Implementing Alignment-Based Learning

Menno van Zaanen

FNWI / ILLC
Cognitive Systems and Information Processing Group
University of Amsterdam
Nieuwe Achtergracht 166
1018 WV AMSTERDAM
The Netherlands
mvzaanen@science.uva.nl
http://www.science.uva.nl/~mvzaanen

Abstract. In this article, the current implementation of the Alignment-Based Learning (ABL) framework (van Zaanen, 2002) will be described. ABL is an unsupervised grammar induction system that is based on Harris's (1951) idea of substitutability. Instances of the framework can be applied to an untagged, unstructured corpus of natural language sentences, resulting in a labelled, bracketed version of that corpus.
Firstly, the framework aligns all sentences in the corpus in pairs, resulting in a partition of the sentences consisting of parts of the sentences that are equal in both sentences and parts that are unequal. Since substituting one unequal part for the other results in another valid sentence, the unequal parts of the sentences are considered to be possible (possibly overlapping) constituents. Secondly, of all possible constituents found by the first phase, the best are selected.

1 Introduction

The unsupervised grammar induction system Alignment-Based Learning (ABL) has been described extensively in (van Zaanen, 2002). In this article we will only give a condensed overview of the theoretical system and then concentrate on the current implementation of ABL.

The ABL system normally consists of two distinct phases. However, the implementation described here subdivides the first phase into two separate ones. All three phases will be described here briefly.

The *alignment learning phase* finds possible constituents, called hypotheses, by aligning pairs of sentences to each other. Following a reversed version of Harris's (1951) implication (*if parts of sentences can be substituted by each other then they are constituents of the same type*) groups of words that are *unequal* in a pair of sentences are considered hypotheses.

Figure 1 illustrates how ABL finds hypotheses. The first sentence is aligned to the second. The underlined words indicate similar parts in the sentences. The dissimilar parts (*Book Delta 128* and *Give me all flights*) are now considered hypotheses (which is indicated by the pair of brackets).

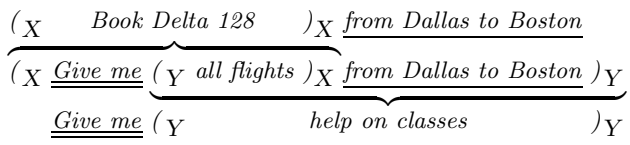

Fig. 1. Overlapping hypotheses

The system also assigns non-terminal types to the hypotheses. If hypotheses with different non-terminals occur in the same context, then the non-terminals are merged. This is normally part of the alignment learning phase, but the implementation contains a separate *cluster phase* that performs this function.

When the second sentence is aligned to the third, it receives another hypothesis, which overlaps with the older hypothesis. Since the underlying grammar is assumed to be context-free, overlapping constituents are unwanted. The *selection learning phase* eliminates the unwanted, overlapping hypotheses.

2 Implementation

The main programs of the current ABL implementation are written in C++, while some of the support programs are written in Perl. The entire package will be made available soon.

The package has three main entry points. These programs represent the main tasks in a particular project. Firstly, files can be converted to different formats. Secondly, the actual learning takes place and finally, the learned corpus can be evaluated against the original corpus. Each of these three steps is represented by a specific program.

The main programs call many sub-programs, each of which perform a specific task. These programs can also be called by the user directly. The main entry points merely make it easier to handle the different options and data files.

2.1 Conversion

The `abl_convert` program can convert files from and to different formats. The current ABL package recognises the following formats:

plain In this format, the file contains plain sentences only.
gold This is the gold format as used in the EVALB program (Collins, 1997).
abl The abl format is the output of all ABL learning programs.
atis, wsj and ovis The format of the ATIS, Wall Street Journal (Marcus et al., 1993) and OVIS treebanks (Bonnema et al., 1997) can be used as input only.

It converts that file from the input into the wanted output format. Even though this program is not necessarily part of the ABL system, it is a useful addition, since these conversions are often necessary.

2.2 Running ABL

The main program of the package is called `abl_build`. It can be told to do one, two or all three phases of ABL. The user can choose if the output of a phase

should be stored for further evaluation or is intermediate and can be discarded afterwards.

Furthermore, one or more different instantiations of each phase can be selected as well as a number of random seeds to be used. This allows the building of several output files (with automatically generated file names) in only one program call.

2.3 Evaluation

The third program of the package, `abl_eval`, takes care of the evaluation. It should be told which files to evaluate and which evaluation metrics should be computed.

Currently, the evaluation program calls the EVALB program to compute most metrics, but the number of learned hypotheses and the number of introduced non-terminal types can also be counted. Additionally, the maximum obtainable scores on all metrics after the cluster phase can be computed by selecting hypotheses only if they occur in the original treebank. This simulates a perfect selection learning phase.

3 Conclusion and Future Work

The current implementation of the ABL grammar induction system, which will become available soon, has three main programs with an easy interface which makes it very user-friendly. A graphical user interface is currently under development. Because it is programmed in C++, it is efficient and easily extendible.

In the near future, we expect to implement new instantiations of all three phases. Special attention will be given to the cluster algorithm. Additionally, a grammar induction phase, which extends the ABL system to a *parseABL* system will be added and finally, we are working on a parallel version of the alignment learning phase, since this is the most time consuming phase of the algorithm.

Bibliography

ACL (1997). *Proceedings of the 35th Annual Meeting of the ACL and the 8th Meeting of the EACL; Madrid, Spain*. ACL.

Bonnema, R., Bod, R., and Scha, R. (1997). A DOP model for semantic interpretation. In ACL (1997), pages 159–167.

Collins, M. (1997). Three generative, lexicalised models for statistical parsing. In ACL (1997), pages 16–23.

Harris, Z. S. (1951). *Structural Linguistics*. University of Chicago Press, Chicago:IL, USA and London, UK, 7th (1966) edition.

Marcus, M., Santorini, B., and Marcinkiewicz, M. (1993). Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):313–330.

van Zaanen, M. (2002). *Bootstrapping Structure into Language: Alignment-Based Learning*. PhD thesis, University of Leeds, Leeds, UK.