SEMI-AUTOMATIC ACQUISITION OF DOMAIN-SPECIFIC SEMANTIC STRUCTURES

Kai-Chung Siu and Helen M. Meng
Human-Computer Communications Laboratory
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Shatin, N.T.,
Hong Kong, China
{kcsiu, hmmeng}@se.cuhk.edu.hk

ABSTRACT

This paper describes a methodology for semi-automatic grammar induction from unannotated corpora belonging to a restricted domain. The grammar contains both semantic and syntactic structures, which are conducive towards language understanding. Our work aims to ameliorate the reliance of grammar development on expert handcrafting or the availability of annotated corpora. To strive for a reasonable model for real data, as well as portability across domain and languages, we adopt a statistical approach. Our approach is also amenable to the optional injection of prior knowledge to aid grammar induction, and subsequent hand editing for grammar refinement. This constitutes the semi-automatic nature of the approach. Experiments with the ATIS corpus showed positive results in semantic parsing, when compared to an entirely handcrafted grammar.

Keywords: grammar induction, semantic processing.

I. INTRODUCTION

Research and development of spoken language systems for restricted domains has been gaining much momentum in recent years [1]. Many approaches to spoken language understanding require a grammar for parsing the input utterance to acquire its semantics. The grammar is often handcrafted by a grammarian and / or a knowledge domain expert. This is a daunting and expensive task, which forms a major bottleneck in the development of spoken language understanding systems. Furthermore, there is no direct control that the grammar so designed will model the target language well – under-generation or over-generation (or both) may be severe when the grammar and parser are applied to realistic spoken queries.

Another related approach is to automate the grammar writing process. Grammar induction is typically corpus-based [2],[3]. Corpus-based approaches may be desirable, as the grammar produced may model real data closely. The corpus may be annotated with some domain-dependent semantic tags, or domain-independent syntactic tags.[1] Various grammar induction algorithms can automatically capture patterns in which syntactic structures and semantic categories interleave into a multitude of surface forms. However, annotation of corpora may be costly.

We wish to devise a methodology to semi-automatically capture language structures from *unannotated* corpora. These structures need to be conducive towards language understanding. This is essentially a grammar induction process. The resultant grammar should contain language structures that may be semantic, syntactic, or a tight coupling of both. Our work is an attempt to expedite the process of grammar design for spoken language understanding in a prescribed domain. We conceive of several desired features for such a methodology:

(i) It may be corpus-based, but should ameliorate reliance on annotated corpora.

(ii) It should be easily portable across different restricted domains, as well as across languages.

(iii) The output grammar should provide reasonable coverage of within-domain data, and reject out-of-domain data.

(iv) The output grammar should be intuitive, and amenable to interactive refinement by a human.

(v) The process should accommodate the optional injection of prior knowledge to aid grammar induction.

We conceive of possible applications of this work in data mining, information extraction, and meta-data abstraction.

II. A STATISTICAL APPROACH

We adopt a statistical approach, which is inspired by previous work on language modeling for speech recognition b McCandless and Glass [6]. We wish to extend a similar framework in order to accomplish understanding of natural language.

An iterative procedure is used to cluster the words from a corpus of sentences in a restricted domain. Clustering is implemented both spatially and temporally. For spatial clustering, we considered the Kullback-Liebler distance -- an information-theoretic distance between two probability distributions p_1 and p_2:

$$D(p_1 || p_2) = \sum_{i=1}^{V} p_1(i) \log \frac{p_1(i)}{p_2(i)} \quad (1)$$

where V is the vocabulary size within the given context. It should be noted that $D(p_1 || p_2) = 0$ if p_1 and p_2 are equivalent. In order to acquire a symmetric distance measure, we used the *divergence* measure:

$$Div(p_1, p_2) = D(p_1 || p_2) + D(p_2 || p_1) \quad (2)$$

All probabilities are estimated by tallying counts from the training sentences, with appropriate smoothing. At the onset of spatial clustering, all the words in the training set (with at least the pre-set minimum occurrence) are considered pair-wise. We compute the distance between a pair of words (or word clusters for later iterations), which is the sum of the divergences of probability distributions to the left and right of the entities (e_1, e_2):

[1] E.g. part of speech tags, as in the Penn Treebank [4], and the Tagged Brown Corpus [5].

$$Dist(e_1,e_2) = Div(p_1^{left}, p_2^{left}) + Div(p_1^{right}, p_2^{right}) \quad (3)$$

The N most similar pairs are clustered, and assigned the spatial cluster label SC_i, where i is a counter which increments automatically as spatial clusters are formed. Subsequently, all the words in the training set are substituted with their corresponding spatial cluster label. Spatial clustering is expected to produce semantic categories. After spatial clustering, the process proceeds to temporal clustering.

