CLASSiC

D2.2: Semantic Decoder which Exploits Syntactic-Semantic Parsing, for the TownInfo Task

James Henderson

Distribution: Public

CLASSiC
Computational Learning in Adaptive Systems for Spoken Conversation
216594 Deliverable 2.2

Feb 2009

Project funded by the European Community
under the Seventh Framework Programme for
Research and Technological Development

The deliverable identification sheet is to be found on the reverse of this page.

Project ref. no.	216594
Project acronym	CLASSiC
Project full title	Computational Learning in Adaptive Systems for Spoken Conversation
Instrument	STREP
Thematic Priority	Cognitive Systems, Interaction, and Robotics
Start date / duration	01 March 2008 / 36 Months

Security	Public
Contractual date of delivery	M12 = Feb 2009
Actual date of delivery	Feb 2009
Deliverable number	2.2
Deliverable title	D2.2: Semantic Decoder which Exploits Syntactic-Semantic Parsing, for the TownInfo Task
Type	Prototype
Status & version	final 1.0
Number of pages	8 (excluding front matter)
Contributing WP	2
WP/Task responsible	UNIGE
Other contributors	Lonneke van der Plas and Paola Merlo at UNIGE and François Mairesse at UCAM
Author(s)	James Henderson
EC Project Officer	Xavier Gros
Keywords	Semantic Decoder, Spoken Language Understanding, Semantic Parsing

The partners in CLASSiC are:

University of Edinburgh HCRC	EDIN
University of Cambridge	UCAM
University of Geneva	UNIGE
Ecole Superieure d'Electricite	SUPELEC
France Telecom/ Orange Labs	FT

For copies of reports, updates on project activities and other CLASSiC-related information, contact:

The CLASSiC Project Co-ordinator
Dr. Oliver Lemon
School of Informatics
Edinburgh University
EH8 9LW
United Kingdom
olemon@inf.ed.ac.uk
Phone +44 (131) 650 4443 - Fax +44 (131) 650 4587

Copies of reports and other material can also be accessed via the project's administration homepage, `http://www.classic-project.org`

Contents

Executive summary

This document describes the Prototype deliverable 2.2, due at month 12 of the CLASSiC project. The prototype is for a Spoken Language Understanding (SLU) module, which has been integrated into the System 1 architecture as part of Task 5.2. This document presents an overview of the SLU model, results of evaluation in the TownInfo domain, and information on how to run the prototype. The SLU model learns to map user utterances (as output by the Automatic Speech Recognition module) to their meaning in terms of the CLASSiC project dialogue act scheme (as required by the Dialogue Manager module). It first parses the utterance with a domain-general syntactic-semantic parser, then extracts informative features from the resulting parse structure(s), and finally applies a set of statistical classifiers to construct the output dialogue act(s). In case of ambiguity, the generative statistical nature of the syntactic-semantic parser allows the SLU module to output a probability distribution over possible dialogue acts, as required by the CLASSiC architecture's treatment of uncertainty. This approach to SLU is largely complementary to the approach developed in Task 2.1, and we report in this document the results of a collaboration which combines the two approaches.

1 Overview of the Spoken Language Understanding Module

Task 2.2 focuses on exploiting UNIGE's domain-general syntactic-semantic parsers within a module for Spoken Language Understanding (SLU). The SLU module takes the output of the Automatic Speech Recognition (ASR) module, and compute the input for the Dialogue Management (DM) module. Within the CLASSiC architecture, ASR computes a probability distribution over possible user utterances, and DM expects a probability distribution over possible dialogue acts. For clarity, we will begin with a presentation of how a single hypothesised utterance is mapped to a single dialogue act, and then we will introduce the mechanisms for producing probability distributions.

