Context-aware Spoken Language Understanding for Human Robot Interaction

Andrea Vanzo(†), **Danilo Croce**(‡), **Roberto Basili**(‡) **and Daniele Nardi**(†)
(†)Sapienza University of Rome
Department of Computer, Control and Management Engineering "Antonio Ruberti"
Via Ariosto 25, 00185 Roma, Italy
(‡)University of Roma, Tor Vergata
Department of Enterprise Engineering, Via del Politecnico 1, 00133 Roma, Italy
{vanzo,nardi}@dis.uniroma1.it, {croce,basili}@info.uniroma2.it

Abstract

English. Robots operate in specific environments and the correct interpretation of linguistic interactions depends on physical, cognitive and language-dependent aspects triggered by the environment. In this work, we present LU4R - *adaptive spoken Language Understanding 4 Robots*, a Spoken Language Understanding chain for the semantic interpretation of robotic commands, that is sensitive to the operational environment. The system has been designed according to a Client/Server architecture in order to be easily integrated with the vast plethora of robotic platforms.

Italiano. *L'interpretazione di comandi espressi nei confronti di piattaforme robotiche è un processo strettamente legato al contesto operativo in cui avviene l'interazione. In questo lavoro, presentiamo LU4R - adaptive spoken Language Understanding 4 Robots, un sistema per l'elaborazione automatica di comandi vocali, dipendente dall'ambiente in cui il comando viene espresso. Il sistema proposto, implementato come una cascata di passi di elaborazione semantica, è stato progettato seguendo un'architettura Client/Server, per ridurre i requisiti di integrazione con le piattaforme robotiche esistenti.*

1 Introduction

End-to-end communication in natural language between humans and robots is challenging for the different cognitive abilities involved during the interaction. As an example, for a robot to react to a command like "*take the book on the table*", a number of implicit assumptions should be met. First, at least two entities, a `book` and a `table`, must exist in the environment and the speaker must be aware of such entities. Accordingly, the robot must have access to an inner representation of the objects, e.g., an explicit map of the environment. Second, mappings from lexical references to real world entities must be developed or made available. In this respect, the *Grounding* process (Harnad, 1990) links symbols (e.g., words) to the corresponding perceptual information. Hence, robot interactions need to be *grounded*, as meaning depends on the state of the physical world and the interpretation crucially interacts with perception, as pointed out by psycho-linguistic theories (Tanenhaus et al., 1995). To this end, the integration of perceptual information derived from the robot's sensors with an ontologically motivated description of the world has been adopted as an augmented representation of the environment, in the so-called *semantic maps* (Nüchter and Hertzberg, 2008). In this maps, the existence of real world objects can be associated to *lexical* information, in the form of entity names given by a knowledge engineer or spoken by a user for a pointed object, as in Human-Augmented Mapping (Diosi et al., 2005). While SLU for Interactive Robotics have been mostly carried out over the only evidences specific to the linguistic level (see, for example, (Chen and Mooney, 2011; Matuszek et al., 2012)), we argue that such process should be context-aware, in the sense that both the user and the robot access and make references to a shared environment. For example, in the above command, "*taking*" is the intended action whenever a book is actually on the table, so that "*the book on the table*" refers to a single argument. On the contrary, the command may refer to a "*bringing*" action, when no book is on the table and *the book* and *on the table* correspond to different semantic roles.

In this paper, we present LU4R an adaptive spoken language understanding chain for the auto-

matic interpretation of robotic spoken commands that is coherent with the above assumptions. The resulting chain is based on the approach proposed in (Bastianelli et al., 2016) that allows to produce interpretations that are consistent with (*i*) the world (with all the entities composing it), (*ii*) the Robotic Platform (with all its inner representations and capabilities), and (*iii*) the linguistic information derived from the user's utterance. LU4R is fully implemented in Java and is released according to a Client/Server architecture, in order to decouple the chain from the specific robotic platform that will use it. It receives as input one or more transcriptions of a spoken command and produces one or more linguistic predicates reflecting the actions intended by the user. Predicates, as well as their arguments, are consistent with a linguistically-motivated representation and coherent with the environment perceived by the robot.

The rest of the paper is structured as follows. Section 2 provides an architectural description of the entire system, as well as an overall introduction about its integration with a generic robot. In Section 3 we demonstrate the applicability of the chain in the interpretation of commands in English and Italian.

2 The overall architecture

The architecture of the proposed system is decoupled into two main macro-components, as shown in Figure 1: the *Robotic Platform* and *LU4R*.

