10.12871/CLICIT2014157

I-ChatbIT: an Intelligent Chatbot for the Italian Language

Arianna Pipitone and **Vincenzo Cannella** and **Roberto Pirrone**
DICGIM - Dipartimento di Ingegneria Chimica, Gestionale, Informatica e Meccanica
University of Palermo
{*arianna.pipitone, vincenzo.cannella26, roberto.pirrone*}*@unipa.it*

Abstract

English. A novel chatbot architecture for the Italian language is presented that is aimed at implementing cognitive understanding of the query by locating its correspondent subgraph in the agent's KB by means of a graph matching strategy purposely devised. The FCG engine is used for producing replies starting from the semantic poles extracted from the candidate answers' subgraphs. The system implements a suitable disambiguation strategy for selecting the correct answer by analyzing the commonsense knowledge related to the adverbs in the query that is embedded in the lexical constructions of the adverbs themselves as a proper set of features. The whole system is presented, and a complete example is reported throughout the paper.

Italiano. *Si presenta una nuova architettura di chatbot per l'italiano che implementa una forma di comprensione di natura cognitiva della query individuando il corrispondente stottografo nella base di conoscenza dell'agente con tecniche di graph matching definite appositamente. Il sistema FCG usato per la produzione a partire dai poli semantici estratti da tutti i sottografi coandidati alla risposta. Il sistrema effettua una disambigazione a partire dalla conoscenza di senso comun sugli avverbi che codificata come un apposito insieme di caratteristiche all'interno delle relative costruzioni lessicali. Si presenta l'intera architettura e viene svolto un intero esempio di funzionamento.*

1 Introduction

In recent years the Question-Answering systems (QAs) have been improved by the integration with Natural Language Processing (NLP) techniques, which make them able to interact with humans in a *dynamic way*: the production of answers is more sophisticated than the classical chatterbots, where some sentence templates are pre-loaded and linked to the specific questions.

In this paper we propose a new methodology that integrates the chatterbot technology with the Cognitive Linguistics (CL) (Langacker, 1987) principles, with the aim of developing a QA system that is able to harvest a linguistic knowledge from its inner KB, and use it for composing answers dynamically. Grammatical templates and structures tailored to the Italian language that are constructions of the Construction Grammar (CxG) (Goldberg, 1995) and a linguistic Italian source of verbs have been developed purposely, and used for the NL production. The result of the methodology implementation is I-ChatbIT, an Italian chatbot that is intelligent not only for the dynamic nature of the answers, but in the sense of *cognitive understanding* and *production* of NL sentences. Cognitive understanding of the NL query is achieved by placing it in the system's KB, which represents the conceptualization of the world as it has been perceived by the agent. The outcome of such a process is the generation of what we call the *meaning activation* subgraph in the KB. Browsing this subgraph, the system elaborates and detects the content of the answer, that is next grammatically composed through the linguistic base. The FCG engine is then used as the key component for producing the answer. Summarily, the work reports the modeling of the two tasks outlined above.

The paper is arranged as follow: in the next section the most popular chatbots are shown, devoting particular attention to the Italian ones. Section 3 describes the implemented methodology explaining

in detail a practical example. Finally, conclusions and future works are discussed in section 4.

2 The Italian Chatbots

There are no many Italian chatbots in literature. We refer to the most recent and widespread ones. QUASAR (Soriano et al., 2005) uses simple pattern matching rules for the answer extraction and it splits the Italian among the provided language. Eloisa and Romana (available at `http://www.eloisa.it/`) are the most recent Italian chatbots, the former speaking on generic arguments (as sports, politics and so on), the latter specifically for history and folklore of Rome city. Both have a basic form of intelligence because they learn new contents during the conversation, even if no learning algorithms have been made mentioned by the authors. Among cognitive QA systems, the best known cognitive technology is Watson (Ferrucci, 2012) from IBM, which was specifically developed for answering questions at the Jeopardy quiz show. The core is the UIMA (Ferrucci and Lally, 2004) framework on which the whole system is implemented. However, this system does not provide Italian language by now. Finally there are many virtual assistants developed for the Italian, but neither of them uses cognitive approaches. The base technology is using controlled NL and pattern matching; however these systems act on specific and restricted tasks as the services provided by telephonic companies, booking flights and so on.

