Improving Transfer-Based MT Systems with Automatic Refinements

Ariadna Font Llitjós, Jaime Carbonell and Alon Lavie
Language Technologies Institute, School of Computer Sciences.
Carnegie Mellon University.
5000 Forbes Ave. Pittsburgh, PA 15217
{aria,jgc,alavie}@cs.cmu.edu

Abstract

In this paper, we describe an Automatic Rule Refiner that, given online post-editing information, traces errors back to problematic translation rules and proposes concrete fixes to such rules. Evaluation results on an English-to-Spanish Transfer-based MT system show that this approach generalizes beyond sentences corrected by bilingual speakers to unseen data. We show that by applying automatic refinements, higher translation accuracy can be achieved as measured by automatic evaluation metrics.

1 Introduction

The approach described in this paper takes post-editing information from non-experts to automatically improve the underlying rules and lexical entries of an existing transfer-based Machine Translation (MT) system. This process can be divided into two main steps. In the first step, an online post-editing tool allows for easy error correction leading to diagnosis and implicit error categorization. In the second step, an Automatic Rule Refiner performs blame assignment and error remediation, by tracking errors and suggesting repairs of lexical and morpho-syntactic nature (such as adding a new entry, sense or form to the lexicon, adding a missing agreement constraint or a constituent to transfer rules or changing the order of the constituents).

This approach directly improves the recall of corrected MT output and, more importantly, it generalizes over unseen data, increasing MT output accuracy for similar sentences that have not been corrected by bilingual speakers.

Such a system is highly relevant for language pairs with little or no parallel data. However, any existing Transfer-Based MT (TBMT) system could be automatically improved by an Automatic Rule Refiner similar to the one described here.

The main goal of this paper is to illustrate the impact of automatic rule refinements on previously unseen data. According to three different automatic metrics (METEOR, BLEU and NIST) the refined system outperforms the baseline system after just a few user corrections ($p=0.0051$).

2 Related Work

Nishida (1988) and colleagues described a Post-Editing Correction information Feedback system (PECOF) in its early stages, which also sought to improve a transfer-based MT system. The main differences between their approach and the one described in this paper are: 1) the use of expert post-editors, whose work is not only to correct MT output but also to formulate correcting procedures corresponding to unseen error patterns, which are then executed by the PECOF system, and 2) the use of two MT systems in order to detect discrepancies between intermediate representations of the source language and the target language side, namely an *original* MT system (Japanese to English) and a *reverse* MT system (English to Japanese) which was applied to the post-edited English translation.

The grammar rules of our TBMT system integrate information from the three components of a typical transfer system: syntactic analysis (parsing), transfer and generation. Thus, in comparison

with the PECOF system, blame assignment becomes highly simplified, since it is more directly inferable from corrections.

Menezes and Richardson (2001) and Imamura et al. (2003) have proposed the use of reference translations to "clean" incorrect or redundant rules after automatic acquisition. The method of Imamura and colleagues consists of selecting or removing translation rules to increase the BLEU score of an evaluation corpus. In contrast to filtering out incorrect or redundant rules, we propose to actually refine the translation rules themselves, by editing valid but inaccurate rules that might be lacking a constraint, for example.

3 Error Correction Extraction

The first step of the rule refinement process is the extraction of error correction information. Our approach relies on bilingual speaker post-editing information, collected via a user-friendly online Translation Correction Tool (TCTool). Users of the TCTool can edit, add and delete words, as well as alignments, and can change word order by dragging words around (Figure 1).

A set of user studies was conducted to discover the right amount of error information that non-expert bilingual speakers can detect reliably when using the TCTool. These studies showed that minimal post-editing can be elicited much more reliably (F1 0.89) than error type information (F1 0.72) (Font Llitjós and Carbonell 2004).

- Modify a word
- Add a word
- Delete a word
- Change word order
- Add an alignment
- Delete an alignment

Figure 1. Basic Correction Actions allowed by the TCTool.

From this user study, it became apparent that correction actions, error and correction words, and alignment information are sufficient for Automatic Rule Refinement purposes.

