648%)			
Age	*18-25*	*26-35*	*36-45*	*46-55*	*over 55*
	75 (73.529%)	11 (10.784%)	7 (6.862%)	5 (4.901%)	4 (3.921%)
Education	*Elementary School*	*Secondary School*	*Bachelor / Master*	*Doctorate*	*Others*
	12 (11.764%)	46 (45.098%)	37 (36.274%)	6 (5.882%)	1 (0.980%)

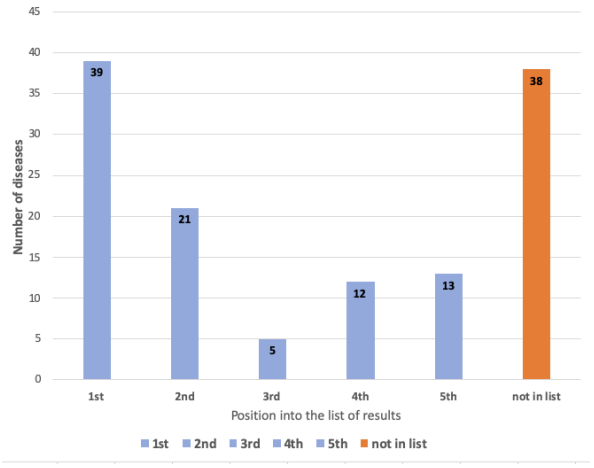

FIGURE 11. Results of the Experiment 2 task. Position of the disease in the list of results provided by the symptom checker.

obtained have been used for assessing the RH2.1 hypothesis. RH2.2 and RH2.3 have been evaluated through the answers to a different questionnaire (Table 8) compiled by users after using HAB for a week (QUEST_b).

1) RESULTS OF EXPERIMENT 2

128 consultations were asked to the Symptom Checker. The results of the evaluation are reported in Fig. 11. Ninety of them are successful (i.e. the module was able to identify the correct disease), while the unsuccessful ones (in which the correct disease was not present in the list of results) are thirty-eight. The Symptom Checker module was, therefore, able to correctly simulate a doctor's diagnosis in 76.271% of cases, i.e. an accuracy value in line with what was observed during Experiment 1. Most of the correct results were obtained by inputting only one symptom in the system, and the average number of symptoms required to conclude a consultation is less than 3 (an average of 2.43 symptoms), as shown in Fig. 12. Observing these results, we can consequently **accept** the **RH2.1** hypothesis: the Symptom Checker module can indeed recognize the patient's disease with few interaction steps and with good accuracy even in a real-world scenario. A qualitative analysis on the performances of the Symptom checker module has been performed by analyzing the answers provided by users to the post-task questionnaire (Table 7 - QUEST_a). The results show that users are not usual with searching on the internet for possible health

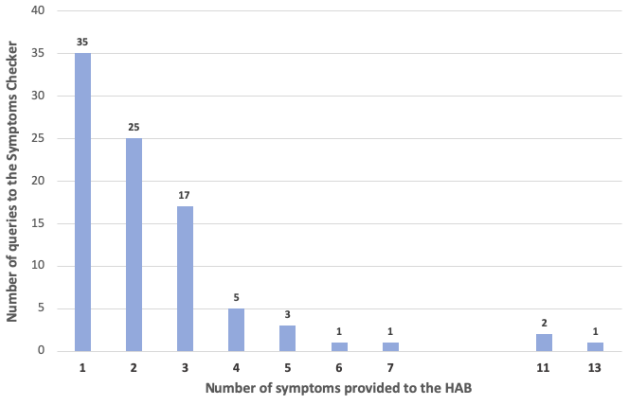

FIGURE 12. Results of the Experiment 2 task. Number of symptoms provided to the HUB for obtaining correct results.

problems (average value 2.71 of the answer to question 2). Anyway they considered the system usefull for them. They were satisfied of the time needed for consultation (average value 4.17 of the answer to question 4), the clarity and quality of the results (answers 9 and 10 with corresponding average values of 4.29 and 3.81).

