Stochastic Parse Tree Selection for an Existing RBMT System

Christian Federmann
DFKI GmbH
Language Technology Lab
Saarbrücken, Germany
cfedermann@dfki.de

Sabine Hunsicker
DFKI GmbH
Language Technology Lab
Saarbrücken, Germany
sabine.hunsicker@dfki.de

Abstract
In this paper we describe our hybrid machine translation system with which we participated in the WMT11 shared translation task for the English—German language pair. Our system was able to outperform its RBMT baseline and turned out to be the best-scored participating system in the manual evaluation. To achieve this, we extended an existing, rule-based MT system with a module for stochastic selection of analysis parse trees that allowed to better cope with parsing errors during the system’s analysis phase. Due to the integration into the analysis phase of the RBMT engine, we are able to preserve the benefits of a rule-based translation system such as proper generation of target language text. Additionally, we used a statistical tool for terminology extraction to improve the lexicon of the RBMT system. We report results from both automated metrics and human evaluation efforts, including examples which show how the proposed approach can improve machine translation quality.

1 Introduction
Rule-based machine translation (RBMT) systems that employ a transfer-based translation approach, highly depend on the quality of their analysis phase as it provides the basis for its later processing phases, namely transfer and generation. Any parse failures encountered in the initial analysis phase will proliferate and cause further errors in the following phases. Very often, bad translation results can be traced back to incorrect analysis trees that have been computed for the respective input sentences. Henceforth, any improvements that can be achieved for the analysis phase of a given RBMT system directly lead to improved translation output which makes this an interesting topic in the context of hybrid MT.

In this paper we present a study how the rule-based analysis phase of a commercial RBMT system can be supplemented by a stochastic parser. The system under investigation is the rule-based engine Lucy LT. This software uses a sophisticated RBMT transfer approach with a long research history; it is explained in more detail in (Alonso and Thurmai,
2003).

The output of its analysis phase is a parse forest containing a small number of tree structures. For our hybrid system we investigated if the existing rule base of the Lucy LT system chooses the best tree from the analysis forest and how the selection of this best tree out of the set of candidates can be improved by adding stochastic knowledge to the rule-based system.

The remainder of this paper is structured in the following way: in Section 2 we first describe the transfer-based architecture of the rule-based Lucy LT engine, giving special focus to its analysis phase which we are trying to optimize. Afterwards, we provide details on the implementation of the stochastic selection component, the so-called “tree selector” which allows to integrate knowledge from a stochastic parser into the analysis phase of the rule-based system. Section 3 reports on the results of both automated metrics and manual evaluation efforts, including examples which show how the proposed approach has improved or degraded MT quality. Finally, we conclude and provide an outlook on future work in this area.
2 System Architecture

2.1 Lucy LT Architecture

The Lucy LT engine is a renowned RMBT system which follows a “classical”, transfer-based machine translation approach. The system first analyses the given source sentence creating a forest of several analysis parse trees. One of these parse trees is then selected (as “best” analysis) and transformed in the transfer phase into a tree structure from which the target text (i.e. the translation) can be generated.

It is clear that any errors that occur during the initial analysis phase proliferate and cause negative side effects on the outcome of the final translation result. As the analysis phase is thus of very special importance, we have investigated it in more detail. The Lucy LT analysis consists of several phases:

1. The input is tokenised with regards to the system’s source language lexicon.
2. The resulting tokens undergo a morphological analysis, which is able to identify possible combinations of allomorphs for a token.
3. This leads to a chart which forms the basis for the actual parsing, using a head-driven strategy\(^1\). Special handling is performed for the analysis of multi-word expressions and also for verbal framing.

At the end of the analysis, there is an extra phase named phrasal analysis which is called whenever the grammar was not able to construct a legal constituent from all the elements of the input. This happens in several different scenarios:

- The input is ungrammatical according to the LT analysis grammar.
- The category of the derived constituent is not one of the allowed categories.
- A grammatical phenomenon in the source sentence is not covered.
- There are missing lexical entries for the input sentence.

