Article

The International Journal of Robotics Research
2017, Vol. 36(5–7) 660–683

DOI: 10.1177/0278364917691112
journals.sagepub.com/home/ijr

Structured learning for spoken language understanding in human-robot interaction

Emanuele Bastianelli[1], Giuseppe Castellucci[2], Danilo Croce[3], Roberto Basili[3] and Daniele Nardi[4]

Abstract

Robots are slowly becoming a part of everyday life, being marketed for commercial applications such as telepresence, cleaning or entertainment. Thus, the ability to interact via natural language with non-expert users is becoming a key requirement. Even if user utterances can be efficiently recognized and transcribed by automatic speech recognition systems, several issues arise in translating them into suitable robotic actions and most of the existing solutions are strictly related to a specific scenario. In this paper, we present an approach to the design of natural language interfaces for human robot interaction, to translate spoken commands into computational structures that enable the robot to execute the intended request. The proposed solution is achieved by combining a general theory of language semantics, i.e. frame semantics, with state-of-the-art methods for robust spoken language understanding, based on structured learning algorithms. The adopted data driven paradigm allows the development of a fully functional natural language processing chain, that can be initialized by re-using available linguistic tools and resources. In addition, it can be also specialized by providing small sets of examples representative of a target newer domain. A systematic benchmarking resource, in terms of a rich and multi-layered spoken corpus has also been created and it has been used to evaluate the natural language processing chain. Our results show that our processing chain, trained with generic resources, provides a solid baseline for command understanding in a service robot domain. Moreover, when domain-dependent resources are provided to the system, the accuracy of the achieved interpretation always improves.

Keywords

Spoken language understanding, human-robot interaction, natural language processing, machine learning for natural language understanding

1. Introduction

As the first robots with vocal interfaces have been released on the market (Cocorobo, 2013; Nao, 2008; Pepper, 2014; Q.bo, 2012), their interaction capabilities become more relevant. This aspect is even more crucial when a high level of interaction and collaboration with a robot is required, as, for example, in service robots or in disaster response robots. In these contexts, the human language represents a natural way of interaction for its expressiveness and flexibility. Moreover, (non expert) users may expect to communicate or have dialog with a robot with the same simplicity as they would with another human. Hence, the ability of a robot to understand a user's spoken command (or, in general, an utterance expressed via natural language) is regarded as an important building block for human-robot interaction (HRI). However, developments towards a dedicated approach that is effective on robots are still relatively limited, as several challenges need to be faced in this context. Speech processing should be robust to environmental disturbances and allow for context-sensitive natural language understanding and grounding capabilities (Scheutz et al., 2011). Although powerful off-the-shelf automatic speech recognition (ASR) components are available, the interpretation of spoken natural language by robots is challenging,

[1]Department of Civil Engineering and Computer Science Engineering, University of Roma Tor Vergata, Italy
[2]Department of Electronic Engineering, University of Roma Tor Vergata, Italy
[3]Department of Enterprise Engineering, University of Roma Tor Vergata, Italy
[4]Department of Computer Science, Control and Management Engineering, Sapienza, University of Rome, Italy

Corresponding author:
Emanuele Bastianelli, Department of Civil Engineering and Computer Science Engineering University of Roma Tor Vergata, Via del Politecnico 1,00133 Roma, Italy.
Email: bastianelli@ing.uniroma2.it

because the input can be very noisy and the intended meaning is often not a straightforward consequence of the ASR output.

Few approaches have been developed to implement language understanding components for robotic applications (Chen and Mooney, 2011; Tellex et al., 2011a). However, these approaches typically focus on dedicated systems, developed for specific tasks (e.g. following route instructions (Kollar et al., 2010)). As a consequence, resources and models, even when suitable for a system, cannot be realistically generalized, i.e. adopted within new applications. On the other hand, recent research into other dialog-based applications, e.g. a personal digital assistant, is driven by general linguistic theories, for example the work by Chen et al. (2013, 2014), but it does not specifically deal with the problems arising in the deployment of dialog systems on robotic platforms.

In this paper, we present an approach to the design and implementation of a spoken language understanding (SLU) workflow for command interpretation that focuses on the following issues: The adoption of state-of-the-art tools that are rooted in contemporary linguistic theories and the use of general linguistic resources; the introduction of a specific approach to deal with the noise in the input and the integration with *grounding* (Harnad, 1990), required to map symbols onto corresponding real world elements, provided by the robot perception system. The resulting system is not tied to a specific robotic application (i.e. the set of robot specific commands and its tasks), and it allows for an effective and reusable command interpretation for spoken human-robot interaction.

The workflow can be decomposed in the following modules: Speech-to-text and morpho-syntactic analysis, performed by:

(a) an off-the-shelf automatic speech recognizer (ASR);
(b) an off-the-shelf statistical parser;
(c) a semantic analysis module comprising of a post processing re-ranking stage acting over the ASR output and a statistical shallow semantic parser both relying on Structured Learning techniques;
(d) a final component that compiles the semantic interpretation of the user input into a robot command.

This last step accomplishes the grounding of the semantics derived from linguistic evidence into the operational environment.

In this work, we also devoted a significant effort to the creation of systematic resources for the training of the system at different stages of the interpretation process, thus obtaining a multi-layered annotated spoken corpus. Several spoken commands have been collected and annotated to compose a realistic SLU corpus for HRI. The annotations characterize linguistic information at various levels and we adopted a domain-independent, linguistically and cognitively-motivated theory (i.e. the frame semantics, Fillmore (1985)) to represent the meaning of utterances. The collection, discussed in the work by Bastianelli et al. (2014b), is called human robot interaction Corpus (HuRIC) and contains speaker utterances in the home robotic domain.

By relying on the HuRIC resource, we have been able to train some of the modules on a dedicated data set and to experimentally evaluate the performance of individual modules of the overall chain, outlining the advantages obtained by the proposed approach. Experimental results suggest that a processing chain can be initialized by only reusing existing and general corpora, such as the FrameNet corpus (Baker et al., 1998), i.e. a large scale collection of sentences annotated according to the linguistic theory we adopted. This first result is noticeable when considering that the only off-the-shelf linguistic resources can be used for a first setup of the chain. In addition, we empirically demonstrate that a small amount of training examples specific for HRI are sufficient to improve the performance of the system, thus showing that a limited effort to collect training examples is worthwhile.

The rest of the paper is organized as follows. Section 2 discusses related work, reporting main paradigms of natural language processing, spoken language understanding and machine learning that are involved in the proposed solution. The proposed approach is presented in Section 3, while Section 4 describes the annotated HuRIC corpus. Sections 5 and 6 discuss the empirical evidence derived by focusing both on the fine grain evaluation of the individual stages of the proposed chain as well as on the evaluation of the system within a concrete end-to-end operational setting. Finally, Section 7 summarizes the results and draws some conclusions.

2. Related work

Studies on interactive robotic systems stand at the cross road of different branches of artificial intelligence (AI). The robot capability of interpreting vocal commands depends on a variety of processing stages, such as decoding, understanding and reproducing spoken language, resolving the lexical references to real world objects and planning the requested action.

The computational treatment of language in fact corresponds to a complex process acting on a linguistic input. Its interpretation poses a set of requirements on the underlying language processing activity. The morphological properties of the individual words in the input are to be detected as hints to capture the grammatical relations between parts of the sentence. Finally, a meaningful representation is built for the sentence and made available to the decision making stage, e.g. a dialog model. In general, language processing is thus decomposed into a cascade of sub-tasks: *Tokenization* and *morphological processing*, that correspond

to the task of segmenting the linguistic input into meaningful units (tokens) and recognizing their basic properties: *Grammatical* or *syntactic analysis*, that produces a syntactic representation of a sentence (e.g. a parse tree) through its rewriting (e.g. parsing) supported by a linguistically motivated grammar: *Semantic analysis* or *parsing* that derives a logical form for each sentence, that is a formula expressing a semantic representation of the sentences meaning, mostly depending on the linguistic theory adopted: *Pragmatic analysis*, where the logical form is linked to a logical representation of the communicative context, such as sequences of facts derived from a document, or the interpretation of a, possibly incomplete, input utterance in the light of previous ones in a dialog.

The result of this process needs to then be linked to the robot behavior, by *grounding* the command into an executable action and its argument into the elements of the environment.

The large body of research interested in the above typical NLP cascade and related to this work can be summarized along three major challenges. First, the related works concerning the language understanding functionalities studied in the specific context of HRI are discussed in Section 2.1. Then, the linguistic theories, models and resources traditionally employed to support a meaningful representation of natural language semantics are introduced (Section 2.2). Finally, the machine learning methods used over linguistic structures, i.e. sequences and trees, are discussed in Section 2.3: these provide the algorithmic foundations of the SLU approach proposed by the present work.

2.1. Spoken language understanding in HRI

Spoken Language Understanding addresses the problem of human language interpretation when it is conveyed by a speech signal (de Mori, 2007). Spoken language is often ungrammatical, fragmented and may present disfluencies. Moreover, automatic speech recognition may introduce transcription errors, leading to sentences possibly lacking a precise and consistent meaning. SLU has been a prolific research field since the nineties.

Following the trends in natural language understanding, two main approaches have been adopted in robotics to develop SLU interfaces: Grammar-based and data-driven.

Grammar-based systems for speech recognition models language phenomena through grammar, that is used to drive the transcription of an audio signal. Often, grammar is enriched with semantic information, i.e. semantic attachments, which is used to build a semantic representation of an utterance during the transcription process (Bos, 2002). Such a method is used, for example, in the work by Bos and Oka (2007), where Godot, an interactive robot for house interaction, is presented. In Godot, the SLU module performs semantic parsing jointly with the ASR process, producing the final interpretation together with the utterance transcription. Other approaches based on formal languages have been applied, as in the work by Kruijff et al. (2007) for spoken dialogues in the context of human-augmented mapping. Combinatory categorial grammar (CCG) is applied over the transcriptions obtained with a grammar based on an ASR engine. Recently, in the work by Perera and Veloso (2015), a flexible template-based algorithm has been proposed in order to extract a semantic interpretation of robotic commands directly operating over their corresponding syntactic parse trees.

As opposed to rule-based methods, data-driven techniques have also been applied to SLU for robotic application. In the work by Chen and Mooney (2011) and MacMahon et al. (2006), a robotic simulated system learns how to follow route instructions in a virtual environment. The parsing is addressed as a statistical machine translation (SMT) task between the human language and an ad hoc robot language. The same problem is addressed using encoders and decoders implemented through recurrent neural networks in the work by Mei et al. (2015). SMT is also applied in the work by Matuszek et al. (2010) to learn a probabilistic translation model between natural language and formal descriptions of the paths in a known map.

Examples of natural language route directions and the corresponding paths described using such a formal language are used for training. Similarly, in the work by Matuszek et al. (2012), a probabilistic CCG is used to parse natural navigational instructions to robot executable commands. The grammar is induced and parameterized using a log-linear model over training data of sentences, paired with the corresponding command, encoded using a specific robot control language.

Kollar et al. (2010) addresses the problem of following natural language directions. Specific meaning structures, called spatial description clauses, are parsed from sentences with a sequence labeling approach performed using conditional random fields (CRFs). In the work by Tellex et al. (2011a), natural language instructions about motion and grasping are mapped into instances of probabilistic graphical models, called generalized grounding graphs (G^3). A beam search is performed in reverse order, across a set of entities described in a semantic map, to find the set of groundings that correspond to the most likely combination in the graphical model. In such a way, the interpretation of an incoming utterance and its grounding in the surrounding world are jointly achieved.

Grounding confidences are learned through a log-linear model on a labeled corpus. Approaches to the specific task of *extracting spatial semantics* from user utterances to build a model of the environment are presented in the work by Hemachandra et al. (2014) and Walter et al. (2013). In the work of Fasola and Mataric (2013a,b), SLU is basically performed using a Bayesian classifier trained over a specific corpus, that exploits different features as, for example, the verb and the preposition used in the sentence and the objects involved in the relation. The classifier learns to parse navigation and pick-and-place instructions that involve spatial language into a formalism representing the command to execute: This expresses actions augmented with spatial

features related to it, e.g. the path and the static relation involved.

In general, the adoption of standard inductive (i.e. machine learning) methods in the design of interactive systems has been shown to be beneficial, as also surveyed in the work by Cuayáhuitl et al. (2013). However, such data-driven systems have been devised to deal with specific situations and domains. A noticeable exception is the work by Misra et al. (2016) where the authors define a probabilistic approach, namely CRFs, to the grounding of natural language instructions within a changing environment. The proposed data-driven approach is based on data about instructions, actions and tasks, collected in an on-line simulated game.

Language learning methods depend on specific design choices for the representation of individual input instances, e.g. feature vectors, and outcomes. General theories about language semantics have been largely studied as they give rise to comprehensive meaning representation formalisms. In turn, theoretically justified representations give rise to cost-effective and accurate language interpretation algorithms, in a variety of different applications. For their role, meaning representation formalisms, as reusable semantic representations for HRI, will be thus discussed in the next section.

2.2. Meaning representation for language understanding

The meaning extracted from natural language during a semantic parsing process should be expressed according to a representation model. The represented semantics can be application or algorithm dependent, offering structures that reflect specific aspects of a domain. On the other hand, meaning representations can be given according to general linguistic theories (e.g. frame semantics (Fillmore, 1985)), as these are more portable across applications and domains. Many works reported in Section 2.1 fall in the former category. In the work by MacMahon et al. (2006) ad hoc structures called *compound action specifications*, model the actions and conditions of route instructions in the robot commands, by representing the surface meaning of a sentence as a predicate-argument structure. The G^3 devised in the work by Tellex et al. (2011a) are dependent on the algorithm, as they are instances of probabilistic graphical models that enable a mapping between words or sentences and concrete objects, places, paths and events in the real world. Moreover, the output of the probabilistic decoding is represented using a simple formalism that is not inspired by any defined theory.

