VOICE-FIRST: Supporting Human Assistants with Real-time Voice Understanding

1st Mario Corrado
IN & OUT S.p.A. Single Shareholder Teleperformance S.E.
Taranto, Italy
mario.corrado@teleperformance.com

2nd Vincenzo Giliberti
IN & OUT S.p.A. Single Shareholder Teleperformance S.E.
Taranto, Italy
vincenzo.giliberti@teleperformance.com

3rd Manuel Gozzi
Isagog Srl
Rome, Italy
m.gozzi@isagog.com

4th Vincenzo Lanzolla
IN & OUT S.p.A. Single Shareholder Teleperformance S.E.
Taranto, Italy
vincenzo.lanzolla@teleperformance.com

5th Guido Vetere
Università Guglielmo Marconi
Isagog Srl
Rome, Italy
ORCID 0000-0002-6703-7276

6th Domenico Zurlo
IN & OUT S.p.A. Single Shareholder Teleperformance S.E.
Taranto, Italy
domenico.zurlo@teleperformance.com

***Abstract*—While AI and automation have made significant strides in customer support, there are still situations where human intervention via voice channels is necessary to provide the best possible customer experience. In fact, although AI and chatbots have become increasingly sophisticated, they may not always be able to handle complex or nuanced customer issues. Human agents can better understand and respond to these situations, providing tailored solutions. At the same time, solving non-trivial customer problems often requires access to knowledge bases and contextual customer information, for which AI is particularly well suited. Hence the idea of integrating human and artificial intelligence in a hybrid solution. We developed an AI system to help human assistants in the process of handling conversations. This system can be viewed as a collaborative bot (cobot). The cobot captures the audio stream of the conversation, converts it to text and analyzes it in real time. The extracted tokens are classified and sent to a reasoning system based on a knowledge graph, that provides information and action suggestions to the human assistant. Assistants are also capable of providing information to the reasoning system, utilizing their human understanding of the client's circumstances as they unfold. While designing a prototypical solution for utility services, we have faced the problem of real-time use of computationally complex procedures, including spontaneous speech understanding and knowledge-based heuristic rules. Moreover, we adopted a standards-based approach and experimented with open source reasoners and publicly available language models. The paper outlines the system architecture and design, and discusses the results of the first experiments.**

***Index Terms*—virtual assistants, natural language understanding, knowledge graph, real time reasoning**

This work has been co-funded by EU and Puglia Region (Italy) R&D Project "VOice Intelligence for Customer Experience (VO.I.C.E. First)", based on AI, NLP and Learning Machine for the multichannel contact center industry [9] [10]

I. Introduction and Overview

The use of intelligent virtual assistants (IVAs) for complex tasks, like assisting clients in resolving issues or accomplishing administrative processes, is considered an essential business requirement. Despite IVAs being capable of executing computations, engaging in seemingly human-like dialogues, responding to queries posed in everyday language, and carrying out tasks on users' devices, a significant number of customer situations still necessitate human involvement. In these cases, it is customary to switch the virtual assistant to a human one. However, this may be frustrating for the customer. Thus, we experimented with a different approach: rather than involving customers in possibly unproductive automated conversations, our system aids human assistants during their interactions, striving to reduce interaction duration and enhance the quality of service provided. This system can be viewed as a collaborative bot (cobot). In order to execute this approach, we had to address the challenge of real-time natural language understanding of spontaneous speech. This challenge includes the following main aspects:

1) Speech recognition: In case of spoken language input, the system needs to convert the audio into text. This is achieved using automatic speech recognition (ASR) technology, which has improved significantly in recent years due to advances in deep learning.
2) Linguistic analysis: The system analyzes the grammatical structure of the input text to identify relationships between tokens and determine their functions within a sentence. The system identifies the user's intention and extracts relevant information from the text. This allows the system to provide an appropriate response or perform

a specific action.

3) Reasoning: Based on the input and analysis, the system generates a response. This may involve selecting a predefined response, generating a response using natural language generation (NLG) techniques, or a combination of both.

