SPOKEN LANGUAGE UNDERSTANDING FROM UNALIGNED DATA USING DISCRIMINATIVE CLASSIFICATION MODELS

F. Mairesse, M. Gašić, F. Jurčíček, S. Keizer, B. Thomson, K. Yu, and S. Young

Cambridge University Engineering Department, Trumpington Street, Cambridge, CB2 1PZ, UK
{f.mairesse, mg436, fj228, sk561, brmt2, ky219, sjy}@eng.cam.ac.uk

ABSTRACT

While data-driven methods for spoken language understanding reduce maintenance and portability costs compared with handcrafted parsers, the collection of word-level semantic annotations for training remains a time-consuming task. A recent line of research has focused on building generative models from unaligned semantic representations, using expectation-maximisation techniques to align semantic concepts. This paper presents an efficient, simple technique that parses a semantic tree by recursively calling *discriminative* semantic classification models. Results show that it outperforms existing generative models, while performance is close to more complex grammar induction techniques. We also show that our method is robust to speech recognition errors, by improving over a handcrafted parser previously used for dialogue data collection.

Index Terms— semantic analysis, spoken dialogue systems, spoken language understanding

1. INTRODUCTION

Most commercial dialogue systems employ keyword spotting techniques for understanding the user input. These methods are robust to noise, but their simplicity prevents them from modelling long-range dependencies within an utterance, as well as scaling to complex semantic representations (e.g., semantic trees). Such keywords—and the way they are mapped to a specific semantic representation—are typically defined by hand, e.g. by specifying a VoiceXML grammar that instantiates slot values.

Within the past 20 years, research on spoken language understanding (SLU) has produced models that alleviate these issues by learning to derive a semantic representation from data. Most work has focused on generative dynamic Bayesian networks that model the semantics of the utterance as a hidden structure on which observed words are conditioned [1, 2, 3, 4, 5]. An advantage of such models is that they can be trained on unaligned data, using expectation-maximisation techniques. However, the Markovian assumption prevents such techniques from explicitly modelling long-range time dependencies. Although a hierarchical hidden state structure can successfully model non-local dependencies in the utterance—such as the Hidden Vector State (HVS) model, which learns a probabilistic push-down automaton [3, 4, 5]—non-trivial structures require computationally expensive inference for training and decoding, and they cannot include a large number of correlated utterance features.

Discriminative models do not make independence assumptions over the feature set, which can lead to improved performance [6]. Support vector machines (SVM) have been used for classifying semantic arguments using syntactic tree features [7], as well as for classifying semantic production rules whose probability is estimated from the classifier's confidence score [8]. Wang & Acero show that linear-chain conditional random fields (CRF) produce the best results when converting the SLU problem into a flat sequential labelling task [9]. A disadvantage of these discriminative methods is that they require training utterances to be semantically annotated at the word-level. Aligning the semantic representation with individual words is a time-consuming task, which results in large maintenance and portability costs during dialogue system development, thereby mitigating the attractiveness of data-driven methods over handcrafted rules. Zettlemoyer & Collins present a grammar induction method that can learn a probabilistic combinatory categorial grammar (PCCG) from utterance-level annotations, by alternatively (a) optimising the grammar's lexicon using the current PCCG and (b) refining the PCCG's weights based on the updated lexicon [10]. They show that their technique produces state-of-the-art performance on the Air Travel Information System (ATIS) dataset [11].

While keeping the same constraints on the level of semantic annotation, this paper presents an efficient yet simple technique that learns discriminative semantic concept classifiers whose output is used to recursively construct a semantic tree, without requiring any alignment information. The next section presents our method in more detail, while Section 3 evaluates its robustness to speech recognition errors and compares it against results reported by He & Young [4] and Zettlemoyer & Collins [10] on the ATIS dataset. Finally, Section 4 concludes with a discussion of future work.

2. METHODOLOGY

This section explains how semantic classifiers can be trained using unaligned data, and how they are used for parsing unseen utterances.

