Dialog Natural Language Understanding using a Generic Textual Inference System

Erel Segal-haLevi, Ido Dagan
Department of Computer Science, Bar-Ilan University
Ramat-Gan, Israel
erelsgl@gmail.com,dagan@cs.biu.ac.il

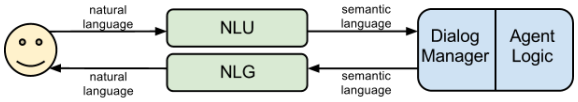

Figure 1: Standard dialog system architecture

Abstract

One of the components of a dialog system is the Natural Language Understanding (NLU) component. This component accepts natural language text, and returns the meaning of that text, in some formal application-specific meaning representation. One of the difficulties in building NLU components is the variability in natural language - the many different ways by which a human can express the same meaning. We propose to tackle this difficulty by using a generic Textual Entailment (TE) system - a system that can calculate, for each pair of texts, whether the meaning of one of them can be inferred from the other. A single TE system can be used for various NLU components in various domains.

1 Introduction and Previous Work

1.1 Natural language understanding

One of the tasks of a dialog system is to understand what the user says. This is the task of the Natural Langauge Understanding component (NLU, figure 1). The *input* to this component is natural language text, which can come directly from the user via a chat interface, or indirectly via an automatic speech recognizer.

(see figure 1). The *output* of this component is the meaning of the text, represented in some formal semantic language. This output is sent to the dialog manager and to an application-specific agent, which use it to decide what to do next - whether to answer the human, argue with the human, ask a question, etc., depending on the application.

Current approaches to NLU can be divided into *shallow* approaches - that only try to capture some of the information intended by the human, and *deep* approaches - that try to capture its entire meaning. These can further be divided into approaches based on *manual* authoring, versus *automatic* machine-learning.

Shallow NLU components rely on the surface form of the human utterance, and try to get a general idea about what the human has said:

Shallow-Manual methods use pattern-matching, especially regular expressions. For example, the regular expression pattern "I (can)? (offer OR give) a salary of (.*) NIS per month" can be used to detect the fact that the human makes a salary offer. Such methods were used in the first documented dialog systems(Weizenbaum, 1966), and are still widely used today, especially in simulation games (Mateas and Stern, 2004) and chatter-bots (Wilcox, 2011).

Shallow-automatic methods use statistical text categorization and machine learning. For example, samples such as "I give a salary of 20000 NIS" and "I can offer 30000 NIS per month" can be used to train a text categorizer that can identify similar sentences such as "I offer a 40000 NIS salary" as belonging to the "offer" class. Systems that use such methods include, for example, virtual humans that answer domain-specific questions (Leuski et al., 2006; Leuski and Traum, 2010) and automatic replies to technical support requests(Marom and Zukerman, 2009).

Shallow methods are relatively easy to implement and are efficient in run-time. However, they are not useful for applications that require deep and detailed understanding of the human intention. As an example, consider a dialog system that plays the role of a job candidate, negotiating his/her job conditions with a human employer. The negotiation dialog may contain an utterance such as "If you agree to my salary offer, I will agree to give you a car". Shallow methods will probably identify the keyword "agree". However, in the context of negotiation, this utterance is not at all an agreement - it is a conditional statement, whose details reveal important information about the preferences of the speaker. In order to detect such fine details, a deeper processing method is required.

Deep NLU components rely on syntactic and semantic analysis of the human utterance, and try to map it into a formal representation of its meaning. There are various possible meaning representations. The most general representation language is first-order logic, but usually a simpler language is used - a list of key-value

$$S \rightarrow I \text{ offer N / OFFER(N)}$$

$$S \rightarrow I \text{ give N / OFFER(N)}$$

$$N \rightarrow salary\,of\,X\,NIS \,/\, SALARY\,=\,X$$

$$X \rightarrow 20000/20000$$

$$X \rightarrow 30000/30000$$

Figure 2: A sample synchronous grammar that can be used for deep NLU. S is the grammar root, N and X are nonterminals. A forward slash (/) separates the source form (natural language) from its translation (semantic language).

pairs, a feature-structure, or an application-specific formal language.

