Graph2Bots, Unsupervised Assistance for Designing Chatbots

Jean-Léon Bouraoui, Sonia Le Meitour, Romain Carbou,
Lina M. Rojas-Barahona, Vincent Lemaire
Orange, 2 Avenue Pierre Marzin,
22300 Lannion, France
{jeanleon.bouraoui,sonia.lemeitour,romain.carbou,
linamaria.rojasbarahona,vincent.lemaire@orange.com}

Abstract

We present Graph2Bots, a tool for assisting conversational agent designers. It extracts a graph representation from human-human conversations by using unsupervised learning. The generated graph contains the main stages of the dialogue and their inner transitions. The graphical user interface (GUI) then allows graph editing.

1 Introduction

In the field of artificial intelligence, dialogue systems are gaining popularity, especially as they benefit from advances in the understanding of conversational contents and contexts. Mobile and home applications such as Siri (Apple), Google Assistant (Google), Cortana (Microsoft) or Alexa (Amazon) are the most popular. To quantify this growing interest in human-machine interfaces, and dialogue systems in particular, let us cite the studies by the analyst firm Gartner[1]. They place dialogue systems among the 10 strategic technologies from 2018 and for the coming years.

One of the current trends is to propose software tools to assist the design of dialogue systems. These tools are customized according to the designer's needs, and the domain of application (e.g. trips reservation). Some solutions allow designing the dialogue architecture through a GUI. However, designers still have to perform this task manually, based on their domain knowledge and eventually the analysis of human conversations on a similar task. Most of the existing solutions do not provide a robust possibility to quickly set up an automated dialogue from human-human conversations.

In this context, we present an unsupervised assistant for the creation and adaptation of dialogue systems when the designer has a corpus of human-human interactions. This is our main contribution : this Proof of Concept can be applied for any application domain, without any prior knowledge. Thus the user will have a first version of the dialogue architecture ; that he can refine it with the GUI.

The remainder of the paper is organized as follows: first we describe the motivation by detailing the problem in Section 2; in Section 3 we present the prototype and its main innovative features. Later in Section 4 we explain the method of unsupervised learning used to build the graph. Section 5 shows the GUI. Finally, we present the conclusions and future work.

2 Rationale

Throughout this document a dialogue is an exchange of information between two speakers (a human or a machine). We are interested in task-oriented dialogues: the speakers will collaborate to achieve a common goal.

Modeling a dialogue agent specialized in a given domain is mostly done manually: either a priori, from the knowledge that the designer has on the task; or a posteriori, from the consultation of existing corpora as shown in the figure 1 below; in both cases, the process is time-consuming.

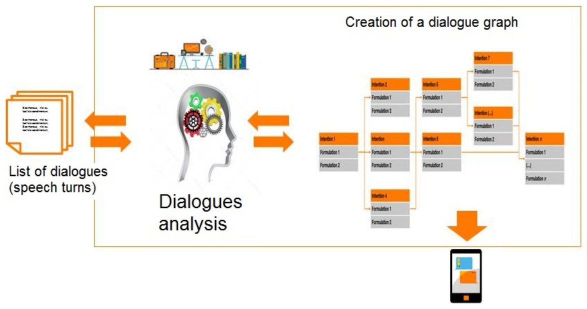


Figure 1: A posteriori Use Case workflow

We work in the "a posteriori" use case. Our goal

[1] https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2019

is to automatically obtain the dialog graph from the corpus.

We call “dialogue corpus” a set of n dialogues related to a particular domain e.g. of transcripts of train reservation dialogues. Each dialogue is composed of t speech turns; a speech turn corresponds to what is said by one of the speakers without any interruption.

We want to automatically determine from the corpus: (i) the different phases of the dialogue (including expressed intentions - hereafter referred to as “themes”. This term corresponds to either generic themes or sub-goals of the dialogue); (ii) transitions between phases. The goal is to obtain an oriented graph showing the main transitions between themes. Our assumption is that, depending on the position in the dialogue, a given turn is more likely to belong to a given phase than another; this information is therefore taken into account during the process.

The obtained graph can be exploited in multiple ways. For instance, it can be used to initialize a dialogue agent. That is to say, it can serve as a basis for modeling a dialogue agent specialized in a given domain, facilitating its design. In addition, the graph, as well as the steps taken to obtain it, will allow the designer, without prior knowledge of the application domain, to have a first understanding of the topics of the dialogues, their structure, and more generally their knowledge. Thus he can quickly get the most relevant information for the conception of the dialogue agent.

