Using DOP for transformations

Andreas van Cranenburgh, Joost Winter

June 4, 2010

Abstract

In this paper, we make an investigation into Data Oriented Transformations, basing ourselves on the existing concepts of Data Oriented Parsing, synchronous grammars, and the Goodman Reduction.

1 Introduction

Data Oriented Parsing (DOP) has proven to be a successful framework for the parsing of natural language and several other applications in computational linguistics. In this paper, we will explore options of using a DOP-based method to be able to perform frequently occurring transformations in natural language. The main example used of such transformations will be that between declarative and interrogative sentences in English, but the developed framework should be generalizable to other transformations, such as:

- The transformation between the 'main clause' and 'auxiliary clause' sentence order in Dutch.
- Transformations between affirmative and negative sentences.
- Similar transformations in other languages.

For this project, we make use of a number of existing frameworks that may prove useful to us. In particular, these include:

- Synchronous grammars, as developed by David Chiang in [Ch].
- Data Oriented Translation framework by Arjen Poutsma in [Po] and [Po2].
- The Goodman-reduction, as developed by Joshua Goodman in [Go].

In section 2, these underlying frameworks (as well as their relevance to our project) will be explained in a somewhat more detailed fashioned; then in section 3, our own approach will be described more comprehensively; in section 4, further details regarding our implementation will be given, and in section 5, the obtained results will be presented.

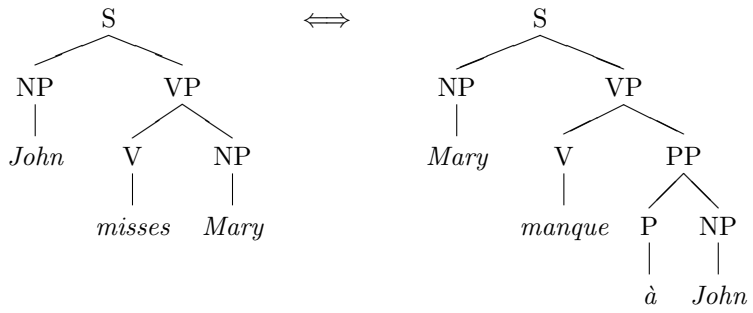
2 Theoretical framework

2.1 Synchronous grammars

In [Ch], the concept of a synchronous grammar is introduced, as a tool to easily extend the notion of a (P)CFG into the domains of translation and transformation. The main idea, here, is that instead of single CFG-rules with a left and right hand side, every rule now has a pair of right-hand sides – one corresponding to the source language, and the other corresponding to the target language. Moreover, there has to be a bijective function between the nonterminal symbols on both right-hand sides.

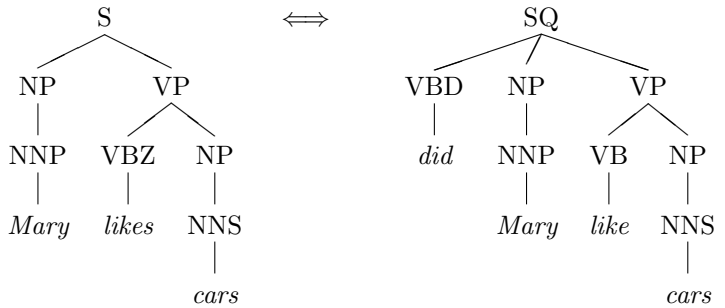
Parsing and translating using synchronous grammars is just as easy as it is for regular PCFGs: to translate, we simply parse using the right-hand sides corresponding to the source language. By reading off the right-hand sides corresponding to the target language, of the rules used in the derivation, the translation is immediately obtained.

Some problems with synchronous grammars are also mentioned in [Ch]. One of these problems is that, in reality, tree structures often do not coincide between the original language and the translated language. For example, a common analysis of



is problematic under the original framework of synchronous grammars, because of the mismatch in the tree structures.

Similarly, when looking at transformations between declarative and interrogative sentences, and basing ourselves on standard analyses of the Stanford parser (which are made using the standards described for the Penn treebank), we encounter the following pairs of trees:



Again, we here are faced with a pair of sentences, where the tree structures are rather different from each other.

As a solution to the problem of a mismatch, the notion of flattening is named. In this case, several CFG-rules are collapsed into a single rule, so that

the mismatch between the trees is resolved.

2.2 DOP for translations

In [Po] and [Po2], synchronous grammars are extended from plain PCFGs to the DOP framework: instead of looking at rules with two right-hand sides and a corresponding set of non-terminals, we now consider pairs subtrees with identical top nodes, and corresponding non-terminal leaf nodes.

These linked subtrees are then combined into a a forest of linked subtrees, which can then be used as a grammar for translations and transformations. Just like in the case of synchronous PCFGs, we can simply parse new sentences using the source 'side' of the grammar, and derive the corresponding target side immediately.

An important contrast with the earlier synchronous grammars for PCFGs, is that in the framework of [Po], we are not ultimately dependent on strong similarities between the tree structures on both sides. This would make translations and transformations, such as in the examples given in section 2.1, possible.