For temporal clustering, we adopt Mutual Information (MI) as our distance measure, to indicate the degree of co-occurrence of two consecutive entities (words, or word sequences).

$$MI(e_1,e_2) = P(e_1,e_2)\log\frac{P(e_2|e_1)}{P(e_2)} \quad (4)$$

Again only words with at least the minimum occurrences are considered. The N pairs of entities with highest MI are selected to form temporal clusters, which are labeled TC_i, where i is a counter which increments automatically as the temporal clusters are formed. The training sentences then undergo a pass whereby appropriate entities are substituted with their TC labels. Temporal clustering is expected to produce phrasal structures. The process then alternates to the next iteration of spatial clustering.

Iterative clustering produces a context-free grammar, which is post-processed with hand editing. Hence, our approach is semi-automatic. The hand-revision process serves to organize grammar non-terminals (SC and TC), identify those that are contributive towards language understanding, and label them with semantically relevant tags. The resultant grammar should reflect the ontology of the domain. The grammar may be evaluated by: (i) comparing the semi-automatically derived language structures with those which are handcrafted, should the latter be available; or (ii) implementing a semantic parser which utilizes the grammar to parse data sets, and examine the language understanding performance.

III. EXPERIMENTAL CORPUS

Our experimental corpus is based on the training and test sets of the ATIS (Air Travel Information System) domain [7]. ATIS is a common task in the ARPA (Advanced Research Projects Agency) Speech and Language Program in the USA. We used the Class A sentences of the ATIS-3 corpus. The disjoint training and test sets consist of 1,564, 448 (1993 test) 444 (1994 test) transcribed utterances respectively. Each utterance is accompanied with its corresponding SQL quer for database retrieval.

IV. UNSUPERVISED ITERATIVE CLUSTERING

For iterative clustering, the minimum count M is set at 5. We experimented with the number of merges per iteration, N, for the values N=1 and N=5.

With N=1, each iteration will produce an SC and a TC, as it searches through the space of all entity-pairs, which is a ver computationally intensive process. Having proceeded through 17 iterations of clustering, our algorithm produced 34 spatial and temporal categories, 2 of which were deemed irrelevant. Examples include:

SC0	→	layover | stopover	
SC3	→	numbers | times	(irrelevant)
SC6	→	cheapest | last	(irrelevant)
SC10	→	could | can	
SC12	→	nashville | toronto	
TC0	→	flights from	
TC8	→	round trip	
TC15	→	los angeles	

Upon investigation, SC3 and SC6 may be pardonable offenses. The words *"numbers"* and *"times"* were merged, due to many instances *of "…flight numbers from…"* and *"…flight times from…"* The words "*cheapest*" and "*last*" were merged, due to many instances of *"…the cheapest flight…"* and *"…the last flight…"*

With N=5, our algorithm produced 46 spatial and temporal categories after only 5 iterations (and 0.2 of the previous processing time). One might expect to obtain more spatial and temporal categories, because 5 iterations each with 5 spatial merges and 5 temporal merges should produce 50 categories. However, there are cases when multiple merges from the same iteration were collapsed. For example, three of the five proposed merges from one iteration were (nashville, toronto), (nashville, tampa) and (detroit nashville). In this case, our algorithm produced a single spatial category, and therefore we are able to quickly generate nonterminals with a greater number of terminals. For example

SC*i* → nashville | toronto | detroit | tampa

Similar phenomena emerged for temporal categories. For example, in one iteration the proposed merges were (salt lake), (lake city), etc. Our algorithm considers across the proposed merges, and looks for cases where the second candidate of a pair coincides with the first candidate of another pair. For these cases, the algorithm exhaustively generates the possible combinations, e.g. "salt lake city". All combinations are considered in decreasing order of MI, while the algorithm also confirms the existence of the sequence in the training data. As a consequence, we generated TC17 → salt lake, performed the subsequent training data replacements, and then considered the proposed (lake city) merge. By now all the occurrences of (lake city) have been replaced, and non-existence renders the proposed merge to be discarded. Finall we proceeded to (salt lake city), given the knowledge of TC17. Since (TC17 city) still occurs in the training set at this point, we generated TC18 → TC17 city.