2.1. Learning semantic tuple classifiers

The input of the learning algorithm presented in this section is a set of utterances and their corresponding semantic trees. Some limitations on the structure of these trees are discussed in Section 2.4. As in previous work [3, 10], we rely on a database characterising entities of interest in the dialogue domain as category/value pairs (e.g., `CITY = New York`; see Section 2.3). An important aspect of our algorithm is the division of each semantic tree into *concept tuples*, which consist of a sequence of l semantic concept nodes linked together within a single branch. For example, the tree `INFORM(FOOD(CHINESE))` contains two tuples of length 2 (i.e., `INFORM`$\rightarrow$`FOOD` and `FOOD`$\rightarrow$`CHINESE`) and one tuple of length 3 (i.e., `INFORM`$\rightarrow$`FOOD`$\rightarrow$`CHINESE`). The learning algorithm consists of the following steps:

This research was partly funded by the UK EPSRC under grant agreement EP/F013930/1 and funded by the EU FP7 Programme under grant agreement 216594 (CLASSiC project: `www.classic-project.org`).

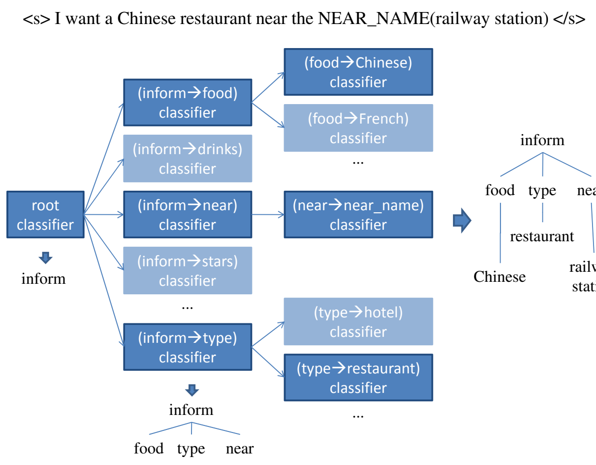

Fig. 1. Semantic tree derivation for an utterance in the TownInfo dataset (see Section 3.1), with positive concept tuple classifications in darker boxes.

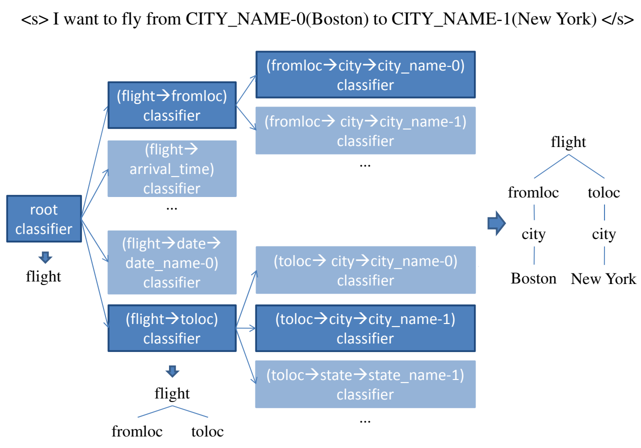
<s> I want to fly from CITY_NAME-0(Boston) to CITY_NAME-1(New York) </s>
root classifier
flight
(flight→fromloc) classifier
(flight→ arrival_time) classifier
...
(flight→date→ date_name-0) classifier
(flight→toloc) classifier
flight
fromloc toloc
(fromloc→city→city_name-0) classifier
(fromloc→ city→city_name-1) classifier
...
(toloc→ city→city_name-0) classifier
(toloc→city→city_name-1) classifier
(toloc→state→state_name-1) classifier
...
flight
fromloc toloc
city city
Boston New York

Fig. 2. Semantic tree derivation for an utterance in the ATIS dataset (see Section 3.1), with a maximum tuple length of 3. Indexed classifiers are used to retrieve non-enumerable database category labels from the utterance, such as `CITY_NAME`.

- **Input:** a set of (utterance, semantic tree) pairs, a maximum tuple length l and a domain database.
- **Output:** a set of semantic tuple classifiers and a domain grammar.

1. Replace database values in the training utterances with category labels (e.g., '*I want to fly from* `CITY_NAME`'; see Section 2.3).
2. Compute relevant lexico-syntactic features for each utterance (e.g., n-gram frequency counts with n from 1 to 3).
3. For each distinct tuple of maximum l concepts in the semantic trees:
   (a) Create a dataset associating each training utterance with a binary class representing whether the tuple occurs in the utterance's semantic tree.
   (b) Train a binary semantic tuple classifier (STC) that predicts the class—i.e., the tuple—from the feature values. The root concept (e.g., dialogue act type) is predicted using a single multi-class classifier.
4. If not provided, construct a domain grammar that matches all trees in the training set.