The Client-Server communication schema between the Robotic Platform (the Client) and LU4R (the Server) allows to maintain the former independent from the latter. It is obvious that the interpretation process must be achieved even when no information about the domain/environment is available, i.e., a scenario involving a *blind* but speaking robot. This is the case when the command "*take the book on the table*" is paired with any additional information and the ambiguity with respect to the evoked predicate, i.e., *Taking* vs. *Bringing*, cannot be resolved. At the same time, the platform allows to specialize the semantic interpretation process to individual situations when contextual information is available. In this case, whenever the sentence "*take the book on the table*" is provided along with information about the presence and position of a `book` on a `table`, the above disambiguation can be solved.

In the following, each macro-component of the architecture in Figure 1 is discussed and analyzed.

2.1 The Robotic Platform

The overall architecture contemplates a generic Robotic Platform, whose task, domain and physical setting are not necessarily specified. In order to make LU4R independent from the above specific aspects, we will assume that the platform requires at least the following modules: (*i*) an Automatic Speech Recognition (ASR) system; (*ii*) a SLU Orchestrator; (*iii*) a Grounding and Command Execution Engine; (*iv*) a Physical Robot. Additionally, the optional component *Support Knowledge Base* is expected to provide the contextual information discussed above. While the discussion about the Physical Robot is out of the scope of this work, all the other components are hereafter shortly summarized.

ASR system. An ASR engine allows to transcribe a spoken utterance into one or more possible transcriptions. In the actual release, the ASR is here performed through an *ad-hoc* Android application that can be deployed on both Android smartphones and tablets. It relies on the official *Google ASR API*[1] that offers valuable performances for an off-the-shelf solution. The acoustic model is here based on deep learning techniques, i.e. Recurrent Neural Networks (Hinton et al., 2012). The Google ASR is publicly available and is rather robust toward some of the complexities of spoken language, such as disfluencies and repetitions. This allowed us to focus on other challenges of spoken language, such as linguistic variations (e.g. synonymy, phonetically similar words, interrogative vs. imperative sentences, ...). Advantages of our lexicalized grounding approach (Bastianelli et al., 2015) is the robustness against variability and sense ambiguity.

SLU Orchestrator. The SLU Orchestrator implements a TCP Server for the Android App, here coded as a ROS node (Quigley et al., 2009) waiting for Client requests. Once a new request arrives (a list of transcriptions for a given spoken sentence), this module is in charge of extracting the perceived entities from a structured representation of the environment (here, a sub-component of the Support Knowledge Base) and sending the list of hypothesized transcriptions to LU4R along with the list of the perceived entities. The communication protocol requires the serialization of such

[1] `https://cloud.google.com/speech/`

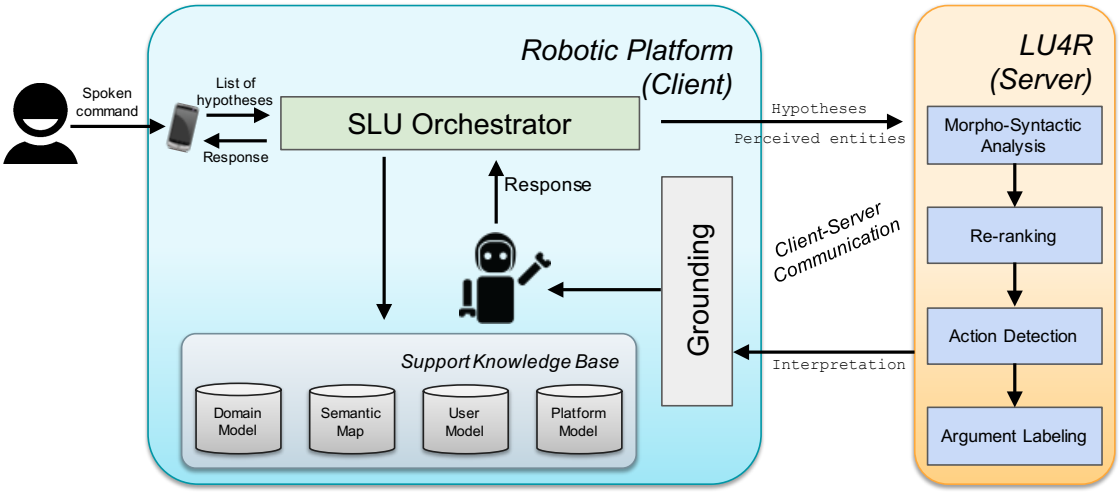

Figure 1: Overall architecture of the system

information in two different JSON objects. Even though this module is actually a TCP Server for the Android App, it represents also the Client interface toward LU4R.

