Iride®: an Industrial Perspective on Production Grade End To End Dialog System

Cristina Giannone, **Valentina Bellomaria**, **Andrea Favalli** and **Raniero Romagnoli**

Language Technology Lab
Almawave srl
[first name initial].[last name]@almawave.it

Abstract

This paper aims at describing, from an industrial perspective, the experience in delivering conversational agents via the development of Iride, a platform able to deploy multi-language task-oriented dialog systems. It has been implemented a set of functionalities that can be aggregated in different ways, in order to build domain independent conversational systems, which are able to satisfy needs of real business cases. Along with algorithms and techniques for end to end Dialog management, such as Natural Language Understanding (NLU), Question Answering (QA) and Dialog State tracking and policy management, the technical insights leveraged into the platform are described by outlining the requirements and constraints emerging from these on the field experiences.[1]

1 Introduction

Over the last years the human computer conversation has been gathering increasing attention due to its promising potentials by opening up a new profits-making market segment. [2]. The benefits of using dialog systems are manifold, these systems can answer to complex questions and also handle hundreds, thousands of conversations at the same time, reducing response times and probability of error in repetitive tasks. In General, developing conversational agents at industrial level requires to manage several issues: (i) The lack of real data: in the majority of the real business cases, in our experience, not enough data are available for training pure learning methods, moreover, the research datasets do not fit the industrial purposes; (ii) Domain updates and system maintenance: The domain requires continuous updates (e.g. the introduction of a new product or service) and the delivered system needs the maintenance, update or changes to correct faults and to improve performance; (iii) User Experience: the conversational agent is the front end of the company, multi-modality (i.e. different user experiences depending on different devices) and what the company aims at communicating must be taken into account; (iv) Runtime latency: is required to add no more than few mini seconds to the entire serving stack; (v) Scale and quality of the text collection: in a voice interaction the system cannot answer with a long text document, but needs to answer with a clear short document passage; (vi) Certified Answers: Being the virtual assistant the voice of the company, it must be controlled (i.e. usually the answers and the messages communicated by the assistant have to be certified by the company); (vii) Human in the loop: Although virtual assistants are becoming more and more intelligent, they are not able to satisfy every user need. In this scenario, it would be better a mixed management, combining the use of virtual agent and human operator.

In this paper, we describe the Almawave's developed solution that allows us to quickly design, write and deploy interactive conversational systems without coding, enabling non-technical users (i.e. conversational designers or domain experts) to design conversational agents, and it leverages Natural Learning Processing (NLP) and Machine Learning (ML) to develop a human-like experience for users. This framework is designed to build multi-turn task-oriented dialog able to solve defined tasks and answer to domain questions.

Following, in section 2 related works will be

[2]https://www.gartner.com/smarterwithgartner/4-trends-gartner-hype-cycle-customer-service-customer-engagement/

discussed: in section 3, the various goals that have leaded the described solution will be discussed; in section 4 the various modules of the architecture will be fully described; finally, in section 5, we formulate some considerations and lessons learned in the conversational agent field.

2 Background

Due to the complexity of task, most studies on Human Machine conversation have addressed individual components such as Intent/Slots detection (Coucke et al., 2018) or Dialog State tracking (Mrksic et al., 2015) about frameworks for building an effective dialog system. Recent works in the end-to-end frameworks are focused on the pure learning approaches, where the sequence of dialog interactions, between the user and the agent, is acquired from large datasets (Wu et al., 2017), (Wen et al., 2017), as well as in the dialog task oriented field (Bordes and Weston, 2016). Although Neural Networks provided a significant improvement in the NLP field, in the conversational agent field, NN end-to-end systems have some limitations, all their components are directly trained on past dialogues, with no assumption on the domain or dialog state structure, thus training with large scale human-human dialog data is required. However, these resources are generally not so easily available for building an end-to-end system. Some works based on NN address on limit the amount of training data: the framework proposed in (Bocklisch, 2017) focused on quickly helping implement machine learning-based dialog management and natural language understanding, the work implements a function to generate, from the input dataset, new data and provided a special function called a story graph that visualize the flow of dialog scenarios in advance. In (Lipton et al., 2017) a deep reinforcement learning algorithm is proposed to tackle a domain extension setting, where new slots can gradually be introduced. On the other hand, in (Lison, 2015), the authors proposed a framework for expressing dialog behaviors as probabilistic rules. The probabilistic rules used in this study consist of conditional statements and actions with probability; these can be made manually or automatically generated by supervised learning or reinforcement learning. Following (Yan et al., 2017), our proposal is toward a platform for the development of a conversational agent able to perform a cold-start with no dialog training data. Other close works address on the building of frameworks in order to allow the development of conversational agents in several scenarios and domains, in (Crook et al., 2016) is proposed a task configuration language, i.e *TaskForm*, which allows to decouple the conversation management issues with the definitions of the target task, and moreover make available a large set of ML algorithms for the NLU tasks. In a recently proposed platform (Sungjin Lee and Gao, 2019), the issue of evaluating the end-to-end conversational agent is approached.

