Learning Strategies In A Grammar Induction Framework

Chin-Chung Wong, Helen Meng, Kai-Chung Siu

Human-Computer Communications Laboratory
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{wongcc, hmmeng, kcsiu}@se.cuhk.edu.hk

Abstract

This work extends a semi-automatic grammar induction approach previously proposed in [1]. We investigate the use of Information Gain (IG) in place of Mutual Information (MI) for grammar induction based on an unannotated training corpus. Experiments using the ATIS-3 training corpus indicate that the use of IG led to better precision and recall of desired semantic categories and at earlier stages in the grammar induction process when compared MI. We also investigate methods to automatically terminate the iterative grammar induction algorithm for grammar output. We define the stopping criterion to be where relative increment in grammar coverage scants 1%. Grammar coverage is measured in terms of coverage of the training corpus vocabulary. We obtain an output grammar based on this extended semi-automatic grammar induction algorithm with *automatic termination*. This grammar compares favorably with the handcrafted and semi-automatic grammars from [1] based on NLU performance using the ATIS-3 test sets.

1. Introduction

Grammar development is an indispensable phase in the implementation of spoken and natural language understanding systems. Conventional approaches involve a grammarian writing grammar rules to cover natural language queries falling within the scope of the application domain. This implies that the grammarian needs to be a domain expert as well. This task of handcrafting grammars is very expensive, and there is no direct control that the written grammar will model the target language well, especially for conversational spoken queries.

An alternative approach is to automatically capture semantic or phrasal structures by corpus-based techniques. Various data-driven approaches have previously been used to automatically capture word classes and multiword units. For instance, [2] performed automatic classification in word clustering by using Simulated Annealing (SA) and the algorithm can terminate automatically when the language model perplexity falls below a threshold. Probabilistic approaches for selecting multiword sequences based on mutual information (MI), perplexity and correlation were studied in [3] [4] and [5]. [6] presented the automatic acquisition framework of salient grammar fragments for the "How may I help you?" (HMIHY) task. Salient grammar fragments are automatically extracted from corpora based on the Kullback-Leibler distance and a significant test, and then used for call-type classification.

We attempt to develop a methodology for semi-automatic grammar induction from un-annotated corpora. Language structures can be learned automatically from domain-specific corpora. These language structures should be interpretable by a human and may be refined and edited by hand, but the demand for manual effort should be much reduced when compared to handwriting grammars. Hence the proposed approach is semi-automatic in nature. Furthermore, since human knowledge may be injected during the grammar induction process, there is lower demand on the amount of training data required. Our approach towards language structure acquisition is inspired by previous work on statistical language modeling [7], and the motivation is similar to efforts in semi-automatic language model acquisition for speech recognition [8].

2. Semi-Automatic Grammar Induction Approach

We reference the semi-automatic grammar induction approach in [1], but include some modifications. This section presents a brief overview of the approach.

The semi-automatic grammar induction approach begins with agglomerative clustering of words in a corpus of un-annotated sentences from a restricted domain, e.g. the Air Travel Information Service (ATIS) domain. Clustering is implemented both spatially and temporally. In spatial clustering, words or multi-word entities with similar left and right linguistic contexts are clustered together. Consider the clustering of entities e_1 and e_2. If p_1 denotes the unigram distribution of words occurring to the left of e_1, and p_2 denotes that for e_2, then we can measure the similarity of the two distributions by the divergence metric *Div* (or symmetrized Kullback-Leibler distance) as shown in Equations (1) and (2):

$$Div(p_1^{left}, p_2^{left}) = D(p_1^{left} \,||\, p_2^{left}) + D(p_2^{left} \,||\, p_1^{left}) \quad (1)$$

where

$$D(p_1 \,||\, p_2) = \sum_{i=1}^{V} p_1(i) \log \frac{p_1(i)}{p_2(i)} \quad (2)$$

V in Equation (2) denotes the corpus vocabulary.
Equations (3) shows the distance metric *Dist* when both the left and right contexts are considered.

