Error Corrective Learning for Semantic parsing

F. Jurčíček, F. Mairesse, M. Gašić, S. Keizer, B. Thomson, K. Yu, and S. Young
Cambridge University Engineering Department, Trumpington Street, Cambridge, CB2 1PZ, UK
{fj228, farm2, mg436, sk561, brmt2, ky219, sjy}@eng.cam.ac.uk

Abstract

a In this paper, we present a semantic parser which transforms initial naive semantic hypothesis into correct semantics by using a ordered set of rules. These rules are learned automatically from the training corpus with no linguistic knowledge.

1 Introduction

Check how you cal STEP. ECL parser?

Semantic parsing is important part of a spoken dialogue system (Williams and Young, 2007; Thomson et al, 2008). The goal of a semantic parsing is to construct formal meaning representation (semantics) which is directly executable by a dialogue manager. Such semantics is usually defined by a grammar, e.g. LR grammar for GeoQuery domain (Wong and Mooney, 2006) **better paper???**, which is designed by a domain expert and easy to interpret by a dialogue manager or question answering system.

Semantic parsing can be understood as machine translation from a natural language to a formal language. First, we do not not have formal grammar for natural language is ungrammatical, include hesitations, and very often only fragments of complete sentences, e.g. "Boston to Miami tomorrow".

2 Related work

There has been a large amount of work done on learning to map sentences into their semantics. Many different techniques have been considered including machine translation (Wong and Mooney, 2006)

techniques using inductive logic programing (**?**)

support vector machines (Mairesse et al, 2009) and tree kernels (Kate, 2008)

markov logic networks (Meza et al, 2008a; Meza et al, 2008b)

automatic induction of combinatory categorical grammar (Zettlemoyer and Collins, 2007). Z & C mention their problems with spoken speech =¿ non-compositionality After Zettlemoyer & Collins implemented second-pass decoding which relaxed constraints on the input utterances, they achieved state-of-the-art results.

(He and Young, 2006) developed a parser for the ATIS domain also takes utterances paired with semantics as input. Their parser approximate a push-down automaton with semantic concepts as non-terminal symbols.

transformation techniques used in the system SILT (Kate, 2005). SILT learns transformation rules which sequentially rewrites an input utterance into its formal representation. Our ECL parser differ in the way the semantics is constructed instead of transforming input utterance. Instead of rewriting an input utterance, we transform initial naive semantic hypothesis. As a result, we can use in input words several times to trigger transformations of the output. This extends the ability to handle non-compositionality phenomena in spoken language.

Transformation-based Error-driven Learning - Some Advances in Transformation-Based Part of Speech Tagging - Brill (1994)

(Kate, 2008)

R. J. Kate, Y. W. Wong, and R. J. Mooney. 2005. Learning to transform natural to formal languages.

In Proc. of AAAI-05, pages 1062?1068, Pittsburgh, PA, July.

L. R. Tang and R. J. Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In Proc. of ECML-01, pages 466? 477, Freiburg, Germany.

Subsection **??**).

3 Transformation-based parser

This section describes the transformation-based parser. First of all, we describe rule templates used to generate rules for the rule inference process. Secondly, we describe the learning process. Finally, we describe parsing algorithm.

3.1 Rule templates

The learning algorithm uses rule templates to instantiate rules which are subsequently tested by the learning algorithm. Each rule template is composed of a trigger and transformation. **A trigger controls whether a transformation of hypothesis can be performed**. Each trigger question either input utterance or output semantics. As a results each trigger contains one or several condition:

- The utterance contains n-gram N?
- The utterance contains skipping[1] bigram B?
- The semantics dialogue act equals to D?
- The semantics contains slot S?

If a trigger contains more than one condition, then all conditions must be satisfied. **Get rid of the questions.**

A transformation performs one of these operation:

- substitute a dialogue act type
- add a slot
- delete a slot
- substitute a slot

As substitution can either substitute a whole slot, an equal sign in the slot, or a slot name.