A. Speech recognition

To get started, we have put in place a service for converting telephone conversations from audio to text. The system used for this component allows for real-time transposition of the customer's portions of the conversation audio, into the text which will then be analyzed by the Natural Language understanding components. The implementation includes a service (Audio Capture Service) installed on the agent's workstation, whose output is sent to some server services (speech to text, `realTime webWorkers`), which converts it into text and cause the agent dashboard to show the transcript in real time. Speech to text conversion resorts on cloud transcription SDK. In a continuous stream, the speech to text module sends the bytes of the conversation to Microsoft services, which return the corresponding text. This text is made up of a set of sentences combined with conversation segments, between which there was a short pause of silence or which lasted 15 seconds. Through session handler module, the text sentences are saved in a cache database, which allows text analysis services to quickly recover the parts of the conversation to which to apply the analysis processes in real time. Finally, the text is sent to the agent dashboard using `SignalR` mechanisms and a unique conversation identifier that is subsequently used to retrieve the source workstation of the converted audio. At the end of the conversation, the set of sentences and analysis metadata is retrieved from the cache database and saved in a NOSql database.

B. Linguistic analysis

We analyzed about 3000 transcripts of real conversations provided by a utility company, using a proprietary web application. This enabled us to acquire transcriptions of audio files from recorded phone conversations, offering capabilities such as anonymizing transcriptions, removing confidential information, and incorporating metadata, which encompasses a high-level categorization of the conversation's objective. Then we chose 2151 fragments containing interesting content and annotated them with a tag set of intents and topics taken from a domain ontology of our conception, whose discussion is out of the scope of the present work. We thus obtained a training set to feed a number of classifiers for conversation segments. In order to apply these classifiers, we had to find a way to efficiently segment user utterances. Shallow parsing (chunking) of Italian texts has been extensively studied in the past decades [6]. With respect to our specific goal, we opted for a simple heuristic based on analyzing coordinated conjunctions. This has proved to be effective when applied to users' expression in real time.

Our classifier adopts a transformer based approach [7] based on a multilingual linguistic model supporting Italian, namely Bertino, by indigo.ai, trained on a large Italian general domain corpus that combines both good performance and small weight compared to other models [8]. Although `GPT3 Davinci` [2] showed better results, we considered, for a real-time system, the problem of relying on remote services via API.

C. Reasoning

The parsed blocks of text are sent to a reasoning service, which makes inferences about the ongoing conversation. This service consists in a Knowledge Graph based on Apache Jena, which is comprised of background domain knowledge and customer's data which is dynamically loaded at the conversation start. The heuristic reasoning on the conversation content is obtained by `Description Logic (DL)` inference (in particular, *instance checking*) and business rules, operating on ontology definitions.

The effectiveness of the ontology plays a crucial role in the system's overall functionality. To ensure this, we devised a foundational layer consisting of a few broad categories and relationships. These categories and relationships were derived from established standards, such as [4] and [1], both featuring fundamental distinctions among events, objects, and non-physical entities. Building upon prior research efforts, like [14] and [12], we further included domain-specific models for dialogue acts and their connections with referenced entities.

We adopted Openllet [1], an open source implementation of the Pellet `DL` reasoner [13], which supports `SWRL` [2] rules for hybrid reasoning. We evaluated the expressiveness of `SWRL` as adequate to represent our heuristic rules [5].

II. Architectural Overview

When the assistant responds to a phone call and identifies the customer, the corresponding customer data is retrieved from a Customer Relationship Management System (CRM) and stored in a ephemeral (temporary) Knowledge Graph that exists in the computer's memory. The transformation of CRM data into KG assertions is granted by a suitable mapping. This temporary graph serves as the system's real-time understanding of the ongoing conversation. Subsequently, the cobot incorporates information into the KG by adding statements based on input received from a Natural Language Processing (NLP) pipeline, which analyzes user utterances, and from the assistant itself, communicated through an internal chat interface.

With each update, the cobot engages in a reasoning process by considering the customer data obtained from the CRM system, the customer expressions interpreted by the NLP pipeline, and any additional information it gathers from the assistant. This collaborative process allows the cobot to make informed decisions based on a comprehensive understanding of the available information.

