Smart Assistance for Students and People Living in a Campus

Salvatore Gaglio, Giuseppe Lo Re, Marco Morana, and Claudio Ruocco
{salvatore.gaglio, giuseppe.lore, marco.morana, claudio.ruocco}@unipa.it
University of Palermo
Viale delle Scienze, ed.6, 90128, Palermo, Italy

Abstract—Being part of one of the fastest growing area in Artificial Intelligence (AI), virtual assistants are nowadays part of everyone's life being integrated in almost every smart device. Alexa, Siri, Google Assistant, and Cortana are just few examples of the most famous ones. Beyond these *off-the-shelf* solutions, different technologies which allow to create *custom* assistants are available. IBM Watson, for instance, is one of the most widely-adopted question-answering framework both because of its simplicity and accessibility through public APIs. In this work, we present a virtual assistant that exploits the Watson technology to support students and staff of a smart campus at the University of Palermo. Some in progress results show the effectiveness of the approach we propose.

Index Terms—Smart Service Systems, Virtual Assistant, Smart Campus

I. INTRODUCTION

Nowadays, people are surrounded by *smart* devices, such as smartphones, smartwatches, and IoT devices in general, capable of interacting with each other in order to provide some useful services.

In such a scenario, *smart environments* can be designed by exploiting Artificial Intelligence and Machine Learning to *understand* the environment itself and the needs of its inhabitants [9]. Some academic institutions, for instance, are trying to enhance their campuses in order to provide students with novel *smart* services that can make everyday activities easier. As an example, in [1] a fog architecture enabling a set of services for an augmented campus is presented. Similarly, [5] proposes a system based on participatory sensing which infers the activities performed by the users of a smart campus in order to understand their habits and deliver ad-hoc services.

In our vision, all these services should be easily enjoyed by means of a virtual assistant, that is a system or a software agent capable of interacting with the users in a natural way, interpreting human speech, and responding through texts or synthesized voices.

Creating such an assistant requires a proper cognitive engine, which can interpret users' inputs and consequently reply with either simple (pre-made) answers, or complex responses obtained by analyzing a collection of documents.

Although the great presence of *closed*, proprietary, assistants (e.g., Amazon Alexa, Apple Siri, Google Assistant, Microsoft Cortana), there is still a significant request for technologies and tools to develop *custom* virtual assistants, that can be fully trained and specialized on specific application scenarios.

Amongst these technologies, in this paper we present a virtual assistant based on the IBM Watson framework, because of its reliability and ease of use.

Originally thought as a "simple" question-answering system, Watson is now one of the most powerful AIs available, being used in several applications all over the world. Accessible by developers since 2014, it now counts almost 3 billion monthly API requests [16].

The suite of services that can be exploited to build an efficient virtual assistant includes *Watson Assistant*, which provides the tools to build a chatbot, and *Watson Discovery*, useful to retrieve information from unstructured data. These services are complemented by a number of other software modules, such as *Speech-To-Text*, which is used to accept human voice as input, *Text-To-Speech*, and *Natural Language Understanding*, which allows to extract key components from texts, e.g. concepts, entities, keywords.

The virtual assistant we propose here is designed to support students of the University of Palermo to move inside the campus (e.g., by easily locating buildings, classrooms and other relevant points of interest) and to answer to some common questions regarding scholarships, taxes, enrollments, and so on. Moreover, the capabilities of the virtual assistant can be further extended allowing it to interact with other intelligent systems in order to perform more complex tasks, e.g., identifying a free spot in a parking or suggesting a classroom with a certain temperature according to data coming from IoT devices [12].

The paper is organized as follows: in Section II related works are presented; Section III presents the characteristics of the question-answering architecture we used. The logical components behind the proposed virtual assistant are described in SectionIV. A case study deployed at the University of Palermo, and the results achieved by the assistant are discussed in Section V-B. Conclusions are provided in Section VI.

II. RELATED WORK

The design of virtual assistants (VAs) has been addressed in many works exploiting different technologies.