Grounding and Command Execution. Even though the grounding process is placed at the end of the loop, it is discussed here as it represents part of the Robotic Platform. In fact, this process has been completely decoupled from the SLU, as it may involve perception capabilities and information unavailable to LU4R or, in general, out of the linguistic dimension. Nevertheless, this situation can be partially compensated by defining mechanisms to exchange some of the grounding information with the linguistic reasoning component. However, grounding is always carried out on board of the robot, as it represents the most general situation. The grounding carried out by the robot is triggered by a logical form expressing one or more actions through (linguistic) predicates. The output of the SLU process embodies the produced logical form: this latter exposes the recognized actions that are then linked to specific robotic operations (primitive actions or plans). Correspondingly, the predicate arguments (e.g., objects and location involved in the targeted action) are detected and linked to the objects/entities of the current environment. A fully grounded command is obtained through the complete instantiation of the robot action (or plan) and its final execution.

2.2 The LU4R system

The language understanding process produces an interpretation of the user's utterance in terms of linguistic predicates as defined in Frame Semantics (Fillmore, 1985). Specifically, we consider the formalization adopted in the FrameNet (Baker et al., 1998) database. According to such theory, actions expressed in user utterances can be modeled as *semantic frames*. These are micro-theories about real world situations, e.g., the action of *taking*. Each frame specifies also the set of participating entities, called *frame elements*, e.g., the THEME representing the object that is taken during the *Taking* action. For example, for the sentence "*take the book on the table*", a potential corresponding parsed version is:

$$[\textbf{\textit{take}}]_{Taking}\ [\textit{the book on the table}]_{\text{THEME}} \quad (1)$$

In a robotic perspective, semantic frames provide a cognitively sound bridge between the actions expressed in the language and the implementation of such actions in the robot world.

As shown in Figure 1, LU4R is composed of four main steps.

Morpho-syntactic analysis is performed over each available utterance transcription, by applying Part-of-Speech tagging and syntactic parsing, providing morphological and syntactic information, essential for further processing.

Whenever more than one hypothesized transcription is available, a **Re-ranking** module can be activated to evaluate a new sorting of the hypotheses, in order to get the best one out of the original ranking. It allows to reuse existent ASR solutions, making the final rank sensitive to the specific robotic domain.

The selected transcription is the input of the **Action Detection** (AD) component. Here all the

frames (i.e. intended actions) evoked in a sentence are detected, according to their triggering lexical units. For instance, given the above example, the AD would produce the following interpretation: $[\textbf{\textit{take}}]_{Taking}$ *the book on the table*.

The final step is the **Argument Labeling** (AL). Here a set of frame elements is retrieved for each frame, detected during the AD step. Such process is, in turn, realized in two sub-steps. First, the *Argument Identification* (AI) aims at finding the spans of all the possible frame elements. Then, the *Argument Classification* (AC) assigns the suitable frame element label to each span identified during the AI, producing the final tagging shown in (1).

An off-the-shelf tool is used for the morpho-syntactic analysis, namely the Stanford CoreNLP suite (Manning et al., 2014). Re-ranking is performed using a learn-to-rank approach, where a Support Vector Machine exploiting a combination of linguistic kernels is applied, as discussed in (Basili et al., 2013). The AD, AI and AC steps are modeled as a sequential labeling task, as in (Bastianelli et al., 2016). The Markovian formulation of a structured SVM proposed in (Altun et al., 2003) is applied to implement the sequential labeler, known as *SVM*^{hmm} . In general, this learning algorithm combines a local discriminative model, which estimates the individual observation probabilities of a sequence, with a global generative approach to retrieve the most likely sequence, i.e. tags that better explain the whole sequence. In other words, given an input sequence $\mathbf{x} = (x_1 \dots x_l) \in \mathcal{X}$ of feature vectors $x_1 \dots x_l$, *SVM*^{hmm} learns a model isomorphic to a k-order Hidden Markov Model, to associate $\mathbf{x}$ with a set of labels $\mathbf{y} = (y_1 \dots y_l) \in \mathcal{Y}$.

A sentence s is here intended as a sequence of words w_i, each modeled through a feature vector x_i and associated to a dedicated label y_i, specifically designed for each interpretation process. During training, the SVM algorithm is devoted to associating words to step-specific labels: linear kernel functions are applied to different types of features, ranging from linguistic to perception-based features, and linear combinations of kernels are used to integrate independent properties. At classification time, given a sentence $s = (w_1 \dots w_{|s|})$, the *SVM*^{hmm} efficiently predicts the tag sequence $\mathbf{y} = (y_1 \dots y_{|s|})$ using a Viterbi-like decoding algorithm.