In order to better understand the causes of unsuccessful classifications of the Symptom Checker, we performed a detailed *error analysis*. We were able to identify three categories of errors occurring during the classification process:

- **Partial symptom matching**: the correct disease contains only some of the symptoms entered by the user; the ranking, therefore, prefers other diseases that contain all the symptoms. It occurs for the following diseases: "flu", "cold" and "migraine";
- **Multiple matching**: The correct disease contains all the symptoms entered by the user, but there are others, even rarer diseases, with the same symptoms. The final order of elements presented to the user is due to chance because we do not have any information about the rarity of the disease. It often occurs for diseases like "flu", "fever", "herniated disc", and "conjunctivitis" that show common symptoms;
- **Incorrect symptom-disease association**: Not all symptoms entered by the user are associated to the reported disease in the knowledge base. It especially occurs for consultations on the following diseases: "gastritis", "allergy", and "bronchitis".

While the last class of errors can be solved by expanding the knowledge base, the first two are consequences of the classification model. In particular, the model does not take into account the "rarity" of the diseases, and the ranking function prefers diseases that match as much symptoms as possible. In future work, we will try to solve these problems by introducing the rarity of the disease in the scoring function, and by extending our knowledge base.

In order to test the RH2.2 hypothesis, we analyzed the answers provided by the subjects to the questionnaire

TABLE 7. Questionnaire provided to users for the evaluation of the effectiveness of the symptom checker module in a real scenario of use. (QUEST_a).

	Questions	Possible Answers				
1	*What's your gender?*	Man	Woman			
2	*What's your age range?*	18-25	26-35	36-45	46-55	over 55
3	*What's your level of education?*	Elementary School	Secondary School	Bachelor/Master	Doctorate	Others
4	*How often do you use the Internet to understand your symptoms and find possible related diseases? (1 never, 5 very often)*	1	2	3	4	5
5	*How often do you use the Internet for medical/health research? (1 never, 5 very often)*	1	2	3	4	5
6	*How easy was it to get the agent to recognize your symptoms? (1 very difficult, 5 very easy)*	1	2	3	4	5
7	*How satisfied are you with the total time taken to complete a single consultation? (1 not at all satisfied, 5 very satisfied)*	1	2	3	4	5
8	*Was the number of steps required to complete a single consultation appropriate? (1 not at all appropriate, 5 very appropriate)*	1	2	3	4	5
9	*The results you got were clear? (1 definitely no, 5 definitely yes)*	1	2	3	4	5
10	*The answers you got from the agent were the ones you were looking for? (1 definitely no, 5 definitely yes)*	1	2	3	4	5
11	*Indicates the level of satisfaction with the service received (1 very low, 5 very high)*	1	2	3	4	5

described in Table 8 (QUEST_b). In particular, we focus on questions about the *satisfaction of use* numbered 4 through 7, and 11 through 14. The first group of questions obtained an average score of 3.86, with a maximum value of 3.95 obtained for question number 4 (*How much has the agent satisfied your needs?*). The second group of answers provides us some suggestions about the user's satisfaction with each specific functionality of HAB. In particular, the Symptom Checker component has been appreciated the most, followed by the doctor recommendation System, and the Treatment Management. The high score obtained for all the questions considered in this study gives us the confidence to confirm the chatbot' ability to satisfy the user needs. Consequently we can **accept RH2.2**.

Finally, in order to evaluate RH2.3, we observed the results provided by the users for questions 8 through 10 of the questionnaire shown in Table 8 (QUEST_b). With an average score of 3.85 out of five, the answers confirm that users were overall satisfied with the interaction, and found the system simple to use and useful. Due to this, we can **successfully accept** the **RH2.3** hypothesis.

V. CONCLUSION AND FUTURE WORK

In this article, we presented HealthAssistantBot, a Telegram-based conversational agent for supporting patients in their daily activities. The agent has been developed with a modular strategy so that new features can easily be attached to it when needed. The system offers users the possibility to monitor their treatments, their biological values, suggest doctors, and perform self-diagnosis. The conversation is performed through a text-based interface, that makes the interaction simple while being robust to errors. The architecture of the proposed platform is divided into three main parts: the interface, the gateway, and the server-side functionalities. Each of these is independent in order to ensure high internal cohesion of its functionality and low overlap with the functionality of the other modules.