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

The probabilities are obtained from the frequency counts in the training corpus. Only the words that have at least *M* occurrences are considered, in order to avoid sparse data problems. All word pairs are considered as described in Equation 3, and the algorithm selects the *N* most similar pairs (i.e. lowest values for *Dist*) to form spatial clusters that are labeled as SC_i, where *i* is a counter of the number of spatial clusters formed. Thereafter, the appropriate word pairs in the training corpus are substituted by their *SC* labels, and the algorithm proceeds to an iteration of temporal clustering.

In temporal clustering, words or multi-word entities that co-occur sequentially are clustered together. Mutual Information (MI) is used as the metric for clustering, as shown in Equation (4).

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

Again, only words that have at least *M* occurrences are considered, and the *N* pairs of entities with highest MI are selected to form temporal clusters labeled as TC_i, where *i* is a counter of the number of temporal clusters formed. Thereafter, the appropriate word pairs in the training corpus are substituted by their *TC* labels, and the algorithm proceeds to another iteration of spatial clustering.

As such, the agglomerative clustering approach produces a context-free grammar. The *SC* and *TC* are non-terminals in the grammar. *SC* clusters tend to be semantic structures, and *TC* clusters tend to be phrasal structures. The grammar is then post-processed by hand-editing, hence the grammar induction approach is deemed semi-automatic.

3. Experimental Corpus

We used the training and test sets of the ATIS (Air Travel Information System) domain as our experimental corpus. ATIS is a common task in the ARPA (Advanced Research Projects Agency) Speech and Language Program in USA. We used the Class A sentences of the ATIS-3 corpus. The disjoint training and test sets consist of 1564, 448 (1993 test) and 444 (1994 test) transcribed utterances respectively. Each utterance is accompanied by its corresponding SQL query for database retrieval. For example:

Utterance: *is there a flight around three p m from charlotte to minneapolis*

Simplified SQL
Select FLIGHT_ID from ORIGIN, DESTINATION
where ORIGIN.CITY_NAME = "charlotte"
and DESTINATION.CITY_NAME = "minneapolis"
and DEPARTURE_TIME = "around three pm"[1]

Table 1. An ATIS utterance with its SQL query.[1]

[1] "around three p m" is mapped to 14:30 – 15:30.

4. Evaluating Induced Grammars

Given the SQL query, we can evaluate the grammar produced at various stages in the induction process by means of precision and recall of semantic concepts, based on the PARSEVAL measures used in [9] and [10]. We transform the SQL query of an input utterance into a set of *reference* brackets. We can also parse the input utterance with the induced grammar *G* and generate a set of *hypothesized* brackets from the candidate parse. The reference and hypothesized brackets are compared to find the number of matching brackets. *Precision* is the proportion of matching brackets found in the set of hypothesized brackets. *Recall* is the proportion of matching brackets found in the set of reference brackets. Table 2 references the input utterance in Table 1 to illustrate the computation of precision and recall.

Bracket delimiters based on input utterance:
*is**0** there**1** a**2** flight**3** around**4** three**5** p**6** m**7** from**8** charlotte**9** to**10** minneapolis**11***

Reference brackets (based on SQL in Table 1):
around three p m (4, 7)*
charlotte (9, 9)*
minneapolis (11, 11)

Excerpt of grammar *G* from the induction process:
SC13 → one | two | three | …
SC24 → …charlotte | chicago | …
TC22 → SC24 to
TC23 → SC13 p m
SC25 → …may | june | …

Hypothesized brackets from candidate parse with *G*:
three (5, 5)
around three pm (4, 7)*
charlotte (9, 9)*
charlotte to (9, 10)

Table 2. Example illustrating the computation precision and recall values. Matching brackets are marked in '*'.