The figure 1 gives an overview of the system. It reads as follows:

[1]https://github.com/Galigator/openllet

[2]https://www.w3.org/Submission/SWRL/

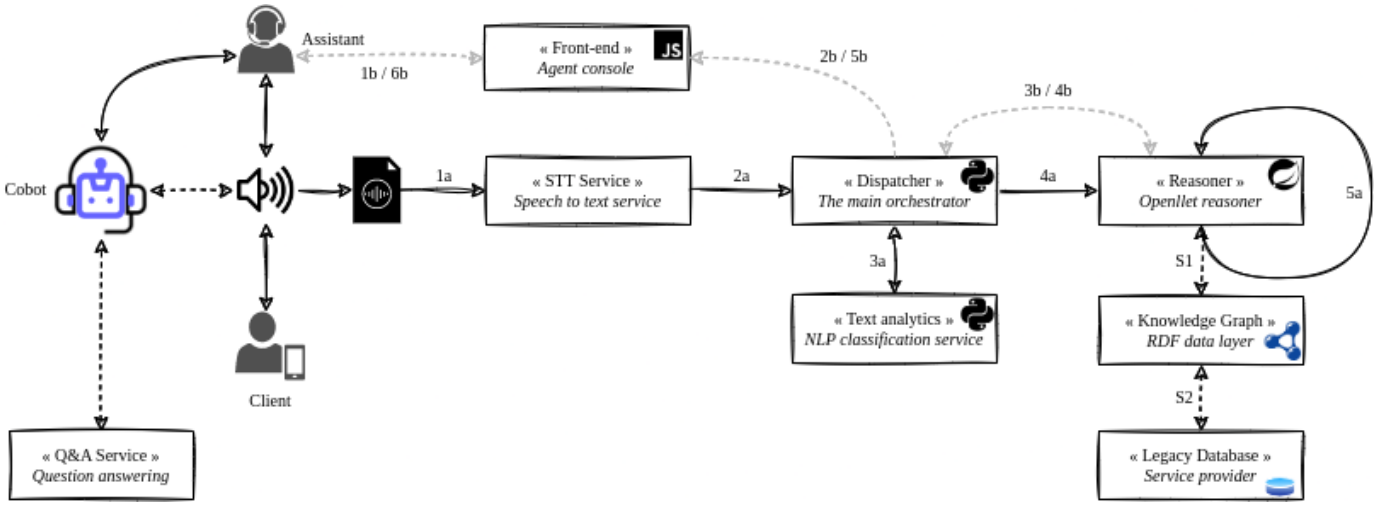

Fig. 1. System Overview

- 1a Assistant and customer talk. A listening service captures the audio stream and sends it in real time to the STT service
- 2a The STT service converts the audio into natural text and propagates the result to the dispatching service
- 3a The dispatching service forwards the input to the text classification service, which identifies the keywords that categorize the act of speech in progress.
- 4a The dispatching service sends the classification result to the reasoning service
- 5a The reasoning service translates the classification results in a suitable update to the KG
- 1b During the whole phone call, the assistant uses a console that summarizes the state of the speech
- 2b The console "asks" the dispatching service for the presence of inferences that can be deduced from the content of the graph itself.
- 3b The dispatching service forwards a request to the reasoning service with the reference to the KG. The reasoning service receives the request and launches a SPARQL query to compute the inferences, thus starting the reasoning process.
- 4b The output of the reasoning process is encoded and sent back to the calling service.
- 5b The dispatching service propagates the result up to the console.
- 6b Once the reasoning result is received by the console, a message for the Assistant is produced in various forms (pop-up, windows opening)

A sequence diagram of the main flow is depicted in figure 2

III. Measurements

It is expected that, during the conversation, the system will be able to provide the assistant with useful information, e.g. recovery operations to be performed on certain devices. Of course, this has to happen in sync with the flow of the conversation. With the help of field operators, we have figured out that, for the cobot to be useful, an acceptable reaction time should not exceed 5 seconds. In practice, this means that up to 5 seconds can elapse between the moment the user utters a revealing phrase and the moment the cobot shows the assistant useful information.
Through some simulations, we measured the average times of each activity phase shown in figure 2, the results are shown in the table I

The results have been obtained by examining a sample of conversations with an average length of 1 minute, from which 8 chunks have been extracted and sent to the reasoner. The infrastructure has been deployed on leading cloud service provider to achieve scalability, reliability and availability, security, disaster recovery and backup. Host characteristics for testing are: 1) Cloud Service Plan (8 vCPU, 32GB RAM, 250GB storage) on which text analysis services, session management, and web features are installed. 2) Cloud-managed Redis service.