The main contribution of this paper is the definition of the Symptom Checker module, that identifies the patient's disease with a certain degree of accuracy, starting from a set of symptoms. This functionality is designed to assist the user in obtaining a set of automatic diagnoses that can be later discussed with her doctor. The classifier is based on a Bayesian algorithm, trained on an artificial data set created

TABLE 8. Questionnaire provided to users for the evaluation of the satisfaction and simplicity of use of the chatbot. (QUEST_b).

	Questions	Possible Answers				
1	*What's your gender?*	Man	Woman			
2	*What's your age range?*	18-25	26-35	36-45	46-55	over 55
3	*What's your level of education?*	Elementary School	Secondary School	Bachelor/Master	Doctorate	Others
4	*How much has the agent satisfied your needs? (1 none of my needs have been met, 5 all my needs have been met)*	1	2	3	4	5
5	*Would you recommend our chatbot to a friend? (1 definitely no, 5 definitely yes)*	1	2	3	4	5
6	*How satisfied are you with the services that you used? (1 not at all satisfied, 5 very satisfied)*	1	2	3	4	5
7	*Have the services you received helped you to deal more effectively with your problems? (1 didn't help at all, 5 helped a lot)*	1	2	3	4	5
8	*Overall, how comfortable did you feel while the interaction with the agent? (1 not at all satisfied, 5 very satisfied)*	1	2	3	4	5
9	*If you ever need one of the agent's services again, would you use the chatbot again? (1 definitely no, 5 definitely yes)*	1	2	3	4	5
10	*Was the interaction with the interface simple, meaningful and useful? (1 definitely no, 5 definitely yes)*	1	2	3	4	5
11	*Which is the service you found the less useful?*	Doctors Recommender System	Symptom Checker	Medical Dictionary	Monitoring	Treatments Management
12	*Which is the service you found the most useful?*	Doctors Recommender System	Symptom Checker	Medical Dictionary	Monitoring	Treatments Management
13	*What is the service that satisfied you most?*	Doctors Recommender System	Symptom Checker	Medical Dictionary	Monitoring	Treatments Management
14	*What was the service that satisfied you less?*	Doctors Recommender System	Symptom Checker	Medical Dictionary	Monitoring	Treatments Management
15	*Now you can write a comment with suggestions or advice on how to improve the agent*	Free Text				

following the real distribution of symptoms for each disease. An in-vitro study and in-vivo user study were performed, both of which produced encouraging results. We measured an F1 score of 0.942 on the synthetic dataset, a success ratio of 76,271% on real use cases. Besides, we noted that the doctor recommendation system and the Treatment Management functionality were widely appreciated. They have proven to be effective and able to satisfy the needs of end-users.

As future work, we are planning to improve the performance of the Symptom Checker module by adding information on the rarity of the diseases. Moreover, we will focus on adding new functionalities, such as the management of medical records, and the automatic suggestion of food and physical activity to perform based on the user's health conditions. Finally, we will perform a more extensive user study once a large enough community of HAB users will be established.

APPENDIX A
QUESTIONNAIRES

See Tables 7 and 8.

ACKNOWLEDGMENT

The authors would like to thank Domenico Vitarella for the support given to this research.

REFERENCES

[1] A. Allen, "Morphing telemedicine-telecare-telehealth-ehealth," Telemed Today, Special, no. 2000, 2000.

[2] S. L. Murphy, J. Xu, K. D. Kochanek, and E. Arias, "Mortality in the United States, 2017," *NCHS Data Brief*, no. 328, pp. 1–8, Nov. 2018.

[3] K. Myers, P. Berry, J. Blythe, K. Conley, M. Gervasio, D. L. McGuinness, D. Morley, A. Pfeffer, M. Pollack, and M. Tambe, "An intelligent personal assistant for task and time management," *AI Mag.*, vol. 28, no. 2, p. 47, 2007.

[4] R. Sarikaya, "The technology powering personal digital assistants," in *Proc. 16th Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH)*, Dresden, Germany. Kolkata, India: ISCA, Sep. 2015, pp. 2022–2026. [Online]. Available: http://www.isca-speech.org/archive/interspeech_2015/i15_4002.html

[5] A. S. Tulshan and S. N. Dhage, "Survey on virtual assistant: Google assistant, Siri, Cortana, Alexa," in *Proc. Int. Symp. Signal Process. Intell. Recognit. Syst.* Cham, Switzerland: Springer, 2018, pp. 190–201.