3 Goals

From our experience, the main objectives of a dialog system for business needs are the usability and the robustness. The system must always be functioning in time and satisfy user needs by operating as few interactions as possible. The conversational platform here proposed was developed with some characteristics concerning those objectives:

3.1 Usability

The usability principles for this kind of framework look to user designers. The main issues to purse the usability goal are described in the following:

Focus on conversation design

Designing a dialog conversation must take into account both what has already been said and what will happen next; it is much more complex than one-off activities, like answering a search query, playing a song and so on. In relation to this, new professions are emerging, such as the Voice User Interface (VUI) designer who curates the conversation, defining the flow and its underlying logic in a detailed design specification that represents the complete user experience, playing an important role from the conceptual phases of the project until its launch (Urban and Mailey, 2019).

However, these profiles are not necessarily developers or data scientist, so it is very important that tools offer to them all the available technology but are easy to use, so that the designer can focus on aspects more related to domain and policies of dialog management. A solution we delivered to solve this problem is a Visual Dialog Editor, hiding the complexity of programming AI components, allowing the user to construct a dialog agent with a visual building block approach, the drawn flow is thus compiled producing the dialog agent

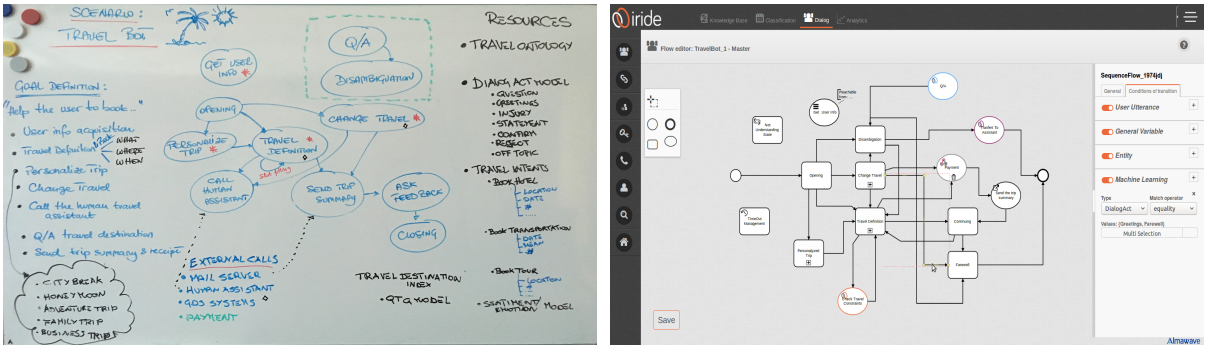

Figure 1: A conversation design process, from conversational map to dialog model in the Iride Conversational Platform

software. In Figure 1 an example of a conversational design process is shown, from a conversational map that highlights the important items to the dialog model drawn with the Dialog Editor.