Our resultant grammar (46 categories, N=5) is a superset of the previous grammar (34 categories, N=1). The extra 12 categories are all relevant, e.g.

SC14	→	francisco | jose
TC15	→	los angeles
TC16	→	san SC14

We concluded that N=5 is a better parameter setting. The merging is more aggressive, and seems to produce an equall good grammar with fewer iterations. (In the future, we will be experimenting with even higher values of N.)

Clustering was allowed to proceed to 100 iterations. We monitored its progress by keeping track of the nonterminal (SC and TC) and terminal categories in the grammar (see Figure 1). We observe that as the grammar grows, the number of terminals saturated at around iteration 50, to a count of 276. This covers a fraction of the vocabulary (531 words in all) from the training set. The remaining words are those which did not meet our minimum count requirement. The number of SCs grew slowly to 129 at iteration 100, and they were mainly semantic categories and inflectional variations. The growth rate of TCs dominated the overall growth rate of the nonterminals, reaching 510 TCs at iteration 100.

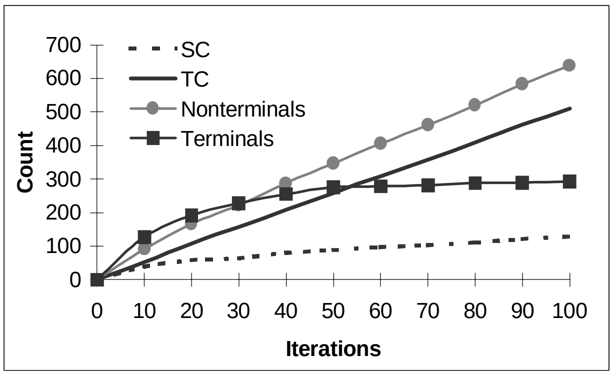

Figure 1. Growth of grammar units along increasing iterations in the grammar induction process.

The TCs were mainly phrasal structure as expected. Examples of SCs and TCs include:

SC4	→	december | february	
SC7	→	nashville | toronto | tampa | detroit ...	(i.e. city names)
SC17	→	june | march	
SC24	→	serve | serves	
SC28	→	monday | wednesday | thursday	
TC39	→	first class	
TC44	→	one way	
TC21	→	to SC7	(i.e. a destination)
TC25	→	SC7 to	(i.e. an origin)
TC26	→	to SC7 to	(i.e. a stopover)
TC145	→	flights from SC7 to SC7	(i.e.a phrase)

It is noticeable that some automatically discovered categories capture domain-specific knowledge that one could have easily given to the algorithm. We also observe rules that are close renditions of one another, e.g.

TC211	→	flights from SC7 to SC12
TC274	→	flights from SC7 to SC29
TC292	→	flights from SC12 to SC7

Since SC7, SC12 and SC29 are all city names, we should collapse TC211, TC274, TC292 into a single category.

Our observations prompted our idea of seeding the clustering algorithm with basic domain-specific knowledge. This should serve to jump-start our grammar induction process, and enable the algorithm to proceed further with even fewer iterations.

V. INJECTION OF PRIOR KNOWLEDGE

As we referenced the grammar nonterminals formed from unsupervised clustering, we selected 20 which we considered to be basic semantic classes for our domain. These include AIRLINE_NAME, AIRPORT_NAME, CITY_NAME, DIGIT, MEAL_DESCRIPTION, FARE_CLASS, etc., which were compiled to become seed categories for initializing our clustering algorithm. Compilation involves the consolidation of multiple SCs into a single semantic class, as well as completion of the set of terminals belonging to a given nonterminal. Seed categories were labeled SC1 to SC20. Clustering was thus initialized to run through 100 iterations with *N*=5. Again we monitored the grammar inference process, as shown in Figure 2.

Figure 2 shows that clustering was initialized with 397 terminals in the seeding categories. These include vocabular entries below the minimum count of 5, as well as inflectional forms of words that have not occurred in the training data. The growth of terminal categories began to saturate within 20 iterations to 485. We found iteration 40 to be a suitable termination point, beyond which over-clustering aggravated and produced many heterogeneous groupings. At this point, we recorded 76 SCs, 214 TCs and 491 terminals.