[6] M. B. Hoy, "Alexa, siri, cortana, and more: An introduction to voice assistants," *Med. Reference Services Quart.*, vol. 37, no. 1, pp. 81–88, Jan. 2018.

[7] G. McLean and K. Osei-Frimpong, "Hey Alexa... examine the variables influencing the use of artificial intelligent in-home voice assistants," *Comput. Hum. Behav.*, vol. 99, pp. 28–37, Oct. 2019.

[8] S. Devi, Z. A. M. Merchant, M. S. Siddiqui, and M. Lobo, "Artificial intelligence based personal assistant," *Asian J. Converg. Technol.*, vol. 5, no. 1, pp. 1–4, 2019.

[9] U. Saad, U. Afzal, A. El-Issawi, and M. Eid, "A model to measure QoE for virtual personal assistant," *Multimedia Tools Appl.*, vol. 76, no. 10, pp. 12517–12537, 2017.

[10] R. W. White, "Skill discovery in virtual assistants," *Commun. ACM*, vol. 61, no. 11, pp. 106–113, Oct. 2018.

[11] D. Rafailidis and Y. Manolopoulos, "The technological gap between virtual assistants and recommendation systems," 2018, *arXiv:1901.00431*. [Online]. Available: http://arxiv.org/abs/1901.00431

[12] L. van Gemert-Pijnen, S. M. Kelders, H. Kip, and R. Sanderman, *eHealth Research, Theory and Development: A Multi-Disciplinary Approach*. Evanston, IL, USA: Routledge, 2018.

[13] H. Oh, A. Jadad, C. Rizo, M. Enkin, J. Powell, and C. Pagliari, "What is eHealth (3): A systematic review of published definitions," *J. Med. Internet Res.*, vol. 7, no. 1, p. e1, Feb. 2005.

[14] J. E. W. C. Van Gemert-Pijnen, O. Peters, and H. C. Ossebaard, *Improving EHealth*. The Hague, The Netherlands: Eleven International Publishing, 2013.

[15] J. Weizenbaum, "Eliza—A computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, pp. 36–45, 1966.

[16] H. Shah and K. Warwick, "Constraining random dialogue in a modern ELISA," in *Proc. Int. Conf. Comput. Philosophy*, Laval, France, May 2006, pp. 247–265.

[17] F. Narducci, P. Lops, and G. Semeraro, "Power to the patients: The HealthNetsocial network," *Inf. Syst.*, vol. 71, pp. 111–122, Nov. 2017.

[18] L. Kelly, H. Suominen, L. Goeuriot, M. Neves, E. Kanoulas, D. Li, L. Azzopardi, R. Spijker, G. Zuccon, H. Scells, and J. Palotti, "Overview of the CLEF eHealth evaluation lab 2019," in *Proc. Int. Conf. Cross-Lang. Eval. Forum Eur. Lang.* Cham, Switzerland: Springer, 2019, pp. 322–339.

[19] N. Karisalmi, J. Kaipio, and S. Kujala, "Encouraging the use of ehealth services: A survey of patients' experiences," in *Improving Usability, Safety and Patient Outcomes with Health Information Technology*, vol. 257, F. Y. Lau, J. A. Bartle-Clar, G. Bliss, E. M. Borycki, K. L. Courtney, A. M. Kuo, A. Kushniruk, H. Monkman, and A. V. Roudsari, Eds. Victoria, BC, Canada: IOS Press, Feb. 2019, pp. 206–211, doi: 10.3233/978-1-61499-951-5-206.

[20] A. Iovine, F. Narducci, and G. Semeraro, "Conversational recommender systems and natural language:: A study through the ConveRSE framework," *Decis. Support Syst.*, vol. 131, Apr. 2020, Art. no. 113250.

[21] F. Narducci, C. Musto, M. Polignano, M. de Gemmis, P. Lops, and G. Semeraro, "A recommender system for connecting patients to the right doctors in the healthnet social network," in *Proc. 24th Int. Conf. World Wide Web Companion (WWW)*, A. Gangemi, S. Leonardi, and A. Panconesi, Eds. Florence, Italy: ACM, May 2015, pp. 81–82, doi: 10.1145/2740908.2742748.