Component Based

A Component Based approach in a SW architecture lead to quality products, rapid development and an increased ability to adapt to change. In contrast to use of end-to-end conversational model that concentrates all the interaction features and capabilities within a monolithic model as black-box, a modular approach allows the potential of system engineering to be exploited for complexity management. An important aspect we focused was to maximize the re-usability of the platform components, such as algorithms or trained models as well as the dialog flows, within conversational agents for different domains, tasks and languages, maintaining a domain and task independent environment. To pursue it, the framework makes various components and algorithms available, in order to have a different level offer views. There are components dedicated to knowledge management, others that realize language understanding, dialog management and multi-modality connection. Even a single module can be seen as the set of sub-modules that realize more specific functionalities.

3.2 Robustness

In a commercial solution the robustness of a system must be guaranteed, and it can be achieved by a combination of different strategies. A significant effort was made in the system to detect and handle a wide range of errors, ranging from the language understanding, the discourse processing and the domain reasoning. But, whatever input understanding strategy is adopted, managing every possible user input is difficult, therefore the platform provides different solutions to improve the reliability managing both not understood and misunderstood inputs.

4 The Conversational Platform Overview

This section describes the overall structure of the platform. In order to pursue the main goals we defined this architecture. It is the result of collaborative effort between working on the different technologies and where the different components can be assembled to produce multiple applications.

The components are described dividing them in 3 logical views: The Design Tools for the conversational agent design, the Dialog Core Modules that implements the underline engine dialog components and, in order to provide analysis over the conversations, an Analytics Module.

4.1 Design Tools

Visual Dialog Editor

Modeling a dialog means defining the flow of the conversation and its underlying logic.

Designers define the behaviours of the agent, defining the dialog script in terms of States, Transitions and Actions. The visual editor facilitates the modeling of the flow of dialog, drawing the transitions between the dialog states and actions using graphical approach, and enable the use of the various types of resources.

Moreover, the editor, provides a graphical interface to the resource management (e.g. ontologies, models, indexes).

Simulator

A conversation simulation environment is provided within the editor for the dialog assessment. This tool enables the testing by the designer and

confirms the correctness of the dialog before deployment.

Through the simulator it is possible to verify some relevant aspects in the realization of virtual assistant. Observing the flow of conversation makes it possible to assess the smoothness and naturalness of the discourse, in relation to the management of waiting times and turn-taking. The simulator also helps to evaluate and balance the use of graphic components such as images, buttons and quick replies, usually used to make the interaction easier. It is also important explore the error management to put in way out and recovery policies.

4.2 Dialog Core Modules

Knowledge Representation

Designers use knowledge representation to build the operational structure of the dialog agent. The concepts of the domain and their relationships are represented by ontologies, taxonomies and dictionaries. If we could develop a dialog agent in a new domain with a rich ontological structure, re-using the knowledge of the existing domain becomes fundamental. The separation of domain knowledge also reduces the complexity of the linguistic components, using both general purpose resources and domain specific ones. Within the conversational platform different types, i.e, dictionaries, ontologies, inference rules, indexes and machine learning models, of knowledge representation are used in combination in order to obtain flexible dialog and dialog agent configurable.

Language Understanding

The platform makes available a proprietary multi-lingual NLP pipeline, composed by several modules that enable language comprehension, providing the language analysis at several levels ranging from morphological to pragmatic and task-dependent analysis.

This pipeline allows an hybrid approach, rule-based and machine learning, depending on needs, that can be both used and combined together, exploiting, for example, the outomes of DL classification into ontological reasoning. Among the several modules, the following Deep Learning models are leveraged:

- A sentence classification model built over pre-trained language models (Devlin et al., 2018) used for several tasks such as Dialog Act Classification (Stolcke et al., 2000) or Question Classification (Li and Roth, 2002);
- A Sequence classification models, for NER task (Chen et al., 2018);
- Intent detection and slot filling jointly classification (Castellucci et al., 2019);
- A sentiment analysis NN model, described in (Bonadiman et al., 2017)

The chosen models benefit from the advantages of the transfer learning techniques (Tan et al., 2018) in order to reduce the amount of required training data. Although this approach provides a relevant advantage in reducing the annotation effort, it might be useful to choose, according to the scenario, the right approach between "good old-fashioned techniques" and deep learning approaches.