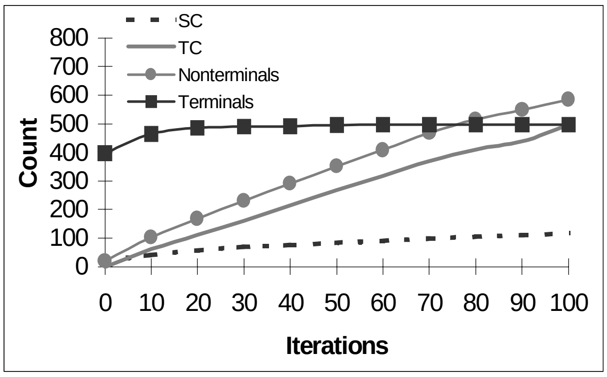

Figure 2. Growth of grammar units alongside increasing iterations, for grammar induction seeded with prior knowledge.

Inspection of the grammar output reveals that seed categories catalyzed the formation of longer phrasal structures with fewer iterations. We attempt to illustrate with the following example rules:

SC2	→	atlanta | baltimore | boston | ...	
SC3	→	monday | tuesday | wednesday | ...	
SC5	→	morning | afternoon | evening | ...	
SC15	→	from | departing from | leave from | ...	
SC16	→	to | arriving to | arrive to | ...	
SC24	→	airfare | fares | ticket	
TC38	→	flights SC15 SC2 SC16 SC2 on SC3	(a phrasal fragment)
TC109	→	round trip SC24 SC15 SC2 SC16 SC2	
TC125	→	flights SC15 SC2 SC16 SC2 on SC3 SC5	

A couple of artifacts were created by the seed categories

DIGIT	→	zero | oh | one | two | ...
DAY	→	first | second | third | ...

The substitution of seed category labels prior to clustering created many instances of "DIGIT dollar" from queries about pricing, as well as the frequent occurrence of "DIGIT way" due to "*one way*". Consequently, we saw the inappropriate merger SC*i* → dollar | way Another case is due to "*first class*", which was seeded to become "DAY class". Subsequent clustering produced TC35 → DAY class, which is also inappropriate. These were corrected during postprocessing.

VII. POSTPROCESSING

The resultant grammar from 40 iterations of clustering (with seeding) was post-processed by hand editing. The procedure involved: (i) replacing some of the T *i* or SC*i* tags with meaningful labels, e.g. city_name, month, etc., (ii) completing the set of terminals for some categories (e.g. days of the week), (iii) consolidating grammar categories which belong to the same semantic class, and (iv) pruning irrelevant nonterminals and terminals. Postprocessing took about an hour, to produce a grammar with 36 nonterminals and 369 terminals.

VIII. EVALUATION

The semi-automatically generated grammar (G_{SA}) was compared with a handcrafted grammar (G_H). G_H was manually designed to capture the key semantic categories from the training set [8]. G_{SA} has 36 nonterminals and 369 terminals. G_H has 6 nonterminals and 483 terminals. The grammars share 20 common nonterminals due to seeding. The remaining 16 nonterminals in G_{SA} were obtained from automatic clustering, and were selected during hand-revision as they have close counterparts in G_H. For example:

FLIGHT_NUMBER → flight DIGIT (e.g. "*flight four seventeen*")

TIME_VALUE → PRE_TIME DIGIT (e.g. *"after ten twenty six")*

The grammars were coupled with a parser, to operate on our data sets for retrieving key semantic concepts. The SCs in our grammars specify the key semantic categories to be extracted from the utterance and entered into a case frame. These are compared with the reference set of semantic categories from the SQL query.

Results on parse coverage are shown in Table 1. "Full Understanding" refers to utterances with exact matches between the semantic categories in the case frame and those in the SQL. "Partial Understanding" refers to partial matches. "No Parse" occurs when no semantic categories were extracted, due to out-of-domain words / word sequences. Our results show that G_H has extremely high coverage and accuracy in understanding. Coverage of G_{SA} is slightly lower, and generally has lower rate in full understanding, which is somewhat compensated by a higher rate in partial understanding.