3 The Architecture

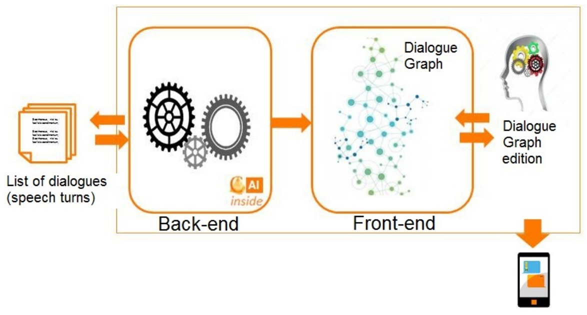


Figure 2: Graph2Bots architecture

Imagine that a designer wants to set up a conversational agent for a specific application domain. He owns a domain-related corpus of dialogues. First, he will use the unsupervised approach offered by our tool (i.e. the back-end) to identify the underlying dialogue phases, and their transitions. He will then edit the obtained graph by using our GUI. We describe these different steps in the following sections. By this way the role of the designer is changed (from the Figure 1 to the Figure 2) from a specialized dialogue analyst to a dialog graph editor.

4 Unsupervised exploitation of large dialogue corpus

In this section we will describe the three steps necessary to generate the dialogue graph from the corpus: the pre-processing to normalize and prepare the data, then the co-clustering to group speech turns in clusters corresponding to the dialogue phases and finally the graph generation.

4.1 Preprocessing

The corpus contains text documents with one speech turn per line identified by: the dialogue it belongs to, its position in the dialogue and the speaker. We begin with an anonymization process to replace the named entities (like customer name, phone number, address, etc.) by a tag. Then, the corpus is filtered in order to remove the “stopwords”. These are words that do not convey semantic information (e.g. prepositions, articles, etc.). We used the list of “stopwords” provided by the NLTK library [2]. Finally we calculate the frequency of the words in the corpus.

4.2 Co-clustering

To identify the dialogue phases, a co-clustering technique (Guigourès, 2013) is used to obtain clusters of speech turns considering the words they contain. We consider that each speech turn corresponds to a text document, being composed of words. This can be represented by a matrix. We use the technique of co-clustering to discover the best reordering and grouping of lines and columns. The method used is based on the Minimum Optimized Description Length (MODL) approach described in (Boullé, 2011) (a tutorial of this approach may be found in (Boullé et al., 2013)). Each speech turn is assigned to one and only one cluster but a cluster can group distinct dialogue turns. This method finds by itself the right number K of clusters and does not require any parameter, which is useful for non-experts. The algorithm maximizes the mutual information between

[2]http://www.nltk.org/nltk_data

the two clusterings (one partition corresponds to the group of speech turns, and the other to the group of words). A post-processing, an Agglomerative Hierarchical Clustering (Guigourès, 2013), allows to reduce the number of clusters and simplify the result.

It then remains to determine the transitions between the dialogue phases.

4.3 Graph Generation

The desired graph has to represent the architecture of the dialogue, that is to say, the succession of the phases from the beginning to the end of the dialog and the various possible paths. The figure 3 illustrates a part of such a graph.

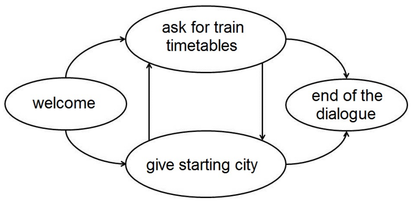


Figure 3: Graph for a train booking application

In our approach, a phase corresponds to a cluster of turns of speech. Ideally, these speech turns are homogeneous in relation to a given theme. After the discovery of the different phases (the clusters) the connections are computed according to the frequency of phase successions in the corpus: the dialogue turns are ”projected” onto the clusters and the number of transitions between the clusters along the dialogues are counted.

This process is illustrated in the figure 4; ’User’ represents a human customer speech turn, ’Agent’ represents a human agent or a bot and T_i corresponds to the cluster identifier. When a speech turn from cluster T_1 is followed by another from cluster T_3 in a dialogue, it results in a transition from T_1 to T_3 in the graph.