Figure 2 illustrates initial and final TCTool snapshots showing the incorrect Spanish translation as produced by the MT system (top) and the final corrected MT output, as post-edited by a bilingual speaker (bottom).

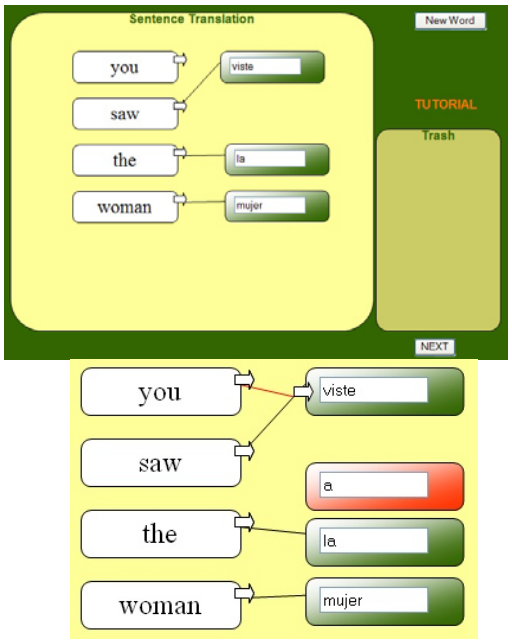

Figure 2. TCTool snapshots showing initial screen with incorrect MT output (top) and corrected MT output (bottom).

Each user correction is stored in a Correction Instance (CI), which is processed and parsed by the Automatic Rule Refiner (ARR) to extract all the relevant correction and error information. Figure 3 shows the CI extracted from the user correction illustrated in Figure 2. In addition to the source language sentence (SL) and target language sentence (TL), CIs also store alignment information (AL).

SL: You saw the woman
TL: *Viste la mujer*
AL: ((2,1),(3,2),(4,3))

Action 1: Add (*a*)

CTL: *Viste **a** la mujer*
CAL: ((1,1),(2,1),(3,3),(4,4))

Figure 3. An example of Correction Instance.

CIs store all correction actions taken by a user, with related error information, into a vector of actions. Actions are processed by the ARR one at a time.

3.1 Collection of Correction Instances

Blame assignment and refinement decisions made by the system fully depend on the correct extraction and processing of error correction information given by bilingual speakers. Since users of the

TCTool are not linguists or translation experts, the need to compare different correction instances and filter out noise becomes even more relevant.

The ARR reads in multiple correction instances affecting multiple translation pairs, and stores them in a Collection. This allows the ARR to compare all the CIs affecting a SL-TL pair and, if they contain equivalent information[1], they are stored only once in the Collection with a weight proportional to the number of different CIs that were found to be equivalent. For each SL-TL pair, the Automatic Rule Refiner only processes the CI with the highest weight (namely, most user support).

Since we want to prioritize correction instances that tackle simpler errors first, the ARR uses a ranking algorithm to order CIs according to their error complexity. This "Tetris" approach is based on the underlying assumption that once simpler errors are fixed, more complex errors will be simplified (thus moving up in the ranking) and become easier to fix automatically. For a more detailed description of error complexity and ranking of errors, see Font Llitjos and Ridmann (2007).

4 Automatic Rule Refinements

After extracting the location of the error in the translated sentence (position in the TL vector) and inferring the implicit error-type information, which corresponds to the correction actions performed by non-expert bilingual speakers, the Automatic Rule Refiner can trace the errors back to incorrect lexical and grammar rules responsible for the errors (blame assignment) and propose concrete fixes to such rules (rule refinement).

In our MT system, translation rules include parsing, transfer, and generation information, similar to the modified transfer approach used in the early Metal system. The constraint-enriched synchronous grammar is parsed using pseudo-unification (Tomita and Knight, 1987). In our approach, automatic refinements only affect the target language side of translation rules, namely transfer and generation information.

After a brief description of the theoretical framework, one concrete example is given for each type of refinement operation.

4.1 Rule Refinement Operations

There are two main refinement operations applicable to both grammar rules and lexical entries: CONSTRAIN, which specializes in handling features, and BIFURCATE, which specializes in handling structure.

