Top-Down Transfer in Example-based MT

Vincent Vandeghinste
Centrum voor Computerlinguïstiek
K.U.Leuven
Belgium
vincent@ccl.kuleuven.be

Scott Martens
Centrum voor Computerlinguïstiek
K.U.Leuven
Belgium
scott.martens@ccl.kuleuven.be

Abstract

In this paper we describe and evaluate a top-down transfer component of a hybrid example-based machine translation system with an architecture similar to that of transfer MT systems, but with automatically derived transfer-rules and dictionary entries based on a parallel treebank. The tests were applied on the translation pair Dutch to English. Evaluation and error analysis have shown that the top-down transfer process has a number of shortcomings on which we wish to report and which we will try to solve in future work by applying bottom-up transfer.

1 Introduction

In this paper[1] we present a top-down transfer component of an *example-based transfer system*. Although there seems to be no clear definition of example-based machine translation, most people would agree that an EBMT system requires a parallel corpus containing example translations. [19] mention three main directions for extending and augmenting the original EBMT ideas in [14]. One is to augment the example database with linguistic annotations and perform the same type of linguistic analysis on the input before matching it with the examples. A second extension is through the use of templates rather than sentences in the example database. In these templates, some parts of the input sentence have been replaced by variables, annotated with linguistic information. A third extension is to use a combination of examples that match fragments of the sentence and recombine their translations into a target language sentence.

Our system uses a parallel corpus as an example database and extends the original EBMT approach along all three of these lines. We annotate the examples using a deep syntactic parse that marks both dependency and constituent structure. We use abstractions over lexical items to deduce a set of templates, which we call transfer rules, and we combine unlexicalized and lexicalized transfer rules - those containing just syntactic categories and relations and those containing both syntactic information and words or roots - into a target language structure (a bag of bags) from which we generate a target language sentence.

The system can also be considered a rule-based transfer system. It conforms to the general architecture of a rule-based system: the source sentence is parsed, the transfer rules and dictionary are applied to generate target language parse trees, and a surface form of the sentence is generated from the target parse tree. The rules and dictionary are automatically induced from a parallel corpus.

The approach we take is in many aspects similar to Data-Oriented Translation (DOT) ([16, 7]), but while DOT uses the specific formalism of Data-Oriented Parse Trees ([3]), we use already existing monolingual parsers that are either rule-based with a stochastic disambiguation component or stochastic parsers trained on a manually parsed or corrected treebank. The DOT approach only uses small corpora and a limited domain, whereas we use large corpora and a general domain (*news*). The extraction of transfer rules resembles the work of [6] and [5], but we map from source tree to target tree, whereas they map from source string to target tree. It is also similar to the work of [2], but they limit their transfer rules to depth 2 to produce synchronous context-free grammars, first introduced by [1].

[1]The research presented in this paper was done in the PaCo-MT project, sponsored by the STEVIN-programme and by the AMASS++ project sponsored by IWT - Vlaanderen.

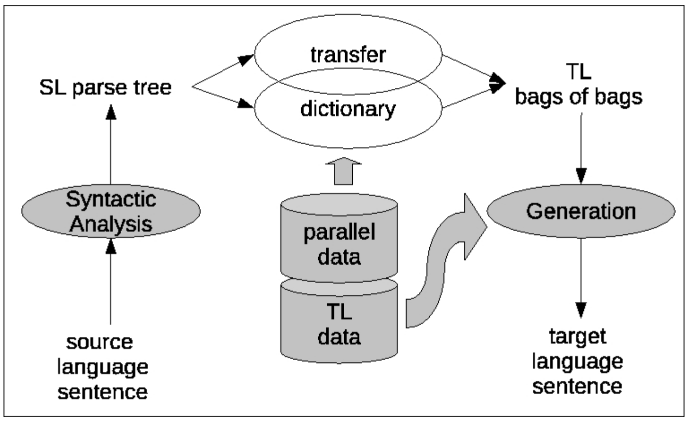

Figure 1: Architecture of our system

At the moment the system is not available in open source as it is too immature and will still undergo many changes. All components and tools used are either open source or freely available, and we are open to sharing the tools we built with other researchers.