2.2. Parsing algorithm

Figures 1 and 2 illustrate the parsing process for the two evaluation domains presented in Section 3.1. The algorithm consists of the following steps:

- **Input:** an utterance, the semantic concept classifiers, the domain grammar and database.
- **Output:** the semantic tree of the input utterance.

1. Replace database values in the utterance with category labels.
2. Compute the utterance's features and filter out those not seen during training (e.g., unseen n-grams).
3. Run all semantic tuple classifiers (STC) on the utterance's features, set T equal to the set of positively classified tuples.
4. Initialise the output semantic tree as the predicted root, and a variable r pointing to the root concept. Then recursively do either:
   (a) *High precision mode*: For each tuple t of T whose root has the same concept as r, append t's child to r in the semantic tree and remove t from T. Start over recursively by setting r equal to t's terminal node.
   (b) *High recall mode*: Start with (a). Then, for each remaining tuple t of T that is dominated by the concept r in the domain grammar, append t to r in the semantic tree. Start over recursively by setting r equal to t's terminal node.
5. Associate each of the tree's terminal node corresponding to a database category label with the corresponding value in the utterance (e.g., `CITY_NAME` becomes `New York`).

The output of this algorithm is a semantic tree of the utterance, with terminal concepts associated with database values. In *high precision mode*, the algorithm relies only on the outputs of the classifiers to expand the tree (e.g., `FOOD`→`CHINESE` can only be appended if the `FOOD` concept is already part of the tree), whereas in *high recall mode* the tree can be expanded as long as the result matches the domain grammar (e.g., `FOOD`→`CHINESE` can be appended to `INFORM` if this combination has been seen during training). We use the high recall mode throughout this paper, since it performs better on our datasets.

The maximum tuple length l parameter effectively controls the trade-off between (a) the accuracy of individual classifiers and (b) the non-ambiguity of the tree reconstruction process. For example, classifiers returning a full branch ($l = \infty$) make the reconstruction process trivial at the expense of classification accuracy, whereas classifiers returning individual semantic concepts ($l = 1$) produce ambiguous parses. Additionally, smaller tuples generalise better to branches that are not seen during training.

2.3. Learning instances of a database category

While our method relies on a domain database to scale to a large set of possible concept values, it prevents the learning algorithm from correctly parsing synonyms of such values (e.g., 'city centre' or 'downtown' for the `AREA`→`CENTRAL` tuple). To address this issue, we divide the set of database attribute concepts into (a) enumerable and (b) non-enumerable attributes. The number of values of an enumerable attribute is bounded regardless of the number of entities in the database (e.g., the TYPE attribute in Figure 1 is only associated with RESTAURANT, HOTEL and BAR), whereas non-enumerable attributes can take any number of values (e.g., CITY in Figure 2). In our algorithm, only non-enumerable values are replaced by their database category, whereas enumerable values are included explicitly in the semantic representation. At decoding time, non-enumerable values are associated with the database value that has the largest word overlap with the utterance.

In some application domains, the user can refer to multiple values of the same database category within a single utterance. In this case, the value extraction process becomes ambiguous. A solution is to number identical categories sequentially in each utterance (e.g., '*from CITY_NAME-0 to CITY_NAME-1*') and learn a classifier for every indexed concept (e.g., TOLOC→CITY→CITY_NAME-1 in Figure 2). When parsing an utterance, each concept with index i is associated with the i^{th} category label in the utterance. We use indexed classifiers on the ATIS dataset (see Section 3.1).

2.4. Limitations

While our method is based on a relatively simple framework, it is important to note some of its limitations. First, our parsing algorithm does not align semantic constituents with individual words that are not in the database. For example, the semantic tree in Figure 2 does not indicate what words are governed by FROMLOC. However, we believe this is not an issue in practice since dialogue managers typically do not make use of alignment information for words that are not database values.

Our method can scale to large semantic representations while keeping the number of possible classifiers to a maximum of $|C|^l$, where C is the set of possible semantic concepts and l the tuple length. However, a second limitation is that none of the tuple classes should have identical child concepts (e.g., FROMLOC→CITY and TOLOC→CITY), since they can produce ambiguous anchor points in the tree. This issue can be overcome by increasing the value of the parameter l, as we do in our experiments on the ATIS dataset (see Section 3.1).

