Reusing a Statistical Language Model for Generation

Kevin Humphreys, Mike Calcagno, David Weise
Natural Language Group, Microsoft Corporation
One Microsoft Way
Redmond, WA 98052, USA
{kevinhum,mikecalc,davidw}@microsoft.com

Abstract

A relatively self-contained subtask of natural language generation is sentence realization: the process of generating a grammatically correct sentence from an abstract semantic / logical representation. We propose a method where sentence realization is carried out using a simplified (context free) version of a large analysis grammar, combined with a statistical language model from the full (context sensitive) version of the same grammar. The statistical model provides a measure of the probability of syntactic substructures, derived from the analysis of a corpus with the full grammar, and is used to guide both subsequent analysis and generation.

1 Introduction

To date, only limited use of statistically-derived resources has been made for realization in natural language generation, notably Knight & Hatzivassiloglou (1995), Langkilde & Knight (1998) and Bangalore & Rambow (2000). This paper reports on new work in that direction, but with an emphasis on reusing resources originally produced for analysis purposes. In particular, a generation grammar is derived from an extensive analysis grammar in such a way as to retain the statistical language model built using the analysis grammar.

2 Statistically-Driven Generation

Work to date on using statistical knowledge for generation has mainly focused on the sub-task of surface (in fact, sentence) realization: the production of a grammatically correct string from an abstract semantic/logical representation of linguistic content. This assumes the existence of a separate higher-level process to produce such a representation, following the canonical pipeline architecture of a full generation system (Reiter, 1994). The approach described here has the same focus, but attempts to more tightly integrate the statistical knowledge in the generation process, and also to avoid the need to create generation-specific resources.

2.1 Nitrogen

The Nitrogen system (Knight & Hatzivassiloglou, 1995; Langkilde & Knight, 1998) made the first significant attempt to integrate statistical knowledge for surface realization. It uses an extremely simple generation-specific grammar and generates a lattice representing all possible strings that the grammar allows for a particular semantic input. Then, in a separate stage, simple bigram statistics are used to rank the alternatives in terms of 'fluency', determined by similarity to word pairs in the training corpus. The language model represented by the bigrams is not used within the generation algorithm itself, rather it acts as a filter on the proposed output of an independent generation system. The simplified grammar is so unconstrained that typically hundreds of thousands of alternative strings are generated for a single input, including many ungrammatical forms. The bigram model then selects the most probable pairwise combination of words to select a sentence, considering non-adjacent words for a fixed set of syntactic relations, but not representing any context to allow for true long-distance dependencies or to avoid multiple expression of the same constituents.

2.2 Fergus

Bangalore & Rambow (2000) build on the approach of the Nitrogen system but use a language model which does encode some structural information. They use an XTAG grammar (XTAG-Group, 1999), which is not generation-specific, with statistically ranked subtree structures associated with lexical entries. An initial set of subtrees is chosen for a particular input, using the model, then a lattice of all possible combinations licensed by the grammar is constructed, where each combination represents an alternative output string. Then, in the same way as Nitrogen, a separate trigram model is used to rank the alternative strings. The subtree combination phase allows the handling of long-distance dependencies, and can more accurately control constraints such as agreement, which Nitrogen must leave entirely to its bigram model.

3 A Reusable Language Model

The approach presented in this paper places a strong emphasis on reusing resources originally developed for analysis applications. Without requiring a fully reversible analysis system (e.g. Neumann & van Noord, 1994), it has proved possible to successfully reuse a language model developed for analysis in a related generation system.

3.1 Analysis

The analysis system is built around a broad-coverage, manually-constructed grammar (a descendant of that described in Jensen et al., 1993). The grammar can be viewed as a context-free backbone of binary phrase-structure rules, together with an extensive set of detailed, potentially context-sensitive, conditions on each rule, referring to lexical, morphological, syntactic and semantic features.

A statistical language model – a lexicalized PCFG (similar to that of Collins, 1997) – is derived from the analysis grammar by processing a corpus using the same grammar with no statistical model and recording frequencies of substructures built by each rule. The sensitivity of the model can be tuned to include any of the features referred to by rule conditions, including neighboring or descendant nodes.

The training phase for the model requires no manual annotation of the corpus, although some manual filtering was done to attempt to exclude any particularly bad parses. For the approximately 200 rule grammar, a corpus of 25,000 sentences was used for training, selected from a variety of sources and genres.