In the context of SLU for robotics, some works do rely on comprehensive linguistic theories. Explicit logic-based representations are adopted in the work by Bos and Oka (2007), where interpretations are given according to discourse representation theory (Kamp, 1981), a logic formalism based on λ-calculus. Similarly, in the work by (Kruijff et al., 2007) utterance meaning is hosted in hybrid logic dependency semantics (Kruijff, 2001) formalisms. The spatial description clauses defined in the work by Kollar et al. (2010) are structures that can hold the result of the spatial semantic parsing in terms of the spatial roles. Although inspired by existing theories, such structures are hybrid representations of action and spatial semantics.

Computational linguistics is research focused on general semantic theories about lexical and textual meaning. Research achievements have also been developed around theories of meaning representation related to large scale resources, such as general purpose knowledge bases, like WordNet (Miller, 1995) or VerbNet (Kipper-Schuler, 2005), as well as annotated corpora, like PropBank (Palmer et al., 2005) or the FrameNet corpus (Baker et al., 1998). The latter, besides inspiring some other HRI research (Thomas and Jenkins, 2012), is built upon a cognitively sound theory about the lexicon, i.e. frame semantics (Fillmore, 1985). A *frame* is a cognitive device that defines a real-world situation and its participants, in terms of a micro-theory about an activity (e.g. *moving*, *selling* or *making*) and its major relations (e.g. the *mover* or the *direction/goal* of the movement, or the *seller*). Frames are rooted in the lexicon and are considered as a theory on how the lexicon for a natural language encodes the cognitive aspects of frames: Verbs (e.g. *to sell*, or even nouns, *acquisition*) trigger (or evoke) frames (i.e. *Selling*) and verb arguments (e.g. the grammatical Subject) activate the semantic roles, thus characterizing the participants (e.g. the *Seller*). Notice that this approach refers to natural languages in general and efforts in the development of FrameNet for different languages exist. As a consequence, the lexical theory of FrameNet allows one to semantically represent a large set of situations with a high coverage of semantic phenomena and a significant level of reusability. Moreover, the FrameNet corpus, including a huge collection of texts labeled in terms of frames and semantic roles, supported the development of data-driven approaches to semantic interpretation tasks, largely known as the semantic role labeling process (Carreras and Màrquez, 2005; Gildea and Jurafsky, 2002).

In the context of data-driven approaches to SLU for HRI, several alternative annotated corpora have been built, but most of them have a task-specific nature representing a limitation for the definition of reusable solutions. In fact, the adopted semantics focuses either on a specific domain (e.g. spatial domain of route instructions) (Kollar et al., 2010; Kuhlmann et al., 2004), or depends on structures that are specific to the learning approach (Tellex et al., 2011b). Some of them are sticking to an independent representation, but do not offer a suitable level of abstraction (Dukes, 2013; Thomas and Jenkins, 2012). Some of them do not even contain any structured meaning representation, as in (MacMahon et al., 2006), where route instruction semantics are provided in the form of environmental changes, which are observable by the agent traversing the environment.

In this work, we promote the adoption of FrameNet. This is beneficial, on the one side, to determine a general meaning representation framework with a large coverage of language and real world phenomena, much wider than the ones required by HRI. FrameNet in fact currently includes about 1000 different frames with a dictionary of about 12,000 lexical entries. On the other side, it promotes the reusability of FrameNet data (i.e. annotated data sets), FrameNet lexicons (i.e. the frame dictionaries) as well as the adoption of learning algorithms, that have been largely experimented and evaluated against the FrameNet corpus.

Moreover, we also extended FrameNet data, by creating a publicly available HRI corpus, called HuRIC. This corpus relies on the frame semantic theory, but refers to service robotic style interactions, i.e. commands and domotic requests. It is composed of a set of recorded utterances, each one manually annotated according to frame semantics phenomena, acting as a training and validation resource in the context of HRI. Training in this scenario makes use of highly complex input structures (e.g. word sequences and trees) and produces structured predictions (i.e. frames and their roles that corresponds to graph structures). As a consequence, complex machine learning algorithms can be trained, as further discussed in the next section.

2.3. Structured learning for spoken language understanding

In the context of SLU for the development of conversational dialog systems, advanced structured learning techniques have been successfully applied. In particular, Kernel methods (Shawe-Taylor and Cristianini, 2004) have been applied in the work by Moschitti et al. (2007). In this case a classifier exploiting a specific formulation of kernel function, namely tree kernels (TKs (Collins and Duffy, 2002)), is used to tag users' utterances with semantic hypotheses. These kernel-based methods are particularly interesting as they can be directly applied over linguistic structures (i.e. syntactic parse trees) obtained with off-the-shelf natural language processing tools. Moreover, they obtain state-of-the-art results in several language processing tasks, as presented in the work by Moschitti (2012).

TKs have been similarly applied in the work by Coppola et al. (2008), where a FrameNet-inspired formalism has been adopted over a corpus specifically created for the development of dialog systems. In the work by Tur et al. (2005), a statistical parser is trained over sentences extracted from the PropBank corpus and used to semantically parse user utterances for a spoken dialog system. The experimental results show that it is sufficient to use task-independent training data to obtain good performances.

In the work by Dinarelli et al. (2011), other advanced NLP techniques have been applied together with TKs. The semantic interpretation problem is modeled as a sequential labeling task using CRFs. The *n*-best list of hypotheses from the ASR is thus tagged, and a re-ranking stage is applied over it. Re-ranking allows computation of a new rank over the target list of items (here a list of ASR transcription hypotheses), according to a ranking function that tries to bring hypotheses that are more likely considered correct up in the rank.

Some specific steps of our SLU chain are similar to the ones proposed in the work by Dinarelli et al. (2011). In particular, we also perform hypothesis re-ranking and semantic parsing by adopting statistical methods for sequence labeling. However, our approach presents some significant differences. First, we rely on a lexical theory whose representation is focused on the linguistic meaning, such as frame semantics, in order to maximize the reusability of the data and algorithms across HRI applications. Second, we apply structured learning techniques over such general representations that are demonstrated to be very effective in a variety of NLP tasks, i.e. a smoothed-partial tree kernel (Croce et al., 2011) and SVM^{hmm} (Altun et al., 2003). As we will show in the next sections, this allows us to reuse a large body of theoretical work and resources, rooting our work on principled semantic phenomena, i.e. frames.

Frame semantics clarify the role of lexical and grammatical knowledge and simplify the design of planning strategies; rather systematic one-too-many mapping exists in fact between frames and robotic actions, as also exploited in the work by Thomas and Jenkins (2012). In addition, we propose a solution for ASR hypothesis re-ranking which, at the best of our knowledge, is a novel contribution in the context of HRI; it emphasizes linguistic structures often neglected in previous work, where basic methods, e.g. stopword removal, are applied to deal with uncertainty in the spoken language. The adoption of FrameNet as a source of training examples in language learning makes the overall learning approach reusable across different styles of interaction, as for the semantic coverage of the frame repository made available. We also show how the achievable performances can be improved by the addition of small amounts of domain examples, better reflecting language phenomena in the robotic domain, e.g. spoken commands to a robot in the house servicing scenario drawn from the HuRIC corpus.

3. The overall processing chain

This section provides the high-level decomposition of our proposed workflow for spoken language understanding. We have developed a processing chain by leveraging on existing tools, such as a free-form ASR and a generic morpho-syntactic parser. A statistical semantic parser built according to Croce et al. (2012b) is then adopted. Figure 1 summarizes the proposed workflow; for each step the produced output is also shown as it represents the input for the next step. We grouped the different steps of the workflow as follows.

1. *Speech and morpho-syntactic analysis*. The user utterance is transcribed into text (in the *speech recognition* step), as for example '*bring the can on the dining table*',

and is parsed by a generic morpho-syntactic parser during the step called *morpho-syntactic analysis*. The quality of the ASR process is then improved through the *speech re-ranking* step over the set of possible transcription hypotheses.

2. *Semantic parsing*. The transcribed text is semantically analyzed in order to recognize the command expressed by the user's utterance; in the *action detection* step the aim is to detect the intended action, e.g. *bringing*, as well as the participating entities in the *full command recognition* step, e.g. *the can*, as the targeted object and the *dining table* as the goal position of the action.
3. *Grounding*. During the *action grounding* step, the semantic information derived in the previous steps is linked to the robot specific action representation formalism; the final grounding is performed in the *argument grounding* step by linking objects and locations referred to by the command with real word entities, e.g. spatial coordinates, as well as the proper robot operation needed to satisfy the user request.

The pragmatic analysis of robotic commands is out of the scopes of this work. In the following section/subsections the above steps are detailed.

3.1. Speech and morpho-syntactic analysis

The first step in the spoken command processing chain is the automatic transcription of vocal commands. In our architecture, we rely on an off-the-shelf ASR system. Even though ASR systems are designed to be robust in many scenarios, our aim is to verify whether their performance on a specific domain can be improved. Roughly speaking, since robots are still programmed to work in controlled environments and for specific tasks, we ask the ASR to provide more than one transcription of an utterance; then, we apply a re-ranking strategy through a model that is specific for an application domain and that can be easily re-trained to work in different domains, without adapting the entire ASR engine, as described in the following subsection. We used the official Google Speech API of the Android environment as the ASR engine (Chelba et al., 2013), that represents a largely available off-the shelf solution with very good accuracy.

3.1.1. Speech re-ranking. The re-ranking module aims to improve the recognition of the free-form ASR engine by applying post-processing on the n-best candidate transcriptions. The objective is to push the correct transcription up in the rank. The selection of the correct sentence is a function over a set of hypotheses (i.e. the n candidates) whose aim is to assign a preference to the correct hypothesis, discarding the wrong ones. We thus propose a re-ranking function based on a set of linguistic properties, which can be directly derived from domain-specific annotated examples and they specialize the workflow to the expected robot scenario. As an example, let us consider a spoken command such as 'bring the can on the dining table'. The following list of candidate hypotheses for such an utterance are yielded by the ASR:

(a) bing the can on the dining table;
(b) bring the can on the dining table;
(c) bring the canon dying table.

While (b) is the correct interpretation, it may not receive the best confidence from the ASR. The proposed re-ranking strategy aims to measure how much sentence (b) violates the smallest number of grammatical and semantic constraints with respect to (a) and (c). The hypothesis violating fewer constraints is supposed to receive the highest score, thus resulting in being the best candidate. Notice that this preference depends on the grammar, as '... the can ... on the dining table' corresponds to a properly formed fragment for a referring expression, while 'bring the canon dying table' is ungrammatical. However, lexical information is also important as '... bing the can ... ' in (a) is implausible.

As in the work by Shen and Joshi (2003), the re-ranking function can be mapped in a classification function that evaluates the plausibility of hypotheses pairs (H_i, H_j), where both H_i and H_j are included in the n-best list obtained from the ASR output. Given an ordering of the n-best list, we can define pairs that reflect the total order, for example, we can build pairs in which the left element is ranked before the right one. In this way, the pair (H_i, H_j) means that H_i is ranked before H_j in the rank of the n-best transcriptions. The ranking function can be modeled through a classification function f, that takes as the input a pair (H_i, H_j) and outputs a score reflecting if the proposed order (i.e. H_i is ranked before H_j) is suitable; in other words, f should discriminate between positive pairs (i.e. reflecting a correct order) from negative ones (i.e. pairs determining an inverted order).

A learning algorithm, here the support vector machine (SVM (Cortes and Vapnik, 1995)), has been used to acquire the classification function f. The prediction $f(H_i, H_j) \geq 0$ means that H_i ranks before H_j[1]. The best hypothesis $\hat{H}$ can thus be derived as the hypothesis that maximizes the chance that a specific hypothesis is ranked higher with respect to all the other hypotheses, i.e.

$$\hat{H} = \arg\max_{H_i} \sum_{k,\, i \neq k} f(H_i, H_k) \tag{1}$$

Notice that, as in our setting we do not have the correct ordering of the n-best list, we built the training dataset by exploiting the correct transcriptions against the incorrect ones. During training, given the n-best list of transcriptions, we can derive positive example pairs as (H_i, H_k), for $k \in [1, \ldots, n]$ where $i \neq k$ and H_i is the correct transcription. Negative instances are obtained as the inverse of the positive ones (H_k, H_i).

Kernels (Shawe-Taylor and Cristianini, 2004) are functions used in machine learning to map instances in complex representation spaces, where convergence of the training algorithm is guaranteed, while more expressive features are

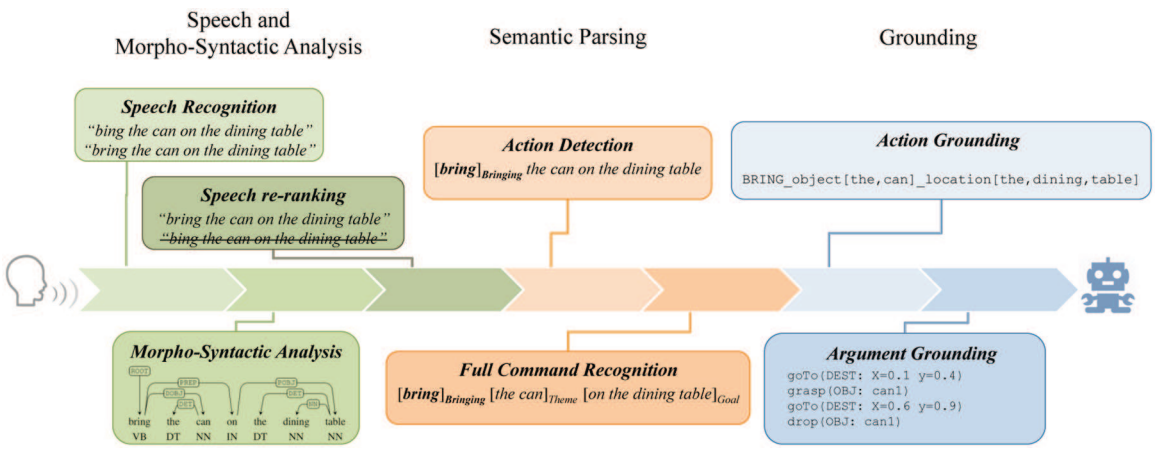

Fig. 1. The SLU workflow for command interpretation.

made available. Kernels can be seen as similarity functions between training examples projected in such spaces. As an example, syntactic kernels, such as TKs (Collins and Duffy, 2002), compute the similarity between two sentences as the number of all meaningful structures and substructures shared between their corresponding parse trees, which encode their syntactic and semantic information. As the syntax can play a crucial role in the re-ranking process, we embed the syntactic representation for each hypothesis into a proper learning scheme. The syntactic regularities seen in the training set can be captured and modeled by TKs and, together with lexical semantics, they can drive the re-ranking process. In fact, the syntactic similarities between seen and unseen trees will assign a lower score to those sentences, among the candidate ones, that present an odd and semantically implausible syntactic structure. In addition, the use of TKs allows us to avoid the process of manually selecting the more discriminative features for this task. As TKs model similarity between two training examples as a function of their shared tree fragments, discriminative information is automatically selected by the learning algorithm, without the need for manual feature engineering.

