Available online at www.sciencedirect.com

ScienceDirect

Computer Speech & Language 47 (2018) 272–297

www.elsevier.com/locate/csl

Speech understanding for spoken dialogue systems: From corpus harvesting to grammar rule induction☆

Elias Iosif[a,b,*], Ioannis Klasinas[c], Georgia Athanasopoulou[c], Elisavet Palogiannidi[c], Spiros Georgiladakis[c], Katerina Louka[d], Alexandros Potamianos[a,b]

[a] *School of Electrical and Computer Engineering, National Technical University of Athens, 15780 Athens, Greece*
[b] *"Athena" – Research and Innovation Center in Information, Communication and Knowledge Technologies, 15125 Athens, Greece*
[c] *School of Electronic and Computer Engineering, Technical University of Crete, 73100 Chania, Greece*
[d] *VoiceWeb S.A., 15124 Athens, Greece*

Received 9 September 2016; received in revised form 16 March 2017; accepted 15 August 2017
Available online 25 August 2017

Abstract

We investigate algorithms and tools for the semi-automatic authoring of grammars for spoken dialogue systems (SDS) proposing a framework that spans from corpora creation to grammar induction algorithms. A realistic human-in-the-loop approach is followed balancing automation and human intervention to optimize cost to performance ratio for grammar development. Web harvesting is the main approach investigated for eliciting spoken dialogue textual data, while crowdsourcing is also proposed as an alternative method. Several techniques are presented for constructing web queries and filtering the acquired corpora. We also investigate how the harvested corpora can be used for the automatic and semi-automatic (human-in-the-loop) induction of grammar rules. SDS grammar rules and induction algorithms are grouped into two types, namely, low- and high-level. Two families of algorithms are investigated for rule induction: one based on semantic similarity and distributional semantic models, and the other using more traditional statistical modeling approaches (e.g., slot-filling algorithms using Conditional Random Fields). Evaluation results are presented for two domains and languages. High-level induction precision scores up to 60% are obtained. Results advocate the portability of the proposed features and algorithms across languages and domains.

Keywords: Spoken dialogue systems; Grammar induction; Corpora creation; Semantic similarity; Web mining; Crowdsourcing

1. Introduction

Natural language understanding (NLU) is at the very heart of spoken dialogue systems (SDS) since its purpose is to transform the output of the speech recognizer into a semantic representation. Such representations are useful for other related tasks, e.g., the identification of speaker intention that drive the module of dialogue management. For example, consider an SDS for air tickets booking and the following example utterance: "I am leaving from Chicago".

☆ This paper has been recommended for acceptance by Roger Moore.
* School of Electrical and Computer Engineering, National Technical University of Athens, 15780 Athens, Greece.
E-mail addresses: iosif.elias@gmail.com, iosife@central.ntua.gr (E. Iosif).

http://dx.doi.org/10.1016/j.csl.2017.08.002
0885-2308/

The salient part of this utterance is the lexical fragment "leaving from Chicago" that can be regarded as an instance of a grammar rule denoted as < DepartureCity > . Such grammar rules enable the understanding of the user input, e.g., the system can infer that 'Chicago is the departing city, and then proceed to other dialogue states for gathering any missing information, such as destination and travel dates. SDS grammars constitute a linguistic formalism that serves as the middleware between the recognized speech and the semantic representation. Speech understanding grammars can be distinguished into two broad categories, namely, finite-state-based (FSM) and statistical. Initial efforts in speech understanding grammar modeling were based on rule-based systems (e.g., Wang, 2001) suffering from poor generalizability and relying on manual updates (Pieraccini and Suendermann, 2012). Better results can be obtained using finite-state-based grammars (Potamianos and Kuo, 2000; Raymond et al., 2006), which enable the integration of automatic speech recognition output with NLU. More recent efforts rely on discriminative models such as Support Vector Machines (SVM) (Vapnik, 1998) and Conditional Random Fields (CRF) (Lafferty et al., 2001) and have been shown to outperform finite-state-based approaches (Raymond and Riccardi, 2007). Lately, top performance has been achieved by Recurrent Neural Networks (RNN) (Mesnil et al., 2015). The manual development of grammars poses an obstacle to the rapid porting of spoken dialogue systems to new domains and languages. The need for machine-assisted grammar induction has been an open research area for decades (Lari and Young, 1990; Chen, 1995) aiming to lower this barrier. Automatic (or semi-automatic) induction algorithms can be distinguished into two main categories, namely, resource-based and data-driven. The main drawback of resource-based approaches is the dependency on knowledge bases, which might not be available for under-resourced languages. This is tackled by the data-driven paradigm that relies (mostly) on corpora.

In this paper, we adopt a data-driven paradigm investigating various algorithms for the creation of text corpora and the induction of finite-state-based grammars. The end goal is to help automate the grammar development process. Unlike previous approaches (Wang and Acero, 2006; Cramer, 2007) that have focused on full automation, we adopt a human-in-the-loop approach where a developer bootstraps each grammar rule or request type with a few examples (seeds) and then machine learning algorithms are used to propose grammar rule enhancements to the developer. The enhancements are post-edited by the developer and new grammar rule suggestions are proposed by the system in an iterative fashion, until a grammar of sufficient quality is achieved. The main approach used for corpora creation is the harvesting of web data via the formulation of web search queries, followed by corpus filtering. The richness of the world wide web and its multilingual character enable the creation of corpora for less-resourced languages and domains. Note that the exploitation of web data is also appropriate for the development of statistical grammars where large amounts of data are required. In addition, various crowdsourcing tasks are used in order to elicit spoken dialogue text data. SDS grammar rules are distinguished into two types, namely, low- and high-level. Low-level rules refer to terminal concepts, e.g., the concept of city name can be represented as < City > → ("New York"|"Boston"). High-level rules are defined on top[1] of low-level rules, e.g., < DepartureCity > → ("fly from < City > "|"departing from < City > "). Two different families of language-agnostic induction algorithms are proposed, one for each type of rules. Greater focus is given to the induction of high-level rules, for which different approaches are proposed exploiting a rich set of features.

This work builds upon our prior research in Klasinas et al. (2013); Georgiladakis et al. (2014); Athanasopoulou et al. (2014); Palogiannidi et al. (2014), adding the following original contributions:

1. Regarding the harvesting of web data for corpora creation, two types of query generation (corpus- and grammar-based) are investigated, extending the work in Klasinas et al. (2013) where only the grammar-based approach was followed. In addition, here, more techniques for corpus filtering are proposed and compared. Detailed experimental results demonstrate that web harvesting is a viable approach for creating corpora intended for grammar induction.
2. In this work, we investigate the induction of both low- and high-level rules. Emphasis is given on the induction of high-level rules, a less researched area, unlike previous studies (Klasinas et al., 2013; Palogiannidi et al., 2014) that dealt only with low-level rules. We show that different similarity metrics and features are appropriate for the induction of low- and high-level rules. In total, four different approaches are proposed and compared for the high-level rule induction, extending the preliminary work in Athanasopoulou et al. (2014).

[1] High-level rules can be also stacked on top of each other, e.g., < DepartureArrivalCity > defined on top of < ArrivalCity > and < DepartureCity > .

3. The portability of the aforementioned approaches and algorithms is verified with respect to two different domains and two languages.
4. A slot-filling statistical approach is investigated for inducing high-level rules and compared with the similarity-based approaches.

The proposed approach for grammar induction is motivated by earlier efforts for low-level rule induction conducted in the framework of Bell Labs Communicator system (Pargellis et al., 2001, 2004). An overall evaluation of this system is presented in Sungbok et al. (2002) based on various dialogue metrics and user satisfaction statistics. In the present work, we adopt the basic idea of Pargellis et al. (2001, 2004) regarding low-level induction, and in addition we investigate features of lexical and semantic similarity for inducing high-level rules. The output of the algorithms considered in this work is exploited for the creation of FSM-based grammars.

The remainder of the paper is organized as follows: In Section 2, we review related work in the areas of corpora creation and grammar induction for SDS. In Section 3, an overview of the proposed approach is given that spans from the creation of corpora to the induction of low- and high-level rules. The two different approaches for corpora creation, namely, web harvesting and crowdsourcing are presented in Section 4. The induction of low-level rules is described in Section 5, while in Section 6 high-level rule induction is presented. Experiments along with the evaluation results are presented in Section 7. Section 8 concludes this work.

2. Related work

Automatic or machine-aided grammar creation for SDS can be broadly divided into two categories (Wang and Acero, 2006): knowledge-based (or top-down) and data-driven (or bottom-up) approaches.

Knowledge-based algorithms rely on domain-specific grammars or lexica. Various sources of domain knowledge are available nowadays in the form of ontologies; such knowledge is increasingly being exploited in dialogue systems (Milward and Beveridge, 2003; Pardal, 2007). In addition, research on ontology lexica (Prévot et al., 2010; McCrae et al., 2012) explores how such domain knowledge can be connected with rich linguistic information. Grammars that are generated from ontology lexica often achieve high precision but suffer from limited coverage. In order to improve coverage, regular expressions and word/phrase order permutations are used, however often at the cost of overgeneralization. Moreover, knowledge-based grammars are costly to create and maintain, as they require domain and engineering expertise, and they are not easily portable to new domains. This led to the development of grammar authoring tools facilitating the creation and adaptation of grammars. One such tool is SGStudio (Semantic Grammar Studio) (Wang and Acero, 2006) that enables (1) example-based grammar learning, (2) grammar controls, i.e., building blocks and operators for building more complex grammar fragments (regular expressions, lists of concepts), and (3) configurable grammar structures, allowing for domain-adaptation and word-spotting grammars. A popular grammar authoring environment for commercial applications is NuGram (NuGram Platform, 0000), however it does not support automatic grammar creation. The Grammatical Framework Resource Grammar Library (GFRGL) (Ranta, 2004) enables the creation of multilingual grammars adopting an abstraction formalism that hides the linguistic details (e.g., morphology) from the grammar developer.

Data-driven (bottom-up) approaches rely solely on corpora of transcribed utterances (Meng and Siu, 2002; Pargellis et al., 2004). The induction of low-level rules consists of two steps: (1) identification of terms (term extraction, named-entity recognition (NER)), and (2) assignment of terms into rules. Standard tokenization techniques can be used for the first step. For multiword terms, e.g., “New York”, gazetteer lookup and NER can be employed (if the respective resources and tools are available), as well as corpus-based collocation metrics (Frantzi and Ananiadou, 1997). Typically, the identified terms are assigned into low-level rules via clustering algorithms using a semantic similarity metric. The distributional hypothesis of meaning (Harris, 1954) is a widely-used approach for estimating term similarity. A comparative study of similarity metrics for the induction of SDS low-level rules is presented in Pargellis et al. (2004), while the combination of metrics was investigated in Iosif et al. (2006). Different clustering algorithms have been applied, including hard- (Meng and Siu, 2002) and soft-decision (Iosif and Potamianos, 2007) agglomerative clustering.

High-level rule induction is a less researched problem that consists of two steps similar to low-level rule induction: (1) the extraction and selection of candidate fragments from a corpus, and (2) the assignment of terms into rules. Regarding the first sub-problem, consider the fragments “I want to depart from $<$City$>$ on” and “depart

from < City > " for the air travel domain. Both express the meaning of departure city, however, the semantics of the latter fragment are more concise and generalize better. Semantic similarity and distributional semantic models (DSMs) can be employed for inducing such semantic classes as for the case of low-level rules (Meng and Siu, 2002; Pargellis et al., 2004). The recent advances of DSMs in the area of compositional semantics (e.g., Marelli et al., 2014) can be applied for estimating the similarity between larger textual chunks, such as the typical high-level rule, which is a harder task compared to the word-level similarity computation. An alternative approach is statistical semantic parsing technology (slot-filling). Semantic parsing refers to the mapping of a natural language sentence to a semantic representation. Several models have been used such as finite state transducers (Raymond and Riccardi, 2007), SVM (Pradhan et al., 2004), hidden Markov Models (HMM), and CRF (Sha and Pereira, 2003; Raymond and Riccardi, 2007; Heck et al., 2013). In this framework, a statistical model is built for each slot through the training of classifiers, while the understanding of recognized utterances is cast as a slot-filling problem. In Mairesse et al. (2009), SVMs were used for the semantic parsing of spoken language using as training data a set of utterances and the respective semantic trees. The basic units of such trees are category–value tuples, such as Food → Chinese. For each tuple type a binary classifier was trained using *n*-gram frequency counts as features that were extracted from the corresponding utterances. SVM were also applied to the problem of dialogue act classification. In Liu et al. (2012), CRFs were employed for segmenting a transcribed spoken language query and assigning semantic labels to the identified segments. This was performed in the context of speech-enabled search interface for movie databases, where segments such as "funny" can be assigned the "Genre" label. CRFs features were extracted from fields such as the movie titles and summaries, the list of actors, etc. An experimental comparison between CRFs and RNNs is provided in Mesnil et al. (2015) for the task of slot-filling with respect to three domains including the ATIS domain. For ATIS, RNNs were found to improve the CRF-based performance by 2% in terms of absolute error reduction. The comparison of several RNNs-based approaches for the task of slot-filling for the ATIS domain can be found in Shi et al. (2016). In Jurčíček et al. (2009), the Transformation-Based Learning proposed by Brill (1995) was adapted for the task of semantic parsing. The key idea was the learning of a set of transformation rules, e.g., an *n*-gram is transformed (mapped) to a semantic category. The adapted algorithm was applied over two different corpora of spoken language, having as prerequisite the availability of semantic categories such as city and airport names. Overall, the aforementioned approaches are closely related to a series of open research issues spanning from the compositional aspects of lexical semantics (Mitchell and Lapata, 2010; Agirre et al., 2012) to unsupervised parsing (Ponvert et al., 2011; Beltagy et al., 2014).