Section 2 describes our MT system, section 3 describes an evaluation of our system, and section 4 concludes the paper.

2 System description

Figure 1 shows the general architecture of our system. The source language for which we tested our system is Dutch. We use the wide-coverage dependency parser Alpino [20] which is freely available, and which generates a deep syntactic analysis of the sentence, containing both the dependency structure as well as the phrase structure. The transfer and dictionary component are described in section 2.1 and the target language generation component is described in 2.2.

2.1 From Source to Target Language

In order to transfer the source language tree into the target language tree, we must identify examples in the example database that match some or all of an input source language tree. Section 2.1.1 describes how the example trees are preprocessed for matching, and section 2.1.2 describes the matching process itself.

2.1.1 Preprocessing the parallel data

The parallel tree data we used is derived from the Dutch and English sections of the Europarl corpus [11]. Dutch was parsed using the Alpino parser ([20]) and English was parsed using the factored models of the Stanford parser[9, 10].[2]

Our system requires sentence and word alignments, but also sub-sentential alignments on intermediate (non-root and non-leaf) nodes. How the sub-sentential alignments were generated is described in [18]. From the alignments, we derive one uniform set of both lexicalized and unlexicalized rules.

[2]Thanks to Jörg Tiedemann for parsing these corpora.

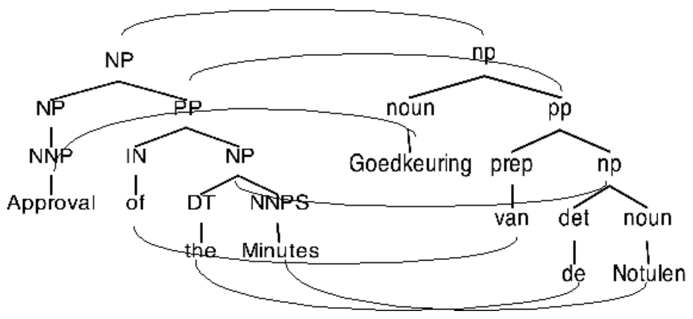

Figure 2: An example alignment containing POS and phrase category information.

For each pair of aligned nodes, we extracted the subtrees rooted at those nodes, in both source and target language, down to a maximum tree depth of 3.[3] Figure 2 shows an example alignment and Figure 3 shows the set of transfer rules extracted from this alignment. Transfer rules are constructed for *each* node, including nodes that are already included in other transfer rules. For example, in Figure 2, in addition to the transfer rule for the entire `NP` *"Goedkeuring van de Notulen"* $\leftrightarrow$ *"Approval of the Minutes"*, the `NP` containing the words *"the Minutes"* is aligned with the `np` containing the words *"de Notulen"* as a separate alignment from which transfer rules are also extracted. We make abstraction over the sequential ordering of the children of a parent, representing all different permutations under the same transfer rule, modeling only structural transfer. Ordering is solved in target language modeling (section 2.2), the transfer rules only capture hierarchical relations.

We extract *unlexicalized* rules by ignoring the *tokens*. We proceed top-down from the root of the source and target alignments, extracting the subtrees of up to depth 3 from each root alignment. The result is up to three source language subtrees and three target language subtrees. We then consider the leaf nodes of each subtree, and in each case where all the leaves of the source subtree align with all the leaves of a target language subtree, we accept that tree pair as a mapping from source to target languages - in effect a transfer rule. A transfer rule is defined by one source language tree, one target language tree, and an alignment table of each leaf node in the source tree to a target language tree leaf node. When a mapping between two subtrees does not include a complete alignment of all the leaves, but is otherwise identical to a rule that does have a complete leaf alignment, its frequency and weight are added to that transfer rule. The procedure for constructing *lexicalized* rules is identical, but we add the words to the leaves of both sides of the transfer rules in those cases where it concerns terminal nodes in the alignment.