The SLU module which we have developed within Task 2.2 consists of three components, preprocessing of the user utterance, syntactic-semantic parsing, and mapping to the output dialogue act. The preprocessing stage proved largely unnecessary because the parser was surprisingly robust to the differences between spoken user utterances and written text, and we were able to improve that ability further through retraining the parser. So preprocessing consists only of removing filler words, such as "um", and adding a period to the end of the utterance. The remaining two stages of processing will be presented in more detail in the rest of this section.

1.1 Domain-General Syntactic-Semantic Parsing

Recently an increasing number of corpora have become available with shallow semantic annotations of written texts (for example, see the CoNLL 2008 and 2009 shared tasks). These semantic annotations are designed to be domain-general, so they should be a valuable resource for semantic interpretation tasks in a wide range of domains. We have investigated exploiting such resources by first training a domain-general semantic parser, and then exploiting the resulting parses within a system trained specifically on the target domain, in our case computing dialogue acts for TownInfo user utterances.

We used the syntactic-semantic parsing method proposed in [1], which produces a merged representation of syntactic constituency trees from the Penn Treebank [2] annotated with Semantic Role Labelling information from PropBank [3]. An example of this output is given in figure 1. This is a statistical parsing model, which uses the neural network parsing architecture of [4] to estimate its probabilities. The output of the parser is a list of the most probable parses, with their generative probabilities. A form of beam search is used to search for the most probable parses, which allows a tradeoff between speed and accuracy by varying the beam width.

The only corpora available with domain-general semantic annotations are written corpora of full sentences. We analysed the performance of the above parser on utterances from the TownInfo domain, and found that it handled many differences with written text robustly. However, it had trouble with sentence fragments (e.g. *"in the east"*), which are very rare in the Penn Treebank but very common in system-driven dialogues like those in TownInfo. We addressed this problem by retraining the parser on a new corpus where we added artificially-generated sentence fragments [5]. To ensure that the distribution in the generated corpus is as close as possible to that found in the TownInfo corpus, we used a three-component model to generate the artificial corpus. The first component is a theory of the types of fragments found in the domain, which we characterised primarily as a constituent category (e.g. NP, noun phrase) and all words within that constituent. The second component is the distributions of these types (e.g. NPs versus full sentences) that we found in the TownInfo corpus. This necessitated the annotation of a subset of TownInfo utterances with these type labels, but not with full parses. The third component is the distribution of parse structures found in the original semantically-annotated corpus. The rules are used to extract fragments from this

Utterance: *in the east*
Parse:

```
(TOP (FRAG
       (PP@AM_LOC
         (IN@AM_LOC in)
           (NP (DT the)
             (NNP east) ) ) (. .) ) )
```

Features: localslot=area localvalue=area,east word=in tagword=in_IN@AM_LOC bigram=^_in label=IN@AM_LOC in_IN@AM_LOC_parent=PP@AM_LOC in_IN@AM_LOC_rgtsib=NP in_IN@AM_LOC_maxp_parent=FRAG in_IN@AM_LOC_maxp_rgtsib=. in_IN@AM_LOC_maxp_rgtsibwrd=. word=east tagword=east_NNP bigram=the_east label=NNP east_NNP_parent=NP east_NNP_lftsib=DT east_NNP_lftsibwrd=the east_NNP_maxp_parent=PP@AM_LOC east_NNP_maxp_lftsib=IN@AM_LOC east_NNP_maxp_lftsibwrd=in
Output: [inform(area=east), 0.973584]

Figure 1: An example of a structure output by the syntactic-semantic parser, the features computed from it, and the dialogue act and probability that these features map to. Semantic role labels are those that follow "@".

corpus, keeping their internal parse structures. The TownInfo distribution was then used to sample from this collection of fragments (and full sentences) to produce our artificial corpus. A parser trained on this artificial corpus was significantly more accurate on gold parsed TownInfo data than the original parser [5].