3 Building I-ChatbIT

Figure 1 shows the I-ChatbIT architecture; the main modules are the *Meaning Activator* and the *Answer Composer*, which are connected to the Knowledge Base (KB) and to the linguistic base (composed by our *Italian Verbs Source* (IVS) and MultiWordnet (Pianta et al., 2002) (MWn)). The whole system is managed by the *Controller*, which acts as the user interface too. The KB contains the inner domain representation owned by the system. We used OWL ontologies for such a component. The KB can be replaced so the system is domain independent. MWn and the IVS form the linguistic base of the system. We are currently expanding the IVS to cover the other parts of speech, and it will become the only Italian dictionary of the system. In this phase MWn is used for retrieving parts of speech other than verbs. The Meaning Activator (MA) implements the meaning-activation process through a graph similarity search between the *query-graph* (a graph representation of the query) and the *conceptual-graph* (a graph representation of the KB): the result of search is a set of subgraphs that correspond to placing the query in the KB. Browsing such subgraphs some facts are detected, and they are the candidates for composing the answer. All the candidate facts are inputted to the Answer Composer (AC) that generates grammatical constructions, and filters them according to the linguistic information that is needed for context disambiguation. Filtered constructions are finally plunged and produced by the Fluid Construction Grammar (FCG) engine (Steels and de Beule, 2006). If the answer is not exhaustive for the user, the Controller involves the Semantic Annotator described in (Pipitone and Pirrone, 2012) that retrieves external contents; such contents are re-elaborated as facts by the AC and the process is iterated. Each component is next carefully described.

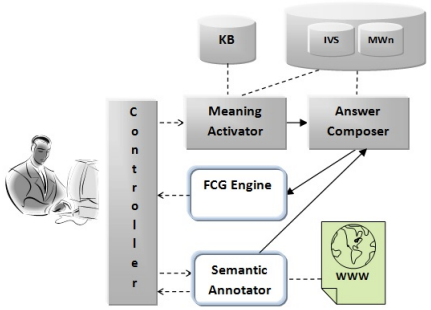

Figure 1: I-ChatbIT Architecture.

3.1 The Meaning Activator

The strategy adopted for implementing cognitive understanding in the MA relies on applying the Graph Edit Distance (GED) method (Zeng et al., 2009) between the query-graph Q and the conceptual-graph C, so that their GED is no larger than a distance threshold τ. In particular, the *query-graph* is the triple $Q = (N_q, E_q, L_q)$ where the nodes set N_q contains the macro-syntactic roles of the NL query, parsed by the Freeling parser (Padr and Stanilovsky, 2012). These nodes are sequentially connected reflecting their position in the query. The labels set L_q are labels nodes, and correspond to the tokens of the query outputted by the parser. For example, the query-graph for the question *"Dov'è nato il famoso Giotto?"* is shown in figure 2. The *conceptual-graph* is the 4-tuple $C = (N_c, E_c, L_c, \sigma)$ where the nodes set

$N_c = C_n \cup R_n$ is the union set of the set C_n containing the concepts in the KB, and the set R_n that contains relations. An edge in E_c connects only a concept-node to a relation-node if the concept is either the domain or the range for the relation itself. The edge is labeled with a progressive number for tracing the entities involved in the relation. σ is a label function $\sigma : N_c \rightarrow L_c$ that associates to each node $n_c \in N_c$ a list of strings $l_c \in L_c$ that are obtained by querying the linguistic sources on-the-fly, as it is next described. An example of conceptual graph is shown in figure 2. For GED computation, we refer to the two following parameters:

- a *similarity measure* between nodes, that is the Jaro–Winkler distance (Winkler, 1990) between the labels associated to them as described in 3.2.1;
- a *graph edit distance* ged between subgraphs, that represents the number of primitive graph edit operations to make them isomorphic. There are six primitive edit operations (Zeng et al., 2009): node insertion and deletion, node label substitution, edge insertion and deletion, and edge label substitution. For our purposes, the above constraints for connecting nodes make label substitution useless, so we refer only to the remaining four operations.

Given Q, C and a distance threshold τ, the problem is *to find a set of subgraphs* $I = \{I_i\}$ *with* $I_i = (N_i, E_i, L_i) \subset C$ *so that* I_i *and* Q *are isomorphic for a number of primitive edit operations* $ged <= \tau$. $M_{act} \equiv \bigcup_i I_i$ corresponds to the meaning activation area of the query. Considering that Q is a linear sequence of nodes and edges, an isomorphism in C will be a sequence too. Threshold τ is necessary for avoiding that the query is sparse in the KB. The τ value has been fixed arbitrarily to 10. The strategy computes the isomorphisms applying the *k*-AT algorithm (Wang et al., 2012), which defines a *q-gram* as a tree consisting of a vertex *v* and the paths starting at *v* with length no longer than q. In our case, the vertexes are nodes from M_{act}, the *k*-AT has been customized for using only four edit operations as explained before.