The **CONSTRAIN** operation consists of modifying an existing overly general rule (R0), by adding one or more agreement constraints, effectively replacing it with a more specific correct rule (R1). For an example see Figure 4.[2]

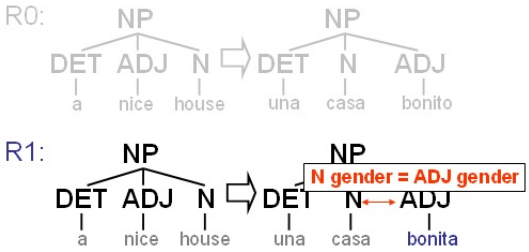

Figure 4: The CONSTRAIN operation adds feature constraints to rules that are overly general (R0) to achieve the right level of specificity (R1).

In the CONSTRAIN case, the new refined rule needs to translate correctly the same sentences as before plus the new user-corrected sentence.

The **BIFURCATE** operation is used if the original rule is correct in general but does not handle special cases. BIFURCATE makes a copy of the original rule (R0) and refines the copy (R1) so that it covers an exception to the general rule. In the BIFURCATE case both the original rule and the refined rule coexist in the grammar (Figure 5).

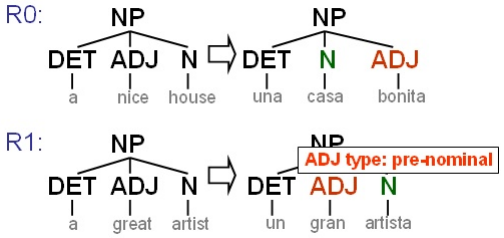

Figure 5: Example of the BIFURCATE operation applied to the general NP rule that deals with post-nominal adjectives in Spanish (R0) to also cover the pre-nominal order (R1).

[1] Equivalent CIs are CIs that in addition to having the same SL-TL and Corrected TL, once the spurious loops have been detected and removed; they also have the same set of correction actions affecting the same words.

[2] Even though the rules in Figures 4 and 5 appear lexicalized, this is just for illustration purposes and most of the rules in our grammar are at the POS level.

This is appropriate for cases when the general rule has correctly applied before, for example in translating *a nice house – una casa bonita,* the grammar already contains the general rule to deal with nouns and adjectives in Spanish , and we want the grammar to also account for an exception to the general rule, namely pre-nominal adjectives.

Our approach does not handle grammar modifications that require the specification of a completely new rule. Corrections requiring new rules that cannot be derived from an existing rule (either by switching the order of the constituents or adding/deleting new constituents) are sent to a Rule Learner (Probst, 2005).

4.2 CONSTRAIN: Adding Constraints

In general, adding a feature constraint to a grammar rule makes the grammar tighter thus preventing spurious translations from being generated. Therefore, while the number of correct translations (recall) is expected to remain the same after applying a constrain operation, the number of incorrect translations is reduced (and precision increases).

{S,74}
S::S : [NP VP] → [NP VP]
((X1::Y1) (X2::Y2)
(x0 = x2)
((y2 subj) = -)
((y1 case) = nom)
((y1 agr) = (x1 agr))
((y2 tense) = (x2 tense))
((y2 agr pers) = (y1 agr pers))
((y2 agr num) = (y1 agr num)))

Figure 6: Translation rule result of two CONSTRAIN operations.

Figure 6 illustrates a translation rule which as been refined after processing the first CI in Figure 10. Refinements, in this case the addition of person and number constraints between the subject and the verb, appear in bold.

4.3 BIFURCATE: Modifying Structure

Consider the CI in Figure 3. The bilingual informant added a word but no alignment to any of the SL words. In this case, the ARR cannot apply any refinements at the lexical level, but the translation tree (generated by the MT system) provides all the necessary information to perform blame assignment (Figure 7), since all translation rules in the lexicon and the grammar are uniquely identified.

For this example, given the translation pair *you saw the woman – viste la mujer* and the user correction of adding the word "a" in front of *mujer*, the ARR detects that "a" is not aligned to any words in the SL sentence, and it proceeds to look at the translation tree to extract the appropriate rule that needs to be refined.