1.2 Dialogue Act Classifier

The dialogue act classifier component of the SLU module computes the output dialogue act. Dialogue acts are defined by the CLASSiC project dialogue act scheme, which specifies the inputs expected by the Dialogue Manager module. Examples are `inform(food=English,type=restaurant,area=east)` for *is there an english restaurant in the east* and `request(phone)` for *could i have the number please*. These dialogue acts are computed from a parse structure output by the syntactic-semantic parser. UNIGE investigated two approaches to mapping from semantic parses to the required SLU dialogue act outputs. In both cases, a set of semantically-informative features are computed from the parses, which are used to choose a dialogue act.

We first investigated a hand-crafted mapping to dialogue acts. As anticipated, we found that a large amount of lexically-specific information was necessary to map from the domain-general shallow semantics of the parser to the specific information required in a dialogue act. This made hand-crafting infeasible.

We then used the data analysis done for the hand-crafted mapping to design a set of patterns for extracting semantically-informative features from a parse. These include lexical features, and some bi-lexical features. The current set of features is summarised in table 1. The resulting vector of features is then used in a statistical classifier to choose a dialogue act.

Inspired by the Semantic Tuple Classifier developed by UCAM in Task 2.1 [6], UNIGE decomposed the

Feature Name	= **Feature Value**
localvalue	string found which matches a database entry
localslot	slot for localvalue
word	word string
label	POS-tag
tagword	pairing of word with POS-tag
norm	normalised word, if different from word
bigram	pair of adjacent words
*w_t*_parent	label of parent of *w_t*, a word-tag pair
*w_t*_lftsib	POS-tag/label of left sibling of *w_t*
*w_t*_rgtsib	POS-tag/label of right sibling of *w_t*
*w_t*_lftsibwrd	word (if any) that is left sibling of *w_t*
*w_t*_rgtsibwrd	word (if any) that is right sibling of *w_t*
*w_t*_maxp_parent	label of parent of *w_t*_maxp, the maximal projection of parent of *w_t*
*w_t*_maxp_lftsib	POS-tag/label of left sibling of *w_t*_maxp
*w_t*_maxp_rgtsib	POS-tag/label of right sibling of *w_t*_maxp
*w_t*_maxp_lftsibwrd	word (if any) that is left sibling of *w_t*_maxp
*w_t*_maxp_rgtsibwrd	word (if any) that is right sibling of *w_t*_maxp
*w_t*_vgov	POS-tag of governing verb of *w_t*
*w_t*_vgovwrd	word of governing verb of *w_t*
*w_t*_vgovroot	label of maximal projection of governing verb of *w_t*

Table 1: The features computed from a parse.

dialogue act into one classifier to choose the dialogue act type (e.g. inform versus request), and separate classifiers for choosing each node in the semantic tree. Differently from UCAM, UNIGE computes the probability of each node conditioned on its parent node, so that the semantic tree is generated by a probabilistic context free grammar. This ensures that the score for each dialogue act is a valid probability.[1] The semantic tree is decomposed as follows. If the dialogue act type takes no arguments, then stop. If the dialogue act type takes a special first argument (e.g. request), then choose that argument. Then for each possible slot name, first choose whether it is absent, positive ("="), or negative ("!="). Then choose its value. There is the option of generating the value as a specific string (e.g. "restaurant"), or as a database type (e.g. "name") which is subsequently substituted with a string found in the utterance (e.g. "Alexander Hotel").

To estimate the probabilities for this model, UNIGE chose to use Maximum Entropy classifiers. These were chosen because they optimise probability estimates. However, they do not work as well with small datasets as the SVMs used by UCAM. Results for this model are shown below as "Parser MaxEnt" in Table 2.

The SLU module developed by UCAM in Task 2.1 and the SLU module developed here share a common level of intermediate representation, namely the vector of features which is input to the dialogue act

[1]Note that for the UCAM system, the "high precision" mode (as used below) also results in valid probabilities, but it makes more independence assumptions.

classifier. Thus we decided to exploit the more sophisticated dialogue act classifier developed at UCAM with the features extracted from the semantic parser. UCAM applied their SVM classifier from Task 2.1 [6] to the feature vectors computed from UNIGE's syntactic-semantic parser. Results for this model are shown below as "Parser SVM" in Table 2.