The extraction of transfer rules bears many resemblances to the work of [2]: both induce lexical and structural rules for building syntactic translation models, respecting the constraints that are imposed by the underlying word alignment and syntax. [2] generates a synchronous context-free grammar based on the alignments, whereas we generate transfer rules with a maximum depth > 2 on both the source and the target side.

```
NP:[NP:[NNP^1] PP:[IN^2 NP^3]] <=> np:[noun^1 pp:[prep^2 np^3]]
NP^1 => np^1
NP:[NP:[NNP^1:[Approval] PP:[IN^2:[of] NP^3]] <=> np:[noun^1:[Goedkeuring] pp:[prep^2:[van] np]]
```

Figure 3: Example of extracted transfer rules with full leaf node alignment. The children of each node on both sides are sorted in alphabetical order, not representing sequential information.

[3]We use a maximum depth of 3 to prevent useless explosion of the ruleset which would not result in many more matches.

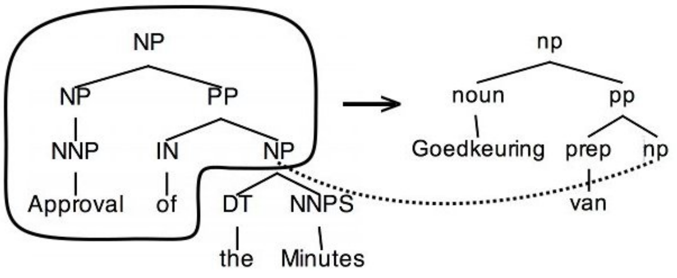

Figure 4: An example tree - half translated.

2.1.2 Matching the Source Language Tree with the Example Database

The result of the preprocessing described above is a set of transfer rules such as the ones in Figure 3 which map a portion of a parse tree in the source language to a portion of a parse tree in the target language. They are stored in a database along with, for each transfer rule, the sum of the confidence weights derived from the alignments, the node-to-node alignments of the leaf nodes in the source and target, the frequencies of the source language tree and the target language tree, and the number of times the two trees are aligned. Each transfer rule is weighted as the product of the average confidence weight[4] and the Dice coefficient of source and target.[5] The resulting weight is further biased to favour deeper trees on both the source and target side and to favour transfers that occur frequently in absolute terms over those that appear only a few times.

Transfer is a breadth-first search with a fixed beam size, starting with the root of the source language sentence, that tries to maximize the product of the weights of the rules used to generate the target language parse tree. Given a node in a source language parse, the transfer engine tries to find rules whose source side matches the top of the subtree rooted at that node, down to as great a depth as possible, up to three levels. Transfer proceeds by starting at the root and transferring the top of the parse tree, then, using the alignment information, determining which subtrees lower in the source sentence attach to which leaves in the partial target language parse tree, as shown in Figure 4.[6]

In Figure 4, the phrase *“Goedkeuring van de Notulen”* is translated by finding a transfer rule with the top three levels of the source parse on the left hand side. On the right is the partial parse tree corresponding to *“Approval of NP”*. The alignment information tells us that the NP in the target side of the rule aligns with the source side NP *“de Notulen”*. In the next iteration of the transfer engine, a rule is found with *“de Notulen”* on the left-side, if any exists, and then inserts the resulting partial parse underneath the corresponding target NP, as in Figure 5.

This process proceeds top-down and from left to right, until all of the source tree has been translated. This procedure is very similar to the generation schemes used in Tree Adjoining Grammar [8].

In many cases, the largest transfer rule translates only a single source node to a single target node (mapping the source labels onto the target labels). This requires special handling, since it means that no rule transfers that node and its children - the rule only specifies, for example, that a source language constituent like NP translates to a target language NP. The children of that node are then translated individually, and placed underneath that node in the target language. Furthermore, when the transfer system attempts to translate an unknown word, it finds a rule to translate its part-of-speech or its frame,

[4]The sum of the individual confidence weights divided by the number of times the source and target are aligned.

[5]The number of times they overlap divided by the total number of sentences where one or the other appears.