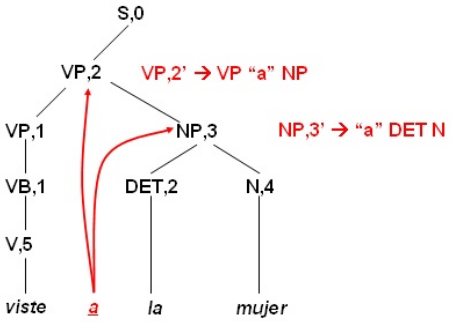

Figure 7: Translation Tree showing user insertion ("a") with two potentially relevant rules highlighted (VP,2 and NP,3).

In this case, since "a" is inserted between "viste" (V) and "la" (DET), there are four candidate rules to undergo the BIFURCATE operation, namely VB,1, VP,1, VP,2 and NP,3. Figure 7 illustrates the modifications required for two of the potentially relevant rules.

Adding an "a" in the right position to any of these rules would have the desired effect for this example. However, only the modification to the VP,2 rule generalizes well to other sentences. If the NP rule was selected, all instances of NP[DET N] would be generated in Spanish with an "a" preceding them, even when the NP is a subject or an oblique; this would result in an unnecessary ambiguity increase and a decrease in precision.

In general, to handle these cases when there is no option for further user interaction, the ARR refines the most specific candidate rule, namely the rule that encodes the most amount of context (Figure 8). This ensures that the refinement applies to syntactic environments most similar to the original corrected sentence. In this case, this means the refinement applies to object NPs only and not to all NPs.

This is still not the ideal level of generalization, since one would want to only add an "a" in front of animate object NPs in Spanish. The ARR could further refine the bifurcated rule to have a value constraint that restricts its application to NPs with *mujer* as a head. However, in the absence of se-

mantic features in the lexicon (such as animacy), not adding any further refinements is the best strategy to strive high accuracy and control unnecessary ambiguity.

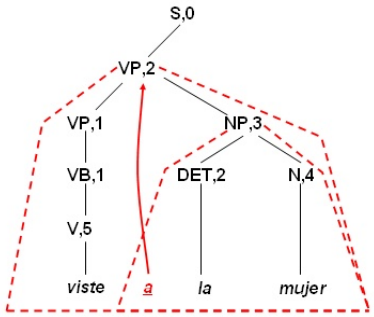

Figure 8: Depicting context captured by each candidate rule. VP,2 encapsulates more context than NP,3, and thus is more specific.

Figure 9 shows the result of bifurcating VP,2 and adding the word "a" as constituent in the right position on the target language side.

```
{VP,21}
VP::VP : [VP NP] -> [VP "a" NP]
( (X1::Y1)  (X2::Y3)
  (x0 = x1)
  ((y1 type) = tr)
  ((y0 agr) = (y1 agr))
  ((y0 tense) = (x0 tense))
  ((y0 subj) = (y1 subj)))
```

Figure 9: Translation rule result of a BIFURCATE operation that added a new constituent ("a").

See Font Llitjós et al. (2005) for more details about the ARR theoretical framework.

5 Evaluation Experiments

The goal of our experiments is to compare performance of a baseline system with an automatically refined system on new data, which has not been corrected by bilingual informants. The experiments reported in this Section were done on English to Spanish MT output. We calculated METEOR scores (Lavie et al. 2004) with v0.5.1, and BLEU (Papineni et al., 2002) and NIST (Doddington 2002) with mteval-v11b.pl.

5.1 Test Data: BTEC

For evaluation on unseen data, we selected the Basic Travel Expression Corpus (BTEC) (Takezawa et al., 2002), which has been used in the Evaluation campaigns in connection with the International Workshop on Spoken Language Translation (IWSLT). Besides still being currently used to build real systems, this corpus contains relatively simple sentences that are comparable to the ones initially corrected and covered by the baseline manual grammar.