The authors of [17], for instance, present a VA that can assist people in Activities of Daily Life (ADL), and whose dialogue system is handled through probabilistic rules (i.e., a Bayesian Network) and if-then-else dialogue construction. In order to resolve conflicting requests, this system also adopts
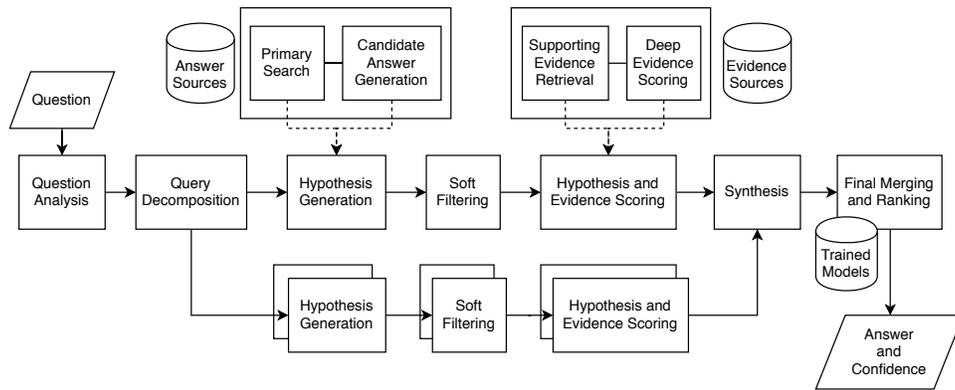
Fig. 1: DeepQA High-Level Architecture [10].

a priority-based approach in which distinct priority values are assigned to three classes of users (young, adult, and elderly).

The *AT&T Watson Speech Recognition Engine* dialogue system is exploited in [18] for detecting natural language understanding errors in an interactive spoken conversation. Authors utilize targeted clarification (TC) for error recovery, in order to keep the dialogue as natural as possible.

IBM Watson is one of the most important technologies used to build custom virtual assistants. Watson has been discussed and analyzed since its early releases in many works in the literature. These works regard its development, possible applications, and even its effects on the human society.

In [14], the analysis process performed by Watson every time it faces a new question is described. This analysis is composed both by a general-purpose parsing and a semantic analysis, even though some adaptations were made in order to win the *Jeopardy!* game [19]. Through this process, Watson is able to understand what is truly being asked and how to best approach the answering process. The authors of [15], on the other hand, describe the communication paths and connections that made Watson able to understand the state of the *Jeopardy!* game without any human intervention.

Other works focusing on the development of Watson have been proposed. [10] describes the IBM DeepQA architecture, giving a deep explanation of how it works and how effective and extensible it is. DeepQA can be seen also as a foundation for evaluating, merging, and improving a wide range of algorithmic techniques to advance the field of question answering (QA). Similarly, in [11] the Prolog interface that Watson uses to connect to Apache UIMA is described.

The best strategies to train IBM Watson are discussed in [16]. Through a classroom experience, where students were divided in different groups, the authors identify a set of guidelines which allow Watson to better index every document and consequently to better answer every question. These tips include the division of every document into sections, the inclusion of relevant keywords and the respect of some formatting rules. It is shown that those projects which did not exactly follow the guidelines achieved worse results, in terms of precision, recall and accuracy.

Also in educational context, but with different purposes, Watson has been adopted in [13] to develop an application for a chosen topic, such as nutrition or tourism. Here, Watson's responses are evaluated by means of a set of metrics, such as the correctness of the answers (correct, partially correct, wrong), and the percentage of correct answers for both variations of the training questions and blind questions.

Interesting insights on the topic of application development through Watson are also provided in [4], where a cognitive advising system to identify question categories and answer accordingly is presented.

Finally, other works focused on the consequences of Watson in the industries. In [3], for example, the deep impact Watson can have on the health care is analyzed. Thanks to its cognitive system which can understand, reason, learn and help people to expand their knowledge base, Watson and cognitive computing could offer the path to more individualized medical decisions, definitely improving doctors' choices.

III. QUESTION-ANSWERING ARCHITECTURE

Watson is a deep learning AI system, originally designed to participate in a famous American television game, *Jeopardy!*, which features rich natural language questions, covering a broad range of general knowledge. In 2011 Watson won against former Grand Champions, Ken Jennings and Brad Rutter.

Initially thought as a "simple" question answering (QA) system, it has been improved throughout the years, taking advantage of new deployment models (IBM Cloud), evolved machine learning techniques, and optimized hardware available. It can now "see", "hear", "read", "talk", "interpret", "recommend".