[6]The transfer rules are completely symmetrical between source and target language, so the translation direction is of no importance here.

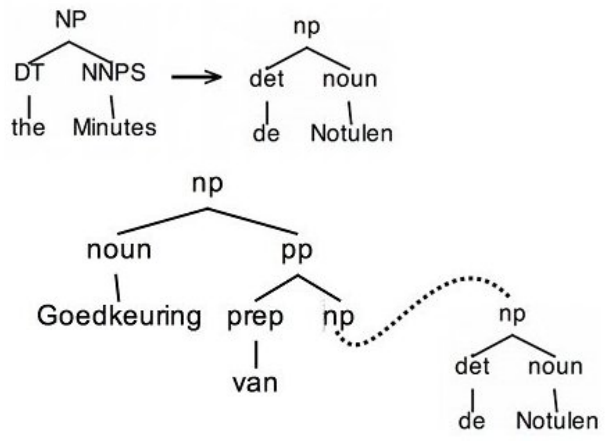

Figure 5: The 2nd step in translation.

depending on the experimental condition (cf. section 3), and then copies the word verbatim.

2.2 Target Language Generation

Target language generation is the component that converts the target language trees (bags of bags) into a target language sentence. The target language parse tree generated by the transfer component described in section 2.1 has no ordering information. For each parent node, the actual order of the daughters is determined by the target language model. It determines word and constituent ordering and can play an important role in lexical selection.

The target language model used here is trained on the target language parts of Europarl, but it can be trained on large monolingual treebanks (like a parsed British National Corpus) as well. From the parse trees in the treebank, we extract the context-free rewrite rules, using the phrase category labels on the left-hand side, but we do not necessarily have to restrict ourselves to the parts-of-speech, the phrase category labels or the tokens on the right-hand side as we distinguish different abstraction levels. For English, we distinguish four abstraction levels, as shown in Figure 6:

1. the *dependency relations* (`Rels`),
2. the *syntactic categories* for non-terminal nodes and the *parts-of-speech* for terminal nodes (`Cat/Pos`),
3. the dependency relations together with the syntactic categories/parts of speech (`Cat+Rel`),
4. the dependency relations together with the syntactic categories/parts of speech as well as the head token information (`Cat+Rel+Token`).

```
Rels            NP : det amod hd
Cat/Pos         NP : AT0 JJ NN1
Cat+Rel         NP : AT0|det JJ|amod NN1|hd
Cat+Rel+Token   NP : AT0|det|a JJ|amod|legal NN1|hd|reason
```

Figure 6: Context free rewrite rules for different abstraction levels

For each node in the target language tree, we check if we find a rewrite rule at the least abstract level (`Cat+Rel+Token`), cascading down to the most abstract level until we find a solution, estimating

Table 1: Evaluation Results

Abstraction	Beam	Tok/Lem	BLEU	NIST	WER	CER	PER	TER
cat/pos	10	lemma	9.70	4.54	82.72	62.25	62.20	75.52
cat/pos	100	lemma	10.17	4.59	82.06	61.68	61.85	75.16
cat/frame	10	lemma	9.82	4.68	82.63	62.21	61.35	75.28
cat/frame	100	lemma	9.88	4.63	82.42	62.12	61.76	75.30
cat/pos	10	token	12.50	5.30	77.55	62.12	54.99	71.84
cat/pos	100	token	13.18	5.43	76.53	**61.36**	54.48	70.92
cat/frame	10	token	**13.53**	**5.70**	76.20	61.91	**52.39**	**70.36**
cat/frame	100	token	13.52	5.61	**76.10**	61.87	53.12	70.76

the probability of different orderings of the daughters of the node under investigation, selecting the most probable, by looking at the frequency of occurrence of the patterns in the training data.

A detailed account of this approach for Dutch as a target language is given in [21], as well as an experiment showing that this approach, at least for Dutch, outperforms a standard trigram target language model.