Since our method can require to run up to $|C|^l$ classifiers on the input utterance, the computational cost of our approach at run-time is limited by the number of distinct tuples in the training data. Using approximately 250 linear support vector machine classifiers implemented in Java on a Quadcore Pentium 2.4 GHz, each utterance was parsed in less than 200 ms on average, for both evaluation datasets. We thus believe our method is suitable for real-time dialogue.

3. EVALUATION

This section evaluates our method on two distinct domains, and compares our results with previous state-of-the-art techniques and a handcrafted rule-based parser.

3.1. 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 [12, 13]. 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 NAME(Char Sue)), and '*I would like a Chinese restaurant*' as INFORM(FOOD(CHINESE) TYPE(RESTAURANT)). Since the TownInfo database is relatively small, all categories are treated as enumerable. The TownInfo training and test sets respectively contain 8396 and 1023 transcribed utterances. The maximum tuple length l is set to 2 to optimise classification accuracy. 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%). We also compare our models with the handcrafted Phoenix grammar [14] used in the trials [12, 13]. The Phoenix parser implements a partial matching algorithm that was designed for robust spoken language understanding.

In order to compare our results with previous work [4, 10], we apply our method to the Air Travel Information System dataset (ATIS) [11]. This dataset consists of user requests for flight information (see example in Figure 2), and a database of concept examples. We use 5012 utterances for training and parameter tuning. 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 tree (e.g., FROMLOC.CITY = New York). He & Young detail the test data extraction process in [3]. Since the ATIS database is relatively large, all categories are treated as non-enumerable.[1] Since many semantic trees contain multiple identical concepts, the maximum tuple length parameter is set to 3 to avoid ambiguous anchor points (see Section 2.4).

3.2. Feature set and learning algorithm

Although any set of lexical, syntactic and semantic features can be used to train the classifiers, we report our first results using n-gram frequency counts computed over the training utterances. Before computing the features, occurrences of non-enumerable database values and numbers are replaced with a generic label. With a maximum n-gram size of 3, we compute a maximum of 16,160 features for the TownInfo dataset and 13,185 features for the ATIS dataset. We do not report results using word stemming as they did not show any improvement, and we leave supervised feature selection as future work.

These features are used to train support vector machine classifiers (SVM), as they were shown to generalise well on various natural language processing tasks requiring a large, sparse set of correlated features [15, 7, 8]. We only report results with *linear* kernel SVMs trained using the LibSVM package [16], as early experiments showed that using other kernels did not improve performance. The maximum n-gram size ($1 \leq n \leq 3$) as well as the SVM misclassification cost parameters are optimised individually for every tuple classifier, by performing multiple cross-validations on the training data.[2]

3.3. Results

Results for both datasets are shown in Table 1. The model accuracy is measured in terms of the percentage of correctly classified dialogue act types, as well as the F-measure of the slot/value pairs. Both the slot and the value must be correct to count as a correct classification. The dialogue act type is the root of the output tree, whereas slot/value pairs are trivially extracted from the branches

[1]We also add all possible time values to the database so that our model can generalise to arbitrary times.

[2]All other parameters are set to their default values.

Parser	DA	Prec	Rec	F
TownInfo dataset with transcribed utterances:				
STC	**94.92**	97.39	94.05	**95.69**
Phoenix	94.82	96.33	94.20	95.26
TownInfo dataset with ASR output:				
STC	**85.15**	94.03	83.73	**88.58**
Phoenix	74.73	90.28	79.49	84.54
ATIS dataset with transcribed utterances:				
STC	**92.63**	96.73	92.37	94.50
He & Young (2006)	–	–	–	90.3
Z & C (2007)	–	95.11	96.71	**95.9**

Table 1. Dialogue act classification accuracy (*DA*), slot/value precision (*Prec*), recall (*Rec*) and F-measure for the ATIS and TownInfo test datasets. Semantic tuple classifiers (STC) are compared with a handcrafted Phoenix parser, as well as results reported by He & Young and Zettlemoyer & Collins (*Z & C*) on the same test set.

