Dependency-Based N-Gram Models for General Purpose Sentence Realisation

Yuqing Guo
NCLT, School of Computing
Dublin City University
Dublin 9, Ireland
yguo@computing.dcu.ie

Josef van Genabith
NCLT, School of Computing
Dublin City University
IBM CAS, Dublin, Ireland
josef@computing.dcu.ie

Haifeng Wang
Toshiba (China)
Research & Development Center
Beijing, 100738, China
wanghaifeng@rdc.toshiba.com.cn

Abstract

We present dependency-based n-gram models for general-purpose, wide-coverage, probabilistic sentence realisation. Our method linearises unordered dependencies in input representations *directly* rather than via the application of grammar rules, as in traditional chart-based generators. The method is simple, efficient, and achieves competitive accuracy and complete coverage on standard English (Penn-II, 0.7440 BLEU, 0.05 sec/sent) and Chinese (CTB6, 0.7123 BLEU, 0.14 sec/sent) test data.

1 Introduction

Sentence generation,[1] or surface realisation can be described as the problem of producing syntactically, morphologically, and orthographically correct sentences from a given semantic or syntactic representation.

Most general-purpose realisation systems developed to date transform the input into surface form via the application of a set of grammar rules based on particular linguistic theories, e.g. Lexical Functional Grammar (LFG), Head-Driven Phrase Structure Grammar (HPSG), Combinatory Categorial Grammar (CCG), Tree Adjoining Grammar (TAG) etc. These grammar rules are either carefully handcrafted, as those used in FUF/SURGE (Elhadad, 1991), LKB (Carroll et al., 1999), OpenCCG (White, 2004) and XLE (Crouch et al., 2007), or created semi-automatically (Belz, 2007), or fully automatically extracted from annotated corpora, like the HPSG (Nakanishi et al., 2005), LFG (Cahill and van Genabith, 2006; Hogan et al., 2007) and CCG (White et al., 2007) resources derived from the Penn-II Treebank (PTB) (Marcus et al., 1993).

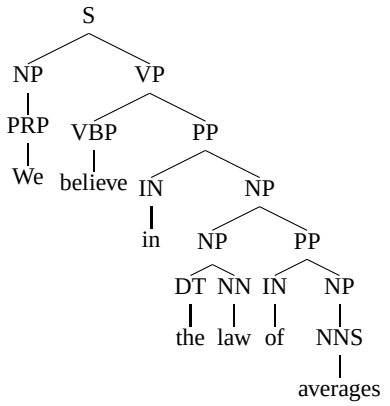
Over the last decade, probabilistic models have become widely used in the field of natural language generation (NLG), often in the form of a realisation ranker in a two-stage generation architecture. The two-stage methodology is characterised by a separation between generation and selection, in which rule-based methods are used to generate a space of possible paraphrases, and statistical methods are used to select the most likely realisation from the space. By and large, two statistical models are used in the rankers to choose output strings:

- N-gram language models over different units, such as word-level bigram/trigram models (Bangalore and Rambow, 2000; Langkilde, 2000), or factored language models integrated with syntactic tags (White et al., 2007).
- Log-linear models with different syntactic and semantic features (Velldal and Oepen, 2005; Nakanishi et al., 2005; Cahill et al., 2007).

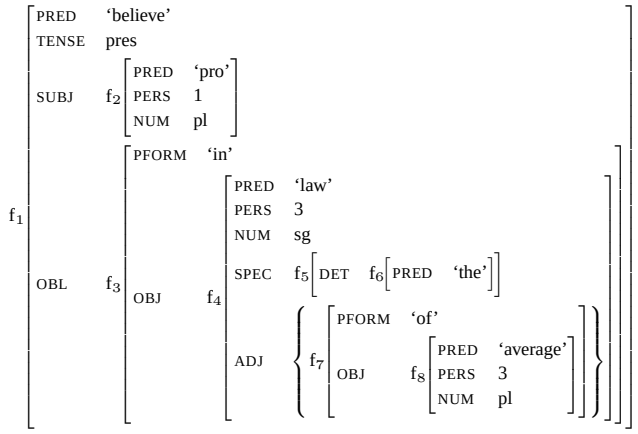
To date, however, probabilistic models learning *direct* mappings from generation input to surface strings, without the effort to construct a grammar, have rarely been explored. An exception is Ratnaparkhi (2000), who presents maximum entropy models to learn attribute ordering and lexical choice for sentence generation from a semantic representation of attribute-value pairs, restricted to an air travel domain.