1.3 Calculating Probabilities

In general, the input to the SLU module is a distribution over ASR hypotheses, and the output is a distribution over dialogue acts. In both cases these distributions are represented as n-best lists where the elements are paired with probabilities. We run the SLU module on each ASR hypothesis independently, and then merge the resulting n-best lists by summing the probabilities of matching dialogue acts, weighted by the probabilities of their associated ASR hypothesis.

Within the SLU module, the parser outputs an n-best list of parses, and we keep all the parses within a threshold of the most probable parse. Usually this results in a single parse, but in some cases two or three parses are kept. The probabilities of these parses are then normalised to produce the appropriate conditional probabilities. Then each parse is processed by the dialogue act classifier. Currently the decoder for this classifier only produces the single best dialogue act with its probability, but in principle it could also produce an n-best list. These probabilities are not normalised because they are already of the right conditional form. The dialogue acts from the different parses are then merged by summing the probabilities of matching dialogue acts.

1.4 Evaluation

The accuracies of the models discussed above on the TownInfo test sets are shown in Table 2. The top half of the table shows accuracies for training and testing on transcribed utterances, and the bottom half shows accuracies for training and testing on the noisy output of an ASR module. For these tests we only use the single best hypothesis from the ASR module, not the n-best hypotheses.

For the models which exploit a syntactic-semantic parser, we used a parser trained with a small vocabulary (813 tag-word pairs) and did decoding with a small beam width (pruning to 5 analyses after each word). These choices made the parser relatively fast, resulting in the full SLU module processing 30 words per second. But it also resulted in a less accurate parser.

These results show that our hand-coded mapping from semantic parses to dialogue acts ("Parser hand-coded") is clearly worse than that of the learned mappings. Between the learned models which exploit a syntactic-semantic parser, there is a clear improvement in using UCAM's SVM classifier ("Parser SVM") over using UNIGE's MaxEnt classifier ("Parser MaxEnt"), in particular on identifying slot-value pairs. This is probably due to the small number of positive examples for many of the slot-value classifiers, as apposed to the relatively large number of datapoints for dialogue act types. SVMs perform particularly well with small datasets. It could also be due to the different ways individual slot/value classifiers are combined in the two systems, particularly because the high-recall version of the SVM classifier was used.

Comparing the results using UCAM's SVM classifier with UNIGE's parser-derived features ("Parser SVM") to the same classifier with UCAM's n-gram features [6] ("N-gram SVM"), very similar results are achieved on transcribed utterances, but the n-gram features perform better on noisy ASR output. This is not an entirely fair comparison, because more fine-tuning of meta-parameters was done for the n-gram features, but the difference on ASR output data probably cannot be accounted for in this way. This difference

suggests that the current parser does not perform as well on noisy input. This limitation might be somewhat mitigated if the evaluations were done using n-best ASR output, rather than only having access to the one-best ASR output as here. Nonetheless, in future work it may be useful to train a syntactic-semantic parser on artificial treebank data where words have been substituted using a confuser that simulates ASR errors.

	Act Type	**Slot-Value**		
	Acc	**Prec**	**Rec**	**F**
	Transcribed utterances			
Parser hand-coded	**67.9**	78.0	88.5	**82.9**
Parser MaxEnt	**93.55**	94.37	85.66	**89.80**
Parser SVM	**94.23**	97.20	92.75	**94.93**
N-gram SVM (T2.1)	**94.92**	97.39	94.05	**95.69**
	ASR output			
Parser MaxEnt	**82.69**	89.71	73.94	**81.07**
Parser SVM	**82.60**	90.97	79.18	**84.67**
N-gram SVM (T2.1)	**85.15**	94.03	83.73	**88.58**

Table 2: Results on the TownInfo test sets for both transcribed utterances and noisy ASR output.