The system's components have been written in different programming languages, but most of them are in Java, C++ or Prolog. Watson runs on the SUSE Linux Enterprise Server 11 operating system using the Apache Hadoop framework to provide distributed computing.

Watson has been developed using the Apache Unstructured Information Management Architecture (UIMA), a framework used to analyze large volumes of unstructured contents (such

as text and video) in order to discover knowledge that may be relevant to an end user. The innermost part of UIMA, which is called Common Analysis Structure (CAS), is a dynamic data structure which contains unstructured data, structured annotations and various user reviews over these annotations. Watson uses a Prolog interface to UIMA, which is described in [11], that translates UIMA CAS into a Prolog format, a knowledge base over which developers can write rules and use inference and the resolution theorem in order to search for annotations and generate new ones. It can also connect to external knowledge bases, such as Prolog WordNet.

In particular, Watson's most important software component, the DeepQA System built upon UIMA, is a parallel probabilistic evidence-based architecture ([10]) that allows to deal with unstructured data and converse with users through some form of intelligent dialog. DeepQA architecture is made-up of multiple steps, during which questions are deeply analyzed and answers are created, ranked, sorted and validated (see Fig. 1).

In particular, after a pre-phase of **content acquisition**, during which the system is trained through identification and gathering of the sources, a first phase of **question analysis** is performed, involving multiple parsings and major features extraction (e.g., entities and relations) in order to understand the question.

Then, **hypothesis generation** module examines these features to produce candidate answers from the system sources. During the **soft filtering** phase, these hypotheses are ranked and sorted through different lightweight scoring algorithms.

The **hypothesis and evidence scoring** module receives only those hypotheses whose score is greater than a threshold, which is automatically determined by learning over training data. During this phase, additional evidence is collected and different algorithms are performed in order to evaluate each candidate.

Last phase, called **final merging and ranking**, includes a further inspection of every hypothesis in order to identify the best answer and the confidence in its correctness, as long as a final merging to group all those answers that, despite being written in different forms, are semantically equivalent. At the end of this phase, all the remaining hypotheses are ranked and sorted according to their merged score.

IV. A VIRTUAL ASSISTANT FOR A SMART CAMPUS

A functional virtual assistant needs to be able to interact with end users in a human-like way, interpreting inputs and producing coherent outputs. Moreover, a *smart* assistant should also be able to provide support on a variety of topics, some of which may require in-depth analysis of a huge number of sources, including unstructured documents, in order to extract relevant pieces of information.

In the virtual assistant we propose here, these two functionalities are covered by two subsystems, namely *conversational assistant* and *content discovery*.

A. Conversational Assistant

The virtual assistant (see Fig. 2) receives users' input through one *integration point*, i.e., a custom application or
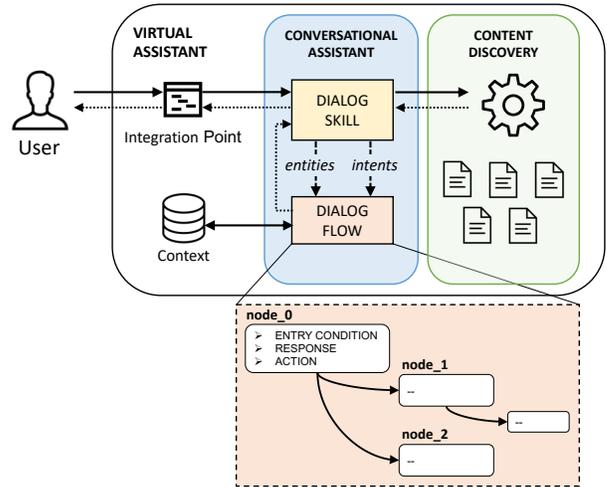

Fig. 2: Virtual assistant architecture. Users interact with the system through an *integration point*, which forwards the requests to the *dialog skill* in order to recognize *intents* and *entities*. Optionally, the *content discovery* module is called to extract relevant information from unstructured documents. The *dialog flow* produces the correct answer, which is later shown to the user through the same *integration point*.

chatbot, and routes it to the *dialog skill* component (a container for training data, dialogs, and the logic that enables the assistant to help the user) in order to perform input interpretation and handle the *dialog flow* accordingly.