[1] In this paper, the term "generation" is used generally for what is more strictly referred to by the term "tactical generation" or "surface realisation".

Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 297–304
Manchester, August 2008



(a.) c-structure



(b.) f-structure

string position	We 1	believe 2	in 3	the 4	law 5	of 6	averages 7
f_1	SUBJ	PRED	OBL				
f_3			PFORM		OBJ		
f_4				SPEC	PRED	ADJ	
f_7						PFORM	OBJ

(c.) linearised grammatical functions / bilexical dependencies

Figure 1: C- and f-structures for the sentence *We believe in the law of averages*.

In this paper, we develop an efficient, wide-coverage generator based on simple n-gram models to directly linearise dependency relations from the input representations. Our work is aimed at general-purpose sentence generation but couched in the framework of Lexical Functional Grammar. We give an overview of LFG and the dependency representations we use in Section 2. We describe the general idea of our dependency-based generation in Section 3 and give details of the n-gram generation models in Section 4. Section 5 explains the experiments and provides results for both English and Chinese data. Section 6 compares the results with previous work and between languages. Finally we conclude with a summary and outline future work.

2 LFG-Based Generation

2.1 Lexical Functional Grammar

Lexical Functional Grammar (Kaplan and Bresnan, 1982) is a constraint-based grammar formalism which postulates (minimally) two levels of representation: c(onstituent)-structure and f(unctional)-structure. As illustrated in Figure 1, a c-structure is a conventional phrase structure tree and captures surface grammatical configurations. The f-structure encodes more abstract functional relations like SUBJ(ect), OBJ(ect) and ADJ(unct). F-structures are hierarchical attribute-value matrix representations of bilexical labelled dependencies, approximating to basic predicate-argument/adjunct structures.[2] Attributes in f-structure come in two different types:

- Grammatical Functions (GFs) indicate the relationship between the predicate and dependents. GFs can be divided into:
  - arguments are subcategorised for by the predicate, such as SUBJ(ect), OBJ(ect), and thus can only occur once in each local f-structure.
  - modifiers like ADJ(unct), COORD(inate) are not subcategorised for by the predicate, and can occur any number of times in a local f-structure.
- Atomic-valued features describe linguistic properties of the predicate, such as TENSE, ASPECT, MOOD, PERS, NUM, CASE etc.

2.2 Generation from F-Structures

Work on generation in LFG generally assumes that the generation task is to determine the set of strings of the language that corresponds to a specified f-structure, given a particular grammar (Kaplan and Wedekind, 2000). Previous work on generation

[2] F-structures can be also interpreted as quasi-logical forms (van Genabith and Crouch, 1996), which more closely resemble inputs used by some other generators.

within LFG includes the XLE,[3] Cahill and van Genabith (2006), Hogan et al. (2007) and Cahill et al. (2007). The XLE generates sentences from f-structures according to parallel handcrafted grammars for English, French, German, Norwegian, Japanese, and Urdu. Based on the German XLE resources, Cahill et al. (2007) describe a two-stage, log-linear generation model. Cahill and van Genabith (2006) and Hogan et al. (2007) present a chart generator using wide-coverage PCFG-based LFG approximations automatically acquired from treebanks (Cahill et al., 2004).