Data			English
BTEC	Train	Sentences Pairs	123,416
		Sentence Length	7.3
		Word Tokens	903,525
		Word Types	12,578
	Test	Sentence Pairs	506
		Word Tokens	3,764
		Word Types	776
		Coverage Test	756 (97%)

Table 1: Corpus Statistics for the target language

As our test set, we used 506 English sentences for which two sets of Spanish reference translations were available. Table 1 shows corpus statistics for the BTEC data.

In order to reduce the number of out-of-vocabulary words, we used phrase alignment techniques to augment the baseline lexicon (Vogel 2005) and manually annotated lexical entries with POS and agreement information.

For the language model, we built a Suffix Array Language Model with the SALM toolkit (Zhang & Vogel, 2006) on 123,416 Spanish sentences (train).

1. I sleep – *Yo *duermen* – Yo duermo
CONSTRAIN: Add person and number agreement constraints (Subj-Verb)

2. I see the red car – *Veo el auto *roja* – …el auto rojo
CONSTRAIN: Add gender agr. constraint (N-ADJ)

3. The girl is tall – *La niña es *alto* – La niña es alta
CONSTRAIN: Add gender agr. constr. (Subj-Compl)

4. I gave John a book – *Yo di un libro Juan – ... a Juan
BIFURCATE: Add "a" to the appropriate Gr. rule

5. You saw the woman – *Viste la mujer – … a la mujer
BIFURCATE: Add "a" to the appropriate Grammar rule

6. Gaudi was a great artist – *Gaudí era un *artista gran* – gran artista.
BIFURCATE: Add NP rule to cover pre-nominal ADJs

7: I would like to go – me gustaría *que* ir – me gustaría ir
BIFURCATE: allow no translation for "to" ([to→ " "]).

Figure 10: Examples of Correction Instances processed by the Automatic Rule Refiner.

5.2 MT System

For our experiments, the refined MT system is the result of having the ARR process fourteen Correction Instances and applying automatic refinements to the baseline grammar and lexicon. The 14 sentence pairs corrected by bilingual informants were simple sentences containing one or two errors. Figure 10 shows half of the Correction Instances collected via the TCTool and processed by the ARR, with information about the Rule Refinement operation that they triggered.

Different sets of CIs were tested on development data, and the final set was chosen since it increased the recall without proportional loses in precision.

The number of rules in the original baseline grammar and the refined grammar, after 14 CIs were processed by the ARR, can be seen on Table 2.[3] The last column shows the number of constraints in the grammar. A total of 30 constraints were added to the grammar via automatic refinements. This means that most of the modifications in the grammar were at the constraint level, and thus resulted from applying the CONSTRAIN operation described in Section 4.1, whereas three BIFURCATE operations were applied to the baseline Grammar and five to the Lexicon.

Data	System	Lex.	Gram.	Constr.
BTEC	Baseline	1732	40	201
	Refined	1737	43	231

Table 2: Number of translation rules and constraints in the translation grammars and the lexicons.

5.3 Oracle Experiment

Oracle scores provide an upper-bound in performance as measured by automatic evaluation metrics. Given 100-best lists for each source language sentence in the BTEC test set, we calculated automatic metric scores for METEOR, BLEU and NIST, and selected the highest scoring translations. Table 3 shows the results of selecting the highest scoring translations according to each automatic metric in turn.

System	METEOR	BLEU	NIST
Baseline	0.6863	0.4068	7.42
Refined	0.6954	0.4215	7.51

Table 3: Automatic metric oracle scores based on a 100-best list

These scores reflect the fact that the system refined automatically (Refined) is able to feed the *n*-best list with better translations than the Baseline system, measured in terms of overlap with two sets of human reference translations ($p=4.42x10^{-6}$).[4]

Even with a small set of independent user corrections, the refined system shows potential improved translation quality as indicated by higher scores for all three automatic evaluation metrics in Table 3.

5.4 Translation Results

The ultimate test consists of seeing whether automatic rule refinements can also improve translation quality of an end-to-end MT system, where the Transfer engine produces multiple alternative translations and a decoder ranks translations according to language model probabilities as well as a fragmentation penalty score.

To this end, we evaluated final translation accuracy of 1-best hypothesis using BLEU, NIST and METEOR metrics.