We consider that the equivalent performance of parser-derived features and n-gram features on transcribed utterances is a promising initial result. The types of phenomena where we would expect semantic parsing to be of help, such as the scope of negation (e.g. “not French or Chinese” versus “not French, Chinese”), are very rare in the current TownInfo corpus, whereas we believe they are more common in human-human dialogue. So the fact that we are doing well even with utterances that are simple enough for n-grams to give state-of-the-art results suggests that semantic parsing should result in an improvement as we move to more sophisticated dialogue systems with more natural interactions and better quality speech recognisers. We will evaluate this potential more precisely in future experiments.

1.5 Conclusions

This task has achieved its main objectives. A prototype (D2.2) which exploits domain-general syntactic-semantic parsing was completed on schedule. Although this method did not outperform the state-of-the-art accuracies achieved in Task 2.1, we consider these to be promising initial results. Further work is anticipated on this topic, to incorporate the new parsing models being developed in Task 2.5, to improve the features computed from semantic parses, and to empirically test hypotheses about the usefulness of semantic parsing in SLU. We also intend to evaluate our models on datasets other than TownInfo, where more complex utterances might be better suited to the advantages of domain-general syntactic-semantic parsing. For example, we anticipate that the Self-Help domain may involve more complex utterances.

2 The SLU Module Prototype

The prototype consists of the Maximum Entropy dialogue act classifier applied to the features extracted from a syntactic-semantic parse, called "Parser MaxEnt" in the above discussion of results. It has been tested under Linux. After unpacking `D2.2_prototype.tar.gz` by running

```
tar -zxf D2.2_prototype.tar.gz
```

enter the directory `D2.2_prototype/src/` and run

```
make
```

Then move back up to the directory `D2.2_prototype/`, where there should now be an executable `srlparser`. From here you can run the prototype interactively by running the script

```
./interactive.sh
```

Simple type sentences, one per line, and the resulting dialogue acts and their probabilities will be displayed. Alternatively, you can run

```
./demo.sh
```

which will display the parses and features computed within the SLU module. The line "DiaAct template:" is the dialogue act before strings from the sentence have been substituted for type names from the database, if any. Both these scripts can also be run by piping sentences through them, as in

```
cat examples.snt | ./interactive.sh
```

The scripts `interactive_asr.sh` and `demo_asr.sh` are the same except they use the classifiers which have been trained on ASR output.
As illustrated inside the above scripts, the prototype can be invoked directly as

`./srlparser` *`paramfile weightname maxentmodelname`*

where the arguments are

paramfile – the parameter file name

weightname – the stem of the names of files which define the semantic parser

maxentmodelname – the stem of the names of files which define the dialogue act classifier

The beam width used by the syntactic-semantic parser for decoding can be adjusted by changing the number which follows "-predbeam" in the parameter file.
This SLU module has also been integrated into the initial System 1, as part of prototype deliverable D5.2.1.

Bibliography

[1] Paola Merlo and Gabriele Musillo. Semantic parsing for high-precision semantic role labelling. In *Proc. 20th Conf. on Computational Natural Language Learning (CoNLL 2008)*, Manchester, UK, 2008.

[2] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[3] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005.

[4] James Henderson. Inducing history representations for broad coverage statistical parsing. In *Proc. joint meeting of North American Chapter of the Association for Computational Linguistics and the Human Language Technology Conf.*, pages 103–110, Edmonton, Canada, 2003.

[5] L. van der Plas, J. Henderson, and P. Merlo. Domain adaptation with artificial data for semantic parsing of speech. Submitted to *NAACL short papers*, 2009.

[6] F. Mairesse, M. Gašić, F. Jurčíček, B. Thomson S. Keizer, K. Yu, and S. Young. Spoken language understanding from unaligned data using discriminative classification models. In *Proc. of IEEE ICASSP*, 2009.