3 Dependency-Based Generation: the Basic Idea

Traditional LFG generation models can be regarded as the reverse process of parsing, and use bi-directional f-structure-annotated CFG rules. In a sense, the generation process is driven by an input dependency (or f-structure) representation, but proceeds through the "detour" of using dependency-annotated CFG (or PCFG) grammars and chart-based generators. In this paper, we develop a simple n-gram and dependency-based, wide-coverage, robust, probabilistic generation model, which cuts out the middle-man from previous approaches: the CFG-component.

Our approach is data-driven: following the methodology in (Cahill et al., 2004; Guo et al., 2007), we automatically convert the English Penn-II treebank and the Chinese Penn Treebank (Xue et al., 2005) into f-structure banks. F-structures such as Figure 1(b.) are unordered, i.e. they do *not* carry information on to the relative surface order of local GFs. In order to generate a string from an f-structure, we need to linearise the GFs (at each level of embedding) in the f-structure (and map lemmas and features to surface forms). We do this in terms of n-gram models over GFs. In order to build the n-gram models, we linearise the f-structures automatically produced from treebanks by associating the numerical string position (word offset from start of the sentence) with the predicate in each local f-structure, producing GF sequences as in Figure 1(c.).

Even though the n-gram models are exemplified using LFG f-structures, they are general-purpose models and thus suitable for any bilexical labelled dependency (Nivre, 2006) or predicate-argument type representations, such as the labelled feature-value structures used in HALogen and the functional descriptions in the FUF/SURGE system.

[3] http://www2.parc.com/isl/groups/nltt/xle/

4 N-Gram Models for Dependency-Based Generation

4.1 Basic N-Gram Model

The primary task of a sentence generator is to determine the linear order of constituents and words, represented as lemmas in predicates in f-structures. At a particular local f-structure, the task of generating a string covered by the local f-structure is equivalent to linearising all the GFs present at that local f-structure. E.g. in f_4 in Figure 1, the unordered set of local GFs {SPEC, PRED, ADJ} generates the surface sequence "the law of averages". We linearise the GFs in the set by computing n-gram models, similar to traditional word-based language models, except using the names of GFs (including PRED) instead of words. Given a (sub-) f-structure F containing m GFs, the n-gram model searches for the best surface sequence S_1^m=$s_1...s_m$ generated by the GF linearisation $GF_1^m = GF_1...GF_m$, which maximises the probability $P(GF_1^m)$. Using n-gram models, $P(GF_1^m)$ is calculated according to Eq.(1).

$$\begin{aligned} P(GF_1^m) &= P(GF_1...GF_m) \\ &= \prod_{k=1}^{m} P(GF_k|GF_{k-n+1}^{k-1}) \end{aligned} \quad (1)$$

4.2 Factored N-Gram Models

In addition to the basic n-gram model over bare GFs, we integrate contextual and fine-grained lexical information into several factored models. Eq.(2) additionally conditions the probability of the n-gram on the parent GF label of the current local f-structure f_i, Eq.(3) on the instantiated PRED of the local f-structure f_i, and Eq.(4) lexicalises the model, where each GF is augmented with its own predicate lemma.

$$P^g(GF_1^m) = \prod_{k=1}^{m} P(GF_k|GF_{k-n+1}^{k-1}, GF_i) \quad (2)$$

$$P^p(GF_1^m) = \prod_{k=1}^{m} P(GF_k|GF_{k-n+1}^{k-1}, Pred_i) \quad (3)$$

$$P^l(GF_1^m) = \prod_{k=1}^{m} P(Lex_k|Lex_{k-n+1}^{k-1}) \quad (4)$$

To avoid data sparseness, the factored n-gram models P^f are smoothed by linearly interpolating the basic n-gram model P, as in Eq.(5).

$$\hat{P}^f(GF_1^m) = \lambda P^f(GF_1^m) + (1-\lambda)P(GF_1^m) \quad (5)$$

Additionally, the lexicalised n-gram models P^l are combined with the other two models conditioned on the additional parent GF P^g and PRED P^p, as shown in Eqs. (6) & (7), respectively.

$$\hat{P}^{lg}(GF_1^m) = \lambda_1 P^l(GF_1^m) + \lambda_2 P^g(GF_1^m) + \lambda_3 P(GF_1^m) \quad (6)$$

$$\hat{P}^{lp}(GF_1^m) = \lambda_1 P^l(GF_1^m) + \lambda_2 P^p(GF_1^m) + \lambda_3 P(GF_1^m) \quad (7)$$

$$where \quad \sum \lambda_i = 1$$

Table 1 exemplifies the different n-gram models for the local f-structure f_4 in Figure 1.