The main challenge for data-driven approaches is data sparseness, which may affect the coverage of the grammar. A popular solution to the data sparseness bottleneck is to harvest relevant data from the web. Recently, this has been an active research area both for SDS systems and language modeling, in general. Data harvesting is performed in two steps: (1) query formulation, and (2) selection (filtering) of relevant documents or sentences (Klasinas et al., 2013). Posing the appropriate queries is important both for obtaining in-domain and linguistically diverse sentences. In Sethy et al. (2002), an in-domain language model was used to identify the most appropriate n-grams to use as web queries. An in-domain language model was used in Klasinas et al. (2013) for the selection of relevant sentences. A more sophisticated query formulation algorithm was proposed in Sarikaya (2008), where from each in-domain utterance a set of queries of varying length and complexity were generated. These approaches assume the availability of in-domain data (even if in limited amount) for the successful formulation of queries; this is also necessary when using a "mildly" lexicalized domain ontology to formulate the queries, as in Misu and Kawahara (2006). Selecting the most relevant sentences returned from the web queries is typically done using statistical similarity metrics between in-domain data and retrieved documents, for example the BLEU metric (Papineni et al., 2002) of n-gram similarity in Sarikaya (2008) or a metric of relative entropy (Kullback–Leibler) in Sethy et al. (2002). When in-domain data is not available, cf. (Misu and Kawahara, 2006), heuristics (pronouns, sentence length, wh-questions) and matches with out-of-domain language models can be used to identify relevant sentences. in Sarikaya (2008), the produced grammar fragments are also parsed and attached to the domain ontology. Harvesting web data can produce high-quality grammars while requiring up to ten times less in-domain data (Sarikaya, 2008). Crowdsourcing is a popular method for various natural language and speech processing tasks (see Callison-Burch and Dredze, 2010 for a survey). Examples include sentence translation from one language to another or gathering annotations on bilingual lexical entries (Ambati and Vogel, 2010; Irvine and Klementiev, 2010), as well as paraphrasing applications (Denkowski et al., 2010; Buzek et al., 2010). Regarding the field of SDS, crowdsourcing has been exploited mainly for system evaluation purposes (Raux et al., 2005; Yang et al., 2010; Jurcıcek et al., 2011; Zhu et al., 2010).

Additional uses of crowdsourcing include the creation of corpora (Wang et al., 2012; McGraw et al., 2011) used for tasks such as language modeling (McGraw et al., 2011). The elicitation of corpora via crowdsourcing used for grammar induction for SDS seems to be a less explored area.

A fully automated bottom-up paradigm for grammar induction has been shown to result in grammars of moderate quality (Wang and Acero, 2006), especially on corpora containing longer sentences and more lexical variety (Cramer, 2007). Grammar quality can be improved by introducing a human-in-the-loop grammar induction paradigm; an expert that validates the automatically created results (Meng and Siu, 2002). However, most automatic grammar induction algorithms work in a batch mode rather than incrementally, failing to efficiently incorporate human feedback. This semi-automatic framework is consistent with the iterative human-centric process adopted in the industry.

3. System overview

In Fig. 1, an overview of the proposed grammar development system is depicted. The system consists of two main modules: (textual) corpora creation and corpus-based grammar induction. Both modules exploit a seed grammar (indicated by different lines in Fig. 1: solid for corpus creation and dashed for grammar induction) that contains a few rules as examples for bootstrapping the process of data collection and induction. Seed grammar rules can be regarded as specifications of domain semantics. The main concept behind grammar induction is to induce rules that are semantically related to the given seeds. The induction of grammar rules is decomposed into two sub-tasks, namely, induction of low-level and high-level rules.

The primary function of SDS grammars is to represent the domain semantics via a formal encoding of their respective lexicalizations. A data-driven paradigm for grammar induction is an efficient approach given the availability of (qualitatively and quantitatively) sufficient data. Wizard-of-Oz (WoZ) sessions have proven to be an appropriate, yet costly, solution for the collection of domain-specific data. A workaround for addressing the shortcomings of the WoZ paradigm is the automatic harvesting of data using the world wide web as a corpus. In the present work, we exploit this solution as the primary approach for corpora creation, followed by a number of filtering techniques for ensuring the in-domainness of the harvested data. In addition, we investigate the potential of crowdsourcing for eliciting spoken dialogue text data, a little-studied area for SDS grammar induction.

An important aspect of the system is the adoption of an iterative human-in-the-loop framework. After the grammar developer has initiated the induction process by providing the seed grammar, the system induces new rules that are post-edited by the developer. The result of post-editing is merged with the seed grammar, i.e., the initial grammar

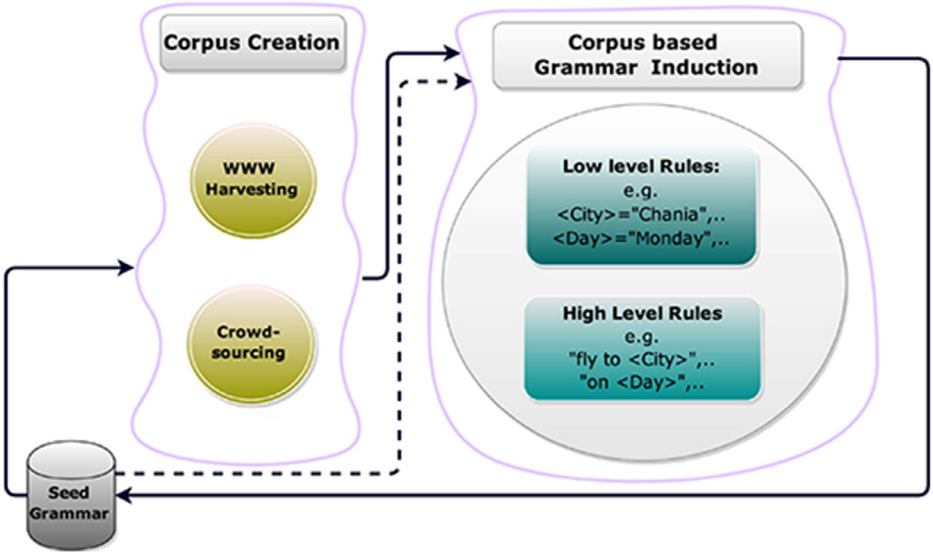

Fig. 1. SDS grammar development system overview.

is enhanced and it can serve as an updated system input. This process can be repeated until a stopping criterion is met, e.g., a grammar of sufficient coverage/quality is achieved. The two modules are fully automatic, nevertheless, their integration with the manual post-edit makes the entire process semi-automatic, in accordance with the cycle of grammar development followed in practice: the grammar developer starts from a basic version of the grammar and incrementally enhances it (often after the system deployment by examining the dialogue logs). In the proposed framework the grammar enhancement is initiated by the system, i.e., new rules are proposed to the developer who is responsible for accepting, rejecting or modifying them.

4. Corpora creation via web harvesting and crowdsourcing

In a typical speech understanding grammar development cycle, the developer starts from user requirements (often expressed as request types or a small corpus) and then encodes this information in a hand-crafted bootstrap grammar. Our goal is starting from this limited-coverage grammar to harvest a corpus using web queries. The end-goal is to enhance the grammar by applying rule induction algorithms over the web-harvested corpus. As far as we know, generating queries from a grammar is a novel idea, although, the method is similar to Sarikaya (2008) where n-gram fragments can be extracted from an already available corpus (also investigated in this paper). A related idea is the harvesting of web search queries instead of web documents. For example, in Tur et al. (2012) harvested search queries were used for building a semantic parser for a movie domain. Web search queries have been also exploited for a series of NLU tasks related to SDS, such as domain Hakkani-Tür et al. (2011, 2012) and intent (Heck and Hakkani-Tür, 2012) detection.

The entire process of corpora creation is illustrated in Fig. 2 consisting of two main steps, namely, query generation and corpus filtering, described in Sections 4.1 and 4.2, respectively. In addition to the harvesting of web data, we investigate the use of crowdsourcing in order to elicit spoken dialogue text data (see Section 4.3).

4.1. Web harvesting: query generation

Two approaches are followed for the generation of web search queries, as follows:

- In the first approach, web queries are extracted from a grammar. Starting from a seed grammar is a more realistic scenario for SDS: the developer typically creates grammars rule by rule and in an incremental way (first a small seed grammar is created and tested and then this grammar is enhanced). If the grammar is small, it might be possible to generate all phrases and feed them to the web search engine. Usually, the size of the grammar prohibits such an exhaustive expansion. Instead, fragments from the grammar itself are created ignoring instantiations of terminal concepts (for example, city names or airline companies) that would increase the number of queries too

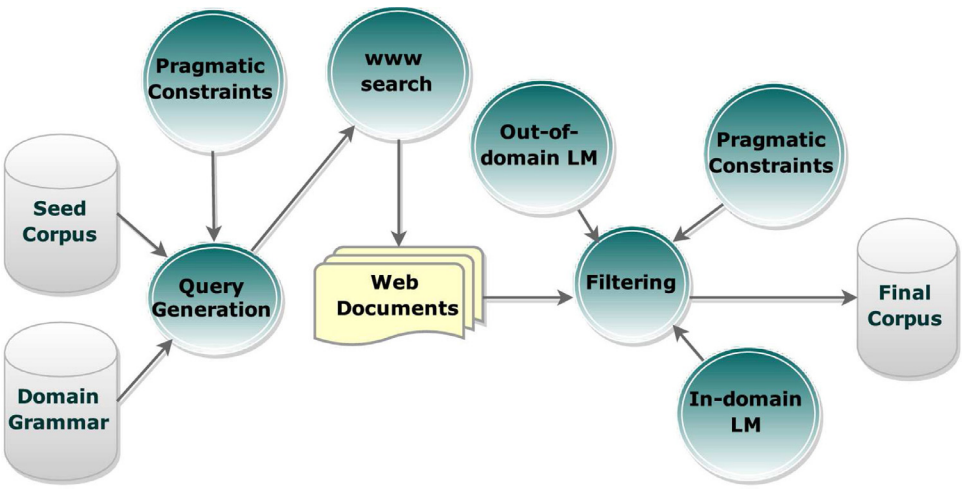

Fig. 2. Corpora creation via web data harvesting.

much. For example, consider the following rule[2] present in the English air travel domain grammar (used in the experiments detailed later in this work): < DepartureCity > → ["depart" | "departing" | "leave" | "leaving" | "left"] ("from" | "between" | "out of") < City >. In this rule, < City > can be replaced with thousands of city names generating tens of thousand of phrases as queries. To overcome this problem, the instances of the terminal rule < City > are ignored, resulting in just 15 queries created for the above rule.

- In the second approach, queries are n-grams extracted from a seed corpus. Not all queries are expected to be equally important; for example, consider the air travel sentence "Tell me the flights leaving from Berlin tomorrow". Both "Tell me" and "flights leaving" are valid queries, however, the first one is a generic English phrase, while the second one describes the domain much better. To estimate the relevance of the query we propose a perplexity-based ranking. The perplexity of a sentence W of length I according to a probability model P is defined as

$$PPL_P(W) = 10^{-\log P(W)/I}. \quad (1)$$

High probability for a given sentence implies that this sentence is similar to the distribution of the model, leading to low perplexity. Query selection is performed as follows: a language model is trained on an out-of-domain corpus and then the perplexity of each query is computed. The queries are then ranked in decreasing order of perplexity and the top ones are kept for web harvesting. The idea is that queries with low perplexity will be generic phrases, while high perplexity queries will be domain-specific phrases (and thus not very common in the out-of-domain corpus).

Query expansion using pragmatic constraints. To further narrow down the retrieved web results, domain-specific pragmatic constraints are manually identified and appended to each query. Such constraints can be regarded as a set of keywords that are related to the domain of interest, e.g., ("airport", "flight") for the air travel domain. For example, the constrained query that corresponds to the aforementioned < DepartureCity > rule is: ("airport" | "flight") ["depart" | "departing" | "leave" | "leaving" | "left"]("from" | "between" | "out of"). We believe that this does not hurt the applicability of the method to different domains/languages, since minimal human intervention is required. These words can also be obtained automatically using an in-domainness metric presented in the next section.

4.2. Web harvesting: corpus filtering

The corpus creation process starts by downloading the top-ranked web documents returned by each query. Then, the content of documents is extracted by removing the HTML tags, as well as embedded code such as JavaScript. Next the most relevant document sentences with respect to the domain of interest are identified (filtered). The corpus is created by aggregating these sentences. We propose two filtering approaches, namely:

Perplexity-based (ppl). Perplexity is a popular criterion (Gao et al., 2002; Bisazza et al., 2010; Ng et al., 2005) for selecting corpora for n-gram language model training. A language model is trained on an in-domain corpus, and the perplexity of each sentence in the downloaded data is estimated. The sentences with the lowest perplexity are selected in order to filter out-of-domain utterances. In previous work (Klasinas et al., 2013), it has been shown that in the absence of an in-domain corpus one can use the downloaded corpus instead.