During the learning phase of f in equation 1 (or its application in the classification step) the SVM algorithm compares two pairs of hypotheses within a suitable kernel function for ranking. Thus, given two pairs of hypotheses $e_1 = (H_{11}, H_{12})$ and $e_2 = (H_{21}, H_{22})$, the re-ranking kernel K_R (also known as the preference kernel, (Shen and Joshi, 2003)) over e_1 and e_2 is defined by

$$\begin{aligned} K_R(e_1, e_2) \quad &= K(H_{11}, H_{21}) + K(H_{12}, H_{22}) \\ &\quad - K(H_{11}, H_{22}) - K(H_{12}, H_{21}) \end{aligned} \tag{2}$$

where K can be any valid kernel function. The definition of kernel K can combine other, more specific, kernels. For example, lexical features (such as the conventional bag-of-words, BoWs) can be captured by a lexical kernel K_{lex}, while the tree kernel, namely K_{TK}, could be integrated through kernel composition (e.g. linear combinations) as

$$K(H_i, H_j) = K_{\text{lex}}(H_i, H_j) + K_{TK}(H_i, H_j) \tag{3}$$

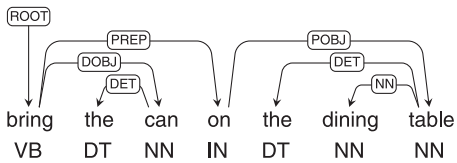

Fig. 2. Example of a dependency graph associated to the sentence 'bring the can on the dining table'.

where utterances are individual sentences. K_{lex} insists on feature vectors for H_i (H_j) that encode lexical information from the utterance (as discussed below), while K_{TK} exploits the grammatical dependencies as expressed into the parse tree of H_i (H_j). In our re-ranking model, we exploit both lexical similarity and syntactic similarity. Regarding the K_{TK}, we explored smoothed partial tree kernels (SPTKs) (Croce et al., 2011) as they smoothly match the trees leveraging on a lexical similarity measure on the leaves.

The tree kernel function is applied to tree structures derived from the original dependency parse structures (see Figure 3), and modeled according to the grammatical relation centered tree (GRCT) representation, shown in Figure 3. Non-terminal nodes reflect syntactic relations, such as `ROOT` and `DOBJ`, pre-terminals are the part-of-speech (POS) tags, such as nouns and verbs, and leaves are lexemes, such as *bring::v* and *table::n*. Each word is lemmatized to reduce data sparseness, but it is characterized with its POS-tag. The SPTK estimates a high similarity with sentences characterized by a strong syntactic analogy and, at the same time, a strong lexical relatedness. For instance, given the example 'bring the can on the dining table', it is expected that an utterance such as 'take the jar on the coffee table' would show a higher similarity with respect to a sentence sharing the same words, but with a different meaning, such as 'the can is on the table'. This approach should assure that, even though the correct transcription is not present in the n-best list, the one with highest degree of syntactic and semantic correctness would occupy the first position in the computed rank.

K_{lex} has been evaluated as the lexical relatedness established between individual words, estimated according to the latent semantic analysis (LSA (Landauer and Dumais, 1997)) technique. A word-by-context matrix M is acquired through large scale corpus analysis in order to obtain a geometrical representation of words that provides a computational generalization of their meaning. In such geometric spaces, called word spaces, semantic relations between words are reflected by the notion of distance between so-called word vectors.

In order to capture paradigmatic relations among words, e.g. quasi-synonymy, entries of the vocabulary V (i.e. the set of words observed in the corpus) are associated to vectors computed as it follows. For every word $w_i \in V$, each dimension of the corresponding word vector counts the co-occurrence of w_i with another word w_j, either in its left or right context in a small window size of size n, e.g. $n = 3$; this value should better capture the paradigmatic lexical properties of words (Sahlgren, 2006). Vector components are weighted through the point-wise mutual information scores. In order to reduce the sparseness in such high dimensional spaces, the LSA technique suggests the decomposition of M through singular value decomposition (SVD, (Golub and Kahan, 1965; Landauer and Dumais, 1997)) into the product of three new matrices: U, S, and V, so that S is diagonal and $M = USV^{\text{T}}$. M is then approximated by $M_k = U_k S_k V_k^{\text{T}}$, where only the first k columns of U and V are used, corresponding to the first k greatest singular values. This approximation projects a generic word w_i into the k-dimensional space using $W = U_k S_k^{1/2}$, where each row corresponds to the representation vectors $\vec{w}_i$. The original statistical information about M is captured by the new k-dimensional space, which preserves the global structure, while removing low-variant dimensions, i.e. distribution noise.

Given two words w_1 and w_2, their semantic relatedness σ is estimated as the cosine similarity between the corresponding projections $\vec{w}_1, \vec{w}_2$ in the LSA-based word space, i.e. $\sigma(w_1, w_2) = \vec{w}_1 \cdot \vec{w}_2 / \|\vec{w}_1\| \|\vec{w}_2\|$. This measure captures second order relations between words and it is an effective model for paradigmatic relations, i.e. synonymy and co-hyponymy between words, such as in *can* vs. *jar* or *direction* vs. *path*. As emphasized in the work by Cristianini et al. (2002), σ is a valid kernel itself and it can be used as a more effective form of the lexical kernel, e.g. K_{lex} in equation (3), as the meaning of a sentence can be expressed by the linear combination of all the LSA vectors corresponding to the nouns, verbs, adjective and adverbs composing the sentence.

The extraction of linguistic information for individual candidates is carried out through the Stanford CoreNLP suite[2] (Manning et al., 2014). It is one of the largely used off-the-shelf morpho-syntactic parser, which includes *tokenization*, *morphological processing* (e.g. part-of-speech tagging) and *grammatical analysis* (e.g. dependency parsing). These allow us to extract linguistic information in the form of linguistic structures as the dependency parse tree shown in Figure 2.

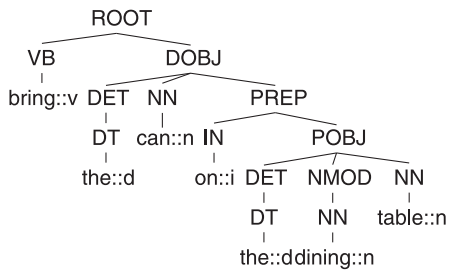

Fig. 3. Example of GRCT representation of the sentence 'bring the can on the dining table'. GRCT representation is presented in the work by Croce et al. (2011).

3.2. Semantic parsing

Several issues arise in translating human generated commands into suitable robotic actions. First, the underlying meaning of an utterance needs to be understood, and then mapped into robot-specific commands, filling the gap between the world representation of the robot and the linguistic information conveyed in the sentences. This is a typical form of semantic parsing, where the output logical form is the crucial support for grounding linguistic expressions into objects, as they are represented in the robot set of beliefs (i.e. robot knowledge).

Existing methodological results and tools suggest a variety of useful semantic representations to express robot commands; we rely on *frame semantics* (Fillmore, 1985). This theory generalizes the actions or, more generally, experiences into *semantic frames*. Each frame corresponds to a real world situation (e.g. the action of *Taking* or *Bringing* an object) that in turn is defined by a micro-theory that specifies the set of participating entities, i.e. the semantic arguments. Arguments play specific roles with respect to the situation described by a frame (e.g. the AGENT involved), so that a a situation is correctly interpreted when the set of core arguments is perfectly recognized.

In frame semantics, *lexical units* (*lu*s, such as verbs, nouns or adjectives) evoke specific frames and they serve as anchors between the textual content and the theory. The semantic arguments of the frame express the participants in the underlying event and, are mapped to *frame elements*. As an example, the frame semantics representation of the sentence 'bring the can on the dining table' is as follows: [***bring***]$_{Bringing}$ [*the can*]$_{\text{THEME}}$ [*on the dining table*]$_{\text{GOAL}}$, where *bring* is the lexical unit evoking the *Bringing* frame. In this structure, the different elements of the *Bringing* event are highlighted, that are the THEME role (the object to be taken) and the GOAL role (the place where the object is to be brought). Frame semantics captures general and application independent information about the situations

and represents a suitable interlingua for connecting language semantics to the perceptual and planning components of the robot.

3.2.1. Semantic role labeling for command semantic parsing. In the workflow for the spoken command interpretation of Figure 1, the task of recognizing semantic roles in transcribed utterances is modeled as the semantic role labeling (Palmer et al., 2010) task, as follows. The action detection (AD), i.e. the detection of the command without its potential arguments, can be seen as the semantic role labeling (SRL) frame prediction (FP) task, i.e. determining and disambiguating all events evoked in a sentence. Given each intended action in a sentence, the full command recognition consists of the complete recognition of all of the arguments in the command. This can be mapped into the other two subtasks of SRL: Argument identification (AI) and argument classification (AC). AI is the task of locating the span of the frame arguments in a sentence. For example, after the *Bringing* frame has been recognized in the sentence 'bring the can on the dining table', the AI should recognize the boundaries [*the can*] and [*on the dining table*] as two different arguments, by identifying the start and the end of each sentence fragment. Finally, the AC corresponds to the assignment of semantic types, such as the THEME and GOAL of the individual detected spans.

Frame Semantics is not only a sound linguistic theory, but it also allows the adoption of large scale annotated resources, that extend (or integrate) the background knowledge of an interactive robot. The FrameNet annotated corpus represents a collection of more than 150,000 annotated sentences.[3] In particular, several approaches for SRL have been proposed since the work by Gildea and Jurafsky (2002), as well as automatic systems, such as Shalmaneser[4] (Erk and Pado, 2006) or LTH[5] (Johansson and Nugues, 2007). In the next subsections the data-driven approach for SRL used in our chain is discussed.

3.2.2. Action detection. The AD problem is a particular instance of the general *predicate disambiguation* task (Croce et al., 2012a), where semantic frames represent predicates. Given a sentence s, AD is modeled as the FP task. Frames (that are to be intended here as command generalizations) are expressed in s by the presence of *lu*s. The problem here is that each *lu* can potentially evoke more than one frame or no frame at all; as an example, the verb *to bring* can evoke both the *Bringing* frame and the *Causation* frame, as in the sentence 'Marriage brings economic and social success'. All of the possible frames that a *lu* may evoke are contained in a resource called the *lu* dictionary that is built at training time. In order to disambiguate over the possible frames, a classification function $g(\cdot)$, following a multi-classification scheme based on the $SVM^{\text{multiclass}}$ approach by Joachims et al. (2009), is adopted. An example for our learning scheme is given by a $\langle lu, s\rangle$ pair, while the target class is the frame $f \in F$. Thus, each sentence s generates as many instances as the number of retrieved *lu*s, and each pair $\langle lu, s\rangle$ in s is classified in F through the multi-classification scheme.

Each instance is modeled as a set of manually engineered features that reflects different linguistic observations.

1. *Lexical features*. These features include the left and right contexts of the lexical unit, i.e. the sets of m words before and after the evoking word.
2. *Shallow syntactic features*. These features include a set of POS *bi*-grams and *tri*-grams of the words before and after the *lu*. Notice that the left and right contexts are analyzed separately, in order to better capture syntactic information.

It is worth noticing that this feature modeling is purely based on the representation of linguistic properties that are always observable in any kind of linguistic input (e.g. a command) and do not change across HRI application scenarios. The proposed features are in fact commonly adopted in the context of semantic parsing in NLP.

Parameters of the function $g(\cdot)$ are learned over an annotated dataset, where gold $\langle\langle lu, s\rangle, f\rangle$ pairs are given. At classification time, the best frame $\hat{f}$ for a pair $\langle lu, s\rangle$ is given by $\hat{f} = \arg\max_{f \in F} g(lu, s)$.

3.2.3. Full command recognition. The full command recognition step is modeled as a sequence labeling task, where each word in the sentence is associated with a specific predicate derived from the AD. Here, a different classification scheme, based on a Markovian formulation of a structured SVMs (i.e. SVM^{hmm} proposed[6] in the work by Altun et al. (2003)) has been adopted. It combines both a discriminative approach to estimate the probabilities in the model and a generative approach to retrieve the most likely sequence of tags that better explains the predicate information. Given an input sequence $\mathbf{x} = (x_1 \ldots x_l) \in \mathcal{X}$ of feature vectors $x_1 \ldots x_l$, the SVM^{hmm} algorithm learns a model isomorphic to a k-order hidden Markov model.