(e.g. `FROMLOC→CITY→New York` becomes `FROMLOC.CITY = New York`). Results on the TownInfo domain show that the semantic tuple classifiers (STC) trained and tested on transcribed utterances perform slightly better than the handcrafted Phoenix grammar. Classifiers trained and tested on the automatic speech recognition (ASR) output show a large improvement over the handcrafted grammar, for both the dialogue act accuracy (10.42% improvement with 85.15% accuracy) and the F-measure (4.04% improvement with F=88.58%).

Concerning the ATIS dataset, Table 1 shows that the STC algorithm produces a 92.63% dialogue act accuracy and a 94.50% F-measure, which represents a 4.2% improvement over results reported by He & Young using the HVS model on the same test set [4], but 1.4% lower compared with Zettlemoyer & Collins' PCCG model [10].

4. DISCUSSION AND FUTURE WORK

This paper presents a new data-driven method for semantic parsing that can learn from *unaligned* semantic trees, thus allowing for faster data collection and dialogue system deployment. Results show that STCs outperform a handcrafted rule-based parser previously used for collecting dialogue data [12, 13], for both clean utterances and noisy speech recognition outputs. Our method also also improves on the HVS model [4] on the ATIS dataset, and it performs only 1.4% worse than more complex iterative grammar induction techniques [10]. It is important to note that the latter requires handcrafting domain-independent parsing rules in the initial grammar lexicon (e.g., PCCG lexicon entries for wh-words).

While we test our parsing technique on two non-trivial datasets, future research should study whether the STC algorithm can scale to arbitrarily large semantic trees, and address limitations on the tree structure (see Section 2.4). Additionally, commercial dialogue systems typically need to detect thousands of database concepts, each of which can be expressed in different ways. While our method currently relies on a database to extract non-enumerable values, we are planning to investigate named entity tagging techniques that do not require word-aligned data. As recent research on data-driven dialogue systems has focused on modelling uncertainty to improve decision making [12, 13], an important future work is to extend our technique to return a distribution over semantic trees based on classification confidence scores, e.g. by mapping the SVM classification margin to a probabilistic value. Finally, while we show that simple n-gram features are robust to noise, we are currently investigating whether higher level syntactic and semantic features can improve performance despite recognition errors.

5. REFERENCES

[1] Esther Levin and Roberto Pieraccini, "Chronus, the next generation," in *Proceedings of the ARPA Workshop on Spoken Language Technology*, 1995.

[2] Richard Schwartz, Scott Miller, David Stallard, and John Makhoul, "Language understanding using hidden understanding models," in *Proceedings of ICSLP*, 1996.

[3] Yulan He and Steve Young, "Semantic processing using the hidden vector state model," *Computer Speech & Language*, vol. 19, no. 1, pp. 85–106, 2005.

[4] Yulan He and Steve Young, "Spoken language understanding using the hidden vector state model," *Speech Communication*, vol. 48, no. 3-4, pp. 262–275, 2006.

[5] Filip Jurcicek, Jan Svec, and Ludek Muller, "Extension of HVS semantic parser by allowing left-right branching," in *Proceedings of ICASSP*, 2008.

[6] Christian Raymond and Giuseppe Riccardi, "Generative and discriminative algorithms for spoken language understanding," in *Proceedings of Interspeech*, 2007.

[7] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, and Daniel Jurafsky, "Shallow semantic parsing using support vector machines," in *Proceedings of HLT-NAACL*, 2004.

[8] Rohit J. Kate and Raymond J. Mooney, "Using string-kernels for learning semantic parsers," in *Proceedings of the Annual Meeting of the ACL*, 2006.

[9] Ye-Yi Wang and Alex Acero, "Discriminative models for spoken language understanding," in *Proceedings of ICSLP*, 2006.

[10] Luke S. Zettlemoyer and Michael Collins, "Online learning of relaxed CCG grammars for parsing to logical form," in *Proceedings of EMNLP-CoNLL*, 2007.

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

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

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

[14] W. H. Ward, "The Phoenix system: Understanding spontaneous speech," in *Proceedings of ICASSP*, 1991.

[15] T. Kudo and Y. Matsumoto, "Chunking with support vector machines," in *Proceedings of the Annual Meeting of the NAACL*, 2001.

[16] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM: a library for support vector machines*, 2001.