During the phrasal analysis, the LT engine collects all partial trees and greedily constructs an overall interpretation of the chart. Based on our findings from many experiments with the Lucy LT engine, phrasal analyses are performed for more than 40% of the sentences from our test sets and very often result in bad translations.

During the phrasal analysis, the LT engine collects all partial trees and greedily constructs an overall interpretation of the chart. Based on our findings from many experiments with the Lucy LT engine, phrasal analyses are performed for more than 40% of the sentences from our test sets and very often result in bad translations.

Each resulting analysis parse tree, independent of whether it is a grammatical or a result from the phrasal analysis, is also assigned an integer score by the grammar. The tree with the highest score is then handed over to the transfer phase, thus pre-defining the final translation output.

2.2 The “Tree Selector”

An initial evaluation of the translation quality based on the tree selection of the analysis phase showed that there is potential for improvement. The integer score assigned by the analysis grammar provides a
good indication of which trees lead to good transla-
tions, as is depicted in Table 1. Still, in many cases
an alternative tree would have lead to a better trans-
lation.

As additional feature, we chose to use the tree
eid distance of each analysis candidate to a stochas-
tic parse tree. An advantage of stochastic parsing
lies in the fact that parsers from this class can deal
very well even with ungrammatical or unknown out-
put, which we have seen is problematic for a rule-
based parser. We decided to make use of the Stanford
Parser as described in (Klein and Manning, 2003),
which uses an unlexicalised probabilistic context-
free grammar that was trained on the Penn Tree-
bank\textsuperscript{2}. We parse the original source sentence with
this PCFG grammar to get a stochastic parse tree that
can be compared to the trees from the Lucy analysis
forest.

In our experiments, we compare the stochastic
parse tree with the alternatives given by Lucy LT.
Tree comparison is implemented based on the Tree
Edit Distance, as originally defined in (Zhang and
Shasha, 1989). In analogy to the Word Edit or Lev-
enshtein Distance, the distance between two trees
is the number of editing actions that are required to
transform the first tree into the second tree. The Tree
Edit Distance knows three actions:

- Insertion
- Deletion
- Renaming (substitution in Levenshtein Distance)

Since the Lucy LT engine uses its own tag set,
a mapping between this proprietary and the Penn
Treebank tag set was created. Our implementation,
called “Tree Selector” uses a normalised version of
the Tree Edit Distance to estimate the quality of the
trees from the Lucy analysis forest, possibly over-
riding the analysis decision taken by the unmodified
RBMT engine. The integration of the Tree Selector
has been possible by using an adapted version of the
rule-based MT system which allowed to communi-
cate the selection result from our external process to
the Lucy LT kernel which would then load the re-
spective parse tree for all further processing steps.

2.3 LiSTEX Terminology Extraction

The LiSTEX extension of the Lucy RBMT engine
allows to improve the system’s lexicon; the approach
is described in more detail in (Federmann et al.,
2011). To extend the lexicon, terminology lists are
extracted from parallel corpora. These lists are then
enriched with linguistic information, such as part-of-
speech tag, internal structure of multi-word expres-

\begin{table}
\begin{tabular}{|l|c|}
\hline
Best Analysis Tree & Percentage \\
\hline
Default (id=1) & 42 (61.76\%) \\
Alternative (id=2-7) & 26 (38.24\%) \\
\hline
\end{tabular}
\caption{Evaluation of Analysis Forests}
\end{table}

353
sions and frequency. For English and German, about 26,000 terms were imported using this procedure.