The framework allows the use of domain dictionaries, ontologies and inferential rules that enable the extraction and inference of semantic concepts. Our framework gains the benefits of each approach by simultaneously applying the rule-based and machine learning approaches combining both techniques to infer complex knowledge structures. It is worth mentioning that with the platform is released a tool that allows, in a simple way, even to non-technical users the training of specific models to customize a system on a given domain.

Dialog Management

The dialog manager (DM) is the core component of the platform. At each turn in the conversation, the dialog management component takes the current dialog state and the user utterance as its inputs, performs different actions based on context, and outputs corresponding results as responses. DM includes two stages: dialog state tracking and dialog policy. The dialog state comprises all that is used when the system makes its decision about what is the next agent action; in this scenario, **the dialog state tracker** updates the context based on the result of the analysis of the last received input, e.g. NLU analysis over the user utterance or the query response of an external knowledge base.

In the proposed approach, the dialog tracking is implemented over hand written probabilistic rules in line with (Lison, 2015),(Wang and Lemon, 2013). The designer draws the flow of interaction as edge transitions between dialog elements

(actions and states) and adding weights for each transition. The resulting transition edges from two states cannot be not mutually exclusive, hence, at time t the tracked state of the dialog, consisting of a representation of the conversation history, the input analysis and the more "weighted" state connected to the previous one. **The dialog policy** generates dialog actions based on the current dialog information state. The system utterances depend on the current action/state, i.e. answers can be randomly selected from a defined list (in a state) or obtained as result of the selected action, as in the QA module. This approach enables a 'cold-start' when past conversation data are not available and the dialog has to be designed from scratch. The tracked state is passed on to the dialog policy module to select the best next action to perform the objective task. A set of predefined and easily customizable actions are available for the dialog design, the platform uses a plug-in mechanism, for each agent the required elements are plugged into the solution. Some of them are:

- **Question Answering**: The Question Answering action follows two steps: it performs a retrieval process over a domain dependent index. The retrieved answers are re-ranked applying NN for learning to Re-Rank process as in the CQA task (Nakov et al., 2016) in line with (Nassif et al., 2016). Moreover the QA action implements clarification strategies in case of ambiguous results.

- **External System Call**: Rest APIs are available for integration with external systems. The conversational designer can graphically draw this action fulfill few input data (e.g. endpoint, authentication and request data)

- **Slot Filling complexion**: The agent engages with the user a set of interactions to fulfill the values of a specified list of entities, e.g. the slot list of an intent or the properties of an ontological concept.

- **Route to Operator**: Under specific conditions, the dialog session can be redirect to a human operator giving to him the visibility of the information acquired up to that time. This action manages specific business cases ensuring robustness and service continuity.

Multimodality

There are different ways of communication and the choice of the users depends on various factors. The platform makes available connectors to different communication channels ranging from social network to legacy systems. The conversational agents can be delivered both through voice and written chat. Moreover, the change of channel is available (e.g. route the chat to operator or vice versa) in order to respond to specific business cases managing the change transparently to the user. Moreover, the conversations based on the different channels, can be equipped with UI components such as images, buttons and quick replies.

4.3 Analytics Module

The analysis of the conversations provides a constant view of how the conversational agent plays the "voice of the company" role. The analytics module allows to extract several insights from the dialog: interaction satisfaction, dialog errors as well as analytics for CX analysis. This one, in addition to provide market information, collects data for the agent maintenance and updates.

5 Industrial Consideration and Conclusion

In this paper we described the experience in building the Iride conversational platform for the design and deployment of task-oriented conversational agents in enterprise environment. The platform has been built taking into account needs and constraints required by an industrial scenario. We focused on a component based architecture able to maximize the re-usability of the components, enforcing a clear separation between the domain-specific aspects of the dialog and domain-independent ones across the several dialog layers (language understanding, dialog management and knowledge management). Moreover, in order to enable the work of conversational designer, the platform offers a suite of tools for conversational designers. Such architectural choices have been verified testing "on the field" the effectiveness and usability of the described solution.

Several conversational agents have been developed with this framework, in different business cases and in different domains and languages; these experiences demonstrate that the platform is efficient and easy-to-use and meets the needs of various types of use cases.