Filtering using pragmatic constraints (FPC). Pragmatic constraints, i.e., words that have high application domain saliency, can be used in the filtering step, to pick the most informative sentences from the downloaded corpus. Instead of manually selecting such words, we propose to find this set of constraints in an unsupervised way. Generally speaking, highly salient domain words would appear much more frequently in an in-domain (foreground) corpus rather than in a general-purpose (background) corpus. In addition, pragmatic constraints should appear in the majority of the in-domain corpus documents, i.e., will be evenly spread in the foreground corpus. Let $P_{for}(w)$ and $P_{bck}(w)$ be the probability of a word according to the foreground and background model, respectively. The ratio of those probabilities multiplied by the percent of in-domain documents that contain this word, $D(w)$, can provide a good

[2] An augmented Backus–Naur Form (BNF) is used to present rules here, where [.] means zero or one occurrences, (.) stands for one occurrence, and < . > denotes concepts.

criterion for selecting salient words:

$$G(w) = D(w)\frac{P_{for}(w)}{P_{bck}(w)}. \quad (2)$$

If an in-domain corpus is not available, the downloaded corpus is used instead. The metric is computed over the vocabulary of the corpus and the most informative words (i.e., the ones with the highest $G(w)$ value) are selected.

4.3. Corpora creation via crowdsourcing

Here, we summarize various methods (tasks) to elicit spoken dialogue text data via crowdsourcing for grammar induction, which are detailed in Palogiannidi et al. (2014).[3] The main difference with traditional crowdsourcing tasks, e.g., Ambati and Vogel (2010), is the different elicitation methods investigated here. Also, in contrast to Raux et al. (2005); Yang et al. (2010); Jurcıcek et al. (2011); Zhu et al. (2010), the focus is not on evaluating SDS, but on creating a corpus useful for the development of a SDS. In order to elicit realistic SDS data, we designed four crowdsourcing tasks that simulate SDS interaction. Hence, the majority of the tasks follows a *question and answer* structure. Specifically, the following tasks were created: (1) *Answers*: collecting answers from questions (SDS prompts), (2) *Paraphrasing*: collecting paraphrases of an (underlined) portion of a sentence (corresponding to a prompt or user input), (3) *Complete the dialogues*: task contributors must insert suitable answers and questions to incomplete dialogues, and (4) *Fill in*: task contributors must fill in the missing part of a sentence, i.e., complete a sentence. Illustrative examples of the four elicitation methods are shown in Fig. 3 for a travel domain.[4] Empty fields must be filled in by the contributor. Note that the filtering techniques described in Section 4.2 can be also applied to the data collected via crowdsourcing.

5. Induction of low-level rules

In this section, we describe a corpus-based approach for the induction of low-level grammar rules. An example of such a rule is $< \text{City} > \rightarrow$ (“New York”|“Boston”) that encodes the concept of city. In essence, the rule can be regarded as a set of semantically similar textual fragments. The end goal is the automatic induction of such rules, starting from a small number of examples that serve as bootstrapping seeds for each rule. The overall process is depicted in Fig. 4, while the main steps are described below. In this figure, the solid blue lines refer to the main processing modules, while the gray lines relate resources (i.e., raw corpus and seed rules) with such modules. The dashed

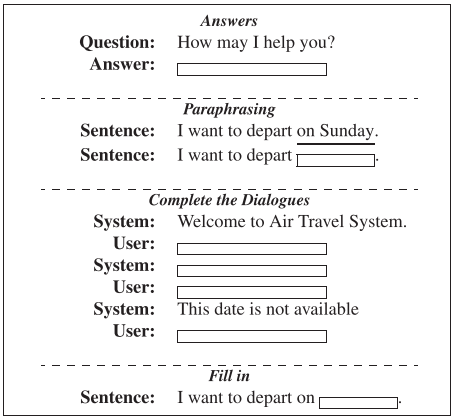

Fig. 3. Examples of the four crowdsourcing tasks used for corpora creation.

[3] The data presented in Palogiannidi et al. (2014) is publicly available at http://www.telecom.tuc.gr/~epalogiannidi/icassp_2014.html.

[4] All tasks were constructed manually, while 85 hits (human intelligence tasks) were used per task type (on average). More details can be found in Palogiannidi et al. (2014).

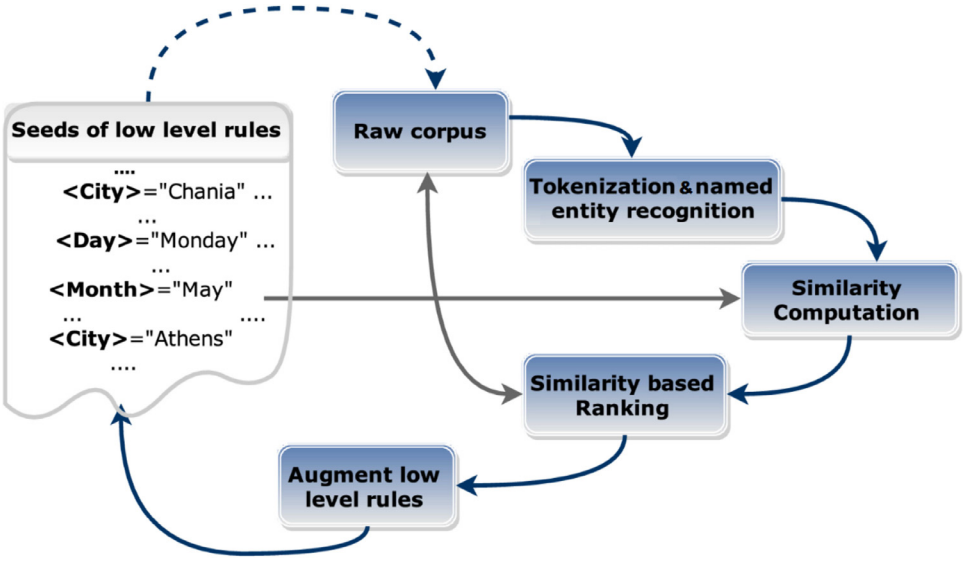

Fig. 4. Induction of low-level rules.

blue line denotes that the corpus instances of low-level rules are substituted by the respective labels (e.g., "May" is substituted by < Month >).

Step 1: Tokenization and named entity recognition. The corpus is tokenized and the multiword terms that correspond to named entities are detected. Such terms are represented as single tokens. For example, the sentence "I want to travel from New York to San Francisco" is transformed to "I want to travel from New−York to San−Francisco".

Step 2: Semantic similarity computation. For a low-level rule, the semantic similarity between seeds and each vocabulary entry (token) is computed. Since more than one seed may be provided for each rule, the similarity between a rule and a token is estimated by averaging the similarities between each of the seeds and the token. The distributional hypothesis of meaning (Harris, 1954), i.e., *similarity of context implies similarity of meaning*, is adopted for the computation of semantic similarity between seeds and tokens. Each word w (seed or token) is considered together with its neighboring words in the left and right contexts: $w_1^L \ w \ w_1^R$. The semantic similarity between two words, w_x and w_y, is estimated as the *Manhattan-norm* (*MN*) of their respective bigram probability distributions of left and right contexts (Pargellis et al., 2004). For example, the left-context *MN* is defined as:

$$MN^L(w_x, w_y) = \sum_{i=1}^{N} |p(w_i^L|w_x) - p(w_i^L|w_y)|, \tag{3}$$

where $V = (w_1, w_2, \ldots, w_N)$ is the corpus vocabulary. Note that $MN^L(w_x, w_y) \equiv MN^L(w_y, w_x)$. The semantic similarity between w_x and w_y is estimated as the sum of the left- and right-context *MN*, i.e., $MN(w_x, w_y) = MN^L(w_x, w_y) + MN^R(w_x, w_y)$.

Step 3: Rule augmentation. For each grammar rule, the tokens are ranked according to their respective semantic similarity, while the top-ranked tokens are used for augmenting (enhancing) the rule. For example, assume that "New York" and "Boston" are used as seeds for the rule < City >, while "Atlanta", and "Toronto" are found to be the two most similar tokens to the seeds. < City > is enhanced as < City > → ("New York"|"Boston"|"Atlanta"|"Toronto").

The process is iterative and Steps 1, 2, 3 are repeated until the desired number of fragments is acquired for each rule. It is also possible to incorporate a human in the induction loop for examining (accept/reject) the decisions of Step 3.

6. Induction of high-level rules

In this section, we present two approaches for inducing high-level rules. The first approach (detailed in Section 6.1) utilizes a rich set of textual features including phrase semantic similarity. For the second approach

(described in Section 6.2), the induction task boils down to a slot-filling problem using statistical models. A simple fusion of the aforementioned approaches is presented in Section 6.3.

6.1. Induction based on semantic similarity

This is a lightly supervised human-in-the-loop module for corpus-based grammar induction. The key idea is that a developer provides a minimal set of examples (typically two to three) for a grammar rule and then the system automatically suggests a set of fragments for enhancing each grammar rule (as for low-level rule induction in Section 5). Our focus is on high-level rules that sit higher in the domain ontology and typically span two to five words. At the core of this module is an algorithm for the *selection* of lexical fragments (n-gram chunks) from a corpus that convey relevant semantic information in an unambiguous and concise manner. For example, consider the fragments "I want to depart from < City > on" and "depart from < City >" for the air travel domain. Both express the meaning of departure city, however, the (semantics of the) latter fragment are more concise and generalize better. Rule-based and statistical approaches are proposed for the fragment selection problem, which are described in Sections 6.1.2 and 6.1.3, respectively. The fragment selection is then combined with a phrase-level semantic similarity metric in order to induce a new set of grammar rules. The overview of the module in Fig. 5 shows the three main phases described also below.

Phase I: Induction of low-level rules. Using the algorithms described in Section 5, low-level rules, such as < City > and < Day >, are induced, and subsequently their corpus instances are substituted by the respective label, e.g., the word "Chicago" is substituted by < City >.

Phase II: Fragment extraction. This component extracts all candidate phrase fragments from the corpus. All n-grams, that contain low-level rules, are extracted, with n ranging between two and five. For example, candidate

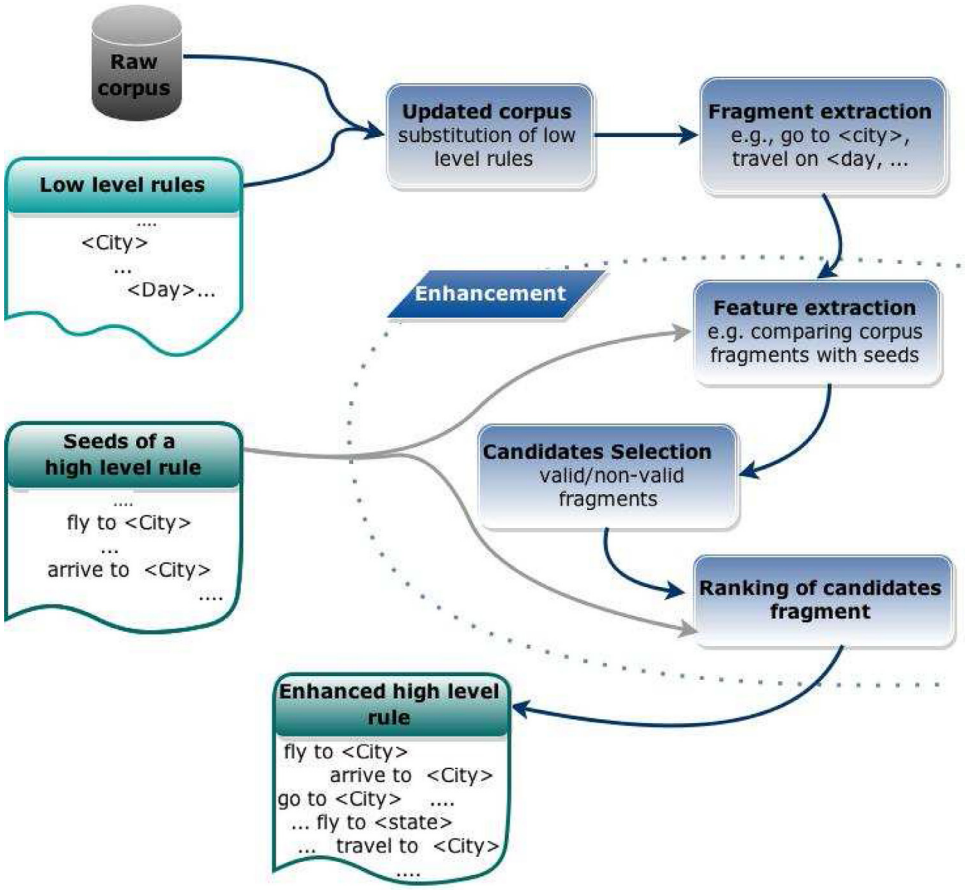

Fig. 5. Induction of high-level rules.

Table 1
Example of ranking the selected fragments during the enhancement of the high-level rule < DepartureCity >.