The model is then used in subsequent analysis with the same grammar to guide bottom-up rule applications to build the most probable substructures first, acting to direct the search through the structures licensed by the grammar.

3.2 Generation

An equivalent guidance is also required in generation, although here structures are built top-down. The analysis grammar cannot be used directly for generation, though the statistical model depends on the rules in this grammar. For generation, then, a simplified grammar is derived from the full form, effectively retaining only the context-free backbone and discarding almost all of the detailed rule conditions. The statistical model is therefore still applicable to the derived generation grammar. Probabilities can be determined for substructures exactly as in the analysis grammar, thus retaining the effects of the rule conditions without requiring reversing and explicit testing when generating.

The derived, simplified, generation grammar, because almost all rule conditions have been discarded, will massively overgenerate.[1] However, the statistical language model, because of the one-to-one correspondence of rules in the full and simplified grammars, provides an immediate way to constrain the overgeneration. Structures allowed by the generation grammar but which are excluded in

[1] This is in addition to the overgeneration known to be part of the original analysis grammar to allow for certain classes of ungrammatical input

analysis will simply be assigned extremely low probabilities, and similarly for ungrammatical structures, which may be allowed by the analysis grammar but which occur only rarely in the training corpus.

At every choice point in the application of the grammar rules, the statistical model is available to indicate a preference.

4 The Generation System

The current implementation of the generation system operates in three distinct stages:

1. the semantic representation (basically, a representation of argument structure) is mapped to an unordered set of syntactic nodes,
2. the generation grammar is used to create a tree structure to order the syntactic nodes and insert any additional, syntactically required, nodes,
3. an inflected form of each leaf node in the final tree is produced, and a final string generated.

The input semantic representation is roughly equivalent to a quasi-logical form (QLF) (Alshawi, 1992), abstracting away from structural syntactic dependencies but making explicit many surface features. In particular, each input is known to be a sentence unit, and all lexical choices, including prepositions and determiners, are fully specified. The remaining generation tasks are therefore the linear ordering of the lexical units, and their inflections, each handled separately by the second and third stages.

The current application context for the generation system is in the restatement of natural language database queries, for clarification and disambiguation. Analysis of the queries produces a QLF representation which is interpreted with respect to a semantic model of the current database. Interpretation results in one or more revised QLFs which are then passed to the realizer for confirmation or selection by the user before translating to the database query language.

4.1 Semantic to syntactic mapping

The first stage translates logical form relations and features to corresponding syntactic terms, referring to lexical entries where necessary. For example, the semantic representation:

```
run (+past)
actor: John
manner: quickly
```

is translated unit-by-unit to a verb phrase, a subject noun phrase, and an adverbial phrase modifier. The syntactic units are linked in a graph structure, but with no ordering constraints between them.

In the current prototype system, this stage in fact enforces a one-to-one mapping of semantic to syntactic features, though there is clear scope for extending the use of the statistical language model to direct the translation here, and to allow for a one-to-many mapping to be ranked subsequently. The analysis grammar builds logical forms compositionally, with individual grammar rules fully specifying their semantic contributions, but at present these specifications are not automatically extracted for the simplified generation grammar, and the language model is not yet fully sensitive to the semantic features in the rules. The extent to which this mapping stage can be automated is currently being investigated.

4.2 Linear ordering

The second stage of the system, to determine a linear ordering for the syntactic nodes using the generation grammar, integrates the statistical language model directly with no adaptation required from its use in analysis.

A root node is selected from the set produced in the previous stage, currently the node produced from the root of the logical form graph, though the selection could be made more flexible. The simplified generation grammar is then checked for all rules which apply to the node, testing features and relations, such as subject, derived from the semantic relations. To access the rule probabilities represented in the language model requires that a substructure is generated from each applicable rule, and the language model then assigns a probability to each substructure. The highest scoring substructure is not immediately selected though – a 'look-ahead' is carried out to evaluate any alternative structures which express the same features.

First the effect of the rule which produced the initial structure is determined, in terms of which features or relations the rule 'expresses' or 'consumes' from the input, e.g. a VP → NP VP rule, which builds a VP with a subject in analysis, will 'consume' the subject from a root VP in generation. Alternative generation paths which consume the same features or relations are then considered, and the language model again used to determine a probability for each substructure expressing the same. If an alternative path receives a higher ranking than the substructure produced initially, the initial substructure is discarded, and the remainder considered.