2.4 Named Entity Handling

Named entities are often handled incorrectly and wrongly translated, such as George Bush → George Busch. To reduce the frequency of such errors, we added a pre- and post-processing modules to deal with named entities. Before translation, the input text is scanned for named entities. We use both HeiNER (Wolodja Wentland and Hartung (2008)) and the OpenNLP toolkit\(^3\). HeiNER is a dictionary containing named entities extracted from Wikipedia. This provides us with a wide range of well-translated entities. To increase the coverage, we also use the named entity recogniser in OpenNLP. These entities have to be translated using the RBMT engine. We save the named entity translations and insert placeholders for all NEs. The modified text is translated using the hybrid set-up described above. After the translation is finished, the placeholders are replaced by their respective translations.

3 Evaluation

3.1 Shared Task Setup

For the WMT11 shared translation task, we submitted three different runs of our hybrid MT system:

1. Hybrid Transfer (without the Tree Selector, but with the extended lexicon)
2. Full Hybrid (with both the Tree Selector and the extended lexicon)
3. Full Hybrid+Named Entities (full hybrid and named entity handling)

Our primary submission was run #3. All three runs were evaluated using BLEU (Papineni et al. (2001)) and TER (Snover et al. (2006)). The results from these automated metrics are reported in Table 2.

<table>
<thead>
<tr>
<th>System</th>
<th>BLEU</th>
<th>TER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid Transfer</td>
<td>13.4</td>
<td>0.792</td>
</tr>
<tr>
<td>Full Hybrid</td>
<td>13.1</td>
<td>0.796</td>
</tr>
<tr>
<td>Full Hybrid+Named Entities</td>
<td>12.8</td>
<td>0.800</td>
</tr>
</tbody>
</table>

Table 3: Manual evaluation scores for WMT11

<table>
<thead>
<tr>
<th>System</th>
<th>Normalized Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Hybrid+Named Entities</td>
<td>0.6805</td>
</tr>
<tr>
<td>Original Lucy</td>
<td>0.6599</td>
</tr>
</tbody>
</table>

3.2 Error Analysis

The selection process following the decision factors as explained in Section 2.2 may fail due to wrong assumptions in two areas:

1. The tree with the lowest distance does not result in the best translation.
2. There are several trees associated with the lowest distance, but the tree with the highest score does not result in the best translation.

To calculate the error rate of the Tree Selector, we ran experiments on the test set of the WMT10 shared task and evaluated a sample of 100 sentences with regards to translation quality. To do so, we created all seven possible translations for each of the phrasal analyses and checked whether the Tree Selector returned a tree that led to exactly this translation. In case it did not, we investigated the reasons for this. Sentences for which all trees created the same translation were skipped. This sample contains both examples in which the translation changed and in which the translation stayed the same.

Table 4 shows the error rate of the Tree Selector while Table 5 contains the error analysis. As one can see, the optimal tree was chosen for 56% of the sentences. We also see that the minimal tree edit distance seems to be a good feature to use for comparisons, as it holds for 71% of the trees, including those examples where the best tree was not scored highest by the LT engine. This also means that additional features for choosing the tree out of the group of trees with the minimal edit distance are required. Even for the 29% of sentences, in which the optimal tree was not chosen, little quality was lost: in 75.86% of those cases, the translations didn’t change.

Table 3 shows that we were able to outperform the original Lucy version. Furthermore, it turned out that our hybrid system was the best-scoring system from all shared task participants.
Best Translation Returned 56%
Other Translation Returned 44%
Best Tree has Minimal Edit Distance 71%
Best Tree has Higher Distance 29%

Table 4: Error Rate of the Tree Selector

at all (obviously the trees resulted in equal translation output). In the remaining cases the translations were divided evenly between slight degradations and and equal quality.