Developing a virtual assistant requires to identify the *intents* of the users, i.e., the goals that are most likely to be expressed by end users.

Once *intents* have been identified, it is necessary to provide, for each of them, few examples of possible users' inputs. The *dialog skill* component is responsible for learning from these examples and deriving patterns and strategies which allow it to correctly identify each intent, even for inputs which have never been seen before.

Moreover, the *dialog skill* is also responsible for the detection of *entities*, i.e., terms or objects which may be relevant to the *intents*, usually providing specific context. For example, a restaurant which provides different menus can identify an *intent* for the users who ask to see the menu, and one *entity* to distinguish between each menu.

Identifying these *entities* requires the developer to list all the possible values they can assume. Optionally, each value can have synonyms or patterns (i.e., *regexps*) which can be used to make their definition more general. Some *intents* and *entities* addressing the case study considered here are shown in Table I.

The *dialog flow*, which defines how the application responds to inputs, is represented by a tree whose nodes correspond to every possible answer. Each node is composed by an *entry condition*, which needs to be evaluated as true to access the node, a *response*, and an *action* to be performed after leaving the node (e.g., waiting for new inputs, jumping to another

Intent Name	Example questions
#enrollment	Can you help me with my enrollment?
	What do I need to do in order to correctly complete the enrollment?
#free_classroom	Is the classroom "A310" currently occupied?
	Are there any lessons at the moment in the classroom "B220"?
#university_info	What do you know about the university of Palermo?
	Tell me about the university
	Where is the headquarter of the university?

Entity Name	Example values
@city	Palermo, Agrigento, Trapani, Marsala, Caltanissetta
@enrollments	TOL, TOL-C, Bachelor's degree, Master's degree

TABLE I: Some of the intents and entities which are used for our virtual assistant.

node).

For every conversation round, the topmost node whose condition is evaluated as true is the one that defines the *conversational assistant*'s answer.

Common conditions are the detected *intents*, *entities*' values, and *context*, that is additional information provided by the application representing the state of the dialog. Usually, the last node contains a condition which is always evaluated as true, namely "*anything_else*", which allows to provide answers to those inputs that are not matched by any of the previous nodes. This is generally used to ask users to rephrase the input.

B. Content discovery

The second subsystem includes a cognitive, search, and content analytics engine, which allows to identify patterns, trends, and insights from unstructured data.

Using data analysis combined with cognitive intuition and natural language processing, this module is able to enrich unstructured data, extracting relevant features such as concepts, relations, entities, sentiments, keywords and so on.

Content discovery needs a set of documents to work with, so a preliminary phase of document collection is required. Once the documents have been collected and uploaded, they are ingested, indexed and enriched, in order to extract relevant pieces of information. At this point, documents can be queried at any time, either in natural language or using a specific syntax, called *Discovery Query Language*, which allows to navigate through documents and filter them. Moreover, it provides aggregate functions in order to produce more complex responses.

However, *content discovery* often needs a training phase in order to improve the relevancy of its answers. This training requires both to provide enough example queries, representative of what the end users are most likely to ask, and to rate the results of each query as *relevant* or *non-relevant*.

Once enough information has been provided, the *content discovery* service will start to learn. It is important to point out that this subsystem does not just memorize shown examples, but it learns from individual queries and applies the patterns it has detected to all new queries. This training will be further inspected in Section V-B.

V. CASE STUDY

In this section we show some of the results achieved, and discuss how the training phases we carried out improved the performances of the proposed virtual assistant.

A. Setting up the virtual assistant

In order to design a virtual assistant, one of the first steps consists in identifying which questions are most likely to be asked. Some of them, referred to common topics, are quite simple, will occur frequently, and can be somehow predictable. These questions, called *short-head*, should be answered immediately in order to optimize the user experience. Conversely, *long-tail* questions require complex answers and their frequency is not very high.

In the assistant we propose, *short-head* questions are handled by the conversational assistant module, since identifying an intent for each of them is easy; *long-tails*, on the other hand, require the content discovery subsystem because of their complexity and variety. However, during its execution, the assistant should also be able to learn from the most asked questions in order to convert *long-tail* questions to *short-heads*, identifying new intents and reducing the response times.