Deep-Manual methods employ hand-written synchronous grammars, or hand-written grammars with semantic tagging. For example, the synchronous grammar in figure 2 can be used to translate the sentence *"I give a salary of 20000 NIS"* to the semantic representation *OFFER(SALARY=20000)*. Grammars used in practical systems range from simple context-free grammars such as *SRGS+SISR*(Van Tichelen and Burke, 2007) and *Phoenix*(Ward and Issar, 1994), to complex context-sensitive grammars such as *Grammatical Framework*(Perera and Ranta, 2007).

Deep-automatic methods use syntactic-semantic parsers trained on manually annotated dialogs, in which each utterance is annotated with the corresponding formal meaning representation. These methods have a potential for building scalable deep dialog systems, however, to achieve acceptable performance they require large dialog corpora. As an example, in the Classic Project, a recent research project of the EU, dialogs were collected in order to train syntactic-semantic parsers (Gesmundo et al., 2009), however, the dialogs collected did not display enough language variability in order to train reliable models, so the researchers had to resort to using the *Phoenix* parser with a manually-written grammar (Henderson et al., 2011).

An additional difficulty with deep methods is that the training data is tied to the application-specific semantic representation, so new training data should be collected for new applications that employ different semantic representations.

1.2 Natural language variability

The task of authoring NLU components, regardless of their type, is greatly complicated by the problem of *natural language variability* - the fact that, in natural language, there are many ways to express the same meaning. For example, the first and second rules in the synchronous grammar of Figure 2 on page 2 are required because the word "offer", in the context of negotiation, means the same as "give". If we build another negotiation dialog system, with a different semantic representation, we again have to include two different rules for "offer" and "give", or, if we use an automatic method, collect training data that cover both "offer" and "give". In practice, of course, there are many more ways to express the notion of an offer, as well as any other predicate of concept relevant for the target dialog domain.

To address this problem, a few dialog systems use *WordNet*(Fellbaum, 1998) - a lexicon providing comprehensive lexical semantic relationships - as an auxiliary resource. For instance, *ChatScript*(Wilcox and Wilcox, 2011) allows authors to create patterns that explicitly refer to WordNet, so, for example, instead of writing two patterns such as "I give salary" and "I offer salary", an author can write a single pattern similar to "I {wordnet:give-1} salary", where the middle token matches all synonyms and hyponyms of the 1st sense of "give" in Wordnet. *NUBEE*(Core and Moore, 2004) used WordNet for unknown words: when the human utterance contained words that are not in the application-specific semantic grammar, the system searched WordNet for synonyms that are in the grammar.

But WordNet is only a single resource. In recent years, the NLP community has developed a much larger collection of semantic knowledge resources, and generic textual inference and similarity engines that utilize them, which are still, to the best of my knowledge, not leveraged for dialog NLU. The dialog field may benefit from using a generic framework for coping with natural language variability - and in particular from the *textual entailment* framework.

1.3 Textual Entailment

Textual Entailment (TE) is a framework for recognizing semantic relations between natural language texts. A TE system focuses on deciding whether a certain natural-language text (called the Text, T) *entails* another natural-language fragment (called the Hypothesis, H), i.e., whether the meaning of H can be inferred from the meaning of T. For example, an ideal TE system can identify that "I offer 20000 NIS per month" entails "I can give you a monthly salary of 20000 NIS", i.e., their meaning is equivalent although the words are different.

TE systems utilize a variety of **entailment knowledge resources**: *Lexical resources*, such as *WordNet*(Fellbaum, 1998) or *VerbOcean*(Chklovski and Pantel, 2004), that describe the semantic relations between single words (i.e. "offer" and "give"); *Lexical-Syntactic resources*, such as *DIRT*(Lin and Pantel, 2001), and *Generic-Syntactic resources*(Lotan, 2012), that describe semantic relations between parts of syntactic parse trees (i.e. "I can give you X" and "you can get X"). TE systems use a variety of methods to integrate all these resources, for example, finding alignments between T and H or trying to "prove" H from T.