Once the isomorphisms are detected, MA probes the KB for retrieving connected facts, for example it adds nodes that are either the domain or the range of some relation node if they are not yet included in the subgraphs, or retrieves adjacent triples to the nodes involved in the isomorphisms. In our example there are two isomorphisms $I_1 = \{Giotto - datanascita\}$ and $I_2 = \{Giotto - luogonascita\}$, and ged is equal to 5 for both of them. They are candidates as possible answers. The AC will provide the correct disambiguation between them. If no disambiguation is possible, the answer is composed by the conjunction of them and results in an expanded sentence.

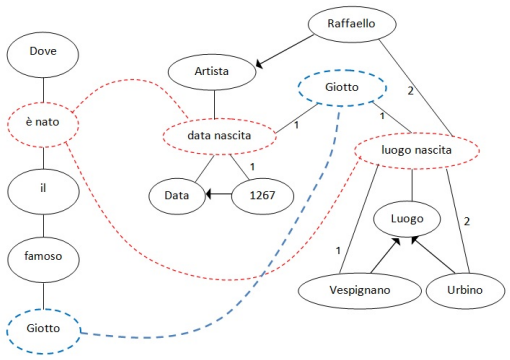

Figure 2: An example of Q and C, joint through the Jaro–Winkler distance, and the computation of the related isomorphisms.

3.2 The Answer Composer

Once the M_{act} subgraphs set is detected by MA, the correct NL sentence has to be composed. For this purpose, we use the FCG engine where system puts the linguistic information about the domain according to the FCG formalism, that is the CxG. Lexical and grammatical descriptions of the domain terms must be represented as *Constructions*, that are form-meaning couples. Form pole contains the syntactic features of terms, while the semantic one contains meaning. Lexical constructions are related to a single word, and conjunctions of them generate grammatical constructions. FCG uses the same set of constructions for both parsing and production, by iterating merging and unification processes on the involved poles (syntatic poles in parsing, semantic ones in production). In this phase, the FCG engine of the system contains lexical constructions for the Italian adverbs and articles, that were manually created. For adverbs, the features embed some commonsense knowledge about them: for example, the lexical construction for the adverb "dove" stores some features like "luogo", "posto", "destra", "sinistra" and so on. Query parsing allows obtaining the semantic poles related to the query, so the probing strategy performed by MA is necessary for retriev-

ing others facts from KB and composing the M_{act} related to the answer. For this reason we use FCG only in production: once the M_{act} is fed to the FCG engine, it unifies the related semantic poles to the correspondent lexical and grammatical constructions, and produces the answer. The opposite way is not possible because we would need to map all possible subgraphs in the KB as facts in the FCG, with a consequent combinatorial explosion.

3.2.1 Filling FCG through linguistic sources

Lexical and grammatical constructions about the domain form the linguistic base of the system and are generated by querying the KB and the linguistic sources (IVS and MWn). In particular, the KB concepts and relations labels are retrieved, and tokenized according to the algorithm described in (Pipitone et al., 2013), that models the cognitive task of reading. As a consequence I-ChatbIT learns the KB content. The system queries either IVS or MWn according to the stem of the label. In case of a verb stem verb, IVS provides all the related information, which includes the related argument structures (Goldberg et al., 2004) and synonyms, as it shown in next section. In the all other cases, the system refers to MWn, and it retrieves synonyms, hypernyms, hyponyms for each label along with the verbal information of the verb included in the definition. The lexical and grammatical constructions of all these terms are generated as described by some of the authors in (Pipitone and Pirrone, 2012) where terms that refer to the same nodes are considered synonymic constructions.

3.2.2 Answer production and disambiguation

FCG contains constructions tailored on the KB. When KB subgraphs are put to the AC, it builds the correspondent meaning poles, and the related constructions fire; all of them are candidates for being used in production. At this point AC applies the *disambiguation process*. Adverb tokens in the query are parsed by the FCG engine, and their corresponding lexical constructions fire. Disambiguation chooses the subgraph that has a link to the adverbial features stored in the construction. In our example, the candidates facts from MA are the subgraphs {Giotto - data nascita - 1267 - is_a Data} and {Giotto - luogo nascita - Luogo}; the lexical construction of the adverb "dove" allows selecting the second subgraph. If the query were "Quando è nato il famoso Giotto?" the first subgraph would be selected using the commonsense knowledge stored in the related lexical construction ("ora", "tempo","data", and so on).