Once questions were identified and categorized, we developed our assistant using the Watson Web Interface and Java SDK. In order to increase the ease of use both for students and staff members, the main user interface looks like a classical web chat. Moreover, the services of the assistant are also provided through a mobile application, so that users can receive help, wherever they are.

Our assistant is capable of answering different types of questions, either with already-prepared answers, or by exploiting the content discovery subsystem and elaborating its answers. Table II shows an example of conversation. The intent behind the first user's input is recognized and the assistant is capable of immediately answering, while the second input requires more efforts. In particular, the content discovery subsystem is called and its response is elaborated into assistant's final answer.

When content discovery is needed, a remote database is queried in order to retrieve useful pieces of information from a collection of documents. The definition of such a collection requires identifying all those documents which can be useful for the assistant in order to answer users' questions.

In our case study, the collection contains the university regulations and some relevant web pages, e.g., the FAQs, retrieved from the University of Palermo's website. These pages were also preprocessed according to guidelines identified by [16], which dictate how to split and modify documents in order to increase results' relevancy.

B. Experimental Results

In order to evaluate the performances of the question-answering system, we selected a group of heterogeneous users that were asked to interact with the assistant and rate its answers.

Welcome! I'm the virtual assistant for Unipa. How can I help you?
Hi, I need help with my enrollment
[Intent Detected : #enrollments_info]
First and foremost, you need to signup on university web site: <u>link</u>.
Make sure you have the followings:
1) a valid ID document
2) your social security number
3) your family ISEE indicator (which will help you to pay less taxes)
4) a valid email and telephone number
5) a picture of yourself, in .jpg format, 420x480 px, less than 200KB
Starting **Wed, 1 August 2018** you can complete your enrollment on our website. Check our curricula page in order to access the full list of courses we are providing for the academic year 2018-2019. Also, make sure to check that your chosen course allows you to signup at this date.
Can you help me locating the central administration building?
[Intent Detected : #discovery]
I cannot answer right now, give me a moment to search through my documents. Please, wait.
These are the most relevant passages I've found:
Passage #1
"(...) The University of Palermo is a statal university, founded in 1806. The central administration building is located in the historical building *Palazzo Chiaramonte-Steri* (...)"
More on: unipa_wiki_introduction.pdf
Passage #2
"(...) After World War II, the university achieved its first 10.000 students. In 1950s, the central administration building was moved to *Palazzo Steri*. (...)"
More on: unipa_wiki_history.pdf

TABLE II: Example of conversation with our virtual assistant (note that actual questions were asked in Italian).
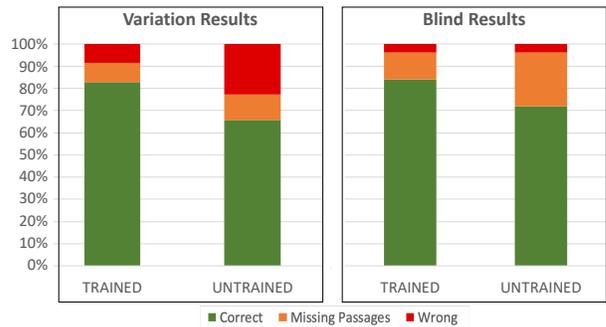

Fig. 3: Results achieved by trained and untrained version of the content discovery subsystem, both for variations (left) and blind questions (right).
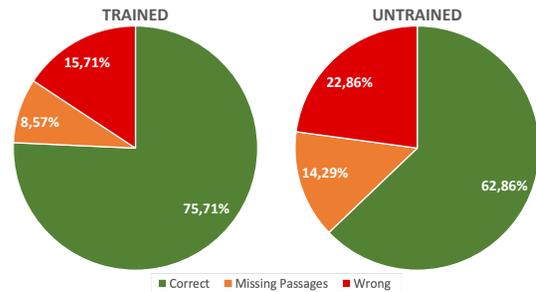

Fig. 4: Overall results achieved by both the trained and untrained systems.

We chose to separately evaluate the *conversational assistant* and *content discovery* subsystems, since their answering capabilities are quite different. Indeed, while the former simply recognizes the intent and provides the matching answer, the latter's engine searches through its collection and returns the most relevant passages and the corresponding documents.

Although the identification of intents and entities allows to immediately query the system, preliminary tests carried out on such a base version of the assistant showed that it was able to correctly answer to 60% of the questions only. As a consequence, we chose to perform a phase of training in order to improve the relevancy of the answers.