For the example input given above, three alternative rules in the generation grammar apply to express the adverbial modifier:

S → AVP S	(Quickly, John ran)
VP → AVP VP	(John quickly ran)
VP → VP AVP	(John ran quickly)

The first rule applies to the initial root node, and the substructure it describes is generated immediately. The other two rules apply at lower levels of the tree on alternative paths, and the respective probabilities obtained from the language model for the substructures generated with these particular lexical items are:[2]

S → AVP S	(0.073)
VP → AVP VP	(0.061)
VP → VP AVP	(0.087)

The best ranked structure is therefore that produced by the third rule, and so the generation paths including the first two are discarded. The remaining paths are then considered for the expression of the subject relation, selecting a highly probable VP → NP VP structure, with unary rules applying at the leaves (VP → Verb, NP → Noun and AVP → Adverb), until no further expansion of nodes can be made with the grammar. For any given node, ungrammatical substructures may be produced, but the language model will always be able to rank them.

The generation algorithm for the linear ordering stage can be sketched as follows:

```
1. Make the syntactic node mapped from the
   root node of the logical form, the root node
   of the new syntactic tree.
2. For each non-terminal leaf node in the tree:
   a. For each generation grammar rule that
      applies to the selected node, testing
      conditions on the semantically-derived
      relations and features (e.g. subject):
      i.   Generate the substructure described
           by the rule.
      ii.  Determine the probability for the
           substructure.
      iii. For each generation grammar rule
           that applies to the selected node at a
           lower level in the tree, and
           expresses the same semantic
           relations/features as the rule at the
           current level:
           1. Generate the substructure
              described by the rule.
           2. Determine the probability for
              the substructure.
      iv.  If a substructure generated at a
           lower level has a higher probability
           than the substructure generated at
           the current level, discard the
           substructure at the current level.
   b. Add the substructure generated at the
      current level with the highest probability
      to the current syntactic tree. If no
      substructures exist at the current level
      (no applicable rules or all discarded),
      step down one level (apply a null rule)
      and repeat from 2.a.
```

The algorithm in fact follows a head-driven node expansion, or search through the grammar, (as in Shieber et al., 1990), with the head of the most recently expanded node being selected for the next expansion (in step 2 of the algorithm above), until a leaf node is produced. However, the nature of the grammar is such that no rule expansion will have side effects on any node other than the head of its substructure, and so any search strategy will produce the same final

[2] Probabilities may be different for other lexical items, e.g. with an unambiguously intransitive verb such as "fall", the VP → AVP VP rule (John quickly fell) receives the highest ranking, based on the current training set.

tree, though more alternative paths may be considered.

The 'look-ahead' in the search (step 2.a.iii), to find other rules expressing the same features as a current rule, means that, although the rule probabilities obtained from the language model are based entirely on local rule substructures, the overall path chosen through the grammar is globally optimal. The current implementation of the look-ahead is not optimal, however, with duplicate substructures being created and evaluated for the same rules along equivalent paths, and an obvious extension would be the addition of a simple caching mechanism for substructures and their probabilities.

One adaptation of the method is to use the language model to produce an overall score for the final tree (a simple product of the substructure scores), and then 'backtrack' to consider alternative derivations. This allows an exhaustive search through the grammar, instead of finding only the best path, as described above. For the example input given above, the grammar licenses 12 alternative tree structures, with many producing the same final strings but via rules which would be excluded by the conditions in the analysis grammar (e.g. AVP → Pronoun) and which are assigned extremely low probabilities by the language model, causing the overall tree scores to differ typically by several orders of magnitude.

4.3 Inflection

Once a complete tree has been produced, each leaf node is passed to the final stage of the realization process to be inflected. Features affecting inflection, mainly Person and Number for agreement in English, are initially obtained from the logical form input or from the lexicon in the initial semantic to syntactic mapping stage, and then passed through to the appropriate syntactic nodes by the rules selected to expand the tree structure. The rules may also introduce additional syntactic features such as Case, which are then also passed through. Inflected forms for the final feature set of each leaf node are then either retrieved from the lexicon or generated by rule. A single final string is then read from the completed tree.

