DCU*at Generation Challenges 2011 Surface Realisation Track

Yuqing Guo
Toshiba Research and Development Center
5/F., Tower W2, Oriental Plaza,
Dongcheng District, Beijing, China
guoyuqing@rdc.toshiba.com.cn

Deirdre Hogan and Josef van Genabith
NCLT/CNGL, School of Computing,
Dublin City University,
Glasnevin, Dublin 9, Ireland.
dhogan,josef@computing.dcu.ie

Abstract

In this paper we describe our system and experimental results on the development set of the Surface Realisation Shared Task. DCU submitted 1-best outputs for the Shallow subtask of the shared task, using a surface realisation technique based on dependency-based n-gram models. The surface realiser achieved BLEU and NIST scores of 0.8615 and 13.6841 respectively on the SR development set.

1 Introduction

DCU submitted outputs for SR-Shallow, the shallow sub-task of the surface realisation shared task, using a surface realisation technique based on dependency-based n-gram models, described in some detail in (Guo et al., 2010).

The generation method captures the mapping between the surface form sentences and the unordered syntactic representations of the shallow representation by linearising a set of dependencies *directly*, rather than via the application of grammar rules as in more traditional chart-style or unification-based generators (White, 2004; Nakanishi et al., 2005; Cahill and van Genabith, 2006; Hogan et al., 2007; White and Rajkumar, 2009). In contrast to conventional n-gram language models over surface word forms (Langkilde-Geary, 2002), we exploit structural information and various linguistic features inherent in the dependency representations to constrain the generation space and improve the generation quality.

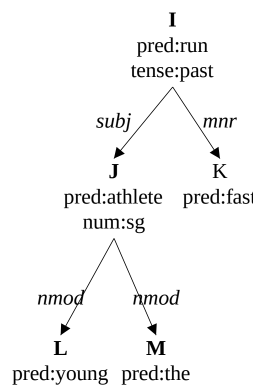

Figure 1: Unordered dependency tree for the input of the sentence: the young athlete ran fast

2 Dependency-based N-gram Models

The shallow input representation takes the form of an unordered dependency tree. The basic approach of the surface realisation method is to traverse the input tree ordering the nodes at each sub-tree based on local information. For each sub-tree the nodes are ordered according to a combination of n-gram models of increasing specificity. At the most general level, for a particular sub-tree, the n-gram model simply models the grammatical relations (including the predicate/head) of the sub-tree. Take for example the sub-tree rooted at node I from Figure 1. The realiser linearises the lemmas at nodes I, J and K by learning the correct order of the syntactic relations (in this case *subj* $\prec$ *pred* $\prec$ *mnr*).

Formally, in our most basic model, for a lo-

*Throughout this document DCU stands for the joint team of Dublin City University and Toshiba (China) Research and Development Center participating in the SR Task 2011.

Proceedings of the 13th European Workshop on Natural Language Generation (ENLG), pages 227–229,
Nancy, France, September 2011.

cal sub-tree t_i containing m grammatical relations (GRs) (including $pred$), generating a surface string $S_1^m = s_1...s_m$ expressed by t_i is equivalent to linearising all the GRs present at t_i. The dependency n-gram (DN-gram) model calculates probabilities for all permutations $GR_1^m = GR_1...GR_m$, and searches for the best surface sequence that maximises the probability $P(S_1^m)$ in terms of maximising $P(GR_1^m)$. Applying the chain rule and the Markov assumption, the probability of the surface realisation is computed according to Eq. (1).

$$P(S_1^m) = P(GR_1^m) = P(GR_1...GR_m) = \prod_{k=1}^{m} P(GR_k|GF_{k-n+1}^{k-1}) \quad (1)$$

The basic dependency n-gram model over bare GRs is not a good probability estimator as it only makes use of a few dozen grammatical function roles. For example there is no way to capture the difference between two nominal modifiers according to the labels of the two GRs. In order to facilitate better decisions, we extend the basic model to a number of more complex DN-gram models incorporating contextual information such as the syntactic relation of the parent of a node, as well as local node information (e.g. $tense$ and $number$ features). In the most specific model all grammatical relations are lexicalised (in the case of subtree rooted at node I from Figure 1 the model learns: *subj(athlete)* $\prec$ *pred(run)* $\prec$ *mnr(fast)*). Log-linear interpolations (LLI) are used to combine the estimates from the different DN-gram models:

$$P^{LLI}(S_1^m) = \prod_i P_i(S_1^m)^{\lambda_i} \quad (2)$$

3 The Realisation Algorithm

In order to generate the surface lexical form corresponding to an input lemma, morphological alternation has to be determined. From the training corpus, we use the grammatical properties like number, part-of-speech tag, tense, and participle feature which are encoded in the input nodes, to learn a mapping from lemma to the appropriate word form in the surface realisation.

The generation process proceeds as follows: Given an input tree T consisting of unordered projective[1] dependencies, the generation algorithm recursively traverses T in a bottom-up fashion and at each sub-tree t_i:

1. instantiates the local predicate $pred_i$ at t_i and performs morphological inflections if necessary
2. calculates DN-gram probabilities of possible GR permutations licensed by t_i
3. finds the most probable GR sequence among all possibilities by Viterbi search
4. generates the surface string s_i according to the best GR sequence as a realisation of t_i
5. propagates s_i up to the parent sub-tree.

4 Experimental Results

Results of the surface generator on the SR development set, trained exclusively on the SR training set, are displayed in Table 1.

BLEU-4	NIST	METEOR
0.8615	13.6841	0.8925

Table 1: Results on the development set

References

Aoife Cahill and Josef van Genabith. 2006. Robust PCFG-based generation using automatically acquired LFG approximations. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yuqing Guo, Haifeng Wang, and Josef van Genabith. 2010. Dependency-based n-gram models for general purpose sentence realisation. *Natural Language Engineering*, 1(1):1–29.

Deirdre Hogan, Conor Cafferkey, Aoife Cahill, and Josef van Genabith. 2007. Exploiting multi-word units in history-based probabilistic generation. In *In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Conference on Computational Natural Language Learning*.

[1]The algorithm assumes all dependencies are projective and therefore has a somewhat inadequate handling of the non-projective dependencies that do exist in the SR data. For example, for the input dependency tree of sentence *Why , they wonder , should it belong to the EC ?* (training set sentId=32553) the algorithm can not generate the original word order. A further pre-processing step is needed to make all dependencies projective.

Irene Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the 2nd International Natural Language Generation Conference (INLG)*.

Hiroko Nakanishi, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic models for disambiguation of an hpsg-based chart generator. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT)*.

Michael White and Rajakrishnan Rajkumar. 2009. Perceptron reranking for ccg realization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-09)*.

Michael White. 2004. Reining in ccg chart realization. In *Proceedings of the 3rd International Natural Language Generation Conference)*.