Referring to Table 2, the delimiters for the brackets follow the order of words in the input utterance. The reference brackets are derived from the SQL corresponding to the input utterance. The grammar *G* is the set of grammar rules obtained during the induction process. Notice that grammar induction has not yet acquired the city name *minneapolis*, so it is ignored in the parse. Parsing with grammar *G* generates the hypothesized brackets. Matches between the reference and hypothesized brackets are marked in *. Hence recall is 2/3 (two matching brackets out of three reference brackets) and precision is 1/2 (two matching brackets out of four hypothesized brackets).

Since a training corpus may not always include the SQL corresponding to the input utterances, we also use another metric to evaluate the grammar *G* produced by the induction process. The induced grammar rules should ideally provide maximum coverage of the words (potential grammar terminals) in the training corpus. Therefore, we also measure the percentage of words in the training corpus that are covered in the grammar. Since the grammar induction approach focuses only on words with at least *M* occurrences, we have incorporated the corresponding adjustments in computing the percentage of word terminals covered in *G*.

5. Information Gain vs. Mutual Information for Temporal Clustering

In this section, we report on our investigation in using Information Gain (IG) in place of Mutual Information (MI) for temporal clustering in grammar induction. It has been pointed out that the use of MI to find co-occurring entities is subjected to estimation errors especially when the occurrences of both entities are rare [11]. We investigate the use of IG, as defined in Equation (5), because it measures the number of bits of information obtained about one entity by knowing the presence or absence of the other entity.

$$IG(e_1,e_2) = P(e_1,e_2)\log\frac{P(e_1,e_2)}{P(e_1)P(e_1)} + P(e_1,\bar{e}_2)\log\frac{P(e_1,\bar{e}_2)}{P(e_1)P(\bar{e}_2)} + P(\bar{e}_1,e_2)\log\frac{P(\bar{e}_1,e_2)}{P(\bar{e}_1)P(e_2)} + P(\bar{e}_1,\bar{e}_2)\log\frac{P(\bar{e}_1,\bar{e}_2)}{P(\bar{e}_1)P(\bar{e}_2)} \quad (5)$$

IG is the sum of mutual information which consider all the combinations of presence or absence of entities e_1, e_2.

$P(e_1,e_2)$ is probability of (e_1,e_2): $\frac{Freq(e_1e_2)}{Freq(Bigrams)}$,

$P(e_1,\bar{e}_2)$ is probability of entity e_1 that is not followed by entity e_2: $\frac{Freq(e_1)-Freq(e_1e_2)}{Freq(Bigrams)}$,

and $P(\bar{e}_1,\bar{e}_2)$ is $1-P(e_1,e_2)-P(e_1,\bar{e}_2)-P(\bar{e}_1,e_2)$.

A coarse comparison between the first *N*(=30) temporal clusters formed using IG with those formed using MI suggests that IG enables us to capture meaningful phrases earlier in the agglomerative clustering process. More specifically, out of the first 30 temporal clusters formed using IG, 15 of them are phrases corresponding to attributes in the SQL, e.g. kansas city, round trip, san francisco, etc. The number of such phrases drops to 12 when MI is used.

For a more detailed comparison, we ran the grammar induction procedure first with both MI and IG for up to 90 iterations each. We compute the recall and precision values at every 10th iteration. Results are shown in Figures 1 and 2 respectively. As agglomerative clustering proceeds, recall values grow and converge while precision values decrease. Throughout the process, however, IG maintains a higher recall and higher precision than MI at the marked iterations. We conducted a paired *t*-test for comparing the use of IG with MI based on recall values at every tenth iteration. The increment in recall when IG is used instead of MI is statistically significant at α=0.01. A similar paired *t*-test applied to precision values also shows that the increment in precision due to IG is also statistically significant at α=0.01.