References

J.; Pawlowski N.; Nichol A. Bocklisch, T.; Faulker. 2017. Rasa: Open source language understanding and dialogue management. *arXiv,*.

Daniele Bonadiman, Giuseppe Castellucci, Andrea Favalli, Raniero Romagnoli, and Alessandro Moschitti. 2017. Neural sentiment analysis for a real-world application. *CLiC-it 2017 11-12 December 2017, Rome*, page 42.

Antoine Bordes and Jason Weston. 2016. Learning end-to-end goal-oriented dialog. *CoRR*.

Giuseppe Castellucci, Valentina Bellomaria, Andrea Favalli, and Raniero Romagnoli. 2019. Multilingual intent detection and slot filling in a joint bert-based model. *CoRR (To Appear)*, abs/1907.XXX.

Lingzhen Chen, Alessandro Moschitti, Giuseppe Castellucci, Andrea Favalli, and Raniero Romagnoli. 2018. Transfer learning for industrial applications of named entity recognition. 12.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

Paul A. Crook, Alex Marin, V. Agarwal, K. Aggarwal, T. Anastasakos, R. Bikkula, D. Boies, Asli Celikyilmaz, S. Chandramohan, Z. Feizollahi, R. Holenstein, M. Jeong, Omar Zia Khan, Young-Bum Kim, E. Krawczyk, X. Liu, D. Panic, V. Radostev, N. Ramesh, J.-P. Robichaud, A. Rochette, L. Stromberg, and Ruhi Sarikaya. 2016. Task completion platform: A self-serve multi-domain goal oriented dialogue platform. ACL - Association for Computational Linguistics, June.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zachary Lipton, Xiujun Li, Jianfeng Gao, Lihong Li, Faisal Ahmed, and li Deng. 2017. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. 11.

Pierre Lison. 2015. A hybrid approach to dialogue management based on probabilistic rules. *Comput. Speech Lang.*, 34(1):232–255, November.

Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Pei-hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. 2015. Multi-domain dialog state tracking using recurrent neural networks. *CoRR*, abs/1506.07190.

Preslav Nakov, Lluís Màrquez, Alessandro Moschitti, Walid Magdy, Hamdy Mubarak, Abed Alhakim Freihat, Jim Glass, and Bilal Randeree. 2016. SemEval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, San Diego, California, June. Association for Computational Linguistics.

Henry Nassif, Mitra Mohtarami, and James Glass. 2016. Learning semantic relatedness in community question answering using neural models. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 137–147, Berlin, Germany, August. Association for Computational Linguistics.

Andreas Stolcke, Noah Coccaro, Rebecca Bates, Paul Taylor, Carol Van Ess-Dykema, Klaus Ries, Elizabeth Shriberg, Daniel Jurafsky, Rachel Martin, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Comput. Linguist.*, 26(3):339–373, September.

Ryuichi Takanobu Xiang Li Yaoqin Zhang Zheng Zhang Jinchao Li Baolin Peng Xiujun Li Minlie Huang Sungjin Lee, Qi Zhu and Jianfeng Gao. 2019. Convlab: Multi-domain end-to-end dialog system platform.

Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. 2018. A survey on deep transfer learning. *CoRR*, abs/1808.01974.

Margaret Urban and Stephen Mailey. 2019. Conversation design: Principles, strategies, and practical application. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI EA '19, pages C26:1–C26:3, New York, NY, USA. ACM.

Zhuoran Wang and Oliver Lemon. 2013. A simple and generic belief tracking mechanism for the dialogue state tracking challenge: On the believability of observed information. In *Proceedings of SIGDIAL 2013*. Association for Computational Linguistics, 8.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, Lina M. Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*, pages 438–449, Valencia, Spain, April. Association for Computational Linguistics.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages

496–505, Vancouver, Canada, July. Association for Computational Linguistics.

Zhao Yan, Nan Duan, Peng Chen, Ming Zhou, Jianshe Zhou, and Zhoujun Li. 2017. Building task-oriented dialogue systems for online shopping.