5 Performance

The current implementation of the generation system is as a prototype subsystem within an existing (C++) analysis system. This allows direct access to the statistical language model built for analysis, and which is used unchanged for generation. The simplified generation grammar is derived automatically from the analysis grammar, and much of the rule application mechanism and representation of tree structures is reused.

The system, including the analysis grammar and the statistical model, is under continued development, but, as is typical of generation systems, the most significant omission is the lack of any formal evaluation methodology. Bangalore & Rambow (2000) propose some interesting initial metrics, but we have not yet attempted any comparative experiments.

Informal evaluation on a non-blind training corpus of 200 logical forms (processed at approximately 40 per second on a 1GHz PC) currently shows a roughly 4% error rate (ungrammatical output) for output sentences with an average length of 7 words (maximum 14). However the vast majority of these errors (85%) are from the bad placement of adjective phrase modifiers, for example "Show [$_{NP}$ the [$_{AJP}$ larger than Monaco] countries]", due to a high probability being assigned for structures with a modifier phrase on the left. This suggests a revision of the statistical model to make it more sensitive to particular modifier types, rather than a revision of the generation algorithm. Indeed, a significant benefit of the reversible approach represented by sharing resources between analysis and generation, is to drive improvements to the resources by identifying weaknesses not apparent in a single processing mode.

6 Future Work

Planned extensions include the use of an additional bigram model to assist in cases where constituent orderings are not constrained by the grammar, such as sequences of adjectives and other modifiers. Such information can be integrated in the initial semantic to syntactic mapping stage where constituent head-words can be compared directly, or during rule selection in the linear ordering stage to indicate a precedence among otherwise equivalent constituents.

A further area of investigation is the effect of retraining the statistical model on specific genre, rather than general, corpora. This has the potential to bias selections in the grammar towards constructions typical of a certain style, such as prepositional phrase fronting in formal writing, etc.

The current generation grammar also excludes punctuation rules, though these are present in the original analysis grammar, and experimentation to determine the ability of the language model to select and place punctuation is planned.

Of course, the input representation used for the realization stage assumes that most of the challenging higher level issues in text and sentence planning have already been dealt with, but this is not yet the case.

7 Conclusion

The tight integration of the statistical language model into the generation process described here allows a 'best first' search through the possible expansions licensed by a simplified and overgenerating grammar. This contrasts with the exhaustive searches through the grammars in Nitrogen (Langkilde & Knight, 1998) and Fergus (Bangalore & Rambow, 2000), where the generation algorithm operates independently of the statistical resources. Such integration has the potential to produce easily tunable generation systems based around a stable comprehensive grammar, as well as indicate precisely which statistical language models are most suited to generation requirements, and whether these requirements differ at all from those of analysis.

References

Alshawi, H., ed., (1992) *The Core Language Engine*, MIT Press, Cambridge, Massachusetts, USA.

Bangalore S. and Rambow O. (2000) *Exploiting a probabilistic hierarchical model for generation*. In "Proceedings of COLING-2000", Saarbrücken, Germany.

Collins, M.J. (1997) *Three generative lexicalised models for statistical parsing*. In "Proceedings of ACL-EACL'97", Madrid, Spain.

Jensen K., Heidorn G.E. and Richardson S.D., eds., (1993) *Natural language processing: the PLNLP approach*, Kluwer Academic Publishers, Boston, Massachusetts, USA.

Knight K. And Hatzivassiloglou V. (1995) *Two-level, Many-Paths Generation*. In "Proceedings of ACL'95", Cambridge, Massachusetts, USA.

Langkilde I. and Knight K. (1998) *Generation that exploits corpus-based statistical knowledge*. In "Proceedings of COLING-ACL'98", Montreal, Canada.

Nuemann G. and van Noord G. (1994) *Reversibility and self-monitoring in natural language generation*. In "Reversible Grammar in Natural Language Processing", Strzalkowski T., ed., Kluwer Academic Publishers, Dordrecht, The Netherlands.

Reiter E. (1994) *Has a consensus NL generation architecture appeared, and is it psychologically plausible?* In "Proceedings of the 7th International Workshop on Natural Language Generation", Maine, USA.

Shieber S.M., van Noord G., Moore R.C. and Pereira C.N. (1990) *Semantic-head-driven generation*. Computational Linguistics, 16(1).

XTAG-Group The (1999) *A lexicalised Tree Adjoining Grammar for English*. Technical report, Institute for Research in Cognitive Science, University of Pennsylvania, USA.