The resulting representation is an oriented graph, whose vertices are the clusters, and whose weighted edges are transitions between clusters. For the interested reader more details of this phase are given in (Bouraoui and Lemaire, 2017).

5 Graphical User Interface

The GUI proposes to visualize interactively in real-time the data processed in the back-end, in the form of graphs. We describe the main features below.

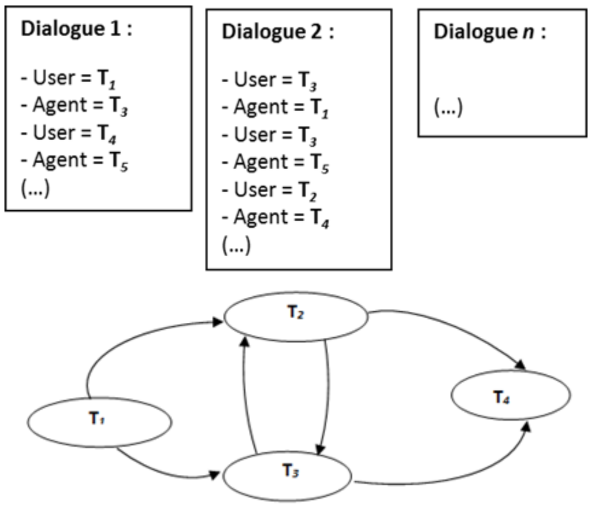


Figure 4: Graph generation

5.1 Real Time and Interactive Display

Here is an example of the list of real time and interactive display functionalities :

• Choice of the granularity of display, according to the size of clusters or the frequency of relations;

• Use of the mouse pointer to “pull” a cluster away from others, to select one or more clusters, to zoom in and out, and so on;

• Ability to rename clusters, which are automatically named with the two most representative words and the speaker; and to browse through their contents: the most representative words, and the corresponding speech turns.

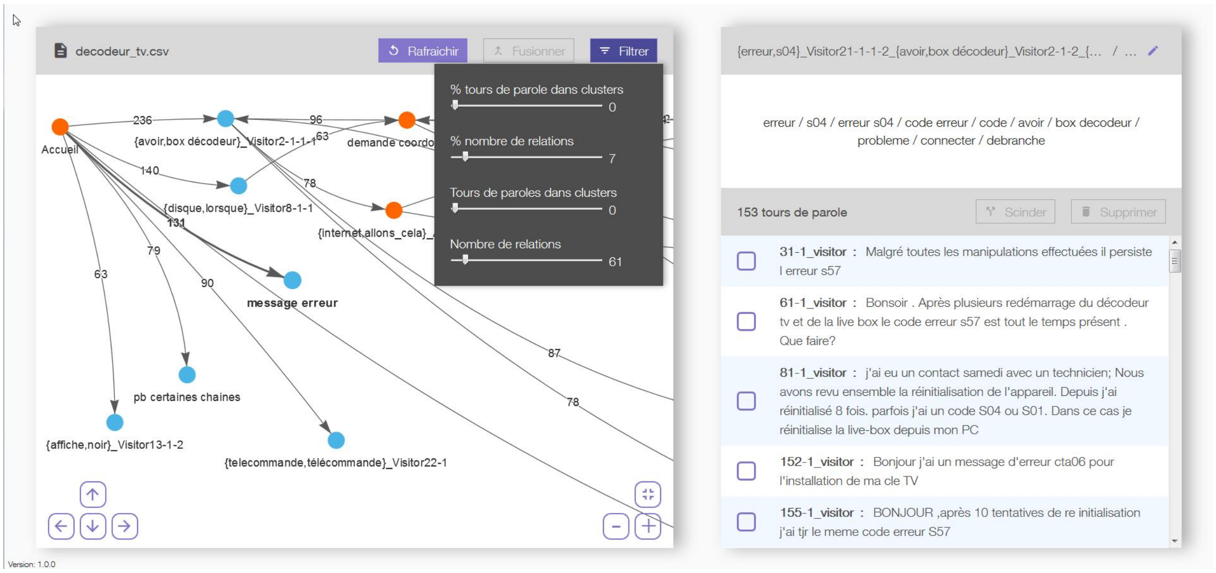
5.2 Dialogue Graph Edition

The designer of the interacting agent can adapt and refine the architecture according to his needs. It is possible to modify:

• The contents of a given cluster by deleting one or several speech turns;

• The architecture itself. Two main features are available. One is the **fusion** of two clusters (if they are thematically similar and therefore redundant). The other is the **split** of a cluster in two groups when speech turns are semantically similar, but heterogeneous with respect to the main theme expressed in the cluster. If any of these features are used, the display of the number of clusters and their connections is updated dynamically.