Rule	Unknown fragment f	$P(r_s\|f)$	$S(f, r_s)$	Total score ($k = 0.8$)
< DepartureCity >	"arrive at < City >"	0.44	**0.57**	0.47
	"depart < City >"	**0.97**	0.48	**0.87**
	"stop at < City >"	0.53	0.52	0.53

bigrams and trigrams for the sentence "arrive to < City > tomorrow" include: {"to < City >", "< City > tomorrow", "arrive to < City >", "to < City > tomorrow"}. Let L denote the set of fragments extracted from this phase.

Phase III: Grammar enhancement. This is the most critical phase dealing with the induction of high-level rules that consists of two steps. It is depicted by the *Enhancement* box in Fig. 5. Let r denote a grammar rule and $\mathcal{F}_r = \{f_{r1}, \ldots, f_{r|\mathcal{F}_r|}\}$ to the set of seed fragments for rule r provided by the developer (typically $|\mathcal{F}_r| = 2$ or 3). We compute two scores for each fragment $f_i \in L$, $i = 1, \ldots, |L|$:

(1) the similarity score between rule r and fragment f_i, $S(r, f_i)$ that is computed as the average similarity (based on Levenshtein distance) between the seed fragments of r, $\mathcal{F}_r$, and the f_i fragment",
(2) the posterior probability that fragment f_i is a good candidate for enhancing grammar rule r, $P(r|f_i, \mathcal{F}_r)$.

Given these two measurements, the two enhancement steps are:

Enhancement-Step 1: Fragment selection. Select fragments from L by setting a threshold θ on $P(r|f_i, \mathcal{F}_r)$, i.e., if $P(r|f_i, \mathcal{F}_r) \leq \theta$ then f_i is removed.[5] The resulting candidate list of fragments is M_r, for rule r.

Enhancement-Step 2: Ranking of selected fragments. Rank the list of candidate fragments, M_r, using the score $R(r, f_j)$ defined as the linear fusion of probability from the previous step and similarity score $S(r, f_j)$:

$$R(r, f_j) = k \cdot P(r|f_i, \mathcal{F}_r) + (1-k) \cdot S(r, f_i), \tag{4}$$

where $j = 1, \ldots, |M_r|$ with $|M_r| \leq |L|$ and $0 \leq k \leq 1$ is a factor that weights the influence of probability and similarity scores. The similarity score, $S(r, f_i)$, is computed as the average similarity between f_j and seed fragments of $\mathcal{F}_r$. The e top-ranked fragments are presented to the grammar developer. An example of fusion procedure is presented in Table 1 for the rule < DepartureCity > → ("leave < City >"|"travel from < City >"| ...) and three unknown fragments.

In order to estimate the probability $P(r|f_i, \mathcal{F}_r)$, for the fragment selection algorithm, labeled training data (i.e., grammar rules) are required. When no such data are available, a rule-based algorithm can be used for fragment selection (detailed below in Section 6.1.2), while (4) can be applied with $k = 0$.

6.1.1. Features for fragment selection

In this section, a series of features are presented, which are used for fragment selection (Athanasopoulou et al., 2014). These features are exploited by rule-based (see Section 6.1.2) and statistical (see Section 6.1.3) induction methods and they can be broadly divided in the following three categories.

Features extracted from corpus statistics. This category includes features such as (1) the probability of fragment f, $P(f)$, computed using statistical n-gram models (Jurafsky and Martin, 2009) trained on the same in-domain corpus used for grammar induction (for $n = 2, \ldots, 4$), (2) the perplexity of fragment f, (3) the number of occurrences of f normalized by the total number of occurrences of all fragments.

Features extracted from corpus parsed with low-level rules. (1) The ratio of low-level concepts over the total number of words in a fragment. For example, for the fragment "traveling from < City >" the feature value is $\frac{1}{3}$. (2) The number of words following the last low-level concept in a fragment (e.g., one for f = "traveling from < City > to"). This feature captures the relative position of low-level concepts in a fragment.

Features extracted from seed fragments. The similarity between two fragments f_q,f_r is estimated using two different metrics: 1) $S_1(f_q, f_r)$: The longest common sub-string lexical similarity metric (Stoilos et al., 2005), and 2) $S_2(f_q, f_r)$

[5] The posterior probability $P(r|f_i, \mathcal{F}_r)$ was computed by a statistical model. For more information see Section 6.1.3, as well as Section 7.3 about the used classification model.

defined below: Let l_a be the (character) length of the larger fragment (between f_q, f_r), l_b the length of the smaller fragment, $d = l_a - l_b$ the difference of the lengths and let $lev(f_q, f_r)$ be the function that computes the Levenshtein distance (or edit distance) of f_q,f_r (Levenshtein, 1966; Wagner and Fischer, 1974), then the similarity of f_q,f_r is computed as:

$$S_2(f_q,f_r) = \frac{l_a - lev(f_q,f_r)}{l_a + d}. \quad (5)$$

To estimate the similarity between fragment f and the set of seed fragments $\mathcal{F}_r$ the average similarity between f and each of the seed fragments in $\mathcal{F}_r$ was computed and normalized by the average score of all fragments in L. Other functions used to compare $f \in L$ with seed fragments in $\mathcal{F}_r$ are the following: (1) modified, pruned $S_2(.,.)$ that takes non-zero values only when two fragments differ by a single word, (2) several binary functions each of which equals to one when: f is a substring of a seed fragment in $\mathcal{F}_r$, f and a seed end with the same low-level rule with one seed (e.g., "at < City >" and "to < City >"), f has exactly the same lexical parts with one seed fragment (e.g., "depart from < City >" and f ="depart from < State >"), f is a substring of a seed with exactly one less word, and f has one extra word within one seed (e.g., "on the < Day >" and "on < Day >").

Next, two fragment selection algorithms are presented, which are applied during the grammar enhancement (i.e., the third phase of the induction process described above). The first algorithm, named SemSim (rule), is described in Section 6.1.2 and it is based on a set of hand-crafted rules. A statistical approach[6] is followed by the second algorithm, SemSim (stat), which is described in Section 6.1.3.

6.1.2. Rule-based fragment selection

This is a heuristic approach, named SemSim (rule), inspired by the manual process of grammar development and fragment selection. A set of features was designed, based on how grammar developers perceive the validity of a fragment. Each fragment, $f \in L$, is compared with seed fragments in $\mathcal{F}_r$. The rule-based fragment selection process is presented in Algorithm 1. The input of the algorithm is the list, L, that contains all fragments extracted from corpus and a set of seed fragments, $\mathcal{F}_r$, of one rule r. For each fragment $f \in L$, the algorithm determines if f is a good candidate for enhancing rule r by comparing it with seed fragments through a series of features. The list of candidate enhancements of rule r is denoted as M_r. For example, if f has exactly the same lexical parts with one of the seed fragments, then it is considered a candidate fragment and added to M_r, e.g., f ="depart from < State >" and $\mathcal{F}_{r_1}$ ="depart from < City >". Another rule checks if f contains at least one of the low-level rules appearing in seed fragments, e.g., for f ="depart from < State > on < Day >" and $\mathcal{F}_r$ ={"depart from < City >", "from < State >"} this is true. Algorithm 1 deterministically selects the candidate list, M_r, independent of the probability threshold value, θ. Thus when using (4) only the similarity scores will influence the ranking among the selected candidate fragments, since $P(r|f, \mathcal{F}_r)$ will be equal to one for all candidates accepted and zero for rejected candidates. The advantage of this algorithm is that it is completely unsupervised, i.e., it utilizes only a set of very few seed fragments to perform fragment selection from any fragment list. However, since no corpus features are utilized (such us the perplexity or context-based features) the selected fragments are often too similar to the seed fragments, which does not allow for high rule variability.

6.1.3. Statistical fragment selection

Given an in-domain corpus and a corresponding hand-crafted grammar, we can generate labeled data in order to train a statistical model for the fragment selection step of the enhancement phase, as follows. For a training rule r and the list L (with the corpus fragments), a feature vector is generated for each fragment $f_i \in L$, using the features proposed in Section 6.1.1. Each f_i is labeled as "valid" only if it belongs in r (in groundtruth grammar), otherwise it is labeled as "non valid". Then, a classifier is trained for selecting the candidate fragments. Note that although the feature extraction process is dependent both on the in-domain corpus (for estimating language model probabilities and perplexity) and on the seed rules (for estimating similarity features), the classifier is both domain and grammar rule independent. Thus, when a statistical model is trained using one in-domain corpus and one set of training rules, it can by used for fragment selection from any corpus and any rule r, providing also the posterior probability, $P(r|f, \mathcal{F}_r)$. The aforementioned approach is named SemSim (stat).

[6] In this work, we used random forest (see Section 7.3).

Algorithm 1. Rule-based fragment selection. Each "if" statement stands as a feature evaluated for candidate fragments. When it evaluates to "true", the respective fragment is added to a list of fragments for enhancing rule r.

Require: L; {Fragments list, i.e. all n-grams from corpus that contain low-level rules}
Require: F_r; {Seed fragments of rule r}
$T_r \leftarrow$ LowLevelRulesOfSeedFragments(F_r);
$M_r \leftarrow \{\}$; {initialization of list with candidate fragments of rule r}
for each fragment $f_i \in L$ **do**
  if f_i has the same lexical parts with at least one fragment in F_r **then**
    $M_r \leftarrow \{M_r, f_i\}$; {addition of f_i to the candidates}
  end if
  if f_i does not contain any low-level rule from the ones included in T_r **then**
    continue to the next fragment;
  end if
  if f_i is substring of at least one fragment in F_r **then**
    $M_r \leftarrow \{M_r, f_i\}$; {addition of f_i to the candidates}
  end if
  if f_i has one less word than one fragment in F_r **then**
    $M_r \leftarrow \{M_r, f_i\}$; {addition of f_i to the candidates}
  end if
  if f_i has one extra word within one fragment in F_r **then**
    $M_r \leftarrow \{M_r, f_i\}$; {addition of f_i to the candidates}
  end if
  if f_i differs by single word with at least one fragment in F_r **then**
    $W_r \leftarrow$ FragmentsThatDifferBySingleWord(F_r, f_i); {$W_r \in F_r$}
    $sim \leftarrow \max_j \left\{\text{SimilarityOfDifferentWords}(W_{r_j}, f_i)\right\}$; {similarity is computed using S_1}
  else
    $sim \leftarrow 0$;
  end if
  if $sim > 0.3$ **then**
    $M_r \leftarrow \{M_r, f_i\}$; {addition of f_i to the candidates}
  end if
  if f_i contains only low-level rules **then**
    $M_r \leftarrow \{M_r, f_i\}$; {addition of f_i to the candidates}
  end if
end for
return M_r;

6.2. Induction based on slot-filling

A popular approach for building SDS statistical grammars is to view the problem as a slot sequence filling application (e.g., Raymond and Riccardi, 2007; Heck et al., 2013). Modeling of slot sequences is typically done using CRF (Lafferty et al., 2001). For a sequence of words $X = x_1, \ldots, x_N, x_i \in V$, where V is the vocabulary and the corresponding tag sequence $Y = y_1, \ldots, y_N, y_i \in C$, and C is the set of labels, the annotation of an utterance according to the grammar is given by: $\widehat{Y} = \text{argmax}_Y P(Y|X)$. The conditional probability is computed using:

$$P(Y|X) = \frac{1}{Z(X)} \exp \sum_k \lambda_k f_k(y_{t-1}, y_t, x_t), \tag{6}$$

where $Z(X)$ is the normalization term and f_k is the set of features used with the associated weights λ_k. We have used six features, modeling the bigram tag sequence $f_k(y_{t-1}, y_t)$ and the tag-word pairs in a size 2 window $f_k(y_t, x_i), t-2 \leq i \leq t+2$. The input vocabulary is composed of words and low-level rules, while for the output the IOB annotation scheme is used, where each token is tagged as O, B, or I, for outside rule, beginning of rule or inside rule, respectively. An example is presented in Table 2.

The algorithm consists of three steps described below, given that the following are available: a corpus where low-level rules have been induced and substituted, a set of seed grammar fragments, and a request of e new fragments.

Table 2
Example of CRF input and output for an unknown utterance (the output is also presented in IOB format).

Initial test (unknown) utterance	Flights	From	Chicago
IN: test utterance with low-level rule substituted	Flights	From	< City >
OUT: test utterance with high-level rule	Flights	< DepartureCity >	
OUT: test utterance with high-level rule (IOB format)	O	B- < DepartureCity >	I- < DepartureCity >

Step 1: CRF training. The sentences of the corpus containing instances of the seed grammar fragments are used to train a CRF classifier. The (high-level) seed fragments are incorporated into those sentences according to the IOB format.

Step 2: Fragment extraction. The classifier is applied on the corpus and the set of candidate fragments is extracted.

Step 3: Grammar enhancement. The extracted fragments are ordered with respect to their frequency of appearance and the top e ones are presented to the grammar developer.

Similarly to the algorithm described in Section 6.1, two constraints are used in the fragment extraction step. Only fragments consisting of two up to five words are considered; fragments that do not contain low-level rules are discarded. This approach is named SlotFill (CRF).