3 System evaluation

We evaluated our system, using well-known automated MT metrics, like BLEU ([15]), NIST ([4]), and TER ([17]), as well as WER (word error rate), PER (position independent word error rate), and CER (character error rate). We have used a test set of 500 Dutch sentences, with two reference translations for each sentence. To give an idea about the difficulty of the test set, it scored 29.96 BLEU on Moses ([12]) trained on the same sentences of Europarl as used in our system[7] and 38.82 BLEU on Google translate.

We trained the translation model on the parsed versions of Europarl ([11]), and the target language model only on the English side of Europarl.

We had three independent variables:

1. the abstraction level used: frames + categories (*cat/frame*) or part-of-speech + categories (*cat/pos*),
2. the beam width for the transfer component (10 or 100), and
3. whether we use tokens or lemmas in our transfer rules.

Table 1 shows the results of these conditions.

All results are worse than what we expected, and we draw the conclusion that top-down processing of the transfer rules is not the way to go. Nevertheless we wanted to report this so others can avoid it.

With respect to the beam size, we expected better results with a wider beam, but this is only the case combined with cat/pos. When using cat/frame, the beam size does not significantly change the scores.

With respect to using tokens or lemmas in the tranfer rules, we expected a wider coverage with the lemmas, but an extra processing step is required to generate the tokens from the lemmas and part-of-speech tags, which can introduce additional errors. Combining cat/pos with lemmas we did not expect good results, as cat/pos in Dutch is clearly not fine-grained enough to allow synthesis of the desired token form. We found no conditions in which lemmas resulted in better scores than tokens.

[7]Some sentences timed out in parsing.

We expected the cat/frame information to be too fine-grained to have a good coverage as compared to the cat/pos information, but results show a clear improvement over all scores when using the frames instead of the pos.

4 Conclusions and Future

The transfer process as described in section 2.1 proved to have a number of shortcomings. Structural matches are relatively easy to find at the lowest levels. Translating individual NPs and simple VPs by this method was not difficult. However, the richness of structures closer to the root of each parse made sparsity of data a major problem, and without sufficient detail about the composition of each phrase, the target language model used to transform parses into sentences was much less accurate. Sparsity also produced some surprising and at first illogical translations due to differences in parsing style between the English and Dutch parses. This model of transfer works best when the parse trees in the source and target language are not just aligned, but where the structures themselves are very similar. Consider, for example, the sentence in English *"He passed the first time."* A reasonable Dutch translation is *"Hij is van de eerste keer geslaagd."* Using a top-down approach to translating this sentence, an English adverbial phrase (or perhaps a noun phrase depending on the underlying formalism) becomes a Dutch prepositional phrase. Using this rule will likely be misleading for many other sentences, and will force the transfer engine to select, among available translations, ones that have the expected phrase category and part of speech labels. Given the structural dissimilarities between parsing strategies in the source and target languages, the imperfect performance of alignment procedures, and the sparsity of the structures in corpora, the errors accumulate very quickly and are propagated downwards, forcing the transfer system to select low probability and flatly wrong translations. This problem becomes more acute as the node and edge labels in the parse tree become more detailed, exacerbating the data sparsity problem even further. We believe this to account for the relatively poor performance of the category and frame test condition in comparison to the category and part-of-speech condition.

Possible remedies include a shift to binary trees for all parses (cf [22]), which may reduce the data sparsity problem because if each node can have no more than two children, the number of feasible structures of any particular size is dramatically reduced. Ensuring that, in so far as is reasonable, the source and target language parse trees are comparable will also likely improve performance. Different theories of syntax yield different approaches to parsing, which in turn yield radically different kinds of parse trees. This lack of uniformity also increases the number of structural mismatches.

One strategy we are actively pursuing is reversing the direction of transfer and performing translations from the bottom up. It is relatively easy to identify every node in a source language parse tree that exactly matches one or more in a bilingual aligned corpus. This is the strategy used by sub-sentential translation memories. We could then use information about the structure of the source sentence, weighted transfer rules extracted from the bilingual aligned corpus, and treebank-derived information about the target language to construct the translation parse tree in much the same way as a monolingual parser. This is much like the synchronous context-free grammar strategy of [2], but integrated with a sub-sentential translation memory.