Both the re-ranking and the language understanding phases can work in two different settings.

In the so-called *basic* scenario, only linguistic information is used during the interpretation task. Perceptual information from the environment is thus neglected and evidences from the user's utterances or linguistic resources are considered.

Conversely, when perceptual information is made available to the chain, a context-aware interpretation is triggered. Such a perceptual knowledge is mainly exploited through a *linguistic grounding* mechanism (Bastianelli et al., 2015). This lexically-driven grounding is estimated through distances between filler (i.e. argument heads) and entity names. Such a semantic distance integrates metrics over word vectors descriptions and phonetic similarity. Word semantic vectors are here acquired through corpus analysis, as in Distributional Lexical Semantic paradigms (Turney and Pantel, 2010). They allow to map referential elements, such as lexical fillers, e.g. *desk*, to entities, e.g. a `table`, by thus modeling synonymy or co-hyponymy. Conversely, phonetic similarities ar smoothing factors against possible ASR transcription errors, e.g. *pitcher* and *picture*. Once links between fillers and entities have been activated, the sequential labeler is made sensitive to additional features, that inject perceptual information both in the learning and the tagging process, e.g. the presence/absence of referred objects in the environment. As a side effect, the above mechanism provides the robot with the set of linguistically-motivated groundings, that can be potentially used for any further grounding process.

Overall, the service provided by LU4R is performed as a black-box component, so that the complexity of each inner sub-task is hidden to the user. The service is realized through a server accepting connections on a predefined port. LU4R is entirely coded in Java and released as a single Jar file[2], along with the required folders containing linguistic models, configurations files and other resources. Hence, it can be run through command line, so that it is easier to integrate it within any architecture.

The LU4R system takes three input parameters: *type* of the understanding process, *output format* and *listening port*. The first parameter defines the type of the interpretation process to be initialized: the `basic` value activates the setting where only linguistic information is adopted, while `simple`

[2] `http://sag.art.uniroma2.it/sluchain.html`

refers to the interpretation where perceptual information is considered. The second parameter specifies the desired output format, e.g., *eXtended Dependency Graph* (Basili and Zanzotto, 2002) or the *Abstract Meaning Representation*, proposed in (Banarescu et al., 2013).

3 Evaluating LU4R

In order to provide evidences about the effectiveness of the proposed solution, we report here a preliminary evaluation of the interpretation process w.r.t. robotic commands in two languages, i.e. English and Italian.

	AD	**AI**	**AC**
English	95.91%	94.29%	95.31%
Italian	82.29%	79.46%	84.49%

Table 1: Experimental evaluation of the semantic interpretation process, in terms of F1

Table 1 shows results obtained over the Human Robot Interaction Corpus (HuRIC) (Bastianelli et al., 2014), a dataset of semantically annotated commands typical of Service Robotics. HuRIC contains 527 sentences in English. Moreover, results w.r.t. to a subset of 188 commands from HuRIC translated in Italian are reported. The results, expressed in terms of F1 measure, focus on the semantic interpretation process, in particular Action Detection (AD), Argument Identification (AI) and Argument Classification (AC) steps. In fact, F1 scores measures the quality of a specific module. While in the AD step the F1 refers to the ability to extract the correct frame(s) evoked by a sentence, in the AI step it evaluates to the correctness of the predicted argument spans. Finally, in the AC step the F1 measures the accuracy of the classification of individual arguments.

We tested each sub-module in isolation, feeding each step with gold information provided by the previous step in the chain. Moreover, the evaluation has been carried out considering the correct transcriptions, i.e., not contemplating the error introduced by the Automatic Speech Recognition system. The results over the Italian dataset refer to the `basic` setting of LU4R, i.e. leveraging just linguistic information. Conversely, the experiments over the English dataset have been carried out with the LU4R setting exploiting also perceptual knowledge. Results against the commands in English are encouraging for the application of LU4R in realistic applications, with a F1 higher than 94% in the recognition of semantic predicates used to express intended actions as well as the involved entities. We speculate that the gap w.r.t. results against the Italian dataset is mainly due to the lack of perceptual knowledge as well as the reduced size of the dataset. A more detailed description of the above evaluation over the English and Italian dataset is available in (Bastianelli et al., 2016) and (Vanzo et al., 2016), respectively.