Understand -ing	Training (G_{SA}, $\mathbf{G_H}$)	1993 Test (G_{SA}, $\mathbf{G_H}$)	1994 Test (G_{SA}, $\mathbf{G_H}$)
Full	78.9%, **84.8%**	67.4%, **82.8%**	62.6%, **73.9%**
Partial	21.0%, **15.2%**	29.3%, **17.2%**	36.0%, **25.5%**
None	0.1%, **0%**	3.3%, **0%**	1.4% **0.6%**

Table 1. Results of semantic parsing based on the semi-automatically generated grammar G_{SA} and the handcrafted grammar G_H. Numbers are percentages.

We also compared the grammars based on semantic sequence evaluation. Both insertions and deletions were considered. Table 2 shows that the two grammars have comparable training set performance; G_{SA} suffers from degradation in test set performance when compared to G_H.

	Error Rate (G_{SA})	Error Rate (G_H)
Training Set	6.9%	6.3%
1993 Test Set	16.1%	8.3%
1994 Test Set	17.1%	13.0%

Table 2. Semantic sequence evaluation based on parsing with the semi-automatically generated grammar G_{SA} and the handcrafted grammar G_H.

IX. PORTABILITY ACROSS LANGUAGES

We have begun to investigate the portability of our approach across languages. As a first step, we translated the ATIS sentences from English to Chinese. Chinese is a character-based language, and a word may range from one to multiple characters. Using a segmentation algorithm based on forward and backward maximum matching, together with a lexicon, we tokenized each Chinese sentence in to a sequence of Chinese words. We have a total of 370 sentences in all.

Clustering proceeds with a minimum count of 5, and at *N*=5 merges per iteration. No prior knowledge was incorporated. At iteration 12, the grammar inferred had 100 nonterminals and 110 terminals. We observe some SCs that are overl heterogeneous, and TCs that are too specific, mainly due to sparse training data. However, we also see the formation of reasonable semantic categories and phrase fragments, which is encouraging. Rule examples include

SC2 → 孟斐斯|紐約|西赤斯特城
SC6 → 三藩市|波士頓|蒙特利爾
SC10 → 列出|TP10, SC16 → 最早|最後
SC21 → 中午|TP8, SC28 → 經|飛往
SC29 → TP45|TP25|那些
TP0 → 既 航班, TP1 → 告訴 我
TP5 → 聯合 航空, TP8 → 中午 時
TP25 → 邊幾班 西北 航空 既 航機 會係 中午 時
TP45 → 我 想 乘搭

X. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented our initial work on semi-automatic grammar induction, for language understanding in restricted domains. Our iterative clustering approach strives to construct a grammar from unannotated corpora, since annotation is time-consuming and expensive. Our approach is semi-automatic, because the grammar inferred is intended to be hand-revised for quality improvement. The clustering algorithm is amenable to initialization with prior domain-specific knowledge to catalyze grammar induction. The algorithm also shows promise in portability across languages. Comparison between G_{SA} with a handcrafted grammar G_H shows that the semi-automatically acquired grammar has a decent coverage in parsing for semantics. Semantic sequence evaluation shows comparable training set performance. Test set performance of G_{SA} suffers some degradation but remained above 83%. Results thus far are encouraging. Future work will be devoted towards improving the grammar quality and examining portability issues.

REFERENCES:

[1] Zue, V., "Conversational Interfaces: Advances and Challenges," The 5th European Conference on Speech Communication and Technology, keynote speech, pp. KN9-18.

[2] Arai, K., J. Wright, G. Riccardi and A. Gorin, "Grammar Fragment Acquisition using Syntactic and Semantic Clustering," Proceedings of the ICSLP 1998.

[3] Chen, S., "Bayesian Grammar Induction for Language Modeling," Proceedings of the COLING/ACL, 1995.

[4] Marcus, M., B. Santorini and M. Marcinkiewicz, "Building a Large Annotated Corpus of English: The Penn Treebank," Computational Linguistics, 19(2), pp. 313-330, 1993.

[5] Francis, W. and H. Kucera, Frequency Analysis of English Usage: Lexicon and Grammar, Houghton Mifflin Company, 1982.

[6] McCandless, M. and J. Glass, "Empirical Acquisition of Word and Phrases Classes in the ATIS Domain," The 3 rd European Conference on Speech Communication and Technology, 1993.

[7] Price, P., "Evaluation of Spoken Language Systems: The ATIS Domain," Proceedings of the ARPA Human Language Technology Workshop, 1990, pp. 91-95.

[8] Meng H., W. Lam and C. Wai, "To Believe is To Understand," these proceedings.