As explained in Section 3.2.1, the full command recognition is performed in two steps. Given a sentence s, a target *lu* and the evoked frame f, the AI phase is modeled as a sequence labeling task over sentence observations, such as lexical properties (i.e. words) and morpho-syntactic properties (e.g. POS tags). Boundaries are represented here using the `IOB` notation that has been successfully adopted in many sequence labeling tasks in NLP, such as *chunking* (Tjong Kim Sang and Buchholz, 2000). The classifier associates a special tag to each word in the sentence, suggesting that it is the beginning (`B`), internal (`I`) or outer (`O`) token with respect to one argument boundary; a correct labeling of the two arguments in the example sentence is represented as

`O`-*bring* `B`-*the* `I`-*can* `B`-*on* `I`-*the* `I`-*dining* `I`-*table*

The AI differs from chunking as sequences here are meant to represent the parts of the sentence playing semantic roles

with respect to the given frame f. It may not be the case that all the spans in the sentence participate to the frame. The AC task aims to assign a label to each of the recognized spans from the AI phase, e.g. THEME to *the can* and GOAL to *on the dining table*. Again, the structured formulation of SVM^{hmm} is applied with different target classes, i.e. the role labels. Notice that these are frame specific, so that a model for each frame has been trained.

More formally, at classification time, the model predicts a tag sequence $\mathbf{y} = (y_1 \dots y_l) \in \mathcal{Y}$ after learning a linear discriminant function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ over input-output pairs. The labeling $f(\mathbf{x})$ is thus defined as

$$f(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y}; \mathbf{w})$$

and it is obtained by maximizing F over the response variable, $\mathbf{y}$, for a specific given input $\mathbf{x}$. F is linear in some combined feature representation of inputs and outputs $\Phi(\mathbf{x}, \mathbf{y})$, i.e. $F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$. In particular, Φ extracts meaningful properties from an observation/label sequence pair $(\mathbf{x}, \mathbf{y})$, i.e. the interactions between the attributes of the observation vectors x_i and a specific label y_i (i.e. emissions of x_i by y_i), as well as interactions between neighbor labels y_i along the chain (transitions). In other words, Φ is defined so that the complete labeling $\mathbf{y} = f(\mathbf{x})$ can be efficiently computed from F, by applying a Viterbi-like decoding algorithm

$$y^* = \arg\max_{y} \{ \sum_{i=1 \dots l} [\sum_{j=1 \dots k} (x_i \cdot w_{y_{i-j} \dots y_i}) + \Phi_{tr}(y_{i-j}, \dots, y_i) \cdot w_{tr}] \}$$

It is worth noting that the adopted Markovian modeling implicitly provides the best possible labeling of a sentence (i.e. the 'best path' obtained by the decoding). This reranking ability is typical of joint global models for SRL, as discussed in the work by Toutanova et al. (2008). The output labeling is the most likely global solution over a sentence, computed by the Viterbi algorithm.

In the training phase, SVM^{hmm} solves the optimization problem described in the work by Altun et al. (2003) that, given training examples $(x^1, y^1) \dots (x^n, y^n)$ of sequences of feature vectors $x^j = (x^j_1, \dots, x^j_l)$ with their correct tag sequences $y^j = (y^j_1, \dots, y^j_l)$, allows us to derive the model parameters $\mathbf{w}$ and ϕ.

In the discriminative perspective of SVM^{hmm}, each word is represented by a feature vector, describing its different observable properties.

1. *Position*. I.e. its relative distance from the target predicate.
2. *Lexical features*. Its lemma and POS tag.
3. *Distributional features*. We used the word space model introduced in Section 3.1.1 also to provide a generalization, that overcomes data sparseness of lexical features. It means that a sentence such as 'bring the jar on the dining table' will have a labeling similar to that of 'bring the can on the dining table', even if they do not share the same words, due to the similarity between the *jar* and *can* vectors in the word space. Thus, each word is projected in such space and the resulting vector is added to each word representation.
4. *Semantic features*. The involved predicate and the underlying frame.
5. *Contextual features*. The left and right lexical contexts, represented by the three words before and after; the left and right syntactic contexts as the POS bi-grams and tri-grams occurring *before* and *after* the word.

As for the AD step presented in Section 3.2.2, the features described above are based only on linguistic properties that are always observable in any command.

3.3. Grounding

The last step of the processing chain takes as the input a fully instantiated semantic frame and turns it into a robot action. This process is usually referred to as grounding. First, a frame is grounded into the corresponding action through the activation of the executable robot plan implementing that action (we call this step *action grounding*). Second, the lexical fillers of the frame elements must be grounded in the real world to instantiate the action arguments, e.g. entities in the environment required by the action, represented by the plan parameters (we call this step *argument grounding*).

Although the proposed solution allows for the development of SLU chains that do not require specific formalisms for expressing the robot's plans, in our experimental evaluation, robot actions are modeled using Petri-net plans (PNPs (Ziparo et al., 2008)). We briefly introduce them below to better clarify the relationship between linguistic frames and plans. A PNP is a plan representation formalism, based on Petri-nets, with a specialized structure suitable to model robot actions, including sensing, parallel executions and interrupts. A PNP is basically composed of the following.

1. *Places.* i.e. the circles in Figures 4 and 5 that represent the execution phases of the actions.
2. *Transitions.* i.e. the gray rectangles in Figures 4 and 5 representing events leading to *places*. Transitions have a label that specify the condition under which the transition is fired.
3. *Edges.* Connecting the *transition* to *places* (i.e. the arrows in Figures 4 and 5).

The black dot is called a *token* and its position represents the state of the system. More than one token can be present in the system. In fact, transitions can consume and/or produce tokens from places according to the so-called 'firing rules', which define the dynamic behavior of the Petri-net. PNPs, in addition to the tokens of a conventional Petri-net, include specific conditions for triggering the execution of

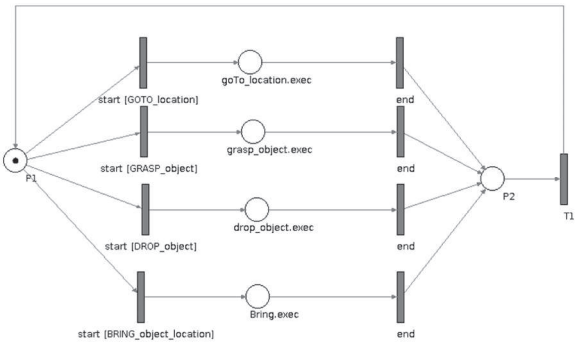

Fig. 4. A fragment of the Petri-net plans modeling the robot behaviors.

actions. Figure 4 shows a snippet of the main PNP modeling of our robot's general behavior. Each branch going out from the initial place (i.e. the one with the token) in the PNP represents a plan, and thus the execution of an action activated when an instantiated frame is received as the input.

3.3.1. Action grounding. A robot plan is usually modeled through a set of one or more *primitive plans* (or *primitive actions*), that can be combined to generate more complex plans. In this way, it is possible to generate plans representing basic actions; for example, the movement to one location in the environment is implemented by the primitive plan `goTo`. Notice how this is linguistically encoded by the *motion* frame. However, more complex actions requiring more processing steps can be modeled. For example, the action of bringing an object to a location may require the execution of a sequence of primitive plans as `goTo`, followed by `grasp` and then `goTo` to reach the destination where the object has to be brought, and finally `drop` to release it. This sequence is represented by the complex plan `Bring`, that is linked to the *Bringing* frame and whose implementation is show in Figure 5. These PNPs can be built in many ways. For example, they can be manually implemented, as well as taught to the robot through spoken interaction (Gemignani et al., 2015). Notice that, despite the fact that a dialog is not the focus of this work, a simple and specific plan is added to the PNPs in order to allow the robot to provide a reply in the case in which no frame is recognized in an utterance during the semantic parsing.

3.3.2. Argument grounding. Once the action specified by the frame is grounded, the argument grounding takes place. The instantiated frame elements need to be grounded, as they represent the possible parameters required for the execution of the selected plan. In our case, this corresponds to retrieving the position of the referenced objects or locations in the space. Specifically, grounding creates the link between the symbols and the robot perception of the environment. In this work, the perceptual representation of the world provided by the robot is represented through a semantic map 'that contains, in addition to spatial information about the environment, assignments of mapped features to entities of known classes' (Nüchter and Hertzberg, 2008). In particular, a semantic map may include representations of objects and locations, their categories (e.g. `newspaper`, `table`, …), their positions with respect to the metric map used for navigation, the perception that the robot has acquired (e.g. images), and possibly other features. In this sense, the semantic map is a resource that enables grounding since it represents the environment acquired through perception. First, the word representing the semantic head of the frame element is extracted from the syntactic tree. Then, the coordinates of the entity having a name that matches the semantic head is retrieved and used to instantiate the plan argument corresponding to the frame element. For example, the THEME frame element of the interpretation [**bring**]$_{Bringing}$ [the can]$_{\text{THEME}}$ [on the dining table]$_{\text{GOAL}}$ is grounded by first extracting the semantic head 'can', and then searching in the semantic map for the entity with that name, e.g. specified through a property as `hasName(c1,can)`. Here we assume that the location can be uniquely identified, otherwise clarification dialogues are needed. The corresponding coordinates are then retrieved, e.g. by firing a query such as `hasCoordinates(c1,X,Y,Z)` on the semantic map. These are used to instantiate the `_object` parameter in the `Bring` PNP, as shown in Figure 5.

4. Creating resources for complex HRI SLU workflow: HuRIC

The computational paradigms introduced so far are based on machine learning techniques and depend strictly on the availability of training data, that is still poor in the HRI SLU context. In order to properly train and test our framework, we developed a collection of datasets that together form the human-robot interaction corpus (HuRIC).[7]

HuRIC is based on frame semantics, and it captures cognitive information about situations and events expressed in sentences, as discussed in Section 3.2. Different from other corpora for SLU in HRI, it is not system or robot dependent both with respect to the kind of sentences and with respect to the adopted formalism. The HuRIC contains information strictly related to natural language semantics and it is decoupled from specific systems. The corpus exploits three situations representing possible commands given to a robot in a house environment. Each situation defines a group of sentences representative of different working conditions.

The HuRIC is composed of three different datasets: Grammar generated (GG), S4R experiment (S4R) and RoboCup (RC). These are given in increasing order of complexity and they are designed to stress the architecture described in Section 3. Each dataset includes a set of audio files representing robot commands, paired with the correct transcription. Each sentence is then annotated with: lemmas, POS tags, dependency trees, frame semantics and

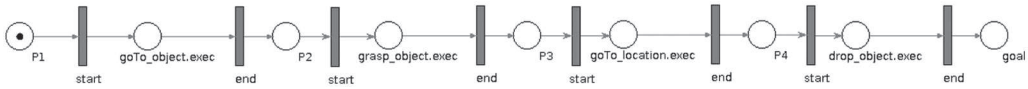

Fig. 5. The PNP of the `Bring.exec` plan.

spatial semantics (Zlatev, 2007). In this way, the HuRIC can potentially be used to train all of the modules of the processing chain presented in Section 3.

The GG dataset contains sentences that have been generated by the speech recognition grammar developed for the 'Speaky For Robots' project,[8] whose aim was to devise a methodology to design vocal user interfaces for robots involved in several applications. Here, we focus on the house-servicing robotics. Sentences have been read and recorded by three different speakers and recorded using a push-to-talk microphone. The acquisition process took place inside a small room, thus with low background noise. The language represented here is free of colloquial forms of interaction. Examples of sentences in this dataset are: 'move near the window', 'go to the bathroom', 'carry the paper near the counter'. Their syntactic structures and substructures are limited, i.e. they represent simple language phenomena. Sentences are mainly imperative commands to a robot, where no modal verbs (e.g. *can* you…) or interjections (e.g. *please*…) are used. The lexicon is also limited and bound to simple objects that can be found in a home environment. This dataset is intended to verify the system behavior in a setting that resembles a controlled scenario, in which linguistic phenomena are simplified.

The S4R dataset has been gathered in two subsequent phases of the 'Speaky for Robots' project. In the first phase, users gave commands to a real robot operating in a real home, and the same microphone used for the GG dataset has been used. In the second phase, users interacted through a web portal to record commands. Images and texts helped them to contextualize general situations in order to produce commands. This time the internal microphone of the computer running the portal has been used. The language represented in this dataset is closer to free spoken English as compared to the GG dataset. Examples of sentences in this dataset are: 'bring the phone to the bedroom', 'find the stove in the studio', 'put the phone on the table in the dining room', 'go to the television in the living room'. Their syntactic structures and sub-structures are slightly more variable with respect to the sentences of the GG dataset.

Finally, the RC dataset has been collected during the RoboCup@Home competition held in 2013, in the context of the RoCKIn project.[9] The recording took place directly in competition venues or in a cafeteria, thus with higher levels of background noise. Here, the same web portal used for S4R has been employed, using again the internal microphone of the PC running it. Expressions exhibit large flexibility in lexical choices and syntactic structures, thus this dataset is much more variable with respect to the previous collections. Examples of sentences in this dataset are: 'can you slowly follow my father', 'please bring the mobile phone to the living room', 'this is a bedroom with one bed and two nightstands'. Colloquial forms and descriptive sentences are frequent, as well as the use of modal verbs. This is mainly due to the freedom given to the users, who chose the sentences and recorded the corresponding audio. Thus, the RC dataset reflects the expectations that users pose on natural interactions with a robotic platform. It is worth noting that this last group of annotated sentences is intended to verify the performances when the complexity grows. The evaluation with this dataset can be seen as a stress test of our architecture, aimed at verifying the robustness of the learning approaches proposed in the previous sections.

The semantic frames and frame elements are used to represent the meaning of the commands, as, in our view, they reflect the actions a robot can accomplish in a home environment. A subset of FrameNet-inspired semantic frames corresponding to the defined robot actions has been selected. Tables 2 and 3 report detailed statistics about the number of frames and corresponding frame elements, for each dataset The information instantiated in a frame encodes what is needed by a robot to understand what action to perform. No sentence has been rejected as it was not covered in FrameNet.

The manual annotation process behind the HuRIC has been decomposed into the three subtasks defined in Section 3.2.1: The frame prediction, the argument identification and the argument classification. Two annotators have been involved during the annotation process. They agreed on most of the annotations. Their agreement was measured and it was high, over 82 points in the F1-score. For detailed information about the corpus, such as annotation procedures and more annotation details, refer to Bastianelli et al. (2014b).