[1]A skipping bigram is bigram which skips one or more words between words in the bigram

3.2 Example of Parsing

The semantic parser transforms initial naive semantic hypothesis into correct semantics by using a ordered set of rules. The parsing is composed of **three** steps:

1. initial semantics is assigned as hypothesis
2. sequentially apply all rules[2]
3. output hypothesis semantics

3.3 Learning

Rules are learned sequentially:

1. initial semantics is assigned as hypothesis to each input utterance
2. repeat as long as the number of errors[3] on training set decreases
   (a) generate all possible rules which correct at least one error in the training set
   (b) measure number of corrected errors for each rule
   (c) select the best rule
   (d) apply the selected rule
3. prune rules

The make the parser more robust, we increase robustness of the parser by the following steps.

First, the number of plausible slot values for each slot is usually very high. As a result, we replace all lexical realizations from the database, available to **a dialogue manager**, by its slot name in the input utterance. For example, in utterance "find all the flights from cincinnati to the new york city" the lexical realization are replaced as follows: "please find all the flights from city-0 to the city-1". Similarly, we replace slot values in the semantics.

Secondly, to limit overfitting the training data, we prune the rules which are learned at the end of the learning. We sequentially apply each rule on the development set. And we chose the number of rules

[2]Input utterance is not modified by rules. As a result, words from the utterance can be trigger several different transformations.

[3]Number of errors include number of dialogue act substitutions, number of slot insertions, number of slot deletions, number of slot substitutions.

for which the parser gets the highest score on the development data.

First of all, very naive rules are learn are for example Classifier learns to correct its errors STEC can delete an incorrect slot STEC can substitute A slot name of an incorrect slot An equal sign of an incorrect slot

To speed up training, we select not only one best rule but also rules which correct at minimum 80% errors of the best rule.

4 Evaluation

In this section, we evaluate our parser on two distinct corpora, and compare our results with the state-of-the-art techniques and handcrafted rule-based parser.

4.1 Datasets

Our first dataset consists of tourist information dialogues in a fictitious town (TownInfo). The dialogues were collected through user trials in which users searched for information about a specific venue by interacting with a dialogue system in a noisy background. These dialogues were previously used for training dialogue management strategies (Williams and Young, 2007; Thomson et al, 2008). The semantic representation of the user utterance consists of a root dialogue act type and a set of slots which are either unbound or associated with a child value. For example, "What is the address of Char Sue" is represented as request(address='Char Sue'), and "I would like a Chinese restaurant?" as inform(food='Chinese',type='restaurant'). The TownInfo training, development, and test sets respectively contain 8396, 986 and 1023 transcribed utterances. The data includes the transcription of the top hypothesis of the ATK speech recogniser, which allows us to evaluate the robustness of our models to recognition errors (word error rate = 34.4%).

In order to compare our results with previous work (He and Young, 2006; Zettlemoyer and Collins, 2007), we apply our method to the Air Travel Information System dataset (ATIS) (Dahl et al, 1994). This dataset consists of user requests for flight information, for example "Find flight from San Diego to Phoenix on Monday is rerepresented as flight(from.city="San Diego",to.city="Phoenix",departure.day="Monday"). We use 5012 utterances for training, and the DEC94 dataset as development data. As in previous work, we test our method on the 448 utterances of the NOV93 dataset, and the evaluation criteria is the F-measure of the number of reference slot/value pairs that appear in the output semantic (e.g., from.city = New York). He & Young detail the test data extraction process in (He and Young, 2005).

For both corpora are available databases with lexical entries for slot values e.g. city names, airport names, etc.

4.2 Improving disambiguation of long-range dependencies

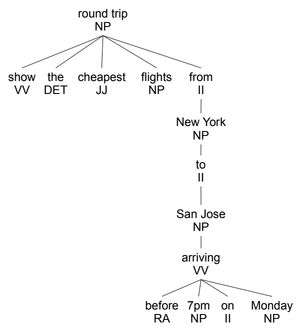

Figure 1: Dependency tree of the sentence "Show the cheapest flights from New York to San Jose arriving before 7pm on Monday" generated by the RASP parser (Briscoe et al, 2006).