The NLP lab at Bar Ilan University has been developing TE technology for several years, and its open-source TE engine, BIUTEE, has achieved state-of-the-art results in the international Recognizing Textual En-

tailment challenge(Stern and Dagan, 2012). In addition to the specific task of recognizing textual entailment, this TE has also been used for applications such as information extraction(Bronstein and Segal-haLevi, 2013). Our current goal is to use TE for dialog NLU.

2 Methodology - Leveraging Textual Entailment to Dialog NLU

Using a generic TE system, the designer of an NLU component does not need to address all the possible forms in which a certain meaning can be expressed, but only a *representative sentence* for each meaning. For example, the sentence "I offer a salary of {number}" may be a representative sentence for the action of offering a salary.

In run time, the TE will get the actual human utterance, check it against all those pre-specified representative sentences, and return only the representatives which are entailed by the human utterance. These representatives can now be translated to the semantic language used by the application. For example, if the human says "I can give a wage of {number}", the TE will detect that it entails the representative sentence "I offer a salary of {number}", and translate it to the appropriate semantic representation, e.g., "OFFER(Salary={number})".

This approach greatly reduces the problem of addressing natural language variability in authoring NLU components. Instead of thinking of all the various ways in which a human may offer a salary, the author may just pick a single way to convey this meaning, and rely on the TE system to do the rest. Thus, the synchronous grammars for translating natural language to semantic language can be much smaller and easier to write.

This method allows each NLU author to take advantage of the work done by many researchers, that have been working on the generic TE problem for many years. Many entailment knowledge resources and entailment recognition algorithms were developed for generic TE, and tested in other applications, such as information extraction. Their power can now be used for DS.

Additionally, even if dialog NLU requires new entailment knowledge resources or new entailment algorithms, it is still more efficient to develop these resources and algorithms once, in the context of the TE framework, so that they may be used in different dialog systems and in different domains, which might be utilizing different semantic representation languages.

Our method can be seen as decoupling the NLU task into two parts: first, convert *generic* natural language into *restricted* natural language, the language of the representative sentences, using generic TE approaches and tools; then, translate the representative sentences from the restricted language into the domain-specific semantic language, using a standard synchrounous grammar (See figure 3).

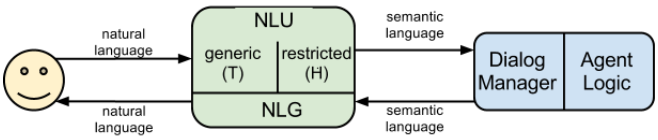

Figure 3: Proposed dialog system architecture. The TE is inside the NLU, converting each T to an H or set of H's with the same meaning.

3 Experimental Framework

3.1 Application

Our main project is building an *automated negotiation agent,* that can negotiate effectively with humans in *natural language*. This project is done in collaboration between Bar Ilan's AI lab and NLP lab. The AI lab develops an automated agent that that can negotiate with humans via a menu interface(Lin and Kraus, 2010), and we at the NLP lab are responsible for adding an NLU capability to that agent.

The agent understands a certain set of actions, such as "Offer(Issue,Value)", "Accept" or "Reject". These actions can be described using a formal semantic language, that can be defined by a context-free grammar. The goal of our NLU component is to translate a given human utterance into the correct semantic representation, which is spanned by the context-free grammar of the agent.

3.2 Data collection

In order to test our method, we collected data in 3 different ways:

1. human-human chat: We built a web-based system, where two humans can negotiate via a chat interface. [1]We let about 20 pairs of students play the game. We gave them exact instructions about the negotiation scenario, the issues under discussion, and the values available in each issue. We collected **136** different natural-language sentences, and manually annotated them with the correct semantic representation.