Table 3 reports the number of audio files and sentences for each dataset. Speakers have been asked to repeat sentences created by others, in order to have different audio versions of the same command. Consequently, each dataset has more than one sentence per audio file. Although the HuRIC contains only 321 sentences, for a total of 570 audio files, it is an ongoing resource, and increasing its size is one of the aims of our research.

5. Experimental evaluation

This section reports and discusses the experimental evaluation of the processing chain presented in Section 3. The evaluations have been designed in order to verify the quality

Table 1. Distribution of the frames and frame elements in each dataset.

Frame	GG	S4R	RC
Attaching	1	0	2
ITEM	0	0	2
GOAL	1	0	0
Being_in_category	0	0	14
CATEGORY	0	0	14
ITEM	0	0	14
Being_located	0	0	20
LOCATION	0	0	11
PLACE	0	0	6
THEME	0	0	20
Bringing	10	22	37
AGENT	0	1	6
BENEFICIARY	2	1	13
GOAL	8	21	24
MANNER	0	0	1
SOURCE	0	0	9
THEME	10	22	37
Change_operational_state	1	3	3
DEVICE	1	3	3
OPERATIONAL˙STATE	1	1	2
Closure	6	0	1
CONTAINER˙PORTAL	2	0	1
CONTAINING˙OBJECT	4	0	0
Entering	4	0	1
GOAL	4	0	1
Following	1	6	30
AREA	0	0	1
COTHEME	1	6	30
GOAL	0	1	5
MANNER	0	3	6
PATH	0	0	1
SPEED	0	0	1
THEME	0	0	6
Giving	0	0	2
RECIPIENT	0	0	2
THEME	0	0	2
Inspecting	0	1	3
DESIRED˙STATE	0	1	1
GROUND	0	1	3
INSPECTOR	0	1	1
Motion	9	25	39
AREA	0	0	1
GOAL	9	25	38
MANNER	2	1	1
PATH	0	1	1
THEME	0	0	8
Perception_active	1	0	0
PHENOMENON	1	0	0
Placing	0	7	10
AGENT	0	0	1
GOAL	0	7	10
THEME	0	7	10
Releasing	0	2	0
GOAL	0	2	0
THEME	0	2	0
TIME	0	1	0
Searching	3	27	24
COGNIZER	0	0	5
GROUND	3	16	7
PHENOMENON	3	27	24
PURPOSE	0	0	5
Taking	12	6	12
AGENT	0	0	4
PURPOSE	0	0	2
SOURCE	10	3	5
THEME	12	6	12

Table 2. Total statistics of frames and frame elements for each dataset.

	GG	S4R	RC
Total number of frames	48	99	198
Average frames per sentence	1.00	1.03	1.12
Total number of roles	74	158	357
Average roles per sentence	1.54	1.65	2.02

Table 3. Number of audio files and sentences.

Dataset	# Audio files	# Sentences	# Audio files per sentence
GG	137	48	∼2.85
S4R	141	96	∼1.46
RC	292	177	∼1.64

achievable by the adopted SLU solution in the robotic context. Moreover, we evaluate whether the data-driven methods can be tailored to the target domain by only extending the annotated material.

Each module is analyzed to determine the main sources of errors to get more insight into the specific subtasks. The three portions of the HuRIC (see Section 4) have been adopted to test the modules and the entire chain in different environmental conditions. All of the morpho-syntactic information, such as POS tags of words and dependency trees of specific utterances have been derived through the use of the CoreNLP 3.3.1 parser (Manning et al., 2014). We determine the statistical significance by using the model discussed in the work by Padó (2006).

In the following, we first address the ASR module and the re-ranking approach in Section 5.1. The semantic parser

is evaluated in Section 5.2. The combination of these modules is presented in Section 5.3, while experiments over the whole chain integrated within a robotic platform are discussed in Section 6. Unfortunately, comparisons to other SLU systems for robotics were not possible, mainly because our approach is not focused on a precise task, like previous works have. Here, we consider a fully operational setup, where the robot is supposed to perform a variety of different actions (i.e. up to 16, as *Bringing*, *Taking* and *Releasing*), not just navigating or grasping.

5.1. The ASR module

All audio files are analyzed through the Google ASR Engine. The service returns n-best lists, with at most 10 different hypotheses for each recorded utterance. In order to reduce the evaluation bias to errors, only the n-best list with an available solution within the 10 input candidates were retained for the experiments. After this pre-processing stage, the corpus was made by 56 utterances for the GG part, 72 utterances in the S4R and 84 utterances in the RC subset. On average, we found about five hypotheses per utterance. Given the hypotheses, we applied the re-ranking strategy discussed in Section 3.1.1. The preference kernel formulation (equation (2)) is used as the global model for re-ranking within a SVM algorithm implemented in the KeLP framework (Filice et al., 2015). Several kernel combinations (see equation (3)) have been tested, based on the following kernels.

1. K_{BOW}: This is a representation reflecting the lexical information of the sentence. Each text is represented as a vector whose dimensions represent boolean indicators of the presence or not of a word in the text. In line with the work by Shawe-Taylor and Cristianini (2004), we investigated the contribution of the polynomial kernel of degree two denoted as K^2_{BOW}, as it defines an implicit space where word pairs also contribute as independent features.
2. K_{LSA}: This refers to the vector representing the candidate sentence, as the linear combination of single word vectors composing it, as projected in the LSA word space. LSA vectors are derived from a word space built from the analysis of the ukWaC data set (Baroni et al., 2009). Notice that this web corpus is not specific to the house service robotics domain. In the word space construction, documents are first analyzed through the CoreNLP to derive lemmas and POS tags of each word. The former is necessary to reduce data sparseness (e.g. *evoked* and *evoking* are considered in their lemmatized form *evoke*). The latter allows us to distinguish between the same word in different syntactic categories. In this sense, the vocabulary V of our word space is composed of $\langle lemma, pos\rangle$ pairs, representing a word w. Each entry of the resulting vocabulary is associated with a co-occurrence vector representation (see Section 3.1.1). In order to construct such vector, we consider a large corpus: a word w_i co-occurs with a word w_t, if w_i appears in the context window of $n = +-3$ words of w_t in the corpus. By considering only vectors corresponding to the words with a frequency ≥ 200, a co-occurrence matrix M is built, with about $45,000$ rows (each corresponding to a $w \in V$). Then, the SVD reduction is applied to M, with a dimensionality cut of $k = 250$. Hence, the final matrix M_k has a dimension of about $45,000 \times 250$. As suggested in the work by Cristianini et al. (2002), we have investigated a radial basis function kernel between such vectors, i.e. $K_{\text{RBF}}(x, y) = e^{-||x-y||^2}$.
3. K_{SPTK}: The smoothed partial tree kernel is adopted to emphasize grammatical information encoded in a parse tree through the generalization allowed by the lexicon in the word space. It computes the similarity between lexical nodes as the similarity between words in the word space. So, this kernel allows generalization over both the syntactic and the lexical dimensions.
4. K_{CONF}: This refers to the one-dimensional vector representing the numerical score corresponding to the confidence output by the ASR. Google ASR reports a confidence score v_0 relative to the first transcription only. In the experiments, a scaling function $v_k = 0.75/(k-1) \cdot v_0$, where v_k is the resulting confidence value for the $k-$th sentence in the ranking) is applied to estimate scores characterizing the other transcriptions. It allows us to define a conservative re-ranking that also considers the ASR confidence in the answer.

For each dataset, we split the data into four subsets; in turn, one set is used as a test set and the remaining three parts are used to train the re-ranker. The best SVM parameters, such as the cost factor, and the best kernel configuration have been estimated on a held-out set for each of the four sets. The configuration that achieved on average the best performance (*BestConfig*) on the four sets is selected. In the GG dataset, the use of K^2_{BOW}+K_{SPTK} shows that in simple sentences, the lexical information provided by the BOWs model and the syntactic/semantic generalization of the SPTK is sufficient to properly re-rank the Google hypotheses. In fact, this is a greedy configuration that ignores the confidence level provided by Google.

In the S4R and RC datasets, i.e. datasets characterized by a higher variability, the K^2_{BOW}+K_{SPTK}+K_{LSA}+ K_{CONF} achieved the best results, showing that a more conservative approach, that considers the Google confidence, and a stronger lexical generalization is beneficial.

Table 4 shows the quality of the ASR module measured in terms of the word error rate (WER (Popović and Ney, 2007)) and the sentence error rate (SER) across the four subsets. The analysis has been performed on each dataset, with and without considering the effect of the re-ranking, the *wRR* and *woRR* rows, respectively, in the table. When this is applied, the WER decreases 1.1 points for the GG dataset, of 2.4 points for the S4R dataset and 1.4 points for the RC dataset. Improvements can be noticed also in

Table 4. Performance of the speech module in terms of the word error rate and sentence error rate.

	GG	S4R	RC
Word error rate			
woRR	2.7%	4.0%	3.8%
wRR	1.6%	1.6%	2.4%
Sentence error rate			
woRR	16.9%	14.5%	20.9%
wRR	7.5%	8.6%	19.2%

Table 5. Performance of the re-ranking module in terms of Precision@1. Results obtained using the setting called BestConfig are statistically significant with respect to the other results ($p < 0.05$).

	Google	Bow	BestConfig
GG	0.834 ± 0.115	0.867 ± 0.101	0.924 ± 0.054
S4R	0.855 ± 0.067	0.841 ± 0.048	0.913 ± 0.028
RC	0.789 ± 0.067	0.717 ± 0.079	0.824 ± 0.100

terms of the SER, especially on the GG dataset and the S4R dataset, with a decrease of 9.4 and 5.9 points respectively. An out-of-vocabulary rate of 0.123, 0.144 and 0.291 has been also evaluated for the GG, the S4R and the RC datasets respectively.

The re-ranker is always beneficial and shows that the performance of a system trained on large sets of examples such as Google, can be still improved. Even if the improvement in the WER can be considered to be minor, it can be significant for further processing. The misinterpretation of a word may have in fact a strong impact on the overall chain. For example, the wrong transcription of the main verb inevitably compromises the recognition of the entire command. We have also applied a stricter measure, shown in Table 5, by estimating the system precision, in terms of the percentage of sentences suggested in the first position by Google or after the re-ranking process. As we adopted a n-fold evaluation scheme, we report the mean and the standard deviation of the performances obtained across the folds. Even considering the precision, the contribution of the re-ranking is effective. This is more noticeable in the GG and in the S4R datasets, where an improvement of about 10 points is measured. In the RC dataset this improvement is lower (four points obtaining a final score of 0.824%). We suppose it is due to the reduced lexical and syntactic variability of examples in the GG and S4R datasets with respect to the RC one.

We introduced also the *Bow* configuration in order to evaluate whether the improvement has been due to the adopted supervised algorithms or to the selected kernel. The *Bow* configuration exploits only the simple lexical kernel K_{BOW} seen above, and it can be considered as a baseline for our method. Notice that when considering these features, the system is able to improve the performance of the Google ASR in the GG scenario, but not in the S4R scenerio. It means that only lexical information is not sufficient when allowing more syntactic variations in the data. However, in the *Bow* and *BestConfig* settings, our system is able to reduce the standard deviation, thus showing that the results are even more stable with respect to the ones obtainable with the pure ASR. Moreover, an error reduction ranging from ∼50% to ∼17% is measured on the precision across the three datasets. Thus, a re-ranking approach can be effective in handling different working conditions, i.e. characterized by an increasing complexity both of the language and of the audio stream involved.

5.2. Evaluating the semantic parsing

In order to verify whether the semantic parsing system is working as expected, we carried out an individual evaluation of its sub-modules. In particular, we analyze the error propagation through the semantic parsing chain under different conditions. Moreover, we investigate where the errors concentrate, thus identifying the areas where more improvement can be achieved. The diverse working conditions are obtained providing a different evaluation with respect to each of the datasets whic compose the HuRIC, considering the correct transcriptions, i.e. not contemplating the error introduced by the ASR system. In this way, we focus on the errors of the semantic parsing module and eliminate the bias introduced by the ASR.

In order to identify where the semantic module fails, we provide an evaluation of single sub-modules where the input is in turn gold and non-gold. It means that we report the performance measures, in terms of precision (P), recall (R) and F1-measure (F1), with respect to the single sub-module, either when its input comes from a previous step of the overall workflow or it comes from the gold annotations. For example, the input of the argument identification phase of the non-gold scenario is the pair ⟨*sentence*, *frame*⟩, where the *frame* is the one recognized in the FP step. The evaluation is carried out by considering two cases: One where only FrameNet training material is adopted (FrameNetOnly, FNO), and another where the same data is augmented with examples coming from one dataset of HuRIC (hybrid, H). This allows us to verify whether the use of data-driven methods based on existing resources is effective for the robotic scenario. The hybrid setting is also intended to also verify whether additional in-domain training data can be beneficial.

In the FP phase, the F1 measures the system quality in correctly recognizing the frame(s), i.e. the robotic action in our scenario, for each sentence. In the AI phase, the F1 quantifies the system ability to recognize the boundaries of each argument; every token (i.e. span) of every argument must be properly detected. In the argument classification phase, F1 measures the correctness of the role label assignment to each span.

Table 6. Semantic parsing subsystem analysis reported in terms of the F1-measure for the FrameNetOnly (FNO) and Hybrid (H) settings. Results obtained using the *Hybrid* setting are statistically significant with respect to the *FrameNetOnly* setting ($p < 0.05$).