Model	N-grams			Cond.
basic (P)	SPEC	PRED	ADJ	
gf (P^g)	SPEC	PRED	ADJ	OBL
pred (P^p)	SPEC	PRED	ADJ	'law'
lex (P^l)	SPEC	PRED['law']	ADJ['of']	

Table 1: Examples of n-grams for f_4 in Figure 1

Besides grammatical functions, we also make use of atomic-valued features like TENSE, PERS, NUM (etc.) to aid linearisation. The attributes and values of these features are integrated into the GF n-grams for disambiguation (see Section 5.2).

4.3 Generation Algorithm

Our basic n-gram based generation model implements the simplifying assumption that linearisation at one sub-f-structure is independent of linearisation at any other sub-f-structures. This assumption is feasible for projective dependencies. In most cases (at least in English and Chinese), non-projective dependencies are only used to account for Long-Distance Dependencies (LDDs). Consider sentence (1) discussed in Carroll et al. (1999) and its corresponding f-structure in Figure 2. In LFG f-structures, LDDs are represented via reentrancies between "dislocated" TOPIC, TOPIC_REL, FOCUS (etc.) GFs and "source" GFs subcategorised for by local predicates, but only the dislocated GFs are instantiated in generation. Therefore traces of the source GFs in input f-structures are removed before generation, and non-projective dependencies are transformed into simple projective dependencies.

(1) How quickly did the newspapers say the athlete ran?

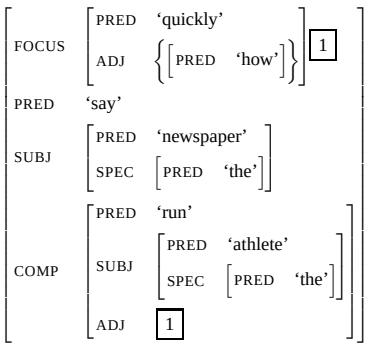


Figure 2: schematic f-structure for *How quickly did the newspapers say the athlete ran?*

In summary, given an input f-structure f, the core algorithm of the generator recursively traverses f and at each sub-f-structure f_i:

1. instantiates the local predicate at f_i and performs inflections/declensions if necessary
2. calculates the GF linearisations present at f_i by n-gram models
3. finds the most probable GF sequence among all possibilities by Viterbi search
4. generates the string covered by f_i according to the linearised GFs

5 Experiments and Evaluation

To test the performance and coverage of our n-gram-based generation models, experiments are carried out for both English and Chinese, two languages with distinct properties.

5.1 Experiment Design

Experiments on English data are carried out on the WSJ portion of the PTB, using standard training/test/development splits, viz 39,832 sentences from sections 02-21 are used for training, 2,416 sentences from section 23 for testing, while 1,700 sentences from section 22 are held out for development. The latest version of the Penn Chinese Treebank 6.0 (CTB6), excluding the portion of ACE broadcast news, is used for experiments on Chinese data.[4] We follow the recommended splits (in the list-of-file of CTB6) to divide the data into test set, development set and training set. The training set includes 756 files with a total of 15,663 sentences. The test set includes 84 files with 1,708

[4]Sentences labelled as fragment are not included in our development and test set.

sentences. The development set includes 50 files with 1,116 sentences. Table 2 shows some of the characteristics of the English and Chinese data obtained from the development sets.