System	METEOR	BLEU	NIST
Baseline	0.6176	0.3425	6.53
Refined	0.6222	0.3513	6.56

Table 4: Automatic metric scores for 1-best decoder hypothesis.

As expected, automatic metric scores for the 1-best hypothesis are significantly lower than the oracle scores for both the Baseline and the Refined system. However, the important thing to notice from these results is that the refined system consistently outperforms the baseline MT system for all three automatic metrics (Table 4).

The difference between the Baseline and the Refined system in terms of 1-best scores is slightly smaller than the difference between oracle scores, which means that the decoder can not fully leverage the improvements made in the grammar. This is also to be expected, since the decoder fails to select the best translation in most cases.

[3] Translation rules in our MT system include parsing, transfer and generation information, which might otherwise be expressed with three different rules in other TBMT systems.

[4] According to the standard paired two-tailed t-Test.

Even though the language model (LM) for the BTEC data is rather small, using a larger LM with additional out-of-domain data from the Europarl training corpus (Koehn, 2005) did not improve these results.

5.5 Error Analysis

After manual inspection, most of the differences between the Baseline and Refined systems were due to three of the 14 CIs processed by the Automatic Rule Refiner, namely 4, 5 and 7 in Figure 10, all of which yielded a BIFURCATE operation.

In 56 cases, the additional generation capabilities of the refined system successfully produced a better translation than the baseline system; 37 of these improvements were ranked as 1-best by the decoder. Table 5 shows examples of the three most common types of fixes yielded by automatic refinements.

5.6 MER Training results

Like in Statistical MT (SMT) systems, in the Transfer engine translations are ranked to their total cost, which is a weighted linear combination of the individual costs. When adding more features to the translation system, a careful balancing of the individual contributions can make a significant difference. However, with each feature added, manually tuning the system becomes less and less practical, and automatic optimization becomes necessary.

We used Minimum Error Rate (MER) training, to find optimal feature weights for the different components in our decoder (*n*-gram language model and fragmentation penalty score). MER improves the match between 1-best translations and given reference translations.

When setting optimal weights in the decoder, both the Baseline and the Refined system get higher scores, not only according to BLEU, which was used as the objective function, but also according to METEOR and NIST (Table 6).

System	METEOR	BLEU	NIST
Baseline	0.6184	0.3609	6.68
Refined	0.6231	0.3780	6.79

Table 6: Automatic metric scores for 1-best hypotheses, after decoder weights have been optimized.

Moreover, the difference between the Baseline and the Refined system after MER training is statistically significant with a p value of 0.0051.[4] For more details on MER training experiments, see Font Llitjós and Vogel (2007).

6 Language Independence of Approach

The same mechanisms to automatically extend and refine an English to Spanish TBMT system are valid when applied to a very different language pair, i.e., Mapudungun-Spanish. The fact that Spanish (fusional and analytic) and Mapudungun (agglutinative and polysynthetic) are typologically very different allows us to argue that our automatic refinement approach is language independent to a large extent.

Initial experiments on Mapudungun-Spanish have shown that automatic rule refinements can be successfully applied (both to the lexicon and the grammar) to translate sentences not originally translated by an unrefined TBMT system.

7 Conclusions and Future Work

Translation rules can be automatically refined to increase overall translation recall and precision. Experiments with an English-Spanish Transfer-Based MT system show that automatic refinements generalize well beyond the specific sentences corrected by users to previously unseen data. Added generation capabilities of the refined MT system led to improved translation quality as measured by different automatic evaluation metrics.

Initial experiments on a Mapudungun-Spanish MT system provide evidence for the language independence of our automatic rule refinement approach.

Source language	Baseline	Refined
is this seat taken ?	está este asiento *cogidas* ?	está este asiento tomado ?
please call the police .	por favor llame la policía .	por favor llame a la policía .
i would like to put my valuables in the safe deposit box .	me gustaría *que* poner mis objetos de valor en la caja fuerte .	me gustaría poner mis objetos de valor en la caja fuerte .

Table 5: MT output examples from the BTEC test set before and after automatic refinements applied to the grammar and lexicon.