Besides simple n-grams and skipping bigrams more complex lexical features can be used. (Kate, 2008) used gold standard word dependencies to capture long-range relationship between words. At its simplest, dependencies tree is one of the most concise ways to describe language syntax. Essentially, each word is viewed as the dependent of one other word, with the exception of a single word which that is the root of the sentence. Kate showed that word dependencies significantly improve semantic parsing because long-range dependencies from an utterance tend to be local in a dependency tree. For example, the words "arriving" and "Monday" are neighbors in the dependency tree but they are four words apart in the sentence (see Figure 1).

Instead of using gold standard word dependencies, we used dependencies provided by RASP dependency parser (Briscoe et al, 2006). First of all, we had to add capitalization and punctuation into the ATIS data to be able to use the RASP parser. The RASP parser without proper capitalization fails to tag ”new“ and ”york“ as NP and instead of this it tags ”new“ as ”JJ“ and 'york' as NP and the dependencies generated by the parser are unsatisfactory. Secondly, we generated new n-gram features from dependency trees. Even though the dependencies generated the RASP parser are no absolutely accurate, the new features increase performance in F-measure on ATIS data.

Secondly, we generated long-range features by using POS tags[4] Our motivation was work of (Meza et al, 2008a; Meza et al, 2008b) who handcrafted features using words ”arrive“, ”arriving“, ”leave“, and ”leaving“. These handcrafted features disambiguate large number of semantic parsing errors in ATIS data because large portion or errors is caused by confusions between concepts ”arrival.time“ and ”departure.time“, ”arrival.day“ and ”departure.day“, etc. To generalize this approach, we want to automatically find features which could disambiguate words like ”Monday“, ”7pm“, and ”Boston“. As a result, we generate a new type of bigrams for a word and the nearest verb, preposition, etc. We use all parts-of-speech provided by RASP and the learning algorithm chose the most discriminative features. Among those learned are not only the words used by Meza-Rui but also words like ”stop“, ”reach“, ”buy“ and prepositions like ”at“, ”from“, ”to“, etc.

Mention why I do not use tree similarity measure as Kate.

4.3 Results

We also compare our models with the handcrafted Phoenix grammar (Ward, 1991) used in the trials (Williams and Young, 2007; Thomson et al, 2008). The Phoenix parser implements a partial matching algorithm that was designed for robust spoken language understanding.

Parser	**Prec**	**Rec**	**F**
TownInfo dataset with transcribed utterances:			
ECL	96.05	94.66	95.35
STC	97.39	94.05	**95.69**
Phoenix	96.33	94.22	95.26
TownInfo dataset with ASR output:			
ECL	92.72	83.42	87.82
STC	94.03	83.73	**88.58**
Phoenix	90.28	79.49	84.54
ATIS dataset with transcribed utterances:			
ECL	96.37	95.12	95.74
STC	96.73	92.37	94.50
HVS	-	-	90.3
MLN	-	-	92.99
PCCG	95.11	96.71	**95.9**

Table 1: Slot/value precision (Prec), recall (Rec) and F-measure (F) for the ATIS and TownInfo datasets. ECL parser is compared with Phoenix parser and STC classifier (Mairesse et al, 2009) on the TownInfo dataset and compared with HVS parser (He and Young, 2006), MLN parser (Meza et al, 2008b), STC classifier, and PCCG parser (Zettlemoyer and Collins, 2007) on the ATIS dataset

Figure 2: The learning curve shows the relation between number of learned rules and the F-measure for both TI and ATIS corpora.