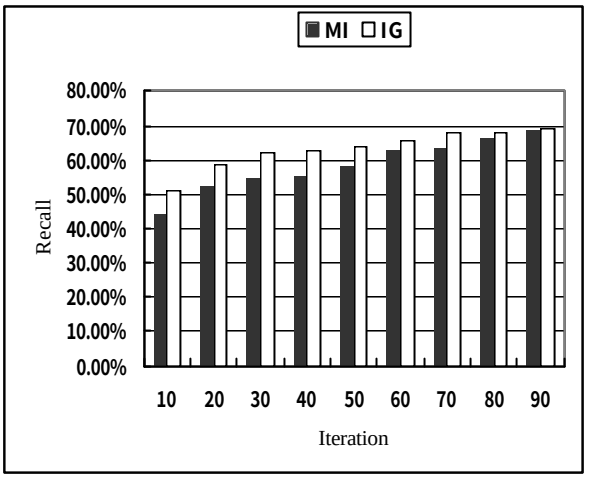

Figure 1. Comparison between Information Gain and Mutual Information in terms of grammar recall values at various stages during the grammar induction process.

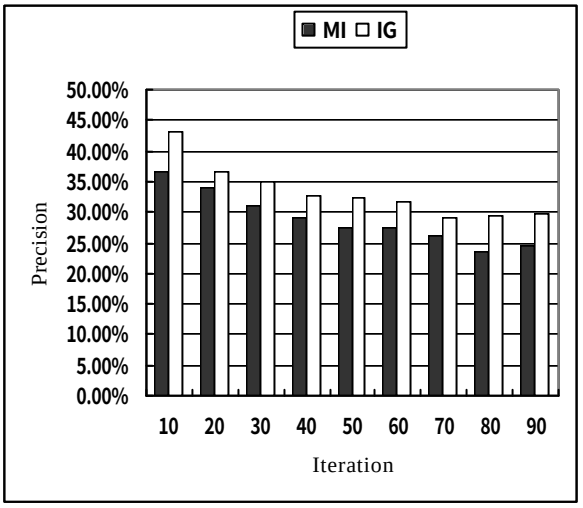

Figure 2. Comparison between Information Gain and Mutual Information in terms of grammar precision values at various stages during the grammar induction process.

6. Defining a Stopping Criterion

The automatic grammar induction is iterative and needs a stopping criterion. A stopping criterion should be defined such that the induced grammar has the highest possible recall or precision or coverage of the training corpus. The computation of recall and precision requires semantic annotations, e.g. those from the SQL, and such a resource tends to be expensive and may not always be available. Hence we attempt to define a stopping criterion based on the number of words (terminals) in the training corpus that

are captured in the induced grammar *G*. For example, the ATIS-3 Class A training set has 531 unique words. Of these, only 300 have occurrences above *M(=5)* counts, and this is the subset of the vocabulary that is processed by our grammar induction algorithm. Running grammar induction for 20 iterations produced a grammar that includes 167 words/terminals. Hence at this point the percentage of terminals covered is 167/300=55.7%. If we monitor the growth of this percentage with respect to the number of iterations, we obtain the graph depicted in Figure 3.

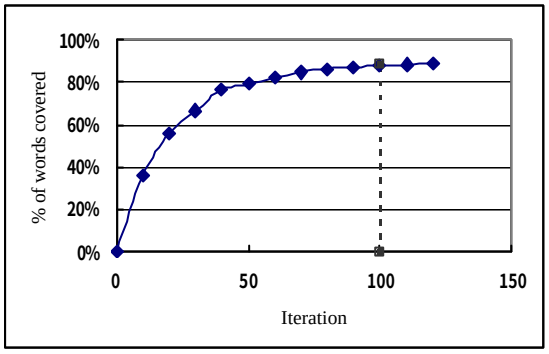

Figure 3. Graph plotting the growth of grammar coverage (measured in terms of the percentage of words/terminals in the training corpus that are captured in the grammar) with respect to the number of iterations. Measurements are taken at every tenth iteration.

We can measure grammar coverage in terms of the percentage of words/terminals in the training corpus that are captured in the grammar. If we define the stopping criterion to be the point where the relative growth in grammar coverage falls below 1%, then we should terminate grammar induction at iteration 100, based on the statistics of Figure 3. Grammar coverage at iteration 100 is 88.0% and grew to 88.3% at iteration 110, hence the relative increment is 0.38% (<1%).