6.3. Combining slot-filling and string similarity

The slot-filling method described in Section 6.2 is based only on context and does not exploit string similarity between the seeds and candidate fragments. A fusion with the rule-based fragment selection algorithm is possible, where context is used to create the candidate fragment list, and string similarity is used for ranking them. Candidate fragments are extracted as described in Section 6.2 (Steps 1 and 2), while (4) with $k = 0$ is applied in Step 3. This approach is named SlotFill (CRF + Sim). An example is given in Table 3, where we present the top six induced fragments using SlotFill (CRF) and SlotFill (CRF + Sim) for the rule < ArrivalCity > and the seed fragments "travel to < city >","arrives at < airport >","to < airport >".

7. Experiments and evaluation

In this section, we first present the details of the corpus creation experimental procedure following the web harvesting approach (see Section 7.1). The low- and high-level induction algorithms are evaluated (see Sections 7.2 and 7.3, respectively) on two domains (air travel, finance) and two languages (English, Greek). The grammar induction evaluation is performed incrementally, i.e., a small set of rules is used to bootstrap the grammars as in a realistic human-in-the-loop SDS application authoring scenario. Such an iterative scenario is described and evaluated in Section 7.4.

Table 3
Example of fragment ranking using SlotFill (CRF) and SlotFill (CRF + Sim) for the enhancement of the high-level rule < ArrivalCity >.

Rank	SlotFill (CRF)	SlotFill (CRF + Sim)
1	To < City >	Travel to < Airport >
2	To < Month >	Fly to < Airport >
3	To < State >	Travel in < City >
4	To < Airline >	Travel to < Airline >
5	Goes to < City >	Arrives at < City >
6	To < Day >	Travel into < City >

7.1. Corpora creation via web harvesting

For corpus creation via the harvesting of web data (and also for grammar induction detailed in the next sections) the experiments were conducted for the following domains and languages: (1) English air travel, (2) Greek air travel, and (3) English finance. For English, a finite-state-based grammar and a seed corpus were used, while for Greek only grammar was available. For Greek air travel, a seed corpus was constructed by manually selecting utterances from the web-harvested corpora. The seed corpus sizes are presented in Table 4. For the English air travel domain, a manually web harvested corpus (described in Klasinas et al. (2013)) and a crowdsourced corpus were also used in the evaluation, comprising of 12K and 25K sentences respectively. The experimental procedure is described next.

Query generation. The queries created for each domain are presented in Table 5. Two methods were investigated for query generation, namely, starting from a seed grammar or from a seed corpus. For the English air travel domain, the extraction of all *n*-grams from the seed corpus up to order seven resulted in a set of 24,714 queries (this approach is denoted as ALL). When restricting the extracted queries to order four (denoted as 4-grams), a set of 7322 queries was obtained. For the English air travel domain, an additional query filtering scheme was implemented that is denoted as PLP. This query set was created by ordering the queries generated by the ALL approach in order of decreasing perplexity (computed with respect to an out-of-domain[7] language model), and keeping the top 10%. For both the English and Greek air travel domain, queries were extracted from the seed grammar resulting in 248 and 4320 queries, respectively (denoted as GRM). The query set for English is smaller because it only includes queries that correspond to grammar rules that exist in the seed set. For the English finance domain, a set of 1036 queries was extracted from the seed corpus following the ALL approach. The following sets of keywords were used for query expansion, serving as pragmatic constraints: (“airport”, “flight”, “travel”) for the English air travel domain, (“αεροδρομιο”, “πτηση”, “ταξιδι”) for the Greek air travel domain, and (“bank”, “account”, “card”) for the English finance domain.

Corpora creation and filtering. The queries were submitted to the Yahoo! web search engine and the 50 top-ranked documents were downloaded. The raw text was extracted (Javaparser 1.4, 0000), and the sentence boundaries were detected (Lingua-Sentence-1.04, 0000). Sentences shorter than five or longer than fifty words were discarded. The downloaded documents were filtered on a per-sentence basis using (1) a set of automatically defined pragmatic

Table 4
Seed corpora for each domain and language.

Domain	Language	# utterances
Air travel	English (EN)	1560
Air travel	Greek (GR)	1107
Finance	English (EN)	416

Table 5
Query generation approaches for two domains (air travel and finance) and two languages (English (EN) and Greek(GR)).

Domain	Language	Name of approach	Queries extracted from		Query filtering	Num. of queries
			Seed corpus	Grammar		
Air travel	EN	ALL	✓(up to 7-grams)	×	×	24,714
		PLP	✓(up to 7-grams)	×	✓(perplexity-based)	2500
		4-grams	✓(4-grams)	×	×	7322
		GRM	✓	✓	×	248
Air travel	GR	GRM	×	✓	×	4320
Finance	EN	ALL	✓(up to 7-grams)	×	×	1036

[7] We have used the English part of the news corpus available at http://www.statmt.org/wmt10/training-monolingual.tgz.

constraints (FPC), and (2) perplexity ranking. The perplexity-based filtering (denoted as ppl) was performed using the seed corpus. A trigram language model was trained and then the procedure described in Section 4.2 was followed. A variation of the ppl filtering method was also investigated denoted as ppl-term, where the instances of terminal rules were substituted by the respective rule labels (e.g., "Chicago" was substituted by $< \text{City} >$) in the corpus used for training the language model. Regarding the FPC filtering approach, the three most informative words were utilized (according to the $G(w)$ value computed in (2)). The following corpora were created via the ppl filtering approach: 50 K and 10 K sentences for the English and Greek travel domain, respectively, while the corpus created for the English finance domain consists of 5 K sentences. These corpus sizes were empirically set for balancing in-domainness and grammar coverage.

For the various corpora created using the query generation and corpus filtering methods outlined above the following statistics are reported in Table 6: (1) fragments per word: the ratio of grammar fragments per word (the fraction of corpus words contained in the grammar, i.e., domain-specific words), (2) number of terminal instances: the number of distinct (unique) terminal rules found in the corpus, and (3) number of non-terminal instances: the number of distinct non-terminal instances found in the corpus. The fragments per word ratio can be regarded as an in-domainness measure (related to precision). The number of terminal and non-terminal instances measures the corpus grammar coverage (related to recall). Regarding the English travel domain, we observe that the seed corpus yields the top performance in terms of in-domainness (the fragments per word ratio equals 0.44). For the case of the Greek travel domain, the highest value of this ratio (0.24) is achieved by the GRM (ppl-term) approach followed by 0.23 that is obtained via the use of the seed corpus. For all domains, the best grammar (considering the number of terminal/non-terminal instances) is obtained by the web-harvested corpora. Random sentence selection (denoted as random) provides good coverage, especially for the English air travel domain where the number of queries is large, however, in-domainness is rather low. Perplexity-based corpus filtering (ppl or ppl-term) improves in-domainness for all domains and languages. For the English air travel domain, the different query generation approaches (ALL, PLP, 4-grams from a corpus) do not impact performance much. However, generating queries from a seed grammar (GRM) results in a corpus with low fragment per word percent (poor in-domainness). This can be attributed to the very small query set size, which leads to a small number of in-domain sentences. The best results for all domains are achieved by employing the ppl-term approach for corpus filtering. The benefit yielded by the ppl-term filtering is

Table 6
Corpora statistics evaluating the coverage of domain grammars. The corpora were created from web data using different query generation and corpus filtering techniques. The statistics are shown for two domains (air travel and finance) and two languages (English (EN) and Greek (GR)).

Domain	Lang.	Corpus creation			Corpus statistics wrt domain grammar		
		Query generation		Corpus filtering	Fragments	# terminal	# non-terminal
		Approach	Pragm.		per word	instances	instances
		Seed corpus			**0.44**	112	89
		Manually harvested web corpus			0.18	428	193
		ALL	×	× (random)	0.08	813	163
		ALL	×	✓(ppl)	0.40	629	167
Air travel	EN	ALL	✓	✓(ppl)	0.41	675	176
		ALL	✓	✓(ppl + FPC)	0.41	701	174
		ALL	✓	✓(ppl-term + FPC)	0.37	**997**	248
		PLP	✓	✓(ppl + FPC)	0.23	834	289
		PLP	✓	✓(ppl-term + FPC)	0.38	980	**367**
		4-grams	✓	✓(ppl + FPC)	0.38	751	181
		GRM	✓	✓(ppl + FPC)	0.12	860	253
		Seed corpus			**0.23**	136	**105**
Air travel	GR	GRM	✓	× (random)	0.04	148	43
		GRM	✓	✓(ppl)	0.12	192	89
		GRM	✓	✓(ppl-term)	0.24	**311**	**105**
		Seed corpus			0.07	27	54
		ALL	✓	× (random)	0.02	5	20
Finance	EN	ALL	✓	✓(ppl)	0.04	18	81
		ALL	✓	✓(ppl + FPC)	0.03	22	93
		ALL	✓	✓(ppl-term + FPC)	**0.11**	**28**	**148**

Table 7
Performance of low-level rule induction (precision) using corpora created via various query generation and corpus filtering techniques. The performance is shown for two domains (air travel and finance) and two languages (English (EN) and Greek (GR)).

Domain	Lang.	Corpus creation			Precision of induction (%)	Average # correctly induced fragments / # induced fragments
		Query generation		Corpus filtering		
		Approach	Pragm.			
		Seed corpus			**34.5**	13.8/40
		Manually harvested web corpus			26.4	10.5/40
		Corpus created via crowdsourcing			23.4	9.3/40
		ALL	×	× (random)	11.5	4.6/40
		ALL	×	✓(ppl)	23.9	9.5/40
Air travel	EN	ALL	✓	✓(ppl)	25.6	10.2/40
		ALL	✓	✓(ppl + FPC)	24.5	9.8/40
		ALL	✓	✓(ppl-term + FPC)	28.6	11.4/40
		PLP	✓	✓(ppl + FPC)	21.1	8.4/40
		PLP	✓	✓(ppl-term + FPC)	18.7	7.5/40
		4-grams	✓	✓(ppl + FPC)	27.4	10.9/40
		GRM	✓	✓(ppl + FPC)	26.7	10.6/40
		Seed corpus			38.5	5.0/13
Air travel	GR	GRM	✓	× (random)	30.8	4.0/13
		GRM	✓	✓(ppl)	**46.2**	6.0/13
		GRM	✓	✓(ppl-term)	30.8	4.0/13
		Seed corpus			11.6	0.9/8
		ALL	✓	× (random)	0	0/8
Finance	EN	ALL	✓	✓(ppl)	**25.8**	2.1/8
		ALL	✓	✓(ppl + FPC)	19.2	1.5/8
		ALL	✓	✓(ppl-term + FPC)	17.2	1.3/8

larger for the English finance and Greek air travel domain. This is probably due to the fact that the seed corpora are smaller for these domains compared to the English travel domain.

7.2. Induction of low-level rules

The low-level induction algorithm was evaluated on the air travel domain for English and Greek, as well as for the English finance domain. The probability distributions of left and right context used in (3) were estimated via n-gram language modeling. A separate model was built for each of the corpora presented in Table 7. The system takes as input an initial set of grammar rules (typically three examples per rule), assumed to be hand-crafted by a grammar developer. The algorithm then induces additional rules (in our evaluation scenario a fixed number of additional rules is requested). For example, consider the low-level rule $<\text{City}>$ and the rule fragment seeds "New York" and "Boston" provided to the algorithm.[8] The following fragments: "Atlanta", "Athens", and "Toronto" are then automatically induced, resulting in the enhancement $<\text{City}> \rightarrow$ ("New York"|"Boston"|"Atlanta"|"Athens"|"Toronto").

The above process uses two experimental parameters: (1) the number of seed fragments that are given as input (note that the seed rules were randomly selected from a groundtruth grammar), and (2) the number of induced fragments that constitute the output of the algorithm. The seed fragments can be regarded as domain knowledge that is made available by the grammar developer. Here, few seeds were used as a way to simulate a minimal human intervention scenario. Specifically, three and two seed fragments were used during the induction process of each rule for the English and Greek air travel domain, respectively. For the same process, two seed fragments were used for the English finance domain. The number of induced fragments per rule was set[9] to ten and thirteen for the English and Greek air travel domain, respectively, while for the English finance domain four fragments were elicited.

For evaluation purposes, grammars (groundtruth) were manually authored by domain experts for each domain and language. The groundtruth for the English and Greek air travel domain includes four and one rules, respectively,

[8] The tokenization task was not addressed in this paper. For practical purposes we have assumed that all tokens have been correctly identified in the corpus. The named entity recognition was implemented via gazetteer lookups. These decisions were made in order to focus on the evaluation of the induction results ignoring any tokenization and NER errors.

[9] These numbers were determined by taking into account the average number of fragments per rule for the rules included in the groundtruth.

while the groundtruth of the English finance domain consists of two rules.[10] The number of rules for each language/domain is different, because only rules with a sufficient number of instances across all evaluation scenarios were selected. Precision[11] was used as the evaluation metric defined as the ratio of the number of correctly[12] induced fragments to the total number of fragments induced by the algorithm. For all cases, the average precision is reported computed by averaging the precision scores over 50 random selections of seeds (runs).

The evaluation results for all domains and languages are presented in Table 7. Table 7 also includes the average number of correctly induced fragments and the total number of fragments included in the groundtruth rules.