5 Discussion

The number of learned rules is very small. As is shown in the figure 2, learning curves for both training data and development data are very steep. Although our current strategy for choosing the final number of rules for decoding is to keep only the rules for which we obtain highest F-measure on the development data, we could use much less rules without scarifying accuracy. For example, we accepted 0.1% lower F-measure on the development data than we would need only YYY rules in comparison with XXX rules if select the number of rules based in the highest F-measure. In contrast, the initial lexicon the CCG parser (Zettlemoyer and Collins, 2007) contains about 180 complex entries for general English words or phrases and yet additional lexical entries must be learned. **explain better**

Also, the number of rules per semantic concept (dialogue act or slot name) is very low. In TI data, we have XXX different dialogue acts and XXX slot and the average number of rules per semantic concept is XXX. In case of ATIS data, we have XXX dialogue acts and XXX slots and the average number of rules per semantic concept is XXX.

Lexical realizations of a slot can overlap with lexical realization of neighbouring slots. It is shows to be important pattern, for example in the trigram (city-0,and,city-1) is very common for sentence including "between city-0 and city-1". The lexical realizations city-0, city-1 respectively would be classified as from.city, and city-1 just because we know the

We found that the dialogue act type recognition accuracy of the STEP parser is lower than STC's; as a result, we tried to use SVM as STC does to classify dialogue act types. We believe that STC is better in dialogue act type recognition better because SVN classifier use all features at one time. Makes decision in one step using all the features rather than making several decisions by several rules as STEP does it.

We hoped for an increase of F-measure as result of increased dialogue act type accuracy. However, we did not get any increase in F-measure.

[4]We used POS tags provided by the RASP parser; however, any POS tagger can be used instead.

Acknowledgments

Do not number the acknowledgment section.

References

J. Williams and S. Young, 2007. *Partially observable markov decision processes for spoken dialog systems*. Computer Speech and Language, vol. 21, no. 2, pp. 231-422.

B. Thomson, M. Gasic, S. Keizer, F. Mairesse, J. Schatzmann, K. Yu, and S. Young. 2008. *User study of the Bayesian update of dialogue state approach to dialogue management*. Proceedings of Interspeech.

W. H. Ward. 1991. *The Phoenix system: Understanding spontaneous speech*. Proceedings of ICASSP.

L.S. Zettlemoyer and M. Collins. 2005. *Online learning of relaxed CCG grammars for parsing to logical form*. Proceedings of EMNLP-CoNLL.

D.A. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, A. Rudnicky, and E. Shriberg. 1994. *Expanding the scope of the ATIS task: The ATIS-3 corpus*. Proceedings of the ARPA HLT Workshop.

Y. He and S. Young. 2005. *Semantic processing using the hidden vector state model*. Computer Speech & Language,vol. 19, no. 1, pp. 85-106.

Y.He and S. Young. 2006. *Spoken language understanding using the hidden vector state model*. Computer Speech & Language, vol. 19, no. 1, pp. 85-106.

Y.W. Wong and R.J. Mooney. 2006. *Learning for Semantic Parsing with Statistical Machine Translation*. Proceedings of HLT/NAACL.

R.J. Kate, Y.W. Wong and R.J. Mooney. 2008. *Learning to Transform Natural to Formal Languages*. Proceedings of AAAI.

R.J. Kate. 2008. *A Dependency-based Word Subsequence Kernel*. Proceedings of EMNLP.

E. Briscoe, J. Carroll and R. Watson. 2006. *The Second Release of the RASP System*. Proceedings of COLING/ACL.

I. V. Meza-Ruiz, S. Riedel and O. Lemon. 2008. *Accurate statistical spoken language understanding from limited development resources*. Proceedings of ICASSP.

I. V. Meza-Ruiz, S. Riedel and O. Lemon. 2008. *Spoken Language Understanding in dialogue systems, using a 2-layer Markov Logic Network: improving semantic accuracy*. Proceedings of Londial.

F. Mairesse, M. Gasic, F. Jurcicek, S. Keizer, B. Thomson, K. Yu, and S. Young. 2009. *Spoken Language Understanding from Unaligned Data using Discriminative Classification Models*. Proceedings of ICASSP.

L. R. Tang and R. J. Mooney 2001. *Using Multiple Clause Constructors in Inductive Logic Programming for Semantic Parsing*. Proceedings of ECML.