Development Set	English	Chinese
num of sent	1,700	1,116
max length of sent (#words)	110	145
ave length of sent (#words)	23	31
num of local fstr	23,289	15,847
num of local fstr per sent	13.70	14.20
max length of local fstr (#gfs)	12	16
ave length of local fstr (#gfs)	2.56	2.90

Table 2: Comparison English and Chinese data

The n-gram models are created using the SRILM toolkit (Stolcke, 2002) with Good-Turning smoothing for both the Chinese and English data. For morphological realisation of English, a set of lexical macros is automatically extracted from the training data. This is not required for Chinese surface realisation as Chinese has very little morphology. Lexical macro examples are listed in Table 3.

lexical macro	surface word
pred=law, num=sg, pers=3	law
pred=average, num=pl, pers=3	averages
pred=believe, num=pl, tense=pres	believe

Table 3: Examples of lexical macros

The input to our generator are unordered f-structures automatically derived from the development and test set trees of our treebanks, which do *not* contain any string position information. But, due to the particulars of the automatic f-structure annotation algorithm, the order of sub-f-structures in set-valued GFs, such as ADJ, COORD, happens to correspond to their surface order. To avoid unfairly inflating evaluation results, we lexically reorder the GFs in each sub-f-structure of the development and test input before the generation process. This resembles the "permute, no dir" type experiment in (Langkilde, 2002).

5.2 Experimental Results

Following (Langkilde, 2002) and other work on general-purpose generators, BLEU score (Papineni et al., 2002), average NIST simple string accuracy (SSA) and percentage of exactly matched sentences are adopted as evaluation metrics. As our system guarantees that all input f-structures can generate a complete sentence, special coverage-dependent evaluation (as has been adopted in most grammar-based generation systems) is not necessary in our experiments.

Experiments are carried out on an Intel Pentium 4 server, with a 3.80GHz CPU and 3GB memory. It takes less than 2 minutes to generate all 2,416 sentences (with average sentence length of 21 words) of WSJ section 23 (average 0.05 sec per sentence), and approximately 4 minutes to generate 1,708 sentences (with average sentence length of 30 words) of CTB test data (average 0.14 sec per sentence), using 4-gram models in all experiments. Our evaluation results for English and Chinese data are shown in Tables 4 and 5, respectively.

Different n-gram models perform nearly consistently in all the experiments on both English and Chinese data. The results show that factored n-gram models outperform the basic n-gram models, and in turn the combined n-gram models outperform single n-gram models. The combined model interpolating n-grams over lexicalised GFs with n-grams conditioned on PRED achieves the best results in both experiments on English (with feature names) and Chinese (with feature names & values), with BLEU scores of 0.7440 and 0.7123 respectively, and full coverage.

Lexicalisation plays an important role in both English and Chinese, boosting the BLEU score without features from 0.5074 to 0.6741 for English, and from 0.5752 to 0.6639 for Chinese.

Atomic-valued features play an important role in English, and boost the BLEU score from 0.5074 in the baseline model to 0.6842 when feature names are integrated into the n-gram models. However, feature names in Chinese only increase the BLEU score from 0.5752 to 0.6160. This is likely to be the case as English has a richer morphology than Chinese, and important function words such as 'if', 'to', 'that' are encoded in atomic-valued features in English f-structures, which helps to determine string order. However, combined feature names and values work better on Chinese data, but turn out to hurt the n-gram model performance for English data. This may suggest that the feature names in English already include enough information, while the value of morphological features, such as TENSE, NUM does not provide any new information to help determine word order, but aggravate data sparseness instead.