	Frame prediction		Boundary detection		Argument classification	
	FNO	H	FNO	H	FNO	H
Gold information at each step						
GG	0.826	0.833	0.684	0.871	0.589	0.822
S4R	0.812	0.817	0.743	0.872	0.736	0.912
RC	0.732	0.758	0.696	0.817	0.701	0.898
Non-gold information at each step						
GG	–	–	0.560	0.680	0.373	0.612
S4R	–	–	0.598	0.712	0.502	0.688
RC	–	–	0.527	0.635	0.451	0.597

Results in terms of the F1-measure for the three phases are reported in Table 6. As expected, a performance drop across the SRL steps is obtained, when non-gold information is provided at each step (bottom of Table 6),[10] compared to the setting where the gold information has been provided (top of Table 6). The former setting reflects a real operating scenario, where the performance drop is due to the error propagation during the semantic understanding process. For example, if we consider the AI phase, a F1 score of 0.684 is measured in the FNO setting with gold-standard information from the FP; this performance drops to 0.560, when enabling error propagation by feeding non-gold information through the modules in the GG scenario. Performances are even worse when considering more complex sentences like the ones in the RC dataset. Here, the most significant performance drops are observed in the AI and AC phase of the FNO setting; there is a true bottleneck for system performance. This problem is partially overcome when SLU specific examples are used for re-training, as in the H setting that exhibits a stable improvement in the F1.

We have also evaluated the semantic parsing systems by using only examples derived from the HuRIC, in order to test the contribution given by the corpus. Hence, no training examples from FrameNet have been considered. This test, called HuRICOnly, has been possible only on the RC dataset. In fact, the *n*-fold strategy used in the evaluation, made the training impractical for the GG and the S4R datasets, as their size was too small.[11] The result obtained in this evaluation can be compared with those reported in Table 6, in the RC row of the configuration where the gold information is used at each step. In the FP phase, the HuRICOnly setting obtained a F1 of 0.730, that is slightly below the F1 obtained in the FNO result. This suggests that a frame predictor (that enables the recognition of intended robotic actions) can be initialized by using general purpose linguistic resources.

In the BD phase, a F1 of 0.785 is achieved, higher with respect to the FNO setting (0.696) but lower with respect to the hybrid setting (0.817). Finally, a F1 of 0.868 is obtained for the AC step, being more tied to the hybrid setting (0.898). The difference with respect to the FNO setting is mainly due to the differences between the typical syntactic/semantic structures of the sentences from FrameNet (composed of assertive and declarative sentences from narrative or news). This language differs from the typical imperative utterances used to express robotic commands. However, this result is important when considering that:

- the FNO setting did not require any additional annotation with respect to the off-the-shelf available resources;
- the addition of a small amount of data to the examples from FrameNet is always beneficial as the Hybrid setting achieves the best F1 score.

The benefits of the Hybrid setting can be seen, for example, in the sentence 'grab the cigarettes next to the phone'. With the FNO setting, the sentence is incorrectly tagged as '**[grab]**$_{Taking}$ *the cigarettes* [*next to the phone*]$_{\text{THEME}}$'. Notice that the phone is tagged as the THEME of the Taking action, i.e., the object t0 be taken. The correct THEME is instead "the cigarettes". On the contrary, when examples from the HuRIC are added, the resulting tagging is '**[grab]**$_{Taking}$ [*the cigarettes*]$_{\text{THEME}}$ [*next to the phone*]$_{\text{SOURCE}}$', corresponding to the right interpretation of the frame elements. Such behavior justifies the increase in the performance for the AI and AC phase. Also the FP phase is biased by the nature of the data. For example, the *Motion* frame expressed in the command 'can you please move near the right lamp' by the *move* verb is not recognized in the FNO setting, invalidating further the AI and AC phases. Again, the frame is correctly tagged when the HuRIC data is made available. Such improvements are due to the inability of the system trained over written texts to deal with structures typical of spoken commands. For instance, phrases such as 'can you please' represent sequences and contexts unseen in FrameNet, that lead to frames and frame elements misinterpreted during the HMM decoding. These results support the assumption that specific corpora are beneficial when training machine learning systems, especially in contexts that are strongly characterized by a domain, as many HRI tasks are. These results are partially in contrast with the findings in the work by Tur et al. (2005).

In this work, the authors state that there is no strict need for in-domain data, even though they affirm that their use would improve the performance of SRL. From our perspective, the use of in-domain data is crucial, and this is noticeable from the F1 scores in Table 6 when non-gold information is provided. Our SLU module is designed to deal with a richer semantic formalism, and thus is more complex, being composed of three processing steps. The use of in-domain material becomes thus fundamental in order to compensate the error propagation across the modules.

Table 7. Recognition performance starting from the audio on the grammar generated set. Results obtained in the *wRR* setting are statistically significant compared to the *woRR* setting ($p < 0.05$).

	Action detection			Full command recognition		
	P	R	F1	P	R	F1
Grammar generated – FrameNetOnly						
woRR	0.753	0.445	0.560	0.173	0.102	0.128
wRR	0.779	0.489	**0.601**	0.186	0.117	**0.143**
Grammar generated – Hybrid						
woRR	0.753	0.445	0.560	0.210	0.124	0.156
wRR	0.779	0.489	**0.601**	0.267	0.168	**0.206**

Table 8. Recognition performance starting from the audio on the S4R set. Results obtained in the *wRR* setting are statistically significant compared to the *woRR* setting ($p < 0.05$).

	Action detection			Full command recognition		
	P	R	F1	P	R	F1
S4R – FrameNetOnly						
woRR	0.889	0.599	0.715	0.354	0.238	0.285
wRR	0.885	0.626	**0.733**	0.365	0.259	**0.303**
S4R – Hybrid						
woRR	0.880	0.599	0.713	0.470	0.320	0.381
wRR	0.876	0.626	**0.730**	0.505	0.361	**0.421**

5.3. The complete chain

In this section we will analyze the performances of an entire chain, i.e. an end-to-end system performing all of the processing steps starting from the audio input. The objective of this evaluation is two-fold. First, we evaluate the adoption of off-the-shelf tools and publicly available resources in the SLU setting. Second, we evaluate the contribution of specific corpora to improve the performances of the whole data-driven system. The system contains the full processing chain where: Starting from the audio file, the Google ASR is used to derive the possible transcriptions; the re-ranking module (as discussed in Section 3.1.1) is applied to the hypotheses generated from the ASR phase; the semantic parsing module is applied to the best interpretation as described in Section 3.2, and the action and the full command are extracted. We focused directly on the system output, by measuring precision, recall and F1-measure of the AD and full command recognition. In the AD, the precision is computed as the percentage of correctly recognized actions among the predicted ones. The recall is computed as the percentage of correctly recognized actions among the ones that should have been retrieved. The F1-measure is the harmonic mean between the precision and recall. In the full command recognition step, performances are computed in the same way, except that a prediction is considered correct only if the action and all its arguments are correctly recognized. This is a far more complex task, as individual words must be assigned to their correct arguments.

In Tables 7, 8 and 9, the performances are reported with respect to the three datasets in the HuRIC. As for previous experiments, two main experimental settings are reported: FNO and Hybrid. In this experiment, we added to the training set only those sentences corresponding to utterances without a correct transcription in the *n*-best list. This strategy is adopted in order to avoid observing sentences that will be used in the test set in the training data.

The contribution of the re-ranking module is also evaluated; the results called *without re-rank* (*woRR*) refer to the case where no re-ranking module is applied, while the results *with rerank* (*wRR*) also consider the contribution of the re-ranking strategy. Notice that the re-ranker has been parameterized in each dataset with the configuration that achieved the best results in Section 5.1, i.e. the *BestConfig* referred to in Table 5.

Table 9. Recognition performance starting from the audio on the RoboCup set. Results obtained in the *wRR* setting are statistically significant compared to the *woRR* setting ($p < 0.05$).

	Action Detection			Full Command Recognition		
	P	R	F1	P	R	F1
RoboCup – FrameNetOnly						
woRR	0.804	0.398	0.533	0.264	0.131	0.175
wRR	0.830	0.432	**0.568**	0.287	0.149	**0.196**
RoboCup – Hybrid						
woRR	0.810	0.480	0.603	0.297	0.176	0.221
wRR	0.830	0.505	**0.628**	0.335	0.204	**0.253**

It is worth noticing that the combination of the woRR and FNO settings corresponds to our baseline, and it is an indicator of the performances obtainable with completely off-the-shelf tools (such as Google ASR and CoreNLP parser) and general purpose resources (such as FrameNet). Other settings are instead meant to verify the contribution of each adaptive processing step described in Section 3.

The F1 scores of the complete chain are lower when compared with the results shown in Table 6, where the gold standard transcriptions are used. This performance drop is mainly due to two reasons. First, the error of the entire chain accumulates across each processing phase. Second, all of the models have been obtained by maximizing the precision metrics, since it is more important in the interaction with a robot. In fact, it is commendable that a robot understands commands with high confidence, i.e. without ambiguity. When low confidence levels are reached, the

robot should ask for confirmation instead of executing an incorrect action. In nearly all the evaluations of the AC step, a precision of 0.80 is achieved, while a lower recall is obtained. This is mainly due to bad transcriptions of the ASR module, as the SVM classifier does not recognize any command with sufficient confidence. This problem is amplified in the full command recognition phase that shows quite low scores, i.e. about 0.3 for the F1-measure.

In the case of the GG dataset (Table 7), the re-ranking strategy and the adoption of in-domain training material are both beneficial, and the F1 of the full command recognition improves from 0.128 to 0.206. The same improvement has not been measured in the AC phase. This was expected as no improvement was also experimented with gold standard transcriptions, as shown by the negligible difference between the FNO and the Hybrid column in the FP phase of Table 6. From a manual analysis of the outcomes most of the errors are due to the incorrect transcription of the main verb from the ASR module.

In the case of the S4R and RC datasets performances follow the same trend as for the GG dataset. In fact, the full processing chain improves from 0.285 to 0.421 and from 0.175 to 0.253 in the S4R and RC datasets, respectively. Even when syntactic and lexical variability is high enough to cause the Google ASR to have a higher WER (see Table 4), our approach is able to partially compensate and improve final performances.

Given the apparently low results of the F1 scores, we compared the proposed chain with another approach. The resulting system used in this comparison has been presented in the work by Bastianelli et al. (2014a); it is based on a grammar based engine, that jointly performs the speech analysis and the semantic parsing stages. The grammar based system has been built using the same grammar underlying the sentences as from the GG dataset. We measure the full command recognition capability. The grammar based approach achieves best results on the GG dataset, with a F1 score of 0.42. This is directly comparable with the best F1 score of Table 7, i.e., 0.206: as expected, the approach in Bastianelli et al (2014a) is performing better, because the grammar is accurate with sentences of its own design domain. However, when the syntactic complexity grows, e.g. in the S4R dataset, performance drops of the F1 are observed, i.e. 0.25 against the 0.42 obtained here. The drop is even more noticeable in the RC dataset, where richer lexical information was employed; in this case, a F1 score of 0.02 is obtained, against the 0.25 showed in Table 9. This performance drop is explained as the grammar doesn't cover rich lexical and syntactic variations. In order to make the grammar suitable for such phenomena, it should be re-engineered.

By looking at the results across the different datasets, one can notice that the F1 scores on the GG dataset are lower than the ones of the S4R and RC datasets. This may seem counter-intuitive, as the GG presents a more regular and simple language. The reason of such a drop derives mainly from the ambiguity of the verbs *take* and *look*, as they may evoke more than one frame. Especially, the first, which can evoke the *Taking* frame or the *Bringing* frame, while the second *Perception_active* or *Searching*. The GG dataset does not contain enough example of the verb *take* for the *Taking* frame, so that it is always assigned to the *Bringing* frame. It is important to understand that dealing with the ambiguity of *take* is very challenging, also for a human. Its meaning in terms of the frames depends also on the environment it is used in. It is enough to consider the different interpretations we may give to a command such as 'take the book on the table', in two different environments, one with a book on a table, and another without. The same applies also for the verb *look*. The systematic error in assigning the right frame to these verbs impacted on the overall performances.

These results emphasize the complexity of the entire task, where a chain composed mainly by off-the-shelf tools shows somewhat unsatisfactory performances. Although such results may appear low, they represent the final outcome of a complex processing chain. The error propagates across the chain, starting from the ASR, passing through the syntactic parsing and the re-ranking, and finishing with the semantic parsing. However, the system improves its performances by adding a limited number of annotated examples. This aspect may become crucial in the development of systems that can acquire novel training material by interacting with the user, and so adapting to the targeted application scenario.

6. Evaluating the processing chain on robotic systems

The evaluations presented in the previous section reported results of the processing chain not deployed on a real robot. They are useful to measure the performances and the errors introduced by each components of the chain (alone and in cascade). To demonstrate the effectiveness of our approach and its independence from specific platforms, we deployed it on several robots available at our labs, shown in Figure 6:

(a) a Videre design erratic[12];
(b) a Turtlebot[13];
(c) a mobile base derived from a Segway[14];
(d) MARRtino,[15] a platform built by the students of the robot programming class.

As a real example, we recorded a short video showing the complete chain at work on a Turtlebot as shown in Extension 1. The footage clearly shows how the re-ranking affects the interpretation process by selecting the right transcription in the two cases when the ASR returned a wrong one in the first position. From the images it is also possible to appreciate how the SRL system is able to deal with sentences presenting colloquial forms by extracting the correct command, e.g. in the command 'I need you to bring the laptop to the cabinet of the corridor'.