2. human-agent chat: We built another web-based system, where a human negotiates with an automated agent(Lin and Kraus, 2010) via a chat interface. Since we don't have a working NLU component yet, we used the wizard-of-oz scheme - the first author hiddenly translated each human utterance, in real time, to the semantic representation of the agent, and sent it to the agent. The agent's replies were automatically translated to natural language by a simple template-based NLG unit. We collected **100** different sentences, which got their correct semantic representation based on the wizard's action.

3. paraphrasing: We created a representative natural language sentence for each semantic representation in the agent's language (each representation that can be

[1]The entry point to the negotiation game is: http://negochat.azurewebsites.net/negochat%5FJobCandidate/beginner

generated by the agent's context-free grammar; totally about 130 sentences). We created an Amazon Mechanical Turk task, where we asked turkers to create paraphrases for each sentence. We collected about 15 paraphrases per sentence, for a total of **2000** sentences.

Each data collection method has its advantages and disadvantages:

1. human-human chats are quite expensive, because they require pairs of humans to enter the web-chat system on the same time. The sentences collected that way are usually long and natural, for example: "i would rather like a 20000 NIS salary. this is mandatory to me to have a good salary as i believe working conditions affect directly my effectiveness".

2. human-agent chats are also expensive, because they require a "wizard" (usually a researcher) to be available while people enter the game. The sentences received are short and simple, because the sentences generated by the NLG are simple, and the human players change their style accordingly, for example: "20000 NIS per month".

3. paraphrasing is the cheapest method: turkers can enter at any time, no coordination with others is required, and the cost is low (less than 0.1 dollar per sentence). However, the sentences collected that way are not natural, since they are not generated in the context of actual negotiation. For example: "The offer I am presenting is a salary of 20000 NIS a month".

3.3 Results

Currently, we don't have a TE system that was built with dialog NLU in mind (see the next section). As a first, preliminary baseline, we ran our NLU scheme based on BIUTEE (Stern and Dagan, 2012), our open-source TE engine, that was designed to solve TE problems in news text. We ran this NLU component on the 3 datasets. For each human sentence in the dataset, we used BIUTEE to detect the set of *all* semantic representations that are entailed by that sentence. We compared this set to the gold standard, and calculated precision, recall and F1.

As expected, the results were poor: about 18% F1 for the human-human dataset and the paraphrase dataset, and 27% F1 for the human-agent dataset (this makes sense because the sentences in the human-agent dataset are simpler). The results are slightly different with different configurations of BIUTEE, but are still in the same order of magnitude.

4 Discussion

Why do we get such low results?

BIUTEE is made of several components, none of them was designed with dialog systems in mind:

1. Standard language preprocessing tools, specifically *syntactic parser*s, were designed and trained on news corpora. Natural language used in dialogs is informal, and often ungrammatical. This is especially true in spoken dialogs that are transcribed automatically by a speech recognizer, but also in chat-based dialogs (which are the focus of our research), as chatters tend to use non-standard shorthand notations, omit punctuation marks and make many spelling mistakes. An example from our human-human dataset is: *"fine ill givbe you no agreement and 8 working hours"* (sic). This makes the standard language processing tools much more prone to errors, which propagate and affect the accuracy of the entailment recognizer. A possible solution is to use other kinds of textual inference engines, that depend less heavily on syntax, such as Semantic Text Similarity (STS)(Bär et al., 2012).

2. Generic entailment resources, such as WordNet or DIRT, are not always useful for NLU in specific domains. For example, WordNet contains the entailment rule *"know" -¿ "accept*", which makes sense in general, but may be misleading in our specific domain. On the other hand, the rule "*suggest*" -¿ "*offer*", which is true in our domain, does not exist in any of our resources. Therefore, one of our research goals is to find ways to create entailment knowledge resources that are useful for a specific domain, while leveraging existing general-domain resources.

3. The textual entailment engine itself was tuned to textual entailment pairs from news-related corpora. Although it can be trained on pairs generated from the negotiation domain, our experiments show that this training doesn't improve performance. A possible reason is that the set of features (types of entailment operations) used by the engine is not fine enough to capture the entailment phoenomena in our corpus. One of our research goals is to refine the feature set by adding new features, for example, lemma-dependent insertions and moves.