WSJ Sec23	Without Features			Feature Names			Feature Names & Values		
Model	ExMatch	BLEU	SSA	ExMatch	BLEU	SSA	ExMatch	BLEU	SSA
baseline	5.30%	0.5074	57.29%	15.27%	0.6842	69.48%	15.15%	0.6829	69.15%
gf	6.62%	0.5318	60.06%	16.76%	0.6969	71.51%	16.68%	0.6977	71.55%
pred	8.03%	0.5697	60.73%	16.72%	0.7035	70.12%	16.76%	0.7042	71.08%
lex	12.87%	0.6741	69.43%	19.41%	0.7384	74.76%	18.96%	0.7375	74.12%
lex+gf	12.62%	0.6611	69.41%	19.70%	0.7388	74.98%	19.74%	0.7405	75.08%
lex+pred	12.25%	0.6569	68.04%	**19.83%**	**0.7440**	**75.34%**	19.58%	0.7422	75.04%

Table 4: Results for English Penn-II WSJ section 23

Test	Without Features			Feature Names			Feature Names & Values		
Model	ExMatch	BLEU	SSA	ExMatch	BLEU	SSA	ExMatch	BLEU	SSA
baseline	8.96%	0.5752	51.92%	11.77%	0.6160	54.64%	12.30%	0.6239	55.20%
gf	9.54%	0.6009	53.02%	12.53%	0.6391	55.78%	13.47%	0.6486	56.60%
pred	10.07%	0.6180	53.80%	13.35%	0.6608	56.72%	14.46%	0.6720	57.67%
lex	13.93%	0.6639	59.61%	15.16%	0.6770	60.44%	15.98%	0.6804	60.20%
lex+gf	14.81%	0.6773	59.92%	15.52%	0.6911	60.97%	16.80%	0.6957	61.07%
lex+pred	16.04%	0.6952	60.82%	16.22%	0.7060	61.45%	**17.51%**	**0.7123**	**61.54%**

Table 5: Results for Chinese CTB6 test data

WSJ Sec23	Sentence length $\leq$ 20 words				All sentences			
	Coverage	ExMatch	BLEU	SSA	Coverage	ExMatch	BLEU	SSA
Langkilde(2002)					82.7%	28.2%	**0.757**	69.6%
Callaway(2003)					98.7%	**49.0%**		**88.84%**
Nakanishi(2005)	90.75%		**0.7733**		83.6%		0.705	
Cahill(2006)	98.65%		0.7077	73.73%	98.05%		0.6651	68.08%
Hogan(2007)	100%		0.7139		99.96%		0.6882	70.92%
White(2007)					94.3%	6.9%	0.5768	
this paper	**100%**	35.40%	0.7625	81.09%	**100%**	19.83%	0.7440	75.34%

Table 6: Cross system comparison of results for English WSJ section 23

6 Discussion

6.1 Comparison to Previous Work

It is very difficult to compare sentence generators since the information contained in the input representation varies greatly between systems. The most direct comparison is between our system and those presented in Cahill and van Genabith (2006) and Hogan et al. (2007), as they also use treebank-based automatically generated f-structures as the generator inputs. The labelled feature-value structures used in HALogen (Langkilde, 2002) and functional descriptions in FUF/SURGE (Callaway, 2003) also bear some broad similarities to our f-structures. A number of systems using different input but adopting the same evaluation metrics and testing on the same data are listed in Table 6.

Surprisingly (or not), the best results are achieved by a purely symbolic generation system—FUF/SURGE (Callaway, 2003). However the approach uses handcrafted grammars which are very time-consuming to produce and adapt to different languages and domains. Langkilde (2002) reports results for experiments with varying levels of linguistic detail in the input given to the generator. The type "permute, no dir" is most comparable to the level of information contained in our f-structure in that the modifiers (adjuncts, coordinates etc.) in the input are not ordered. However her labelled feature-value structure is more specific than our f-structure as it also includes syntactic properties such as part-of-speech, which might contribute to the higher BLEU score of HALogen. And moreover, in HALogen nearly 20% of the sentences are only partially generated (or not at all). Nakanishi et al. (2005) carry out experiments on sentences up to 20 words, with BLEU scores slightly higher than ours. However their results without sentence length limitation (listed in the right column), for 500 sentences randomly selected from WSJ Sec22 are lower than ours, even at a lower coverage. Overall our system is competitive, with best results for coverage (100%), second best for BLEU and SSA scores, and third best overall on exact match. However, we admit that automatic metrics such as BLEU are not fully reliable to compare different systems, and results vary widely depending on the coverage of the systems and the specificity of the generation input.