3.3 The Italian Verbs Source

The IVS contains approximately five thousands verbs, classified into distinct groups. They represent the most common verbs usually used in a common conversation in Italian. All inflexions of each verb have been stored and annotated. The storage adopts a compressed description of verbs. Each inflextion is derived by combining the root of the verb with the corresponding suffix. Suitable rules choose the proper inflexion on the basis of tense, person, number and gender are used to choose the proper inflexion. Verbs have been grouped on the basis of their suffix class, according to the base rules of Italian grammar. A finer grouping has been made according to the origin of the verb. This choice allows a more compact description of the verbs' conjiugations. Irregular verbs have been treated using ad hoc rules for producing their inflections.
Each tense is described by a construction, containing tense, person, number, and gender as its features. Each compound form is described by a single construction, and not as combination of other constructions. All possible active, passive, and reflexive forms have been stored. All verbs have been classified as transitive, intransitive and semitransitive. This information is stored into each verb construction too. Finally, each verb is joined to a list of possible synonymies and analogies.

4 Conclusions and future works

A novel chatbot architecture for the Italian language has been presented that is aimed at implementing cognitive understanding of the query by locating its correspondent subgraph in the agent's KB by means of a GED strategy based on the k-AT algorithm, and the Jaro–Winkler distance. The FCG engine is used for producing replies starting from the semantic poles extracted from the candidate answers' subgraphs. The system implements a suitable disambiguation strategy for selecting the correct answer by analyzing the commonsense knowledge related to the adverbs in the query that is embedded in the lexical constructions of the adverbs themselves as a proper set of features. Future works are aimed at completing the IVS, and using explicit commonsense knowledge inside the KB for fine disambiguation. Finally, the graph matching strategy will be further tuned.

References

David Ferrucci and Adam Lally. 2004. Uima: An architectural approach to unstructured information processing in the corporate research environment. *Nat. Lang. Eng.*, 10(3-4):327–348, September.

David A. Ferrucci. 2012. Introduction to "this is watson". *IBM Journal of Research and Development*, 56(3):1.

Adele E. Goldberg, Devin M. Casenhiser, and Nitya Sethuraman. 2004. Learning Argument Structure Generalizations. *Cognitive Linguistics*, 15(3):289–316.

A. E. Goldberg. 1995. *Constructions: A construction grammar approach to argument structure*. University of Chicago Press, Chicago.

Ronald W. Langacker. 1987. *Foundations of cognitive grammar: Theoretical Prerequisites*. Stanford University Press, Stanford, CA. Vol 1, 1987(Hardcover), 1999(Paperback).

Llus Padr and Evgeny Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May. ELRA.

Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. 2002. Multiwordnet: developing an aligned multilingual database. In *Proceedings of the First International Conference on Global WordNet*, January.

Arianna Pipitone and Roberto Pirrone. 2012. Cognitive linguistics as the underlying framework for semantic annotation. In *ICSC*, pages 52–59. IEEE Computer Society.

Arianna Pipitone, Maria Carmela Campisi, and Roberto Pirrone. 2013. An a* based semantic tokenizer for increasing the performance of semantic applications. In *ICSC*, pages 393–394.

Jos Manuel Gmez Soriano, Davide Buscaldi, Empar Bisbal Asensi, Paolo Rosso, and Emilio Sanchis Arnal. 2005. Quasar: The question answering system of the universidad politcnica de valencia. In Carol Peters, Fredric C. Gey, Julio Gonzalo, Henning Mller, Gareth J. F. Jones, Michael Kluck, Bernardo Magnini, and Maarten de Rijke, editors, *CLEF*, volume 4022 of *Lecture Notes in Computer Science*, pages 439–448. Springer.

Luc Steels and Joachim de Beule. 2006. A (very) brief introduction to fluid construction grammar. In *Proceedings of the Third Workshop on Scalable Natural Language Understanding*, ScaNaLU '06, pages 73–80, Stroudsburg, PA, USA. Association for Computational Linguistics.

Guoren Wang, Bin Wang, Xiaochun Yang, and Ge Yu. 2012. Efficiently indexing large sparse graphs for similarity search. *IEEE Trans. on Knowl. and Data Eng.*, 24(3):440–451, March.

William E. Winkler. 1990. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. In *Proceedings of the Section on Survey Research*, pages 354–359.

Zhiping Zeng, Anthony K. H. Tung, Jianyong Wang, Jianhua Feng, and Lizhu Zhou. 2009. Comparing stars: On approximating graph edit distance. *PVLDB*, 2(1):25–36.