4 Conclusions

In this paper, we presented LU4R, an adaptive SLU processing chain focused on the interpretation of commands in the Mobile Service Robotics domain. The proposed solution relies on Frame Semantics and Distributional Lexical Semantics to support example-driven machine learning algorithms that map individual sentence transcriptions to meaningful commands. Statistical learning, i.e. SVM^{hmm} , is applied by transforming the interpretation process into a cascade of sentence annotation tasks. The use of Frame semantics enables the reuse of large repositories of examples (i.e. (Baker et al., 1998)), supporting the recognition of up to more than 1,000 different semantic frames, currently defined in the FrameNet database. The robustness of the sentence interpretation, as measured in this paper, is rather good, where language variability is tackled by relying on distributional lexical models that generalize semantics for large vocabularies, well beyond the training set dictionaries. Moreover, even though LU4R is completely decoupled from the Robotic Platform, the final interpretation is made dependent on the robot's environment, by designing perceptual knowledge through feature modeling. Perceptual knowledge is derived from the robot's semantic map and translated into feature values. The corresponding space allow to synthesize information about existence and position of named entities, useful to disambiguate predicates and role assignment.

The results gathered during repeated empirical investigation campaigns confirm the effectiveness of the proposed tool and, in particular, the benefits provided by the injection of perceptual Knowledge into the understanding (i.e. labeling) process. LU4R is thus an effective example of grounded language learning framework, whereas

predicates and semantic roles are acquired through an integration between linguistic (e.g. lexical and grammatical) properties and perceptual information (e.g. distances between entities in a map): this makes LU4R an interesting topic for future research, such as the extension of command interpretation process to complex interaction patterns, such as in interactive question answering or dialogue.

References

Yasemin Altun, I. Tsochantaridis, and T. Hofmann. 2003. Hidden Markov support vector machines. In *Proc. of ICML*.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of ACL and COLING*, pages 86–90.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria, August. Association for Computational Linguistics.

Roberto Basili and Fabio Massimo Zanzotto. 2002. Parsing engineering and empirical robustness. *Nat. Lang. Eng.*, 8(3):97–120, June.

Roberto Basili, Emanuele Bastianelli, Giuseppe Castellucci, Daniele Nardi, and Vittorio Perera. 2013. Kernel-based discriminative re-ranking for spoken command understanding in hri. In *AI*IA*, volume 8249, pages 169–180. Springer.

Emanuele Bastianelli, Giuseppe Castellucci, Danilo Croce, Roberto Basili, and Daniele Nardi. 2014. Huric: a human robot interaction corpus. In *Proceedings of LREC 2014*, Reykjavik, Iceland, may.

Emanuele Bastianelli, Danilo Croce, Roberto Basili, and Daniele Nardi. 2015. Using semantic models for robust natural language human robot interaction. In *AI* IA 2015, Advances in Artificial Intelligence*, pages 343–356. Springer International Publishing.

Emanuele Bastianelli, Danilo Croce, Andrea Vanzo, Roberto Basili, and Daniele Nardi. 2016. A discriminative approach to grounded spoken language understanding in interactive robotics. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York*.

David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the 25th AAAI Conference on AI*, pages 859–865.

Albert Diosi, Geoffrey R. Taylor, and Lindsay Kleeman. 2005. Interactive SLAM using laser and advanced sonar. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005, April 18-22, 2005, Barcelona, Spain*, pages 1103–1108.

Charles J. Fillmore. 1985. Frames and the semantics of understanding. *Quaderni di Semantica*, 6(2):222–254.

S. Harnad. 1990. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346.

Geoffrey Hinton, Li Deng, Dong Yu, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath George Dahl, and Brian Kingsbury. 2012. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 29(6):82–97, November.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Cynthia Matuszek, Evan Herbst, Luke S. Zettlemoyer, and Dieter Fox. 2012. Learning to parse natural language commands to a robot control system. In Jaydev P. Desai, Gregory Dudek, Oussama Khatib, and Vijay Kumar, editors, *ISER*, volume 88 of *Springer Tracts in Advanced Robotics*, pages 403–415. Springer.

Andreas Nüchter and Joachim Hertzberg. 2008. Towards semantic maps for mobile robots. *Robot. Auton. Syst.*, 56(11):915–926.

Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. 2009. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*.

M. Tanenhaus, M. Spivey-Knowlton, K. Eberhard, and J. Sedivy. 1995. Integration of visual and linguistic information during spoken language comprehension. *Science*, 268:1632–1634.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *J. Artif. Int. Res.*, 37(1):141–188, January.

Andrea Vanzo, Danilo Croce, Giuseppe Castellucci, Roberto Basili, and Daniele Nardi. 2016. Spoken language understanding for service robotics in italian. In *15th International Conference of the Italian Association for Artificial Intelligence*, page to appear.