Cross-checking with the recall values (see black curve in Figure 4) shows that at iteration 100, the induced grammar *G* attained a recall of 69.5%, which lies within the convergence region of Figure 4.

Recall that the semi-automatic grammar induction approach allows hand-editing of the grammar rules to improve the grammar. We have devoted a small manual effort for this purpose – we first allow the grammar induction algorithm to run for 20 iterations to obtain the grammar G_{20}, which contains 160 nonterminals (*SCs* or *TCs*) and 167 terminals. Among these we selected 17 nonterminals and 121 terminals to be preserved. The 17 nonterminals were labeled as SC0 to SC16, and contain categories such as AIRLINE_NAME, AIRPORT _NAME, DIGIT, ONE_WAY, etc. Of these we selected 7 nonterminals for each of which we inserted a complete set of terminals. The 7 nonterminals are CLASS_NAME, DAY_NAME, PERIOD, MEAL_DESCRIPTION, MONTH, DAY AND NUMBER. A total of 117 word terminals have been added by hand. Examples of the grammar rules include:

CLASS_NAME → business class | economy class| …
DAY_NAME → monday | tuesday | wednesday | …
PERIOD → afternoon | breakfast time | lunch time | …
MONTH → january | february | march | april | …
NUMBER → oh | zero | one | two | …

With this simple manual procedure, we were able to transform G_{20} into an enhanced seed grammar G_{H20} (where the subscript *H* denotes *hand-editing* of the grammar from iteration 20). We then seed the grammar induction algorithm with G_{H20} and continue to run until iteration 100 at which point the relative increment for recall is 0.27% (i.e. our automatic stopping criterion of <1% is met). At this point, the recall achieved is 86.5% (see gray curve in Figure 4).

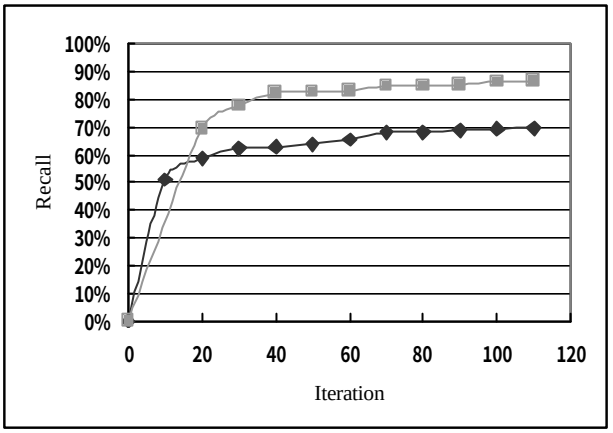

Figure 4. Graph plotting recall rates against the number of iterations. The proposed stopping criterion identifies iteration 100 as the termination point, giving 69.5% recall (see black curve which corresponds to the white bars in Figure 1), which increases to 86.5% if the grammar rules are hand-edited for improvement (see gray curve).

7. Evaluation based on Testing Corpora

The ultimate goal of semi-automatic grammar induction is to produce a grammar for natural language understanding (NLU). Hence this section reports on the NLU results based on the ATIS3 test sets 1993 and 1994.

The SQL corresponding to each utterance provides us with a reference semantic frame for evaluation. Referring to the example in Table 1, the attribute labels and values in the reference semantic frame will be:

ORIGIN: *charlotte*
DESTINATION: *minneapolis*
DEPARTURE_TIME: *14:30 – 15:30*

We parse each test utterance with an induced grammar *G* to produce a generated semantic frame. We follow the evaluation scheme described in [1] where "Full Understanding" refers to utterances with exact matches between the reference and generated frames. "Partial

Understanding" refers to partial matches. "No Understanding" occurs when no semantic categories were extracted, due to out-of-domain words or word sequences. As described previously, the grammar induction algorithm generates temporal clusters using Information Gain, and the iterative process is allowed to run for 20 iterations, at which point we hand-edited the grammar to produce G_{H20}. Seeding with G_{H20} and allowing the grammar to run until the stopping criterion is met produces out final output grammar G_F.