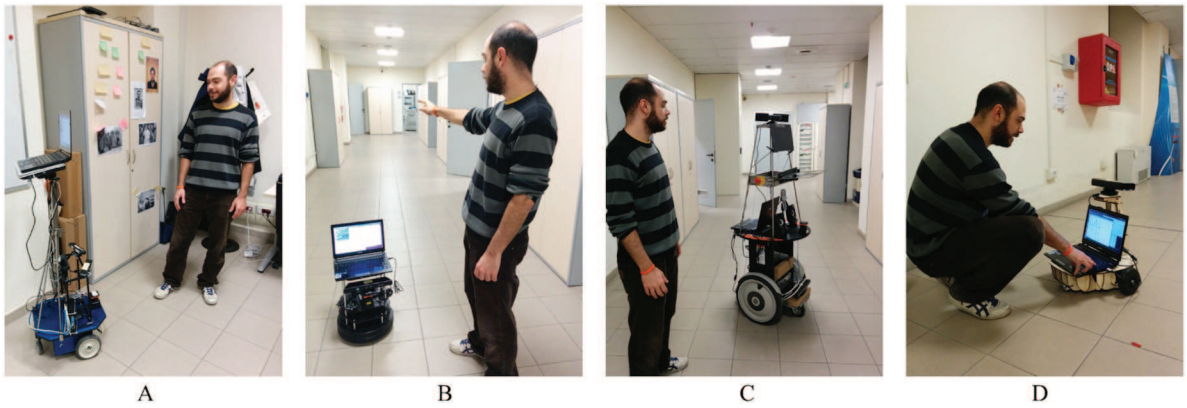

Fig. 6. The four platforms used to test the complete processing chain.

All of the robotic platforms used in the tests can perform a set of predefined PNPs derived from the set selected during the 'Speaky For Robots' project, hence they are related to tasks that a domestic robot should perform, e.g. moving, bringing and checking. This set of actions has been selected without considering any specific robotic platform. Thus, their implementation on the robot will depend on its ability to move autonomously in the space or to interact with objects (e.g. grasping or pushing). Since all of these platforms are without grippers, we simulated the interaction with objects as a request from the robot, asking the user to operate the action on its behalf, e.g. asking to put or remove objects from the loading tray, as it is possible to notice in the video in Extension 1.

In addition to the real demonstration, a preliminary evaluation of the whole system has been carried out, considering the end-to-end process starting from the audio input and ending in the action performed by the robot. We used a simulated environment to run the experiment. This choice has also been mandatory as the HuRIC is modeled on a house domain, so we had to reproduce a similar scenario, and this has been possible only using a simulated house. Figures 7 and 8 show the metric map and the associated semantic map, here split into rooms and furniture, used during the simulation. Table 10 reports the objects present in the house with their position, encoded in the semantic map as well. Furthermore, the semantic map also contains some facts about basic static spatial relations between objects and places, e.g. 'the table of the kitchen' or 'the book on the bed'.

The simulated platform has been modeled exactly as a Videre design erratic, by reproducing its exact size, and has been equipped with a Hokuyo laser range finder used for navigation purposes. The software, e.g. the Petri-net plans, discussed in Section 3.3 and used to model the robot behavior, has been developed in the Robot Operating System (ROS), and it relies on `move_base` as the path planner and the Adaptive Monte Carlo Localization (AMCL) localizer. The test has been run within the ROS stage simulator with the aforementioned map. The simulator has been tuned to reproduce also odometry errors and other noise coming from the perception sensors, e.g. the laser range finders, in order to take into account potential issues arising in a real application.

Table 10. Location of the objects in the house.

Object	#	Location
Book	3	Living_room:couch, Bedroom:bed, Studio:book_shelf
Jar	2	Kitchen:fridge, Living_room:table
Newspaper	1	Bedroom:nightstand
Apple	2	Kitchen:fridge, Living_room:table
Bottle	2	Kitchen:fridge, Studio:table
Cigarettes	1	Studio:table
Knife	1	Kitchen:sink
Jacket	2	Bedroom:cabinet, Living_room:cabinet
Phone	1	Living_room:table
Mobile phone	1	Bedroom:bed
Box	1	Living_room
Tv set	2	Living_room, Kitchen:counter
Pillow	1	Bedroom:bed

Fig. 7. Map of the house: Sampling of the space representing the rooms.

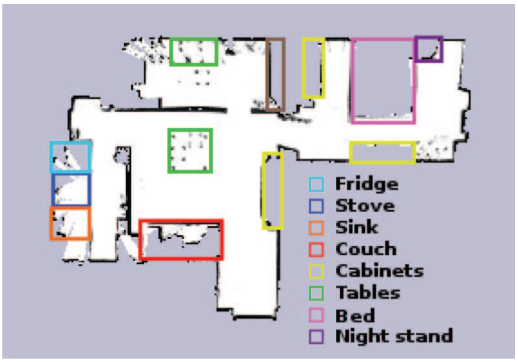

Fig. 8. Map of the house: Position of the furniture.

For the final evaluation, five users have been asked to give vocal commands to the simulated platform. They have been provided with a sketched map of the house, reporting the names of the rooms, as well as the locations of the objects and furniture. Each user has been quickly trained about the robot capabilities, explaining which kind of actions the robot could perform. The users have been requested to give about five commands each: the final number of commands used in the evaluations is 22. The vocal input has been captured with the internal microphone of the laptop running the SLU chain, by simulating the position of the microphone on the final robot. The model for the re-ranking has been trained on the RC dataset, while the ones involved in the semantic parsing step have been trained over the hybrid setting including FrameNet and the RC dataset.

First, we evaluated the percentage of correct commands executed by the robot. A command is considered correctly performed if the corresponding plan is activated and all of the involved arguments are properly grounded. This corresponds to an action that is correctly executed from the beginning to the end. Second, we evaluated the percentage of primitive plans correctly grounded. As reported in Section 3.3, some plans have been decomposed in a sequence of primitive plans, e.g. the `BRING` plan is composed of a `goTo`, a (simulated) `grasp`, another `goTo` and a final `drop`. Even though our final target is the correctness of the complete plan we also evaluated the primitive actions of the decomposition, that is the correctness of the grounding of individual primitives. In real scenarios, the correct generation of the first set of actions is still valuable as the user can potentially adjust the part misunderstood by the robot during the execution of the plan.

Table 11 reports the results in terms of precision, recall and F-measure. The precision is the percentage of the actual plans (or primitive actions) correctly grounded. Here, the wrong or partial grounding can be due to many causes, such as the wrong interpretation by the ASR with no re-ranking possibilities or errors during the parsing process. The recall, instead, is the percentage of expected plans (or primitive actions) correctly grounded, and thus performed by the robot. The recall should account for the cases where the robot does not respond to some commands. The last measure is the harmonic mean of the first two.

Table 11. Performance of the experiment in a simulated environment.

	Precision	**Recall**	**F-Measure**
Plans	68.75%	47.83%	56.41%
Primitives	94.44%	53.13%	68.00%

Differences can be noticed with respect to the results obtained in Table 9. This is due mainly to the bias derived from the user training about the robot capabilities. This led the users to interact in a more schematic way, without accounting for the colloquial forms too much, and thus introducing less noise in linguistic terms. Similarly, the lexicon used in the interaction has been implicitly partially constrained by letting the users know which objects and rooms where present in the environment. Finally, the experiment took place in a room without strong background noise, that is a very different scenario with respect to the noisy conditions reproduced in the RC dataset. These numbers are not directly comparable with the results obtained by other works that adopt machine learning based techniques to process natural language in a HRI context. First, some of them do not account for the error introduced in the ASR phase, as they process the string of text typed by the user. Second, there is a significant difference in the cardinality of the classes to be predicted. Our system deals with 17 different classes in the FP step of the SRL (one per frame plus `no_frame`), three in the AI step (`B`, `I` and `O`) and ∼4*17 in the AC step (one for each frame element plus `no_fe`). Other works, instead, use semantic representations with fewer classes to be predicted. Finally, some works compute their statistics about the percentage of successfully executed plans only considering those sentences that were correctly analyzed during the previous language processing phases, while we do not filter out any sentence.

As a final evaluation, we computed the average processing time of the chain during the experiment. Time represents a crucial constraint in interactive systems, where promptly replying is a distinctive feature of a proper interaction. Unfortunately, the Google ASR needs the Internet, and potential latencies introduced by the connection quality made the evaluation in terms of time insignificant. Hence, we evaluated the average time that the re-ranking process needed to re-rank a *n*-best list, both considering or ignoring the previous syntactic parsing stage. For the former case, the system took 1.364 ± 0.682 s to tag and re-rank a *n*-best list. The re-ranker alone instead needed 1.255 ± 0.567 s to evaluate a new rank. Each *n*-best list contained five hypotheses on average. Notice that no specific optimization methodology is applied here (e.g. caching or parallelization), and this time can be massively reduced.

The time needed by the semantic parser to tag the best hypothesis is negligible. In fact, as reported by Croce et al.

(2012b), it can tag ∼40 sentences per second, thus it takes 0.025 s to tag a sentence. The entire evaluation has been carried out on machine equipped with an Intel© Core® 2640M i7 @ 2.8 Ghz with 8 G of RAM.

7. Conclusions

In this paper, we propose an approach to the design and implementation of natural language interfaces for HRI, with the aim of enabling interactive robotic systems to generalize across a variety of scenarios in which natural interaction is required. We defined a processing chain translating spoken commands into a sequence of actions needed to satisfy the user request.

The proposed system is characterized by three main novel features. First, we propose an algorithm for the re-ranking of the transcription hypotheses generated by a general ASR model. We apply state-of-the-art machine learning techniques such as SVMs, complex kernels (e.g. SPTKs) and preference learning for re-ranking. Second, we design a statistical semantic parser based on frame semantics over user utterances. Semantic frames can be considered as a general meaning representation formalism, making them suitable for representing also the information of robotics commands. This allows the reuse of a large body of evidence made available by existing resources produced by linguistic studies, e.g. FrameNet. Third, the proposed SLU chain in a real robotic platform supports the grounding of commands into the world representation of the robot (i.e., the semantic map). Frames and their arguments can be flexibly mapped into robotic actions and real world objects, respectively.

Through the adoption of general semantic theories, the pipeline can be initialized just reusing existing resources, i.e. the FrameNet labeled corpus. However, specific performance measures for the training of individual SLU stages have been carried out by the development of a HRI-specific annotated spoken corpus, namely HuRIC. As it can be used to train each component, we also measured the improvements achieved, when domain (i.e. HRI) specific training examples are used, by adding the HuRIC evidence to the previous learning settings. Despite the HuRIC having a limited size, the reported results suggest that domain-dependent resources coupled with general ones have a significant beneficial impact on performances, that increase in all different tests. This is an important outcome of the present work, as it shows how the optimization of a new system is made possible even when limited in-domain resources are made available.

The SLU chain proposed in this work opens the way for a variety of research directions. First, it will be used to explore more expressive forms of integration between the linguistic processing level and other cognitive dimensions, such as the perception of the environment or planning. This, in fact, is crucial for a richer and more natural interaction with human users. In this direction, the contribution of more expressive grounding functions (as proposed in the work by Bastianelli et al. (2015) or Misra et al. (2016)) will be investigated.

The grounding of lexical symbols (e.g. arguments) or actions in the perceived environment has been shown to improve the accuracy of the interpretation chain. It makes language understanding systematically consistent with objects and their position in the environment, where the interaction takes place. Second, the role of spatial information, as this can be made available by the robot's semantic map, will be investigated as a bias towards suitable command interpretation and grounding. The machine learning perspective proposed here will allow the adoption of a strictly integrated approach to SLU, whereas the perception acts directly *during* the language learning (through features used to constrain the interpretation).

Finally, new frames will be investigated, among the large repository made available by FrameNet. Although the study presented here focused on a language of imperative commands, FrameNet is representative of a much larger set of semantic phenomena. It is thus able to express other aspects of human-robot interactions, such as belief updates, communication actions as well as recognition and categorization or world entities and properties.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work has been partially funded by the RoCKIn project (http://rockinrobotchallenge.eu) of the European Union VII Framework, Grant Agreement Number 601012.

Notes

1. Notice that a higher absolute value of $f(\cdot,\cdot)$ corresponds to a higher confidence in the classification result.
2. `http://nlp.stanford.edu/software/corenlp.shtml`
3. http://framenet.icsi.berkeley.edu/fndrupal/
4. http://www.coli.uni-saarland.de/projects/salsa/shal/
5. http://nlp.cs.lth.se/software
6. http://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html
7. Available at: http://sag.art.uniroma2.it/huric
8. http://labrococo.dis.uniroma1.it/?q=s4r
9. http://rockinrobotchallenge.eu/
10. Notice that the non-gold setting for the FP is not provided as it is the first step in the considered chain.
11. In the GG and S4R datasets only 48 and 96 sentences for 16 frames are respectively given.
12. sers.rcn.com/mclaughl.dnai/index.htm
13. http://www.turtlebot.com/
14. http://rmp.segway.com/
15. http://www.dis.uniroma1.it/∼spqr/MARRtino

References

Altun Y, Tsochantaridis I and Hofmann T (2003) Hidden Markov support vector machines. In: Tom F and Nina M (eds) *Proceedings of the international conference on machine learning*

(ICML), Washington D.C., USA, 21–24 August, pp.3–10. Palo Alto, CA, USA: AAAI Press.

Baker CF, Fillmore CJ and Lowe JB (1998) The Berkeley FrameNet project. In: *Proceedings of ACL and COLING*, Montreal, Quebec, Canada, 10–14 August 1998, pp.86–90. San Francisco, CA, USA (until 2008). Burlington, MA, USA (actual): Morgan Kaufmann Publishers / ACL 1998.

Baroni M, Bernardini S, Ferraresi A, et al. (2009) The wacky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resource and Evaluation (LRE)* 43(3): 209–226.

Bastianelli E, Castellucci G, Croce D, et al. (2014a) Effective and robust natural language understanding for human-robot interaction. In: *Proceedings of 21st European Conference on Artificial Intelligence*, Prague, Czech Republic, 18–22 August 2014, pp.57–62. Amsterdam, The Netherlands: IOS Press.

Bastianelli E, Castellucci G, Croce D, et al. (2014b) HuRIC: A human robot interaction corpus. In: *Proceedings of the 9th edition of the language resources and evaluation conference*, Reykjavik, Iceland, 26–31 May 2014, pp.4519–4526. Paris, France.

Bastianelli E, Croce D, Basili R, et al. (2015) Using semantic maps for robust natural language interaction with robots. In: *INTERSPEECH 2015, 16th annual conference of the international speech communication association*, Dresden, Germany, 6–10 September 2015, pp.1393–1397. Baixas, France: International Speech Communication Association (ISCA).