Regarding the English air travel domain,[13] perplexity-based corpus filtering is observed to have a positive effect on performance. Specifically, the ALL and 4-grams approaches for query formulation, combined with the expansion of queries using pragmatic constraints, result in the creation of corpora that yield precision (28.6 and 27.4%) higher than that of manually harvested corpora. For the Greek air travel domain, the best results (46.2% precision) are obtained by the ppl approach for corpus filtering. This observation also holds for the English finance domain, where the best performance is 25.8%.

7.3. Induction of high-level rules

The induction algorithm for high-level rules was evaluated on the air travel and finance domains for both English and Greek. We followed the same procedure as for low-level induction described in Section 7.2. In a similar fashion with low-level induction, for a high-level rule, few fragments are given to the algorithm as seed examples. A high-level rule is then enhanced by augmenting the set of seeds with their respective induced fragments. For example, consider the high-level rule $<$DepartureCity$>$. Assume that the rule fragments "fly from $<$City$>$" and "departing from $<$City$>$" are the seeds provided to the algorithm. The algorithm may (automatically) induce fragments such as "flight from $<$City$>$", and "departure from $<$City$>$". After the induction, the rule $<$DepartureCity$>$ is enhanced as $<$DepartureCity$>$ $\rightarrow$ ("fly from $<$City$>$"|"departing from $<$City$>$"|"flight from $<$City$>$"|"departure from $<$City$>$"). A prerequisite for this process is that the low-level grammar rules, e.g., $<$City$>$, have already been induced (and corrected by the grammar developer).

There are two experimental parameters: (1) the number of seed fragments (input), and (2) the number of induced fragments that constitute the output of the algorithm. Again, few seeds were used, three for all domains/languages. The number of induced fragments per rule was set to twelve and eight for the English and Greek air travel domain, respectively, while twelve fragments were used for the English finance domain. For the extraction of features based on corpus statistics, which are used for fragment selection (see Section 6.1.1), the SRILM language modeling toolkit (Stolcke, 2002) was used. Regarding the statistical fragment selection described in Section 6.1.3, a random forest classifier was used that utilized twenty trees. For this model, the fusion weight k (used in (4)) was set to 0.8, while the probability threshold θ was set to 0, based on previous experiments (Athanasopoulou et al., 2014). A single model was trained with respect to the English air travel domain including features extracted from a corpus that was created by following the PLP and ppl-term + FPC approaches for query formulation and corpus filtering, respectively. During the induction process (i.e., testing) this model was applied across three domains/languages in order to investigate its portability.

A hand-crafted grammar was created by experts for each domain/language and used as groundtruth. For evaluation purposes,[14] the five most common rules were used for the travel domain (for both English and Greek), while for

[10] The low-level rules used for evaluation per domain and language are as follows: Air travel (EN): (1) $<$City$>$, (2) $<$Day$>$, (3) $<$Airline$>$, (4) $<$Date$>$; Air travel (GR): $<$City$>$; Finance (EN): (1) $<$Account-type$>$, (2) $<$Card-type$>$.

[11] The recall was not computed since a fixed number of fragments is requested to be induced.

[12] Excluding seed fragments.

[13] We also evaluated the performance of word2vec (Mikolov et al., 2013) (using the implementation available at http://www.code.google.com/archive/p/word2vec/) for computing the cosine similarity between the contextual embeddings of seeds and candidate fragments. It was found that this requires the tuning of the word2vec parameters that is corpus-dependent. For example, for the English air travel domain and the seed corpus, the use of the default parameters yielded 22.2% precision (for context size set to one). For the same example, the highest precision achieved by word2ec was found to be 38.5% and it was achieved after exhaustively searching the parameter space. In particular, the following non-default parameter values were used: window=1, sample=0, min-count=1, and iter=3 (the default settings were preserved for the rest parameters).

[14] The high-level rules used for evaluation per domain and language are as follows: Air travel (EN): (1) $<$DepartureCity$>$, (2) $<$DepartureDate$>$, (3) $<$ArrivalCity$>$, (4) $<$Time$>$, (5) $<$DepartureTime$>$; Air travel (GR): (1) $<$DepartureCity$>$, (2) $<$DepartureFlightti$r$$>$, (3) $<$Time$>$, (4) $<$Date$>$, (5) $<$StopoverCity$>$; Finance (EN): (1) $<$AccountAction$>$, (2) $<$CardAction$>$.

Table 8
Performance of high-level rule induction (precision) using corpora created via various query generation and corpus filtering techniques. The performance is shown for two domains (air travel and finance) and two languages (English (EN) and Greek (GR)).

Domain	Lang.	Corpus creation			Precision (%) of induction			
		Query generation		Corpus filtering	SemSim	SemSim	SlotFill	SlotFill
		Approach	Pragm.		(rule)	(stat)	(CRF)	(CRF + Sim)
		Seed corpus			24.7	29.0	14.2	16.8
		Manually harvested web corpus			**33.5**	37.0	25.5	**29.7**
		ALL	×	× (random)	15.6	20.6	16.4	17.3
		ALL	×	✓(ppl)	30.2	32.3	13.1	21.1
Air travel	EN	ALL	✓	✓(ppl)	30.4	30.6	14.5	20.1
		ALL	✓	✓(ppl + FPC)	30.6	33.0	16.8	20.9
		ALL	✓	✓(ppl-term + FPC)	24.7	29.0	25.2	**28.8**
		PLP	✓	✓(ppl + FPC)	31.3	38.6	24.5	26.0
		PLP	✓	✓(ppl-term + FPC)	33.0	**37.7**	**26.7**	28.3
		4-grams	✓	✓(ppl + FPC)	31.1	35.0	20.1	21.9
		GRM	✓	✓(ppl + FPC)	28.3	31.4	22.7	23.7
		Seed corpus			**40.2**	**45.4**	26.8	26.8
Air travel	GR	GRM	✓	× (random)	13.9	18.2	12.6	13.3
		GRM	✓	✓(ppl)	39.3	41.6	27.6	29.9
		GRM	✓	✓(ppl-term)	34.6	39.2	**28.1**	**31.6**
		Seed corpus			15.6	16.1	8.3	9.7
		ALL	✓	× (random)	10.5	15.3	6.5	6.8
Finance	EN	ALL	✓	✓(ppl)	**22.2**	21.2	24.7	27.6
		ALL	✓	✓(ppl + FPC)	21.8	19.2	23.2	24.2
		ALL	✓	✓(ppl-term + FPC)	21.2	**24.6**	**36.8**	**30.3**

the English finance domain two rules were used.[15] The precision[16] of induction was used for evaluation purposes defined as in the case of the low-level rule induction. Evaluation results for all domains and languages are presented in Table 8 for various corpora created via various query generation and corpus filtering techniques.[17] The results are shown for four induction approaches:

- SemSim (rule). The induction algorithm is based on the selection of candidate fragments, ranked according to their similarity with the seed fragments (see (4) in Section 6.1). This approach adopts a rule-based model for the fragment selection as described in Section 6.1.2.
- SemSim (stat). Similar to the previous approach, except that the rule-based model is replaced by a statistical one that is described in Section 6.1.3.
- SlotFill (CRF). Induction is treated as a slot filling problem based on CRF as defined in Section 6.2.
- SlotFill (CRF + Sim). This is an enhancement of the SlotFill (CRF) approach, which incorporates the similarity between the seeds and candidate fragments (see Section 6.3).

Regarding the SemSim-based approaches, the English travel domain was used for training SemSim(stat) and developing the rules used by SemSim(stat). This model and rules were applied across all domains/languages. For all approaches the average precision is reported, computed by averaging the precision scores over 50 random selections of seeds (runs).

Regarding the English air travel domain, the best performance (37.7%) is obtained by the SemSim (stat) approach exceeding the precision yielded by the seed corpus. Both SemSim (rule) and SemSim (stat) are shown to outperform the CRF-based approaches approximately by a factor of 10% precision. The SemSim (stat) approach performs

[15] Results are reported on a subset of the rules in order to make meaningful comparisons between languages and domains. The proposed algorithms are scalable to a larger set of rules with similar performance, e.g., similar results have been achieved in the English travel domain when using 23 high-level rules (Athanasopoulou et al., 2014). This also hold for the case of low-level rules, e.g., see Iosif et al. (2006), where 38 rules were used for the English travel domain.

[16] Again, the recall was not computed since a fixed number of fragments is requested.

[17] The χ^2 test was applied for the low- and high-level rule induction with respect to the seed corpus and the best-performing web-harvested corpora for the English air travel domain. The differences in performance yielded by these corpora are statistically significant at $p < 0.05$.

slightly better than SemSim (rule). When the similarity is utilized by the CRF-based algorithm, i.e., SlotFill (CRF + Sim), the performance is improved compared to the SlotFill (CRF) approach. The observations above also hold for the Greek air travel domain, for which the highest precision (45.4%) is achieved by the SemSim (stat) approach. This score is higher compared to the performance yielded when using the seed corpus. The fact that the web-harvested corpora do not improve on the performance of the seed corpus implies that the quality of the downloaded data is lower for the Greek language, probably due to the small availability of Greek air travel corpora in the web. The relative performance of the SemSim- and SlotFill-based approaches observed for the air travel domain (English and Greek) is reversed for the English finance domain for some corpora. The highest precision (36.8%) is achieved by the SlotFill (CRF) approach followed by the performance of SlotFill (CRF + Sim).[18] All approaches outperform the precision yielded by the seed corpus.

Regarding crowdsourcing, we have focused on designing rules for a finite-state-based SDS grammar in the travel domain for English. Only a subset of the grammar rules were targeted, namely eliciting data for (1) Date, and (2) DepartureCity concepts. The Crowdflower platform was used to gather the data. The major problem during this process was quality control (Crowdflower's mechanism for automatic quality control), the nature of the tasks did not allow for the use of gold standard data. We used the flagging mechanism in order to exclude contributors providing irrelevant data. In addition, we experimented with varying the payments, starting from 2 cents and converging to 0.6 cents per unit (Human Intelligent task), as well as with restricting the maximum number of units that a contributor could submit. In Table 9, we compare the precision of high-level rule induction using the corpora created via the several crowdsourcing tasks described in Section 4.3. We observe that the best performance (45.0%) is obtained by the SemSim (stat) approach. This performance corresponds to the corpus that resulted by merging the corpora created during the various tasks. Also, the performance of the CRF-based approach is improved by using similarity (SlotFill (CRF + Sim)). Overall, web-harvested corpora were observed to yield performance that is higher (or at least equal) to the respective performance of seed corpora. Another important finding is that the use of pragmatic constraints (either for query formulation or corpus filtering) improves performance.

To better understand the impact of the number of seeds on performance, we applied all four induction algorithms over the best automatically web-harvested corpus for each domain/language using varying number of seeds. The results are plotted in Fig. 6 in terms of precision. For the air travel domain, for both languages, it is observed that the SemSim-based approaches outperform the SlotFill-based approaches when few seeds (approximately three) are available. The relative performance of the SlotFill-based approaches is higher when more seeds are utilized. Both SemSim (rule) and SemSim (stat) perform poorly in comparison to SlotFill (CRF) and SlotFill (CRF + Sim) for the case of the English finance domain. The utilization of similarity in SlotFill (CRF + Sim) improves the performance for the air travel domain (English and Greek) when compared to SlotFill (CRF).

Regarding the three experimental datasets used in the present work, the most widely-studied dataset is the English travel domain (ATIS). For example, in Pargellis et al. (2004) the induction of low-level rules was investigated with

Table 9
Performance of high-level rule induction (precision) using several crowdsourced corpora for the English (EN) air travel domain.

Domain	Lang.	Corpus creation			Precision (%) of induction			
		Query generation		Corpus filtering	SemSim	SemSim	SlotFill	SlotFill
		Approach	Pragm.		(rule)	(stat)	(CRF)	(CRF + Sim)
Air travel	EN	*Seed corpus*			32.5	35.0	32.3	35.1
		Manually harvested web corpus			35.0	42.5	30.9	36.3
		PLP	✓	✓(ppl + FPC)	31.3	38.6	24.5	26.0
		Crowdsourcing: all tasks			**40.0**	**45.0**	**32.5**	34.5
		Crowdsourcing task: answers			33.3	35.8	20.7	27.1
		Crowdsourcing task: paraphrasing			30.8	32.5	23.6	31.4
		Crowdsourcing task: complete the dialogues			34.2	38.3	22.6	27.5
		Crowdsourcing task: fill in			48.3	38.3	31.5	36.3

[18] This can be attributed to the fact that SemSim was developed and tuned using the English air travel domain, while SlotFill is trained for each domain.