Figure 5 shows the current version of the prototype GUI. The graph was obtained on an extract from the Datcha corpus (Damnati et al.,



% tours de parole dans clusters 0

% nombre de relations 7

Tours de paroles dans clusters 0

Nombre de relations 61

{erreur,s04}_Visitor21-1-1-2_{avoir,box décodeur}_Visitor2-1-2_{... / ...

erreur / s04 / erreur s04 / code erreur / code / avoir / box decodeur / probleme / connecter / debranche

153 tours de parole

31-1_visitor : Malgré toutes les manipulations effectuées il persiste l erreur s57

61-1_visitor : Bonsoir . Après plusieurs redémarrage du décodeur tv et de la live box le code erreur s57 est tout le temps présent . Que faire?

81-1_visitor : j'ai eu un contact samedi avec un technicien; Nous avons revu ensemble la réinitialisation de l'appareil. Depuis j'ai réinitialisé 8 fois. parfois j'ai un code S04 ou S01. Dans ce cas je réinitialise la live-box depuis mon PC

152-1_visitor : Bonjour j'ai un message d'erreur cta06 pour l'installation de ma cle TV

155-1_visitor : BONJOUR ,après 10 tentatives de re initialisation j'ai tjr le meme code erreur S57

Figure 5: A chatbot architecture displayed by our solution

2016). The extract contains 4,000 chats between customers and call-center agents, restricted to conversations dealing with set-top box problems. This corresponds to 95,000 speech turns. The max number of clusters obtained is 497 at the most detailed level in the hierarchy ; we have empirically chosen a level with 49 clusters. The clusters (nodes of the graph) are displayed in two different colors corresponding to the two kinds of speakers. The user has filtered the display by setting the number of transitions. The cluster ”error message” is selected; on the right hand the user can consult the most relevant words for the group and the speech turns belonging to the cluster. A video is available at the url: `https://www.dailymotion.com/video/x6j6xi9`.

We used this prototype for the usecase of a bot specialized in online assistance to the users of a phone company. An other application field was also experimented: a personal home assistant (Bouraoui and Lemaire, 2017).

Once the designer has refined the architecture, he can use it to feed the dialog flow in a chatbot creation tool. The speech turns may constitute examples for the intent detection; the most representative words may correspond to entities.

6 Conclusion and future work

We presented Graph2Bots, an unsupervised assistant for dialogue designing. It is able to extract a graph representation from a corpus of conversations by using unsupervised learning, namely co-clustering, and to allow graph editing. The graph displays the main stages of the dialogues and their transitions. Our approach is independent of the domain and the language.

In the future, we would like to improve temporality management in the co-clustering (taking speech turn indexes into account); to simplify the graph when necessary (loops and other specific sub-graphs removal); to enrich its state model (e.g. detection of non-communicative actions entwined in the dialogue); and to instantiate a chatbot automatically via an interoperable format of the graph generated.

References

M. Boullé. 2011. Data grid models for preparation and modeling in supervised learning. In *Hands-On Pattern Recognition: Challenges in Machine Learning*, volume 1, pages 99–130. Guyon, I. and Cawley, G. and Dror, G. and Saffari, A.

M. Boullé, D. Gay, and A. Bondu. 2013. The many faces of data grid models. `https://bit.ly/2Ef6dDb`.

J. L. Bouraoui and V. Lemaire. 2017. Cluster-based graphs for conceiving dialog systems. In *Workshop DMNLP at European Conference on Machine Learning (ECML)*.

Géraldine Damnati, Aleksandra Guerraz, and Delphine Charlet. 2016. Web chat conversations from contact centers: a descriptive study. In *LREC*.

Romain Guigourès. 2013. *The Application of Co-clustering in Exploratory Data Analysis*. Ph.D. thesis, Université Panthéon-Sorbonne - Paris I.