[22] F. Narducci, C. Musto, M. Polignano, M. de Gemmis, P. Lops, and G. Semeraro, "Recommending doctors and health facilities in the healthnet social network," in *Proc. Workshop Social Media Personalization Search Co-Located 39th Eur. Conf. Inf. Retr. (SoMePeAS@ECIR)*, vol. 1840, L. Boratto, A. Kaltenbrunner, and G. Stilo, Eds. Aberdeen, U.K.: CEUR-WS.org, Apr. 2017, pp. 1–7.

[23] Q. Han, I. M. de Rituerto de Troya, M. Ji, M. Gaur, and L. Zejnilovic, "A collaborative filtering recommender system in primary care: Towards a trusting patient-doctor relationship," in *Proc. IEEE Int. Conf. Healthcare Informat. (ICHI)*, Jun. 2018, pp. 377–379.

[24] D. Gujar, R. Biyani, T. Bramhane, S. Bhosale, and T. P. Vaidya, "Disease prediction and doctor recommendation system," *Int. Res. J. Eng. Technol., Tamilnadu, India*, vol. 5, no. 3, pp. 3207–3209, Mar. 2018.

[25] P. Cordero, M. Enciso, D. López, and A. Mora, "A conversational recommender system for diagnosis using fuzzy rules," *Expert Syst. Appl.*, vol. 154, Sep. 2020, Art. no. 113449.

[26] F. Narducci, P. Basile, A. Iovine, M. de Gemmis, P. Lops, and G. Semeraro, "A framework for building chat-based recommender systems," in *Proc. Workshops Co-Located 27th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, vol. 2482, A. Cuzzocrea, F. Bonchi, and D. Gunopulos, Eds. Turin, Italy: CEUR-WS.org, Oct. 2018, pp. 1–2.

[27] M. Wiesner and D. Pfeifer, "Health recommender systems: Concepts, requirements, technical basics and challenges," *Int. J. Environ. Res. Public Health*, vol. 11, no. 3, pp. 2580–2607, Mar. 2014.

[28] S. Hors-Fraile, O. Rivera-Romero, F. Schneider, L. Fernandez-Luque, F. Luna-Perejon, A. Civit-Balcells, and H. de Vries, "Analyzing recommender systems for health promotion using a multidisciplinary taxonomy: A scoping review," *Int. J. Med. Informat.*, vol. 114, pp. 143–155, Jun. 2018.

[29] A. O. Afolabi and P. Toivanen, "Recommender systems in healthcare: Towards practical implementation of real-time recommendations to meet the needs of modern caregiving," in *Handbook of Research on Emerging Perspectives on Healthcare Information Systems and Informatics*. Hershey, PA, USA: IGI Global, 2018, pp. 323–346.

[30] D. Jannach, A. Manzoor, W. Cai, and L. Chen, "A survey on conversational recommender systems," 2020, *arXiv:2004.00646*. [Online]. Available: http://arxiv.org/abs/2004.00646

[31] L. J. Bisson, J. T. Komm, G. A. Bernas, M. S. Fineberg, J. M. Marzo, M. A. Rauh, R. J. Smolinski, and W. M. Wind, "How accurate are patients at diagnosing the cause of their knee pain with the help of a web-based symptom checker?" *Orthopaedic J. Sports Med.*, vol. 4, no. 2, 2016, Art. no. 2325967116630286.

[32] T. Morita, A. Rahman, T. Hasegawa, A. Ozaki, and T. Tanimoto, "The potential possibility of symptom checker," *Int. J. Health Policy Manage.*, vol. 6, no. 10, p. 615, 2017.

[33] H.-C. Kao, K.-F. Tang, and E. Y. Chang, "Context-aware symptom checking for disease diagnosis using hierarchical reinforcement learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–9.

[34] K.-F. Tang, H.-C. Kao, C.-N. Chou, and E. Y. Chang, "Inquire and diagnose: Neural symptom checking ensemble using deep reinforcement learning," in *Proc. NIPS Workshop Deep Reinforcement Learn.*, 2016, pp. 1–9.

[35] D. Braun, A. Hernandez-Mendez, F. Matthes, and M. Langen, "Evaluating natural language understanding services for conversational question answering systems," in *Proc. 18th Annu. SIGdial Meeting Discourse Dialogue*, 2017, pp. 174–185.