5 Summary and conclusions

Textual entailment has a potential to make life easier for developers of dialog NLU systems. To realize the potential, a lot of work needs to be done. New language tools, entailment resources and entailment engines need to be developed.

References

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures. In *SemEval*.

Ofer Bronstein and Erel Segal-haLevi. 2013. Using Textual Entailment with Variables for KBP Slot Filling Task. Technical report.

Timothy Chklovski and Patrick Pantel. 2004. VerbO-

cean: Mining the Web for Fine-Grained Semantic Verb Relations. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 33–40, Barcelona, Spain, July. Association for Computational Linguistics.

Mark G. Core and Johanna D. Moore. 2004. Robustness versus Fidelity in Natural Language Understanding. In Robert Porzel, editor, *HLT-NAACL 2004 Workshop: 2nd Workshop on Scalable Natural Language Understanding*, pages 1–8, Boston, Massachusetts, USA, May. Association for Computational Linguistics.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press, May.

Andrea Gesmundo, James Henderson, Paola Merlo, and Ivan Titov. 2009. A latent variable model of synchronous syntactic-semantic parsing for multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 37–42, Stroudsburg, PA, USA. Association for Computational Linguistics.

James Henderson, Francois Mairesse, Lonneke van der Plas, and Paola Merlo. 2011. Two trained semantic decoders for the Appointment Scheduling task. Technical Report Deliverable 2.4, January.

Anton Leuski and David Traum. 2010. Practical Language Processing for Virtual Humans. In *IAAI*.

Anton Leuski, Ronakkumar Patel, David Traum, and Brandon Kennedy. 2006. Building effective question answering characters. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, SigDIAL '06, pages 18–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

Raz Lin and Sarit Kraus. 2010. Can automated agents proficiently negotiate with humans? *Commun. ACM*, 53(1):78–88, January.

Dekang Lin and Patrick Pantel. 2001. DIRT - discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '01, pages 323–328, New York, NY, USA. ACM.

Amnon Lotan. 2012. A Syntactic Rule-base for Textual Entailment. Master's thesis, Bar Ilan University.

Yuval Marom and Ingrid Zukerman. 2009. An Empirical Study of Corpus-Based Response Automation Methods for an E-mail-Based Help-Desk Domain. *Computational Linguistics*, 35(4):597–635, October.

Michael Mateas and Andrew Stern. 2004. Natural Language Understanding in Façade: Surface Text Processing. In *Proceedings of the Conference on Technologies for Interactive Digital Storytelling and Entertainment*.

Nadine Perera and Aarne Ranta. 2007. Dialogue system localization with the GF resource grammar library. In *Proceedings of the Workshop on Grammar-Based Approaches to Spoken Language Processing*, SLP '07, pages 17–24, Stroudsburg, PA, USA. Association for Computational Linguistics.

Asher Stern and Ido Dagan. 2012. BIUTEE: A Modular Open-Source System for Recognizing Textual Entailment. In *Proceedings of the ACL 2012 System Demonstrations*, pages 73–78, Jeju Island, Korea, July. Association for Computational Linguistics.

Luc Van Tichelen and Dave Burke. 2007. Semantic Interpretation for Speech Recognition (SISR) Version 1.0. Technical report.

Wayne Ward and Sunil Issar. 1994. Recent improvements in the cmu spoken language understanding system. In *Proceedings of the ARPA Workshop on Human Language Technology*.

Joseph Weizenbaum. 1966. ELIZA: A Computer Program for the Study of Natural Language Communication between Man and Machine. *Communciations of the ACM*, 9(1).

Bruce Wilcox and Sue Wilcox. 2011. Suzette, the Most Human Computer (GamaSutra.com website).

Bruce Wilcox. 2011. Beyond Façade: Pattern Matching for Natural Language Applications (GamaSutra.com website).