The training was differently performed on the two main subsystems. Indeed, as for the *conversational assistant*, if it detects a wrong intent, it is required to choose the correct one and the system learns from its mistakes. Conversely, *content discovery* requires a multi-step training during which different sample queries are provided and their results are individually evaluated as relevant or non-relevant.

Starting with *conversational assistant*, we identified a small set of intents and provided few examples for each of them. Then, we asked the assistant different questions in order to evaluate its performances, comparing the detected intent with our ground-truth. We also monitored users' interactions with the system in order to get as much data as possible. These training steps led to a great increase in the performances of the *conversational assistant* subsystem, setting the precision to 85%.

The *content discovery* subsystem, on the other hand, was provided with almost 50 documents regarding the University of Palermo, including FAQs, guides and news about the university itself. Experiments carried out both on a trained and untrained version of the system, where the training was done providing about 60 sample queries and evaluating the results. For each query, we labeled the answers as follows:

- **correct**: there's a correct, close answer to the question in at least one of the passages;
- **missing passages**: passages are wrong (i.e., they do not contain the answer), but the corresponding documents are the ones we expected to be returned;
- **wrong**: both passages and documents are wrong.

Furthermore, inspired by [13], we identified and used three categories of questions: *variation*, *blind*, and *out-of-scope*. The first category contains all those questions which are slightly different from the ones used for training. The *blind* category refers to all those questions which were not used for the training, yet having a close answer contained within the provided documents. Finally, *out-of-scope* includes all the other questions, whose answers are not contained in the document collection.

Results for the first two categories, presented in Fig. 3, show that the relevancy training actually improves the discovery performances, bringing to a 15% decrease in wrong answers for the *variation* category, and a 10% increase in correct answers for the *blind* category.

However, both the trained and untrained versions of the system made the exact same mistakes regarding *out-of-scope* questions, showing problems which may also be related to the language. In this regard, it is worth noting that we want our assistant to be able to speak Italian, which currently is not fully supported by the Watson services.

In order to deal with *out-of-scope* questions, one common strategy implies the identification of intents and questions which need to be redirected to the discovery module, so as to provide a default answer to all the others.

Overall results, obtained by combining both the *conversational assistant* and the *content discovery*, are presented in Fig. 4. These results confirm the importance of the training, which increases the overall correct answer percentage by almost 15%.

VI. CONCLUSIONS

Nowadays, virtual assistants can be found on every smart device and are constantly used by people for a variety of purposes. In this work we presented a virtual assistant that can assist people within a university campus.

To this aim, a conversational assistant, capable of answering to common questions, has been combined with a content discovery engine, that is more suitable for finding the proper answers from a collection of heterogeneous sources.

Despite the Italian language being not fully supported by the IBM services we relied on, the evaluation of the assistant, performed by a group of students and staff members, brought to a 75% of correct answers, also increased by a relevancy training.

Once the assistant is deployed and used on a large-scale, one important factor to be considered is its functioning cost. The IBM services have a specific cost for each API call, so an optimization strategy is required to reduce the interactions between the assistant and the Cloud. For instance, a possible solution to reduce the calls to the content discovery subsystem would involve storing its answers, and repeating them in the case that other users ask the exact same question.

As other future work, we plan to develop a new subsystem dedicated to document pre-processing, so that new documents do not have to be manually inspected and elaborated to be *well-formed*. Moreover, in order to obtain close answers to any type of question, a post-processing module capable of further analyzing responses coming from the *content discovery* engine could be designed. We also noticed that, in some cases, *out-of-scope* questions could be asked by untrustworthy users, that are interested on testing/stressing the system by providing unpredictable inputs. To overcome this limitation, a trust management mechanism could be adopted [6], [2].

Finally, the virtual assistant functionalities could be extended to provide the staff of the campus with a smart interface to some management tasks, such as easily monitoring the energy consumption within the university buildings [8], and controlling the environment by sending commands to a set of actuators [7].

REFERENCES

[1] Vincenzo Agate, Federico Concone, and Pierluca Ferraro. Wip: Smart services for an augmented campus. In *The 4rd IEEE International Conference on Smart Computing (SMARTCOMP 2018)*, Italy, June 2018.