<table>
<thead>
<tr>
<th>Other Translation: Selected Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree 1 (Default) 31</td>
</tr>
<tr>
<td>Tree 2-7 (Alternatives) 13</td>
</tr>
</tbody>
</table>

Table 5: Evaluation of Tree Selector Errors

In the cases when the best tree was not chosen, the first tree (which is the default tree) was selected in 70.45% . This is due to a combinations of robustness factors that are implemented in the RBMT system and have been beyond our control in the experiments. The LT engine has several different indicators which may throw a time-out exception, if, for example, the analysis phase takes too long to produce a result. To avoid getting time-out errors, only sentences with up to 50 tokens are treated with the Tree Selector. Additionally the Tree Selector itself checks the processing time and returns intermediate results, if this limit is reached. This ensures that we receive a proper translation for all sentences.4

3.3 Examples

Using our stochastic selection component, we are able to fix errors which can be found in translation output generated by the original Lucy engine.

Table 6 shows several examples including source text, reference text, and translations from both the original Lucy engine (A) and our hybrid system (B). We will briefly discuss our observations for these examples in the following section.

1. Translation A is the default translation. The parse tree for this translation can be seen in Figure 1. Here the adjective alleged is wrongly parsed as a verb. By contrast, Figure 2 shows the tree selected by our hybrid implementation, which contains the correct analysis of alleged and results in a correct translation.

2. Word order is improved in the Example 2.

3. Lexical items are associated with a domain area in the lexicon of the rule-based system. Items that are contained within a different domain than the input text are still accessible, but items in the same domain are preferred. In Example 3, this may lead to the incorrect disambiguation of multi-word expressions: the translation of to blow up as in die Luft fliegen was not preferred in Translation A due to the chosen domain and a more superficial translation was chosen. This problem is fixed in Translation B. Our system chose a tree leading to the correct idiomatic translation.

4. Something similar happens in Example 4 where the choice of preposition is improved.

5. These changes remain at a rather local scope, but we also have instances where the sentence improves globally: Example 5 illustrates this well. In translation A, the name of the book, “After the Ice”, has been moved to an entirely different place in the sentence, removing it from its original context.

6. The same process can be observed in Example 6, where the translation of device was moved from the main clause to the sub clause in Translation A.

7. An even more impressive example is Example 7. Here, translation A was not even a grammatically correct sentence. This is due to the heuristics of the Lucy engine, although they could also create a correct translation B.

These examples show that our initial goal of improving the given RMBT system has been reached and that a hybrid MT system with an architecture similar to what we have described in this paper does in fact perform quite well.
<table>
<thead>
<tr>
<th>Table 6: Translation Examples for Original (A) and Improved (B) Lucy</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>2</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>3</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>4</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>5</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>6</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>7</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
4 Conclusion and Outlook

The analysis phase proves to be crucial for the overall quality of the translation in rule-based machine translation systems. Our hybrid approach indicates that it is possible to improve the analysis results of such a rule-based engine by a better selection method of the trees created by the grammar. Our evaluation shows that the selection itself is no trivial task, as our initial experiments deliver results of varying quality. The degradations we have observed in our own manual evaluation can be fixed by a more fine-grained selection mechanism, as we already know that better trees exist, i.e. the default translations.

While the work reported on in this paper is a dedicated extension of a specific rule-based machine translation system, the overall approach can be used with any transfer-based RBMT system. Future work will concentrate on the circumvention of e.g. the time-out errors that prevented a better performance of the stochastic selection module. Also, we will more closely investigate the issue of decreased translation quality and experiment with other decision factors that may help to alleviate the negative effects.

The LiSTEX module provides us with high quality entries for the lexicon, increasing the coverage of the lexicon and fluency of the translation. As a side-effect, the new terms also help to reduce parsing errors, as formerly unknown multiword expressions are now properly recognised and treated. Further work is being carried out to increase the precision of the extracted terminology lists.

The addition of stochastic knowledge into an existing rule-based machine translation system is an example of a successful, hybrid combination of different MT paradigms into a joint system. Our system turned out to be the winning system for the English—German language pair of the WMT11 shared task.

Acknowledgements

The work described in this paper was supported by the EuroMatrixPlus project (IST-231720) which is funded by the European Community under the Seventh Framework Programme for Research and Technological Development.

References