[36] P. Voigt and A. von dem Bussche, *The EU General Data Protection Regulation (GDPR): A Practical Guide*, 1st ed. Cham, Switzerland: Springer, 2017.

[37] *The ICD-10 Classification of Mental and Behavioural Disorders: Diagnostic Criteria for Research*, World Health Organization, Geneva, Switzerland, 1993, vol. 2.

[38] R. Navigli and S. P. Ponzetto, "BabelNet: Building a very large multilingual semantic network," in *Proc. 48th Annu. Meeting Assoc. Comput. Linguistics*, 2010, pp. 216–225.
[39] J. Ramos, "Using TF-IDF to determine word relevance in document queries," in *Proc. 1st Instructional Conf. Mach. Learn.*, vol. 242, Piscataway, NJ, USA, Dec. 2003, pp. 133–142.
[40] P. Ferragina and U. Scaiella, "TAGME: On-the-fly annotation of short text fragments (by Wikipedia entities)," in *Proc. 19th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2010, pp. 1625–1628.
[41] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, "Facing the cold start problem in recommender systems," *Expert Syst. Appl.*, vol. 41, no. 4, pp. 2065–2073, Mar. 2014.
[42] M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro, "Semantics-aware content-based recommender systems," in *Recommender Systems Handbook*. Cham, Switzerland: Springer, 2015, pp. 119–159.
[43] P. Lops, C. Musto, F. Narducci, and G. Semeraro, *Semantics in Adaptive and Personalised Systems: Methods, Tools and Applications*. Cham, Switzerland: Springer, 2019, doi: 10.1007/978-3-030-05618-6.
[44] P. Lops, M. de Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender Systems Handbook*. Cham, Switzerland: Springer, 2011, pp. 73–105.
[45] J. Konečný, H. Brendan McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: http://arxiv.org/abs/1610.02527
[46] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *Int. J. Med. Informat.*, vol. 112, pp. 59–67, Apr. 2018.
[47] D. T. Handler, L. Hauge, A. Spognardi, and N. Dragoni, "Security and privacy issues in healthcare monitoring systems: A case study," in *Proc. 10th Int. Joint Conf. Biomed. Eng. Syst. Technol.* Porto, Portugal: SciTePress Digital Library, 2017, pp. 383–388.
[48] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, Mar. 1997.
[49] D. R. Cox, "The regression analysis of binary sequences," *J. Roy. Stat. Soc., B (Methodol.)*, vol. 20, no. 2, pp. 215–232, Jul. 1958.
[50] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[51] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter, "The multilayer perceptron as an approximation to a Bayes optimal discriminant function," *IEEE Trans. Neural Netw.*, vol. 1, no. 4, pp. 296–298, Dec. 1990.

MARCO POLIGNANO received the Ph.D. degree in computer science and mathematics from the University of Bari Aldo Moro, Italy, in 2018, with the thesis titled an affect-aware computational model for supporting decision-making through recommender systems. He is currently a Postdoctoral Research Fellow with the Semantic Web Access and Personalization (SWAP) Research Group, Department of Computer Science, University of Bari Aldo Moro. His research interests include recommender systems, natural language processing, machine learning, and user profiling. He was a Program Committee Member and a reviewer for many journal and international conferences, the Local Organizing Committee Member for the Ai*iA 2017 and the CLiC-it 2019 conferences, an Organizer of the Evalita 2018 Challenge—ABSITA about the aspect-based sentiment analysis and the exUm 2020 Workshop at UMAP 2020 about user modeling and explanation. In 2016 and 2018 he was a Marie Skłodowska-Curie Research and Innovation Staff Exchange (MSCA-RISE) Fellow, involved in the project titled Seo-Dwarf: Semantic EO Data Web Alert and Retrieval Framework.

FEDELUCIO NARDUCCI received the Ph.D. degree, in 2012, with a dissertation on knowledge-enriched representations for content-based recommender systems. From April 2012 to July 2014, he worked as a Postdoctoral Researcher with the University of Milano-Bicocca on the Services and Meta-services for smART eGovernment (SMART) Project with a specific focus on cross-language information retrieval and filtering. He is currently an Assistant Professor with the Politecnico di Bari, Italy, where he is also a member of the SisInf Laboratory—Information Systems Laboratory Research Group. He is also a member of the Semantic Web Access and Personalization (SWAP) Research Group, University of Bari Aldo Moro. He has published more than 60 articles and has served as a reviewer and a co-reviewer for international conferences and journals in the areas of AI, recommender systems, user modeling, and personalization. His primary research interests include artificial intelligence, machine learning, personalization, user modeling, recommender systems, and conversational agents.