Given the structural richness of any sophisticated written language and the relative sparsity of treebank structured data, it is undoubtedly necessary to consider strategies that involve imperfect structural matches between a source language sentence and a bilingual aligned treebank, or a ruleset derived from one. We are actively pursuing approaches along this line which may also deliver benefits in handling structural mismatches between the source and target due to different parsing methods. This requires more sophisticated indexing strategies like those described in [13].

References

[1] Alfred V. Aho and Jeffrey D. Ullman. Syntax directed translations and the pushdown assembler. *J. Comput. Syst. Sci.*, 3(1):37–56, 1969.

[2] Vamshi Ambati, Alon Lavie, and Jaime Carbonell. Extraction of Syntactic Translation Models from Parallel Data using Syntax from Source and Target Languages. In *Proceedings of MT Summit XII*, 2009.

[3] R. Bod. A Computational Model of Language Performance: Data-Oriented Parsing. In C. Boîtet, editor, *Proceedings of the fifteenth International Conference on Computational Linguistics (COLING'92)*, pages 855–859, 1992.

[4] G. Doddington. Automatic Evaluation of Machine Translation Quality using N-gram Co-occurrence Statistics. In *Proceedings of the Second Human Language Technology Conference (HLT)*, pages 138–145, 2002.

[5] Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of 44th ACL*, 2006.

[6] Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What's in a translation rule? In *Proceedings of HLT-NAACL04*, 2004.

[7] M. Hearne. Data-Oriented Models of Parsing and Translation. Master's thesis, Dublin City University, 2005.

[8] Aravind K. Joshi, L.S. Levy, and M. Takahashi. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1), 1975.

[9] D. Klein and C. Manning. Accurate Unlexicalized Parsing. In *Proceedings of ACL-2003*, pages 423–430, 2003.

[10] D. Klein and C. Manning. Fast Exact Inference with a Factored Model for Natural Language Parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pages 3–10, Cambridge, MA, 2003. MIT Press.

[11] P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of MT Summit X*, 2005.

[12] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL*, Prague, 2007.

[13] Scott Martens. Quantitative analysis of treebanks using frequent subtree mining methods. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing (TextGraphs-4)*, pages 84–92, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[14] M Nagao. A framework of a mechanical translation btween japanese and english by analogy principle. In A. Elithorn and Banerji R., editors, *Artifical and Human Intelligence*, pages 173–180. 1984.

[15] K. Papineni, S. Roukos, T. Ward, and W. Zhu. BLEU: a method for automatic evaluation of Machine Translation. In *Proceedings of ACL-2002*, pages 311–318, 2002.

[16] A. Poutsma. Data-Oriented Translation. In *Ninth Conference of Computational Linguistics in the Netherlands*, 1998.

[17] M. Snover, B. Dorr, R. Schwartz, L Micciula, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of AMTA-2006*, pages 223–231, 2006.

[18] J. Tiedemann and G. Kotzé. A Discriminative Approach to Tree Alignment. In *Proceedings of RANLP-2009*, 2009.

[19] D. Turcato and F. Popowich. What is Example-Based Machine Translation. In Michael Carl and Andy Way, editors, *Recent Advances in Example-based Machine Translation*, chapter 2, pages 59–81. Kluwer, 2003.

[20] G. van Noord. **At** **L**ast **P**arsing **I**s **N**ow **O**perational. In *In Proceedings of TALN 2006 Verbum Ex Machina*, pages 20–42, Leuven, 2006.

[21] V. Vandeghinste. Tree-based Target Language Modeling. In *Proceedings of EAMT-2009*, pages 152–159, Barcelona. Spain, 2009.

[22] Wei Wang, Kevin Knight, and Daniel Marcu. Binarizing Syntax Trees to Improve Syntax-Based Machine Translation Accuracy. In *Proceedings of EMNLP - CoNLL*, 2007.