IV. Conclusion

Not surprisingly, the bottleneck of the cobot's answering process is logic inference, as the table I confirms. To support heuristic reasoning, we adopted SWRL, a powerful combination of OWL-DL and Horn clauses known to be computationally demanding, up to the limit of undecidability [11]. On the other hand, modeling business rules in a non-trivial domain seems to require such high expressiveness. From this experience, we draw the conclusion is that the possibility to support real-time human decision processes in a scenario such as customer assistance, by means of knowledge-representation methods, strongly depends on how the business knowledge is represented and organized. In particular, carefully designing and testing rules emerges as is a key requirement. In turn, drawing efficient rules depends on how the ontology represents key business properties.

Different approaches, such as deriving the cobot's behavior from tuning large neural language models [3], may be far more efficient and maybe easier to implement, but this comes at the cost of loosing control, transparency, and auditability on the answering process, as well as the burden of integrating automated interactions with external systems. Furthermore, to reason about the current situation of actual customers, it is essential to have access to real-time data, including factors like consumption rates or billing status. These dynamic data points need to be jointly considered in the reasoning process. It is important to acknowledge that, conversely, the training process of neural networks relies on the assumption of data having a certain level of stability or static nature.

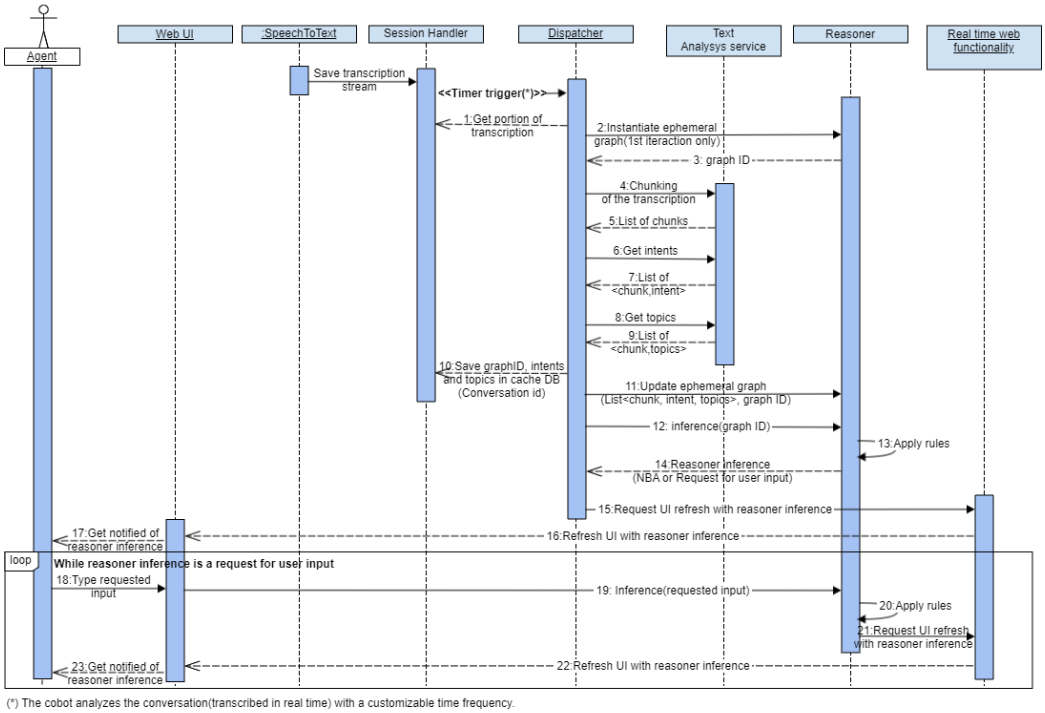

Fig. 2. Sequence diagram

Step	Description	System	Notes	Time
1	Obtain from the session cache the portion of the transcript to be analyzed	Session Handler	Redis cache	50
2	Instantiate ephemeral graph for the conversation identified by conv_ID	Reasoner	Performed only once per conversation	103
4	Identify chunks	Text Analysis Service		156
5	Calculate intent for each chunk	Text Analysis Service	Parallel task with topic extraction; NLP fine tuned Bert for text classification(intent)	345
6	Calculate topic for each chunk	Text Analysis Service	Parallel task with intent understanding; NLP fine tuned Bert for topic extraction	256
7	Save session data	Sessione Handler	Redis cache	150
8	**Update ephemeral graph**	**Reasoner**	(intent, topic)	**2847**
9	Inference for graph_ID	Reasoner	(topic)	138
12	Request UI refresh with reasoner inference	Real Time Web Functionality	Next Best Action or request for user input	75
13	Display Reasoner inference	Cobot's web UI		120

TABLE I
MEASUREMENT TABLE