6.2 Error Analysis and Differences Between the Languages

Though our dependency-based n-gram models perform well in both the English and Chinese experiments, we are surprised that experiments on English data produce better results than those for Chinese. It is widely accepted that English generation is more difficult than Chinese, due to morphological inflections and the somewhat less predictable word order of English compared to Chinese. This is reflected by the results of the baseline models. Chinese has a BLEU score of 0.5752 and 8.96% exact match, both are higher than those of English. However with feature augmentation and lexicalisation, the results for English data exceed Chinese. This is probably because of the following reasons:

Data size of the English training set is more than twice that of Chinese.

Grammatical functions are more fine-grained in English f-structures than those in Chinese. There are 32 GFs defined for English compared to 20 for Chinese in our input f-structures.

Properties of the languages and data sets are different. For example, due to lack of inflection and case markers, many sequences of VPs in Chinese have to be treated as coordinates, whereas their counterparts in English act as different grammatical functions, e.g. (2).

(2) 投资 百万 兴建 这个 工程
invest million build this construction
'invest million yuan *to* build the construction'

This results in a total of 7,377 coordinates (4.32 per sentence) in the Chinese development data, compared to 2,699 (1.12 per sentence) in the English data. The most extreme case in the Chinese data features 14 coordinates of country names in a local f-structure. This may account for the low SSA score for the Chinese experiments, as many coordinates are tied in the n-gram scoring method and can not be ordered correctly. Examining the development data shows different types of coordination errors:

- syntactic coordinates, but not semantic coordinates, as in sentence (2).
- syntactic and semantic coordinates, but usually expressed in a fixed order, e.g. (3).

  (3) 改革 开放
  reform opening-up
  'reform and opening up'

- syntactic and semantic coordinates, which can freely swap positions, e.g. (4).

  (4) 充沛的 精力 和 敏捷的 思维
  plentiful energy and quick thinking
  'energetic and agile'

At the current stage, our n-gram generation model only keeps the most likely realisation for each local f-structure. We believe that packing all equivalent elements, like coordinates in a local f-structure into equivalent classes, and outputing n-best candidate realisations will greatly increase the SSA score and may also further benefit the efficiency of the algorithm.

7 Conclusions and Further Work

We have described a number of increasingly sophisticated n-gram models for sentence generation from labelled bilexical dependencies, in the form of LFG f-structures. The models include additional conditioning on parent GFs and different degrees of lexicalisation. Our method is simple, highly efficient, broad coverage and accurate in practice. We present experiments on English and Chinese, showing that the method generalises well to different languages and data sets. We are currently exploring further combinations of conditioning context and lexicalisation, application to different languages and to dependency representations used to train state-of-the-art dependency parsers (Nivre, 2006).

Acknowledgments

This research is funded by Science Foundation Ireland grant 04/IN/I527. We thank Aoife Cahill for providing the treebank-based LFG resources for the English data. We gratefully acknowledge the feedback provided by our anonymous reviewers.

References

Bangalore, Srinivas and Rambow, Owen. 2000. Exploiting a Probabilistic Hierarchical Model for Generation. *Proceedings of the 18th International Conference on Computational Linguistics*, 42–48. Saarbrücken, Germany.

Belz, Anja. 2007. Probabilistic Generation of Weather Forecast Texts. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, 164–171. New York.

Cahill, Aoife, Burke, Michael, O'Donovan, Ruth, van Genabith, Josef and Way, Andy. 2004. Long-Distance Dependency Resolution in Automatically

Acquired Wide-Coverage PCFG-Based LFG Approximations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, 320-327. Barcelona, Spain.

Cahill, Aoife and van Genabith, Josef. 2006. Robust PCFG-Based Generation Using Automatically Acquired LFG Approximations. *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 1033–1040. Sydney, Australia.