Results reported in this paper are based on a rather small set of Correction Instances. Future experiments will gather different sets of user corrections on training data and will measure their effects on test data drawn from the same distribution.

On a larger scale, and even though the Automatic Rule Refiner takes several steps to filter out noise early in the process, automatic refinements are still entirely based on bilingual speaker corrections, which ultimately means that there is no guarantee that automatic refinements will always increase translation accuracy.

In this respect, having shown that a decoder can pick up grammar and lexicon improvements generated by automatic refinements, we have closed the feedback loop. This allows such a system to validate refinements that lead to measurable improvements on translation accuracy.

A more realistic grammar would include rule probabilities. In a probabilistic setting, user corrections could also be used to adjust rule weights.

Finally, post-editing information gathered via the TCTool can be used to train statistical MT systems to improve translation accuracy of Rule-Based MT systems, as recently shown by Simard and colleagues (2007).

Acknowledgements

We would like to thank Stephan Vogel for his help with the evaluation, especially with MER training results, and William Ridmann for his help with the implementation of the Automatic Rule Refiner. This research was funded in part by NSF grant number IIS-0121-631.

References

Doddington G. 2002. *Automatic evaluation of machine translation quality using n-gram co-occurrence statistics*. HLT, San Diego, USA.

Font Llitjós A. and S. Vogel, 2007. *A Walk on the Other Side: Adding Statistical Components to a Transfer-Based Translation System*. Syntax and Structure in Statistical Translation Workshop at HLT-NAACL, Rochester, USA.

Font Llitjós, A. and W. Ridmann. 2007. *The Inner Works of an Automatic Rule Refiner for Machine Translation.* METIS-II Workshop, Leuven, Belgium.

Font Llitjós, A., J. Carbonell and A. Lavie. 2005. *A Framework for Interactive and Automatic Refinement of Transfer-based Machine Translation*. EAMT 10th Annual Conference, Budapest, Hungary.

Font Llitjós, A. and J. Carbonell. 2004. *The Translation Correction Tool: English-Spanish user studies.* LREC, Lisbon, Portugal.

Imamura, K., E. Sumita and Y. Matsumoto. 2003. *Feedback cleaning of Machine Translation Rules Using Automatic Evaluation*. ACL.

Koehn, P. 2005. *Europarl: A Parallel Corpus for Statistical Machine Translation*. MT Summit X, Phuket, Thailand.

Lavie, A., K. Sagae and S. Jayaraman. 2004. *The Significance of Recall in Automatic Metrics for MT Evaluation*. AMTA.

Menezes, A, & Richardson, S. D. 2001. *A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora*. Workshop on Example-Based MT, in MT Summit VIII.

Nishida, F.; S. Takamatsu, T. Tani and T. Doi. 1988. *Feedback of Correcting Information in Posteding to a Machine Translation System*. COLING.

Papineni, K, S. Roukos, T. Ward, and W. Zhu. 2002. *BLEU: a method for automatic evaluation of machine translation*. 40th ACL, Philadelphia, USA.

Probst, K. 2005. *Automatically Induced Syntactic Transfer Rules for Machine Translation under a Very Limited Data Scenario*. Ph.D. Thesis. Carnegie Mellon University.

Simard, M., C. Goutte and P. Isabelle. 2007. *Statistical Phrase-based Post-editing*. HLT-NAACL-07, Rochester, NY, USA.

Takezawa, T, E. Sumita, F. Sugaya, H. Yamamoto, and S. Yamamoto, 2002. *Toward a Broad-Coverage Bilingual Corpus for Speech Translation of Travel Conversations in the RealWorld*. LREC, Las Palmas, Spain.

Tomita, Masaru and Kevin Knight. 1987. *Pseudo-Unification and Full-Unification*. CMT Technical Report (88), Carnegie Mellon University.

Vogel, S. 2005. *PESA: phrase pair extraction as sentence splitting*. In Proceedings of the Machine Translation Summit X, Phuket, Thailand.

Zhang, Y and S. Vogel. 2006. *Suffix Array and its Applications in Empirical Natural Language Processing,*. In the Technical Report CMU-LTI-06-010, Pittsburgh PA, USA.