Bos J (2002) Compilation of unification grammars with compositional semantics to speech recognition packages. In: *Proceedings of the 19th international conference on computational linguistics – vol. 1, COLING '02*, Taipei, Taiwan, pp.1–7. Stroudsburg, PA, USA: Association for Computational Linguistics.

Bos J and Oka T (2007) A spoken language interface with a mobile robot. *Artificial Life and Robotics* 11(1): 42–47.

Carreras X and Màrquez L (2005) Introduction to the CoNLL-2005 shared task: Semantic role labeling. In: *Proceedings of CoNLL-2005*, 29–30 June 2005, Ann Arbor, Michigan, pp.152–164. Stroudsburg, PA, USA: Association for Computational Linguistics.

Ciprian C, Xu P, Pereira F, et al. In: *IEEE transactions on audio, speech and language processing*, vol. 21, pp. 1158–1169. Piscataway, NJ, USA: IEEE Press.

Chen DL and Mooney RJ (2011) Learning to interpret natural language navigation instructions from observations. In: *Proceedings of the 25th AAAI conference on AI*, San Francisco, California, USA, 07–11 August 2011, pp.859–865. Palo Alto, CA, USA: AAAI Press.

Chen Y, Wang WY and Rudnicky AI (2013) Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing. In: *2013 IEEE Workshop on automatic speech recognition and understanding*, Olomouc, Czech Republic, 8–12 December 2013, pp.120–125. Piscataway, NJ, USA: IEEE Press.

Chen Y, Wang WY and Rudnicky AI (2014) Leveraging frame semantics and distributional semantics for unsupervised semantic slot induction in spoken dialogue systems. In: *2014 IEEE spoken language technology workshop, SLT*, South Lake Tahoe, NV, USA, 7–10 December 2014, pp.584–589. Piscataway, NJ, USA: IEEE Press.

Cocorobo (2013) Sharp. Available at: http://www.sharp.co.jp/cocorobo/ (accessed 11 October 2015).

Collins M and Duffy N (2002) Convolution kernels for natural language. In: Dietterich T, Becker S and Ghahramani Z (eds) *Advances in Neural Information Processing Systems 14*, pp.625–632. London, England, UK: MIT Press.

Coppola B, Moschitti A, Tonelli S, et al. (2008) Automatic FrameNet-based annotation of conversational speech. In: *Proceedings of IEEE-SLT 2008*, Goa, India, 15–19 December 2008, pp.73–76. Piscataway, NJ, USA: IEEE Press.

Cortes C and Vapnik V (1995) Support-vector networks. *Machine Learning* 20(3): 273–297.

Cristianini N, Shawe-Taylor J and Lodhi H (2002) Latent semantic kernels. *Journal of Intelligent Information Systems* 18(2-3): 127–152.

Croce D, Basili R, Moschitti A, et al. (2012a) Verb classification using distributional similarity in syntactic and semantic structures. In: *Proceedings of the 50th annual meeting of the association for computational linguistics: Long papers - vol. 1, ACL '12*, Jeju, Jeju Island, South Korea, 8–14 July 2012, pp.263–272. Stroudsburg, PA: Association for Computational Linguistics.

Croce D, Castellucci G and Bastianelli E (2012b) Structured learning for semantic role labeling. *Intelligenza Artificiale* 6(2): 163–176.

Croce D, Moschitti A and Basili R (2011) Structured lexical similarity via convolution kernels on dependency trees. In: *Proceedings of the conference on empirical methods in natural language processing, EMNLP '11*, 27–31 July 2011, Edinburgh, Scotland, UK, pp.1034–1046. Stroudsburg, PA, USA: Association for Computational Linguistics.

Cuayáhuitl H, van Otterlo M, Dethlefs N, et al. (2013) Machine learning for interactive systems and robots: A brief introduction. In: *Proceedings of the 2nd workshop on machine learning for interactive systems: Bridging the gap between perception, action and communication, MLIS '13*, Beijing, China, 3–4 August 2013, Beijing, China, 3–4 August 2013, pp.19–28. New York, USA: ACM.

de Mori R (2007) Spoken language understanding: A survey. In: *IEEE workshop on automatic speech recognition & understanding, ASRU 2007*, Kyoto, Japan, 9–13 December 2007, pp.365–376. Piscataway, NJ, USA: IEEE.

Dinarelli M, Moschitti A and Riccardi G (2011) Discriminative reranking for spoken language understanding. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*. 20: 526–539.

Dukes K (2013) Train robots: A dataset for natural language human-robot spatial interaction through verbal commands. In: *ICSR. Embodied communication of goals and intentions workshop*. Bristol, England, UK, 27–29 October 2013, pp. 26–31. Turin, Italy: Istituto Italiano di Tecnologia.

Erk K and Pado S (2006) Shalmaneser - a flexible toolbox for semantic role assignment. In: *Proceedings of LREC 2006*, Genoa, Italy, 22–28 May 2006, pp.527–532. Paris, France: ELRA.

Fasola J and Mataric MJ (2013a) Using semantic fields to model dynamic spatial relations in a robot architecture for natural language instruction of service robots. In: *International conference on intelligent robots and systems (IROS), 2013*, Tokyo, Japan, 3–7 November 2013, pp.143–150. Piscataway, NJ, USA: IEEE/RSJ.

Fasola J and Mataric MJ (2013b) Using spatial semantic and pragmatic fields to interpret natural language pick-and-place instructions for a mobile service robot. In: *Social robotics:*

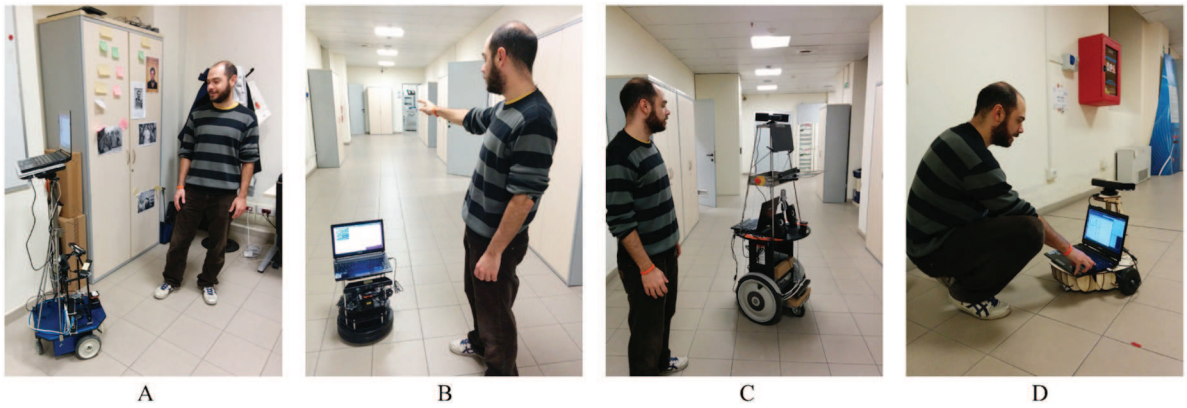

Fig. 6. The four platforms used to test the complete processing chain.

All of the robotic platforms used in the tests can perform a set of predefined PNPs derived from the set selected during the 'Speaky For Robots' project, hence they are related to tasks that a domestic robot should perform, e.g. moving, bringing and checking. This set of actions has been selected without considering any specific robotic platform. Thus, their implementation on the robot will depend on its ability to move autonomously in the space or to interact with objects (e.g. grasping or pushing). Since all of these platforms are without grippers, we simulated the interaction with objects as a request from the robot, asking the user to operate the action on its behalf, e.g. asking to put or remove objects from the loading tray, as it is possible to notice in the video in Extension 1.

In addition to the real demonstration, a preliminary evaluation of the whole system has been carried out, considering the end-to-end process starting from the audio input and ending in the action performed by the robot. We used a simulated environment to run the experiment. This choice has also been mandatory as the HuRIC is modeled on a house domain, so we had to reproduce a similar scenario, and this has been possible only using a simulated house. Figures 7 and 8 show the metric map and the associated semantic map, here split into rooms and furniture, used during the simulation. Table 10 reports the objects present in the house with their position, encoded in the semantic map as well. Furthermore, the semantic map also contains some facts about basic static spatial relations between objects and places, e.g. 'the table of the kitchen' or 'the book on the bed'.

The simulated platform has been modeled exactly as a Videre design erratic, by reproducing its exact size, and has been equipped with a Hokuyo laser range finder used for navigation purposes. The software, e.g. the Petri-net plans, discussed in Section 3.3 and used to model the robot behavior, has been developed in the Robot Operating System (ROS), and it relies on `move_base` as the path planner and the Adaptive Monte Carlo Localization (AMCL) localizer. The test has been run within the ROS stage simulator with the aforementioned map. The simulator has been tuned to reproduce also odometry errors and other noise coming from the perception sensors, e.g. the laser range finders, in order to take into account potential issues arising in a real application.

Table 10. Location of the objects in the house.

Object	#	Location
Book	3	Living_room:couch, Bedroom:bed, Studio:book_shelf
Jar	2	Kitchen:fridge, Living_room:table
Newspaper	1	Bedroom:nightstand
Apple	2	Kitchen:fridge, Living_room:table
Bottle	2	Kitchen:fridge, Studio:table
Cigarettes	1	Studio:table
Knife	1	Kitchen:sink
Jacket	2	Bedroom:cabinet, Living_room:cabinet
Phone	1	Living_room:table
Mobile phone	1	Bedroom:bed
Box	1	Living_room
Tv set	2	Living_room, Kitchen:counter
Pillow	1	Bedroom:bed

Fig. 7. Map of the house: Sampling of the space representing the rooms.

Palmer M, Gildea D and Xue N (2010) Semantic role labeling. *Synthesis Lectures on Human Language Technologies* 3(1): 1–103.

Pepper (2014) Aldebaran. Available at: http://www.aldebaran-robotics.com/(accessed 11 October 2015).

Perera V and Veloso MM (2015) Handling complex commands as service robot task requests. In: *Proceedings of the twenty-fourth international joint conference on artificial intelligence, IJCAI 2015*, Buenos Aires, Argentina, 25–31 July 2015, pp.1177–1183. Palo Alto, CA, USA: AAAI Press.

Popović M and Ney H (2007) Word error rates: Decomposition over pos classes and applications for error analysis. In: *Proceedings of the second workshop on statistical machine translation, StatMT '07*, Prague, Czech Republic, 23–30 June 2007, pp.48–55. Stroudsburg, PA, USA: Association for Computational Linguistics.

Qbo (2012) The corpora robot company. Available at: http://thecorpora.com (accessed 11 October 2015).

Sahlgren M (2006) *The word-space model*. PhD Thesis, Stockholm University, Sweden.

Scheutz M, Cantrell R and Schermerhorn PW (2011) Toward humanlike task-based dialogue processing for human robot interaction. *AI Magazine* 32(4): 77–84.

Shawe-Taylor J and Cristianini N (2004) *Kernel Methods for Pattern Analysis*. Cambridge, England, UK, Cambridge University Press.

Shen L and Joshi AK (2003) An SVM based voting algorithm with application to parse reranking. In: *Proceedings of HLT-NAACL 2003 - vol. 4, CONLL '03*, Edmonton, AB, Canada, 27 May-1 June 2003, pp.9–16. Stroudsburg, PA: ACL.

Tellex S, Kollar T, Dickerson S, et al. (2011a) Approaching the symbol grounding problem with probabilistic graphical models. *AI Magazine* 34(4): 64–76.

Tellex S, Kollar T, Dickerson S, et al. (2011b) Understanding natural language commands for robotic navigation and mobile manipulation. In: Burgard W and Roth D (eds) *AAAI*. Palo Alto, CA, USA: AAAI Press, pp.1507–1514.

Thomas BJ and Jenkins OC (2012) RoboFrameNet: Verb-centric semantics for actions in robot middleware. In: *2012 IEEE international conference on robotics and automation (ICRA)*, St Paul, MN, USA, 14–18 May 2012, pp.4750–4755. Piscataway, NJ, USA: IEEE.

Tjong Kim Sang EF and Buchholz S (2000) Introduction to the CoNLL-2000 shared task: Chunking. In: *Proceedings of the 2nd workshop on learning language in logic and the 4th conference on computational natural language learning - vol. 7*, Lisbon, Portugal, 13–14 September 2000, ConLL '00, pp.127–132. Stroudsburg, PA, USA: Association for Computational Linguistics.

Toutanova K, Haghighi A and Manning CD (2008) A global joint model for semantic role labeling. *Computational Linguistics* 34(2): 161–191.

Tur G, Tur DH and Chotimongkol A (2005) Semi-supervised learning for spoken language understanding using semantic role labeling. In: *Proceedings of the IEEE workshop on automatic speech recognition and understanding*, San Juan, PR, USA, 27 November-1 December 2005, pp.232–237. Piscataway, NJ, USA: IEEE Press.

Walter MR, Hemachandra S, Homberg B, et al. (2013) Learning semantic maps from natural language descriptions. In: *Proceedings of robotics: Science and systems (RSS)*, Berlin, Germany.

Ziparo VA, Iocchi L, Nardi D, et l. (2008) Petri net plans: A formal model for representation and execution of multi-robot plans. In: *Proceedings of the 7th international joint conference on autonomous agents and multiagent systems - Vol.1, AAMAS '08*, Estoril, Portugal, 12–16 May 2008, pp.79–86. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Zlatev J (2007) Spatial semantics. In: *The Oxford Handbook of Cognitive Linguistics*. pp.318–350. Oxford, England, UK: Oxford University Press.

Appendix: Index to multimedia extension

Archives of IJRR multimedia extensions published prior to 2014 can be found at http://www.ijrr.org, after 2014 all videos are available on the IJRR YouTube channel at http://www.youtube.com/user/ijrrmultimedia

Extensions	Media type	Description
1	Video	Video showing the complete chain at work on a Turtlebot