In conclusion, a full reliance on cloud services may not ensure optimal performance and user experience. However, by integrating resources through the acquisition of isolated virtual machines, significant improvements can be achieved. The cost-benefit trade-off of such provisioning should be carefully considered. Generally speaking, on-premise solutions could be justified by better performance control, as well as the potential benefits on the capabilities of the reasoner to automate tasks on external systems. These factors depend on the operating environment in which the system is deployed. We are currently in the process of planning field experiments to evaluate the effectiveness of the presented approach in real-world applications. We are well aware that reasoning on knowledge bases with ontologies and expressive rules is computationally demanding, and in some cases it may not be possible to meet real-time constraints. However, in our application scenario, real time should be intended as the time required for the system to provide the assistant with useful information during the ongoing conversation. Latencies are therefore tolerable within certain limits. On the other hand, the ability to automatically reason on customer data in the background could relieve the assistant of the burden of extensively analyzing this data and mitigate the risk of missing relevant details. This makes us believe that it is worth pursuing

our research.

REFERENCES

[1] Robert Arp, Barry Smith, and Andrew D. Spear. *Building Ontologies with Basic Formal Ontology*. The MIT Press, 2015.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[3] Erik Brynjolfsson, Danielle Li, and Lindsey R Raymond. Generative ai at work. Working Paper 31161, National Bureau of Economic Research, April 2023.

[4] Aldo Gangemi, Nicola Guarino, Claudio Masolo, Alessandro Oltramari, and Luc Schneider. *Sweetening Ontologies with DOLCE*, pages 166–181. Springer, Berlin, Heidelberg, 2002.

[5] Abba Lawan and Abdur Rakib. The semantic web rule language expressiveness extensions-a survey. *CoRR*, abs/1903.11723, 2019.

[6] Alessandro Lenci, Simonetta Montemagni, and Vito Pirrelli. Chunk-it: An italian shallow parser for robust syntactic annotation. 01 2001.

[7] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. Deep learning–based text classification: A comprehensive review. *ACM Comput. Surv.*, 54(3), apr 2021.

[8] Matteo Muffo and Enrico Bertino. Bertino: An italian distilbert model. In *CLiC-it*, 2020.

[9] Gabriele Papadia, Massimo Pacella, and Vincenzo Giliberti. Topic modeling for automatic analysis of natural language: A case study in an italian customer support center. *Algorithms*, 15(6), 2022.

[10] Gabriele Papadia, Massimo Pacella, Massimiliano Perrone, and Vincenzo Giliberti. A comparison of different topic modeling methods through a real case study of italian customer care. *Algorithms*, 16(2), 2023.

[11] Bijan Parsia, Evren Sirin, Bernardo Grau, Edna Ruckhaus, and Daniel Hewlett. Cautiously approaching swrl. 05 2005.

[12] J. Silva, D. Melo, and I. Rodrigues. An ontology based task oriented dialogue. *In Proceedings of the 13th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2021)*, 2.

[13] Evren Sirin, Bijan Parsia, Bernardo Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical owl-dl reasoner. *SSRN Electronic Journal*, 01 2007.

[14] Michael Wessel, Girish Acharya, James Carpenter, and Min Yin. *OntoVPA: An Ontology-Based Dialogue Management System for Virtual Personal Assistants: 8th International Workshop on Spoken Dialog Systems*, pages 219–233. 01 2019.