ANDREA IOVINE (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Bari Aldo Moro, Italy, and he is also investigating several aspects related to conversational recommender systems and dialog agents in general, under the supervision of Prof. Giovanni Semeraro and Dr. Fedelucio Narducci. He is also a member of the Semantic Web Access and Personalization (SWAP) Research Group, University of Bari Aldo Moro. His primary research interests include natural language processing, dialog agents, conversational recommender systems, user modeling, and personalization.

CATALDO MUSTO received the Ph.D. degree, in 2012, with a thesis entitled Enhanced Vector Space Models for Content-based Recommender Systems. He was a Visiting Researcher with the Philips Research Center, Eindhoven, The Netherlands, in 2011, where he worked on the personalization of electronic program guides (EPG). He was involved in various national and international research projects that dealt with natural language processing and recommender systems. From 2016 to 2019 he acted as the Project Leader for a funded project regarding Semantic Holistic User Modeling for Personalized Access to Digital Content and Services. He is currently an Assistant Professor with the Department of Informatics, University of Bari. He is one of the authors of the textbook *Semantics in Adaptive and Personalized Systems: Methods, Tools and Applications* (Springer). His research interests include the adoption of natural language processing techniques and models for fine-grained semantic content representation in recommender systems and user modeling platforms. Since 2009, he has been publishing around 70 scientific articles in top venues and journals. He regularly acts as a PC Member for several top-tier conferences and co-organizes or co-chairs a number of workshops. He received the most inspiring Contribution Award at UMAP 2013 and the Best Paper Nominee at RecSys 2016.

MARCO DE GEMMIS received the Ph.D. degree in computer science from the Department of Computer Science, University of Bari Aldo Moro, Italy, in 2005. He is currently an Associate Professor with the Department of Computer Science, University of Bari Aldo Moro. He has authored over 100 scientific papers published in international journals and collections, proceedings of international conferences and workshops, and book chapters. His primary research interests include content-based recommender systems, natural language processing, information retrieval, text mining, and in general personalized information filtering. He was a Program Committee Member for international conferences, including the ACM Recommender Systems and the User Modeling, Adaptation and Personalization (UMAP). He was an Marie Curie Fellow of the SEO-DWARF Project, funded by the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska-Curie Grant. He has served as a reviewer for international journals, including *User Modeling and User-Adapted Interaction* and the *ACM Transactions on Internet Technologies*. He was an invited speaker at several universities, including the University of Roma 3, the University of the Basque Country, San Sebástian, the University of Cagliari, the University of Milano-Bicocca, the University of Naples Federico II, and at the Workshop on Semantics-Enabled Recommender Systems at ICDM 2016.

GIOVANNI SEMERARO was a Visiting Scientist with the Department of Information and Computer Science, University of California at Irvine, in 1993. He is currently a Full Professor of computer science with the University of Bari Aldo Moro, Italy, where he teaches intelligent information access, natural language processing, and programming languages. He leads the Semantic Web Access and Personalization (SWAP) Antonio Bello Research Group. He has been a Principal Investigator with the University of Bari, in several European, national, and regional projects. He is an author of more than 400 publications in international journals, conference and workshop proceedings, as well as for three books, including the textbook *Semantics in Adaptive and Personalized Systems: Methods, Tools and Applications* (Springer). His research interests include machine learning; AI and language games; recommender systems; user modelling; intelligent information mining, retrieval, and filtering; semantics and social computing; natural language processing; the semantic web; and personalization. In 2015, he was selected for the IBM Faculty Award on Cognitive Computing for the project Deep Learning to boost Cognitive Question Answering. He was one of the founders of the Italian Association for Computational Linguistics (AILC) and on the Board of Directors till 2018. From 2006 to 2011 he was on the Board of Directors of the Italian Association for Artificial Intelligence (AI*IA).