Cahill, Aoife, Forst, Martin and Rohrer, Christian. 2007. Stochastic Realisation Ranking for a Free Word Order Language. *Proceedings of the 11th European Workshop on Natural Language Generation*, 17–24. Schloss Dagstuhl, Germany.

Callaway, Charles B.. 2003. Evaluating Coverage for Large Symbolic NLG Grammars. *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, 811–817. Acapulco, Mexico.

Carroll, John, Copestake, Ann, Flickinger, Dan and Poznanski, Victor. 1999. An efficient chart generator for (semi-)lexicalist grammars. *Proceedings of the 7th European Workshop on Natural Language Generation*, 86–95. Toulouse, France.

Crouch, Dick, Dalrymple, Mary, Kaplan, Ron, King, Tracy, Maxwell, John and Newman, Paula. 2007. XLE Documentation. Palo Alto Research Center, CA.

Elhadad, Michael. 1991. FUF: The universal unifier user manual version 5.0. Technical Report CUCS-038-91. Dept. of Computer Science, Columbia University.

Guo, Yuqing and van Genabith, Josef and Wang, Haifeng. 2007. Treebank-based Acquisition of LFG Resources for Chinese. *Proceedings of LFG07 Conference*, 214–232. Stanford, CA, USA.

Hogan, Deirdre Cafferkey, Conor Cahill, Aoife and van Genabith, Josef. 2007. Exploiting Multi-Word Units in History-Based Probabilistic Generation. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and CoNLL*, 267–276. Prague, Czech Republic.

Kaplan, Ronald and Bresnan, Joan. 1982. Lexical Functional Grammar: a Formal System for Grammatical Representation. *The Mental Representation of Grammatical Relations*, 173–282. MIT Press, Cambridge.

Kaplan, Ronald and Wedekind, Jurgen. 2000. LFG Generation Produces Context-free Languages. *Proceedings of the 18th International Conference on Computational Linguistics*, 425–431. Saarbrücken, Germany.

Langkilde, Irene. 2000. Forest-Based Statistical Sentence Generation. *Proceedings of 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, 170–177. Seattle, WA.

Langkilde, Irene. 2002. An Empirical Verification of Coverage and Correctness for a General-Purpose Sentence Generator. *Proceedings of the Second International Conference on Natural Language Generation*, 17–24. New York, USA.

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

Nakanishi, Hiroko and Nakanishi, Yusuke and Tsujii, Jun'ichi. 2005. Probabilistic Models for Disambiguation of an HPSG-Based Chart Generator. *Proceedings of the 9th International Workshop on Parsing Technology*, 93–102. Vancouver, British Columbia.

Nivre, Joakim. 2006. Inductive Dependency Parsing. Springer.

Papineni, Kishore, Roukos, Salim, Ward, Todd and Zhu, Wei-Jing. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311-318. Philadelphia, USA.

Ratnaparkhi, Adwait. 2000. Trainable methods for natural language generation. *Proceedings of NAACL 2000*, 194–201. Seattle, WA.

Stolcke, Andreas. 2002. SRILM-An Extensible Language Modeling Toolkit. *Proceedings of International Conference of Spoken Language Processing*. Denver, Colorado.

van Genabith, Josef and Crouch, Dick. 1996. Direct and underspecified interpretations of LFG f-structures. *Proceedings of the 16th conference on Computational linguistics*, 262–267. Copenhagen, Denmark

Velldal, Erik and Oepen, Stephan. 2005. Maximum entropy models for realization ranking. *Proceedings of the MTSummit '05*.

White, Michael. 2004. Reining in CCG Chart Realization. *Proceedings of the third International Natural Language Generation Conference*. Hampshire, UK.

White, Michael, Rajkumar, Rajakrishnan and Martin, Scott. 2007. Towards Broad Coverage Surface Realization with CCG. *Proceedings of the MT Summit XI Workshop*, 22–30. Copenhagen, Danmark.

Xue, Nianwen, Xia, Fei, Chiou, Fu dong and Palmer, Martha. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2): 207–238.