[2] Vincenzo Agate, Alessandra De Paola, Giuseppe Lo Re, and Marco Morana. *A Simulation Framework for Evaluating Distributed Reputation Management Systems*, pages 247–254. Springer International Publishing, Cham, 2016.

[3] Mohamed Nooman Ahmed, Andeep S Toor, Kelsey O'Neil, and Dawson Friedland. Cognitive computing and the future of health care cognitive computing and the future of healthcare: the cognitive power of ibm watson has the potential to transform global personalized medicine. *IEEE pulse*, 8(3):4–9, 2017.

[4] Christopher Asakiewicz, Edward A Stohr, Shrey Mahajan, and Lalitkumar Pandey. Building a cognitive application using watson deepqa. *IT Professional*, 19(4):36–44, 2017.

[5] Federico Concone, Pierluca Ferraro, and Giuseppe Lo Re. Towards a smart campus through participatory sensing. In *4th IEEE International Workshop on Sensors and Smart Cities (SSC 2018)*, Italy, June 2018.

[6] C. Crapanzano, F. Milazzo, A. De Paola, and G. Lo Re. Reputation management for distributed service-oriented architectures. In *Self-Adaptive and Self-Organizing Systems Workshop (SASOW), 2010 Fourth IEEE International Conference on*, pages 160–165, 2010.

[7] A. De Paola, A. Farruggia, S. Gaglio, G. Lo Re, and M. Ortolani. Exploiting the human factor in a wsn-based system for ambient intelligence. In *Complex, Intelligent and Software Intensive Systems, 2009. CISIS '09. International Conference on*, pages 748–753, 2009.

[8] A. De Paola, G. Lo Re, M. Morana, and M. Ortolani. An intelligent system for energy efficiency in a complex of buildings. In *Sustainable Internet and ICT for Sustainability (SustainIT), 2012*, pages 1–5, 2012.

[9] Alessandra De Paola, Pierluca Ferraro, Salvatore Gaglio, and Giuseppe Lo Re. Context-awareness for multi-sensor data fusion in smart environments. In *AI*IA 2016 Advances in Artificial Intelligence: XVth International Conference of the Italian Association for Artificial Intelligence, Genova, Italy, November 29 – December 1, 2016, Proceedings*, pages 377–391. Springer, 2016.

[10] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.

[11] Paul Fodor, Adam Lally, and David Ferrucci. The prolog interface to the unstructured information management architecture. *arXiv preprint arXiv:0809.0680*, 2008.

[12] S. Gaglio and G. Lo Re. Advances onto the internet of things: How ontologies make the internet of things meaningful. *Advances in Intelligent Systems and Computing*, 260, 2014.

[13] Ilianna Kollia and Georgios Siolas. Using the ibm watson cognitive system in educational contexts. In *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on*, pages 1–8. IEEE, 2016.

[14] Adam Lally, John M Prager, Michael C McCord, Branimir K Boguraev, Siddharth Patwardhan, James Fan, Paul Fodor, and Jennifer Chu-Carroll. Question analysis: How watson reads a clue. *IBM Journal of Research and Development*, 56(3.4):2–1, 2012.

[15] Burn L Lewis. In the game: The interface between watson and jeopardy! *IBM Journal of Research and Development*, 56(3.4):17–1, 2012.

[16] Syed Shariyar Murtaza, Paris Lak, Ayse Bener, and Armen Pischdotchian. How to effectively train ibm watson: Classroom experience. In *System Sciences (HICSS), 2016 49th Hawaii International Conference on*, pages 1663–1670. IEEE, 2016.

[17] Kenzhegali Nurgaliyev, Dario Di Mauro, Nawaz Khan, and Juan Carlos Augusto. Improved multi-user interaction in a smart environment through a preference-based conflict resolution virtual assistant. In *Intelligent Environments (IE), 2017 International Conference on*, pages 100–107. IEEE, 2017.

[18] Svetlana Stoyanchev and Michael Johnston. Localized error detection for targeted clarification in a virtual assistant. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5241–5245. IEEE, 2015.

[19] Gerry Tesauro, DC Gondek, Jon Lenchner, James Fan, and John M Prager. Simulation, learning, and optimization techniques in watson's game strategies. *IBM Journal of Research and Development*, 56(3.4):16–1, 2012.