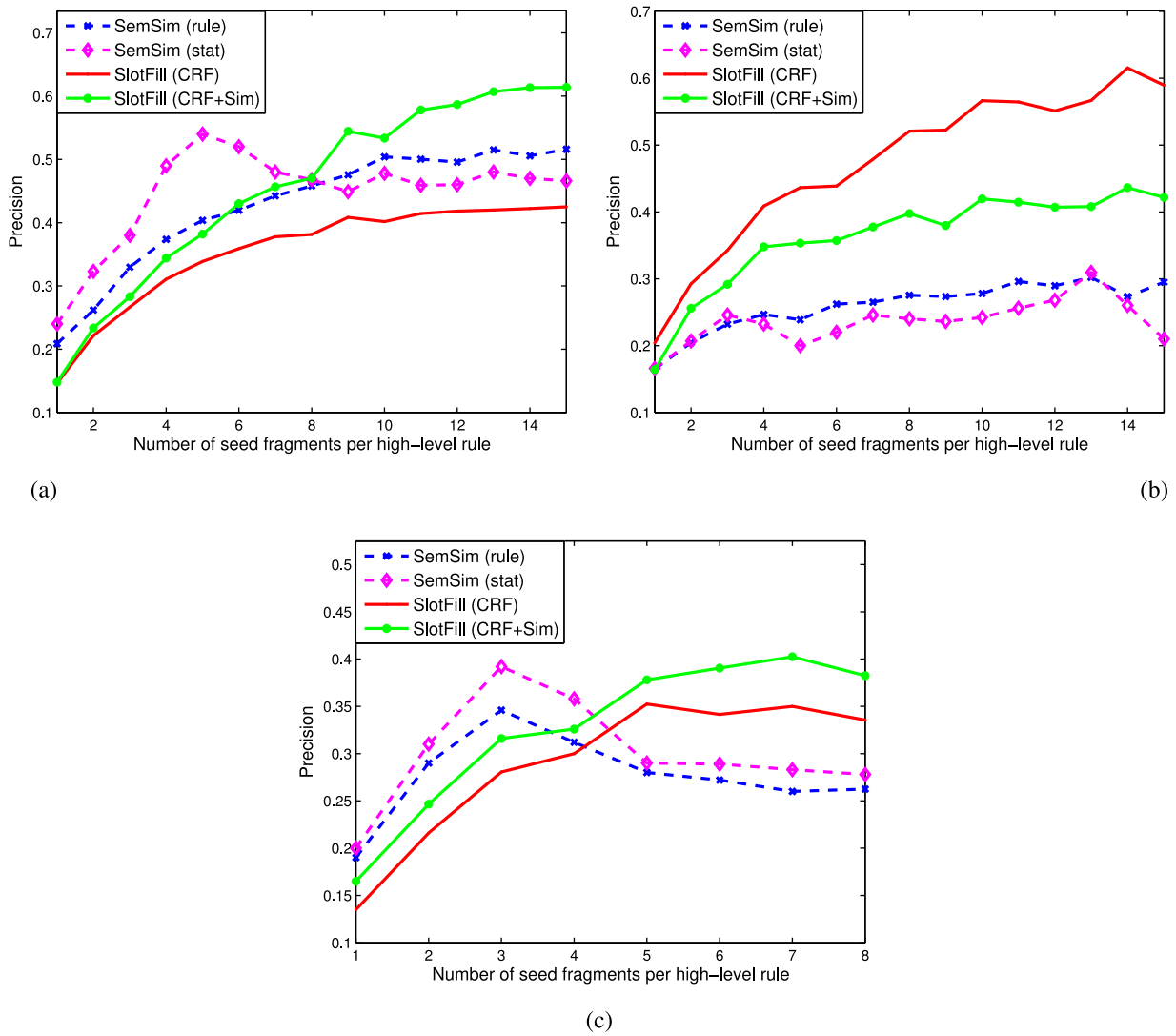

Fig. 6. Performance for high-level rule induction as a function of number of seeds: (a) English air travel domain for five rules, (b) English finance domain for two rules, and (c) Greek air travel domain for five rules. The results correspond to the best performing web-harvested corpora that were created as follows. For (a), query generation: PLP augmented with pragmatic constraints; corpus filtering: ppl-term combined with pragmatic constraints. For (b), query generation: ALL augmented with pragmatic constraints; corpus filtering: ppl-term combined with pragmatic constraints. For (c), query generation: GRM augmented with pragmatic constraints; corpus filtering: ppl-term. SemSim-based approaches were trained only with respect to the English travel domain, while the SlotFill-based approaches were trained for each domain/language.

respect to various similarity metrics, while the correctness of the induced rules was evaluated by human subjects. In Meng and Siu (2002), the induced low- and high-level rules were used for semantic parsing, so, the respective performance was reported in terms of parse coverage. Also, the coverage was not reported separately for each rule type. In addition, a number of approaches based on deep neural networks have been recently proposed in the literature, where the evaluation results are reported for the task of slot-filling without distinguishing low- an high-level rules. Such approaches mainly deal with RNN (e.g., Mesnil et al., 2015) and related variants, such as RNN with external memory (B. Peng and K. Yao, 2015), combination of RNN and structured Support Vector Machines (RSVM) (Shi et al., 2016), long-short-term-memory networks (Yao et al., 2014). Models based on deep neural networks (DNNs) have been shown to perform better than CRFs for the task of slot-filling in various domains including ATIS (exceptions include an entertainment-related domain reported in Mesnil et al. (2015), the MEDIA corpus dealing with hotel reservation and tourist information (Vukotic et al., 0000)). For the ATIS domain the performance is as follows: CRFs

yielded 92.9% F1 score, while the best F1 score (95.5%) was reported for the case of RSVM (Shi et al., 2016). The aforementioned F1 scores were reported in the framework of a comparative study based on an experimental setup consisting of 4978 and 893 training and test utterances, respectively (also used in other works). This setup is different compared to the setting adopted in the present work, where the key idea is the exploitation of few seeds. Here, we investigate the case with sparse data, where the CRF-based models are expected to yield similar performance to DNNs.

7.4. The human-in-the-loop induction paradigm

In this section, we present the evaluation results of grammar induction according to the human-in-the-loop iterative paradigm. The basic idea is that the induction process starts with a set of seed rules determined by the grammar developer. The seeds are used by the aforementioned algorithms for rule induction. Then, the automatically induced rules are manually approved or rejected by the developer, while the approved rules are added to the set of seeds. The updated set of seeds is again used for a new induction cycle, followed by the manual approval/rejection, and so on. The process is manually terminated by the developer.

For evaluation purposes, this process was (independently) followed by ten grammar developers for the air travel domain in English. Each developer was instructed to apply the induction algorithms for both low- and high-level rules. For each level, a separate process was performed. Two seed rules were provided by each developer (not necessarily the same) at the beginning of the process, while the number of requests per iteration was not fixed. A graphical user interface was built for assisting the approval/rejection of the automatically induced rules. The process was terminated by the developer according to his/her (subjective) estimate regarding the coverage of the induced grammar. Regarding the induction algorithms, the best web-harvested corpus was used (created according to the PLP, ppl-term + FPC approach), while the SemSim (rule) high-level induction algorithm was applied.

The evaluation results are shown in Table 10 for the low- and high-level rule induction, after averaging the scores across the ten developers. In addition to the average precision,[19] the results include the average number of iterations and induced rule fragments, as well as the average duration of the process. Slightly higher precision is achieved for the high-level induction (55.0%) compared to the low-level (51.3%). The developers are shown to have requested more than double the number of fragments for the case of high-level induction (49.0 vs. 23.5). Also, the human-in-the-loop approach enables the induction of more precise rules compared to the automatic algorithms. For example, for the case of low-level rules, the 51.3% precision (see Table 10) achieved via the human-in-the-loop approach outperforms the 18.7% precision (see Table 7) obtained by the automatic algorithm. Regarding high-level rules, the respective scores are 55.0% (see Table 10) vs. 33.7% precision (see Table 8). This difference was expected since approvals/rejections were manually made by the developers at the end of each induction cycle. Considering the development of grammar rules as a part of a broader process (also including intermediate validation tests, reviews, etc), the utilization of the induction algorithms was (empirically) found to reduce the overall effort (in terms of time) by more than 50%[20] when compared to the entirely manual process. Also, the effective exploitation of these algorithms was observed to positively correlate with the developer experience.

Table 10
Use of grammar induction algorithms following the human-in-the-loop paradigm: evaluation results.

Rule type	Average # of system iterations	Average # of induced fragments	Average precision (%)	Average duration (min)
Low-level	5.9	23.5	51.3	6.0
High-level	12.5	49.0	55.0	8.3

[19] The average precision was computed across the precision of rules induced by each developer.

[20] Regarding low-level rules, the time of the entire manual process was reduced from 10 to 3 person-days, while the for the case of high-level rules the required time was reduced from 30 to 15 person-days. This reduction was facilitated by the automatic induction exhibiting 51.3 and 55.0% precision for low- and high-level rules, respectively (see Table 10).

Most similar to the proposed "human-in-the-loop" approach is the work of Meng and Siu (2002). in Meng and Siu (2002), low- and high-level rules were automatically induced at each cycle of an iterative process using the same features and metric (a variation of (3)) for both rule types. After a number of iterations, which was empirically set, the process was terminated and the resulting rules were manually post-corrected. The present work has the following key differences in comparison to Meng and Siu (2002): (1) The induction of low- and high-level rules is considered separately, while different features and metrics are utilized for each rule type, (2) The human post-corrections take place at the end of each iteration enabling better control of the induction results. in Meng and Siu (2002), the total time for inducing and correcting a grammar for a similar domain (i.e., travel domain in English using the ATIS corpus) was 5 h resulting into 36 and 446 high- and low-level fragments, respectively. Regarding the induction of both low- and high-level rules, in this work, less time is required ($\frac{6.0+8.3}{23.5+49.0} = 0.19$ min per rule, on average) compared to [26] where $\frac{5\times60}{446+36} = 0.62$ minutes per rule were needed.

8. Conclusions

In this work, we investigated data harvesting and grammar induction algorithms for spoken dialogue systems. The main technique used for corpora creation was the harvesting of web data, while the potential of crowdsourcing was also studied. Two variants of language-agnostic algorithms were employed for inducing low- and high-level grammar rules exploiting various features of lexical and semantic similarity. The induction framework was formulated as an example-driven process where few grammar rules were provided as seeds for initiating the automatic induction algorithms.

Regarding grammar rule induction, the main finding is that different features and similarity metrics should be applied for low- and high-level rules. The (widely-used) similarity of contextual features that is based on the distributional hypothesis of meaning was found to be appropriate for the case of low-level rules. Unlike low-level rules, the induction of high-level rules proved to be a more complex problem consisting of two sub-tasks: the identification of valid text chunks that should be included in the grammar and their ranking. An important finding regarding high-level induction is that the statistical approach, i.e., SemSim (stat), performs better than SemSim (rule). Despite the fact that the SemSim-based approaches were trained on the English travel domain, they were shown to perform well when applied on the respective Greek domain. The differences between the SemSim- and SlotFill-based approaches can be attributed to the fact that the latter were trained for each domain/language. For all domains in English, the precision achieved via the exploitation of (the majority of) web corpora exceeds the respective precision of seed corpora. The low-level induction using the best harvested web corpus outperformed the precision yielded by almost all baseline corpora for both domains. The performance of the low-level rule induction is affected by the in-domainness of the selected sentences as indicated by the varying precision scores obtained for different filtering techniques. Regarding corpora creation, a good filtering scheme can lead up to 100% relative improvement in rule precision compared to the absence of filtering (i.e., random selection of sentences). The number of examples that are used as seeds was found to significantly affect performance, i.e., using more seeds leads to better performance. This is especially true for high-level rule induction. Both types of induction algorithms were successfully applied within the human-in-the-loop framework yielding good results during sessions of reasonable time duration.

Based on the experimental results, harvesting is shown to be a plausible approach for corpora creation for both domains and languages investigated. Specifically for the travel domain it was shown that in terms of richness the automatically harvested corpora outperformed the in-domain baseline corpora. Regarding web query creation, we have demonstrated that it is possible to estimate the quality of queries using a cheap yet effective method that relies only on a generic corpus and is directly applicable across languages and domains. Among the two features employed for the filtering of corpora, sentence perplexity was found to be superior compared to using (the pragmatic filtering-based) salient word/terms. The quality of the harvested corpora was further evaluated taking into account the precision of the induction algorithms. The web corpora were found to yield comparable or sometimes higher precision compared to the in-domain corpora. The same observation holds for the crowdsourced corpora (detailed in previous work Palogiannidi et al., 2014), where the quality of the collected data plays a major role regarding the performance of the induction algorithms.

Web harvesting techniques and evaluation procedures presented in this paper are also relevant for training statistical grammars for spoken dialogue systems, e.g., for call routing applications. We are also working towards an integrated interface for grammar induction and authoring that champions an incremental human-in-the-loop approach

utilizing the research results from this paper. Algorithmic improvements, especially in the feature extraction and fusion between different algorithms presented here are also possible in future work. It is also important to investigate how to include spontaneous speech in the automatically induced grammars.

Overall, we have shown that the proposed algorithms for web harvesting and grammar induction can produce good results and are portable across domains and languages.

Acknowledgments

This work has been partially funded by the projects SpeDial (www.spedial.eu) and PortDial (www.portdial.eu), supported by the EU-IST 7-th Framework Programme (FP7), with grant numbers 611396 and 296170, respectively. The authors wish to thank Dr. Manolis Tsangaris and Vassiliki Kouloumenta for their contribution to the development of the frontend of the grammar induction system according to the human-in-the-loop paradigm.

References

Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., 2012. Semeval-2012 task 6: a pilot on semantic textual similarity. In: Proceedings of the First Joint Conference on Lexical and Computational Semantics, pp. 385–393.

Ambati, V., Vogel, S., 2010. Can crowds build parallel corpora for machine translation systems? In: Proceedings of the NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, pp. 62–65.