G_F is used to generate frames for the test set utterances. NLU results are shown in Table 3. This table also shows that G_F compares favorably with the semi-automatically induced grammar G_{SA} and handcrafted grammar G_H as reported in [1]. G_{SA} was obtained with seeding and used MI for temporal clustering.

1993 Test Set			
	G_F (%)	G_{SA}(%)	G_H(%)
Full Understanding	84.4	80.4	85.5
Partial Understanding	15.0	16.5	14.5
No Understanding	0.7	3.1	0
1994 Test Set			
Full Understanding	77.0	76.8	78.6
Partial Understanding	21.4	21.8	20.2
No Understanding	1.6	1.4	1.1

Table 3. Comparison of the final output grammar (G_F) with seeding from G_{H20}, semi-automatically induced grammar (G_{SA}) and handcrafted grammar (G_H), based on fraction of queries that achieved *Full Understanding*, *Partial Understanding* and *No Understanding*.

8. Summary and Conclusions

This work extends the semi-automatic grammar induction approach previously proposed [1] in two ways: (1) investigating the use of IG in place of MI for grammar induction based on an unannotated training corpus; and (2) defining a stopping criterion to automatically terminate iterative grammar induction for grammar output.

Experiments based on the ATIS-3 training corpus indicate that the use of IG led to better precision and recall of desired semantic categories and at earlier stages in the grammar induction process when compared MI. We measure improvements in precision / recall values at every tenth iteration, and investigation shows that improvements for IG over MI is statistically significant.

To find a termination point for the iterative grammar induction algorithm, we define the stopping criterion to be where relative increment in grammar coverage scants 1%. We measure grammar coverage in terms of the vocabulary coverage in the training corpus. We obtain an output grammar based on this extended semi-automatic grammar induction algorithm with automatic termination. This grammar compares favorably with the handcrafted and semi-automatic grammars from [1] based on NLU performance in the ATIS test sets. Future work will mainly focus on applying this methodology to various domains to achieve portability.

9. Acknowledgments

This research is supported by the Hong Kong SAR Government Research Grant Council Earmarked Grant No. CUHK/4177/00E.

References

1. Meng H. and K. C. Siu, "Semi-Automatic Acquisition of Domain-Specific Semantic Struectures", IEEE Transactions on Knowledge and Data Engineering, in press.
2. Smaili K., A. Brun, I. Zitouni and J.P. Haton. "Automatic and Manual Clustering for Large Vocabulary Speech Recognition: A Comparative Study", Proceedings of EUROSPEECH, 1999.
3. Giachin E., P. Baggia, G. Micca, "Language Models for Spontaneous Speech Recognition: A Bootstrap Method for Learning Phrase Bigrams", Proceedings of ICSLP, 1994.
4. Suhm B. and A. Waibel. "Towards Better Language Models for Spontaneous Speech", Proceedings of ICSLP, 1994.
5. Kuo J. and W. Reichl. "Phrase-Based Language Models for Speech Recognition", Proceedings of EUROSPEECH, 1999.
6. Wright J., A. Gorin and G. Riccardi. "Automatic Acquisition of Salient Grammar Fragments for Call-Type Classification", Proceedings of EUROSPEECH, 1997.
7. McCandless, M. and J. Glass, "Empirical Acquisition of Word and Phrases Classes in the ATIS Domain", The 3rd European Conference on Speech Communication and Technology, 1993.
8. Akiba T. and K. Itou, "Semi-Automatic Language Model Acquisition without Large Corpora", Proceedings of the ICSLP, 2000.
9. Manning, C. D. and H. Schutze, *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge, Massachusetts 1999.
10. Potamianos, A. and J. Kuo, "Statistical Recursive Finite State Machine Parsing for Speech Understanding", Proceedings of the ICSLP, 2000.
11. Wu, M. W. and K. Y. Su, "Corpus-based Automatic Compound Extraction with Mutual Information and Relative Frequency Count," Proceedings of ROCLING 1993.