Of course, it should be noted that, in this framework, it is unlikely that *all* subtrees on the source side are linked to subtrees on the target side: especially in the (far from unlikely) case where the total number of subtree differs, this has to be trivially false.

A problem with the DOT model, as described in [Po], consists of the fact that the complexity of parsing new sentences can be quite huge, as the number of subtrees can be (at worst) exponential in the number of nodes. For standard DOP-based parsing the Goodman reduction, which will be described in the next section, has been found to resolve such issues: however, because we do not have a complete set of subtrees available to use, it is impossible to use this reduction right away. This question of finding a suitable reduction for the DOT model as described in [Po] and [Po2], will be addressed at a later stage in this report.

2.3 Goodman reduction

A significant problem with DOP parsing, is the fact that the number of subtrees that have to be included is exponential in the size of the tree. In [Go], a solution to this problem is presented, that is able to yield equivalent parse probabilities (but not equivalent derivation probabilities) to regular DOP parsing, while limiting the number of rules to a number that is linear in the number of tree nodes.

Here, the subtrees are replaced by a set of PCFG-rules, where nodes can be either 'addressed' or 'non-addressed': in every tree, every node has a unique address, and the addressed nodes only match with that specific address, whereas the non-addressed nodes match with any node of the type.

It is proven in [Go] that the PCFG resulting from the Goodman reduction leads to equivalent parse probabilities as the underlying STSG:

Theorem 2.1. *This construction [the Goodman-reduction] produces PCFG trees homomorphic to the STSG trees with equal probability.*

3 Approach

3.1 A reduction for synchronous DOP

We are now faced with the situation, that we cannot simply apply the Goodman reduction, reducing subtree pairs to PCFG rules, because of the reason that not every subtree will always be a part of a subtree pair. In particular, the PCFG-rules themselves, the 'building blocks' of the Goodman reduction, may not be a part of a subtree pair. This leads us to wonder whether it is possible to, in some way or another, still perform a Goodman-like reduction on the forest of linked subtrees.

It turns out that, indeed, a reduction is possible over the forest of linked subtrees. For this, consider the notion of a *minimal linked subtree pair*:

Definition 3.1. For a pair of linked trees $\langle T_1, T_2 \rangle$, a *minimal linked subtree pair* is a linked subtree pair $\langle t_1, t_2 \rangle$, s.t. $t_1 \subseteq T_1$, $t_2 \subseteq T_2$, such that there is no other linked pair $\langle s_1, s_2 \rangle$ where $s_1 \subsetneq t_1$ and $s_2 \subsetneq t_2$.

Theorem 3.2. *Every pair of linked trees can be reconstructed from the minimal linked subtree pairs derived from it.*

Proof. Assume that there is a pair of linked trees $\langle T_1, T_2 \rangle$ for which this is not possible, given a certain forest of linked subtree pairs. Then there has to be some linked pair of subtrees $\langle t_1, t_2 \rangle$ for which:

1. It is not possible to construct $\langle t_1, t_2 \rangle$ from minimal linked subtree pairs.
2. All pairs of subtrees $\langle s_1, s_2 \rangle \subsetneq \langle t_1, t_2 \rangle$ can be constructed from minimal subtree linked pairs.

It is impossible that this pair of subtrees is itself minimal: if it were, it could trivially be constructed from linked minimal subtree pairs. So, for this case to hold, the linked subtree pair $\langle t_1, t_2 \rangle$ must have a proper subset $\langle s_1, s_2 \rangle$.

... (incomplete but I know how to finish) □

In other words: just like the PCFG rules can be used as the 'building blocks' for the Goodman reduction, the minimal linked subtree pairs as defined above can be used as the 'building blocks' for a similar reduction. It should be noted that, for all means and purposes, such minimal subtrees can be worked with, as if they were simple CFG-rules, that just happen to have an additional internal structure (which is ignored by the parser).

Like in the Goodman-reduction, we replace every elementary tree (i.e. every minimal linked subtree pair) with $2^{(k+1)}$ new rules, where k is the number of nonterminals occurring as leaf nodes (which has to be equal on the left and right hand sides). For the root node, and for every nonterminal node, either an 'addressed' version or a 'non-addressed' version can occur.

Theorem 3.3. *A 'Goodman-like' reduction based on minimal linked subtree pairs leads to equivalent parse probabilities (but not equivalent derivation probabilities) as the original synchronous DOP model.*

4 Implementation

5 Results

6 Conclusions

References

[Ch] Chiang, David, *An Introduction to Synchronous Grammars*

[Po] Poutsma, Arjen, *Data-Oriented Translation*, presented at COLING 2000

[Po2] Poutsma, Arjen, *Data-Oriented Translation*, master's thesis, 2000

[Go] Goodman, Joshua, *Efficient Parsing of DOP with PCFG-reductions* in *Data Oriented Parsing* (2002), Bod, Rens; Sima'an, Khalil; and Scha, Remko, eds.