Athanasopoulou, G., Klasinas, I., Georgiladakis, S., Iosif, E., Potamianos, A., 2014. Using lexical, syntactic and semantic features for non-terminal grammar rule induction in spoken dialogue systems. In: Proceedings of the Spoken Language Technology Workshop (SLT) Workshop.

Beltagy, I., Erk, K., Mooney, R., 2014. Semantic parsing using distributional semantics and probabilistic logic. In: Proceedings of the of Association for Computational Linguistics Workshop on Semantic Parsing (ACL-SP 2014).

Bisazza, A., Klasinas, I., Cettolo, M., Federico, M., 2010. FBK @ IWSLT 2010. In: Proceedings of the International Workshop on Spoken Language Translation (IWSLT), pp. 53–58.

Brill, E., 1995. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. Comput. Linguist. 21 (4), 543–565.

Buzek, O., Resnik, P., Bederson, B., 2010. Error driven paraphrase annotation using mechanical turk. In: Proceedings of the NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, pp. 217–221.

Callison-Burch, C., Dredze, M., 2010. Creating speech and language data with Amazon's mechanical turk. In: Proceedings of the NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, pp. 1–12.

Chen, S., 1995. Bayesian grammar induction for language modeling. In: Proceedings of the Association for Computational Linguistics (ACL), pp. 228–235.

Cramer, B., 2007. Limitations of current grammar induction algorithms. In: Proceedings of the Association for Computational Linguistics (ACL): Student Research Workshop, pp. 43–48.

Denkowski, M., Al-Haj, H., Lavie, A., 2010. Turker-assisted paraphrasing for english-arabic machine translation. In: Proceedings of the NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, pp. 66–70.

Frantzi, K., Ananiadou, S., 1997. Automatic term recognition using contextual cues. In: Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI).

Gao, J., Goodman, J., Li, M., Lee, K., 2002. Toward a unified approach to statistical language modeling for Chinese. ACM Trans. Asian Lang. Inf. Process. (TALIP) 1 (1), 3–33.

Georgiladakis, S., Unger, C., Iosif, E., Walter, S., Cimiano, P., Petrakis, E., Potamianos, A., 2014. Fusion of knowledge-based and data-driven approaches to grammar induction. In: Proceedings of the Interspeech.

Hakkani-Tür, D., Tur, G., Heck, L., Celikyilmaz, A., Fidler, A., Hillard, D., Iyer, R., Parthasarathy, S., 2011. Employing web search query click logs for multi-domain spoken language understanding. In: Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding.

Hakkani-Tür, D., Tur, G., Iyer, R., Heck, L., 2012. Translating natural language utterances to search queries for SLU domain detection using query click logs. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing.

Harris, Z., 1954. Distributional structure. Word 10 (23), 146–162.

Heck, L., Hakkani-Tür, D., 2012. Exploiting the semantic web for unsupervised spoken language understanding. In: Proceedings of the IEEE Spoken Language Technology Workshop.

Heck, L., Hakkani-Tur, D., Tur, G., 2013. Leveraging knowledge graphs for web-scale unsupervised semantic parsing. In: Proceedings of the Interspeech.

Iosif, E., Potamianos, A., 2007. Unsupervised semantic similarity computation using web search engines. In: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, pp. 381–387.

Iosif, E., Tegos, A., Pangos, A., Fosler-Lussier, E., Potamianos, A., 2006. Unsupervised combination of metrics for semantic class induction. In: Proceedings of the IEEE/ACL International Workshop on Spoken Language Technology (SLT).

Irvine, A., Klementiev, A., 2010. Using mechanical turk to annotate lexicons for less commonly used languages. In: Proceedings of the NAACL-HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, pp. 108–113.

Javaparser 1.4. http://www.code.google.com/p/javaparser/.
Jurafsky, D., Martin, J., 2009. Speech and Language Processing an Introduction to Natural Language Processing, Computational Linguistics, and Speech. Pearson Education Inc.
Jurcıcek, F., Keizer, S., Gašic, M., Mairesse, F., Thomson, B., Yu, K., Young, S., 2011. Real user evaluation of spoken dialogue systems using Amazon mechanical turk. In: Proceedings of the Interspeech, pp. 3061–3064.
Jurčíček, F., Gašić, M., Keizer, S., Mairesse, F., Thomson, B., Yu, K., Young, S., 2009. Transformation-based learning for semantic parsing. In: Proceedings of the Interspeech, pp. 2719–2722.
Klasinas, I., Potamianos, A., Iosif, E., Georgiladakis, S., Mameli, G., 2013. Web data harvesting for speech understanding grammar induction. In: Proceedings of the Interspeech, pp. 2733–2737.
Lafferty, J.D., McCallum, A., Pereira, F.C.N., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289.
Lari, K., Young, S., 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. Comput. Speech Lang. 4 (1), 35–56.
Levenshtein, V., 1966. Binary codes capable of correcting deletions, insertions and reversals. Sov. Phys. Dokl. 10, 707.
Lingua-Sentence-1.04, http://search.cpan.org/~achimru/Lingua-Sentence-1.04/.
Liu, J., Cyphers, S., Pasupat, P., McGraw, I., Glass, J., 2012. A conversational movie search system based on conditional random fields. In: Proceedings of the Interspeech, pp. 2454–2457.
Mairesse, F., Gašić, M., Jurčíček, F., Keizer, S., Thomson, B., Yu, K., Young, S., 2009. Spoken language understanding from unaligned data using discriminative classification models. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4749–4752.
Marelli, M., Bentivogli, L., Baroni, M., Bernardi, R., Menini, S., Zamparelli, R., 2014. SemEval-2014 Task 1: evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In: Proceedings of the International Workshop on Semantic Evaluation (SemEval).
McCrae, J., de Cea, G.A., Buitelaar, P., Cimiano, P., Declerck, T., Gomez-Perez, A., Garcia, J., Hollink, L., Montiel-Ponsoda, E., Spohr, D., 2012. Interchanging lexical resources on the semantic web. Lang. Resour. Eval. 46 (4), 701–719.
McGraw, I., Glass, J., Seneff, S., 2011. Growing a spoken language interface on amazon mechanical turk. In: Proceedings of the Interspeech, pp. 3057–3060.
Meng, H., Siu, K., 2002. Semi-automatic acquisition of semantic structures for understanding domain-specific natural language queries. IEEE Trans. Knowl. Data Eng. 14 (1), 172–181.
Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Tur, G., Yu, D., Zweig, G., 2015. Using recurrent neural networks for slot filling in spoken language understanding. IEEE/ACM Trans. Audio Speech Lang. Process 23 (3), 530–539.
Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781.
Milward, D., Beveridge, M., 2003. Ontology-based dialogue systems. In: Proceedings of the Third Workshop on Knowledge and Reasoning in Practical Dialogue Systems – 18th International Joint Conference on Artificial Intelligence.
Misu, T., Kawahara, T., 2006. A bootstrapping approach for developing language model of new spoken dialogue systems by selecting web texts. In: Proceedings of the Interspeech, pp. 9–12.
Mitchell, J., Lapata, M., 2010. Composition in distributional models of semantics. Cognit. Sci. 34 (8), 1388–1429.
Ng, T., Ostendorf, M., Hwang, M., Siu, M., Bulyko, I., Lei, X., 2005. Web data augmented language models for Mandarin conversational speech recognition. In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 589–592.
NuGram Platform. http://nugram.nuecho.com/welcome/.
Palogiannidi, E., Klasinas, I., Potamianos, A., Iosif, E., 2014. Spoken dialogue grammar induction from crowdsourced data. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3211–3215.
Papineni, K., Roukos, S., Ward, T., Zhu, W., 2002. BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the Fortieth Annual Meeting on Association for Computational Linguistics, pp. 311–318.
Pardal, J.P., 2007. Dynamic use of ontologies in dialogue systems. In: Proceedings of the NAACL-HLT 2007 Doctoral Consortium. ACL, pp. 25–28.
Pargellis, A., lussier A. Potamianos, E.F., Lee, C.-H., 2001. A comparison of four metrics for auto-inducing sematnic classes. In: Proceedings of the Automatic Speech Recognition and Understanding Workshop.
Pargellis, A., Fosler-Lussier, E., Lee, C.H., Potamianos, A., Augustine, T., 2004. Auto-induced semantic classes. Speech Commun. 43 (3), 183–203.
Peng, B., Yao, K., 2015. Recurrent neural networks with external memory for language understanding, arXiv preprint arXiv:1506.00195.
Pieraccini, R., Suendermann, D., 2012. Data-driven methods in industrial spoken dialog systems. In: Lemon, O., Pietquin, O. (Eds.), Data-driven Methods for Adaptive Spoken Dialogue Systems: Computational Learning for Conversational Interfaces. Springer, pp. 151–170.
Ponvert, E., Baldridge, J., Erk, K., 2011. Simple unsupervised grammar induction from raw text with cascaded finite state models. In: Proceedings of the Association for Computational Linguistics (ACL), pp. 1077–1086.
Potamianos, A., Kuo, H.-K. J., 2000. Statistical recursive finite state machine parsing for speech understanding. In: Proceedings of the Interspeech, pp. 510–513.
Pradhan, S., Ward, W., Hacioglu, K., Martin, J., Jurafsky, D., 2004. Shallow semantic parsing using support vector machines. In: Proceedings of the NAACL-HLT, pp. 233–240.
Prévot, L., Huang, C., Calzolari, N., Gangemi, A., Lenci, A., Oltramari, A., 2010. Ontology and the lexicon: a multi-disciplinary perspective. Ontology and the Lexicon: A Natural Language Processing Perspective. Cambridge University Press, pp. 3–24.
Ranta, A., 2004. Grammatical framework: A type-theoretical grammar formalism. J. Funct. Program. 14 (2), 145–189.

Raux, A., Langner, B., Bohus, D., Black, A., Eskenazi, M., 2005. Let's go public! Taking a spoken dialog system to the real world. In: Proceedings of the Interspeech.

Raymond, C., Béchet, F., Mori, R.D., Damnati, G., 2006. On the use of finite state transducers for semantic interpretation. Speech Commun. 48 (3–4), 288–304. Spoken Language Understanding in Conversational Systems

Raymond, C., Riccardi, G., 2007. Generative and discriminative algorithms for spoken language understanding. In: Proceedings of the Interspeech, pp. 1605–1608.

Sarikaya, R., 2008. Rapid bootstrapping of statistical spoken dialogue systems. Speech Commun. 50 (7), 580–593.

Sethy, A., Narayanan, S., Ramabhadran, B., 2002. Data driven approach for language model adaptation using stepwise relative entropy minimization. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 177–180.

Sha, F., Pereira, F., 2003. Shallow parsing with conditional random fields. In: Proceedings of the NAACL-HLT, pp. 134–141.

Shi, Y., Yao, K., Chen, H., Yu, D., Pan, Y.-C., Hwang, M.-Y., 2016. Recurrent support vector machines for slot tagging in spoken language understanding. In: Proceedings of the NAACL-HLT, pp. 393–399.

Stoilos, G., Stamou, G., Kollias, S., 2005. A string metric for ontology alignment. In: Proceedings of the Semantic Web-ISWC 2005. Springer, pp. 624–637.

Stolcke, A., 2002. SRILM-an extensible language modeling toolkit. In: Proceedings of the Interspeech.

Sungbok, L., Ammicht, E., Fosler-Lussier, E., Kuo, H.-K., Potamianos, A., 2002. Spoken dialogue evaluation for the bell labs communicator system. In: Proceedings of the Second International Conference on Human Language Technology Research, pp. 275–279.

Tur, G., Jeong, M., Wang, Y.-Y., Hakkani-Tür, D., Heck, L., 2012. Exploiting the semantic web for unsupervised natural language semantic parsing. In: Proceedings of the Interspeech.

Vapnik, V., 1998. Statistical Learning Theory. Wiley.

Vukotic, V., Raymond, C., Gravier, G., Is it time to switch to word embedding and recurrent neural networks for spoken language understanding? In: Proceedings of the Interspeech.

Wagner, R., Fischer, M., 1974. The string-to-string correction problem. J. ACM (JACM) 21 (1), 168–173.

Wang, W., Bohus, D., Kamar, E., Horvitz, E., 2012. Crowdsourcing the acquisition of natural language corpora: methods and observations. In: Proceedings of the Spoken Language Technology Workshop (SLT), pp. 73–78.

Wang, Y., Acero, A., 2006. Rapid development of spoken language understanding grammars. Speech Commun. 48, 390–416.

Wang, Y.-Y., 2001. Robust spoken language understanding in MiPad. In: Proceedings of the Eurospeech.

Yang, Z., Li, B., Zhu, Y., King, I., Levow, G., Meng, H., 2010. Collection of user judgments on spoken dialog system with crowdsourcing. In: Proceedings of the Spoken Language Technology Workshop (SLT), pp. 277–282.

Yao, K., Peng, B., Zhang, Y., Yu, D., Zweig, G., Shi, Y., 2014. Spoken language understanding using long short-term memory neural networks. In: Proceedings of the IEEE workshop on Spoken Language Technology, pp. 189–194.

Zhu, Y., Yang, Z., Meng, H., Li, B., Levow, G., King, I., 2010. Using finite state machines for evaluating spoken dialog systems. In: Proceedings of the Spoken Language Technology Workshop (SLT), pp. 478–483.