Hybrid Rule-Based – Example-Based MT: Feeding Apertium with Sub-sentential Translation Units

Felipe Sánchez-Martínez[†] Mikel L. Forcada[†‡] Andy Way[‡]

† Dept. Llenguatges i Sistemes Informàtics
Universitat d'Alacant
E-03071 Alacant, Spain
{fsanchez,mlf}@dlsi.ua.es

‡ School of Computing
Dublin City University
Dublin 9, Ireland
{mforcada,away}@computing.dcu.ie

Abstract

This paper describes a hybrid machine translation (MT) approach that consists of integrating bilingual chunks (sub-sentential translation units) obtained from parallel corpora into an MT system built using the Apertium free/open-source rule-based machine translation platform, which uses a shallow-transfer translation approach. In the integration of bilingual chunks, special care has been taken so as not to break the application of the existing Apertium structural transfer rules, since this would increase the number of ungrammatical translations. The method consists of (i) the application of a dynamic-programming algorithm to compute the best translation coverage of the input sentence given the collection of bilingual chunks available; (ii) the translation of the input sentence as usual by Apertium; and (iii) the application of a language model to choose one of the possible translations for each of the bilingual chunks detected. Results are reported for the translation from English-to-Spanish, and vice versa, when marker-based bilingual chunks automatically obtained from parallel corpora are used.

1 Introduction

Rule-based machine translation (RBMT) systems perform the translation task using linguistic knowledge, most of which is explicitly encoded by human experts in formats that can be understood by the machine translation (MT) engine. Unlike corpus-based machine translation (CBMT) systems, which rely on the availability of very large sentence-aligned bilingual corpora, RBMT systems can be built for languages lacking those extensive resources by eliciting and encoding translation knowledge directly, a rather costly task. The main advantage of RBMT systems compared to CBMT systems is their predictability; lexical and structural selection is consistent, errors are predictable and may usually be diagnosed and assigned to a particular module in the system, and predictability of errors makes real-life post-editing easier in dissemination tasks. CBMT systems—such as statistical MT (SMT) systems—rely heavily on statistical target-language models, whereas RBMT systems tend to produce fewer translations which are deceptively fluent but in fact prove to be inadequate.

Even if errors can be easily attributed to a particular module or data set in the system, there is no trivial way to incorporate the hard work of post-editors into an RBMT system, because of the need for painstaking analysis and encoding of the changes into data that can be used by the system. Furthermore, one may be interested in having a different translation for a given source-language (SL) segment, not because the translation provided by the MT system is wrong, but because it is not an adequate translation in the specific domain of the text being translated at that precise moment.

Some researchers have set out to do *automatic statistical post-editing* by training SMT systems to "translate" the output of an RBMT system into the reference translations provided [7, 16]; the automatic evaluation indicators that compare the output translations against the references show improvements which may be clearly attributed to improvement in certain linguistic choices [7], but two important issues

Mikel L. Forcada, Andy Way (eds.)
Proceedings of the 3rd Workshop on Example-Based Machine Translation, p. 11–18
Dublin, Ireland, November 2009

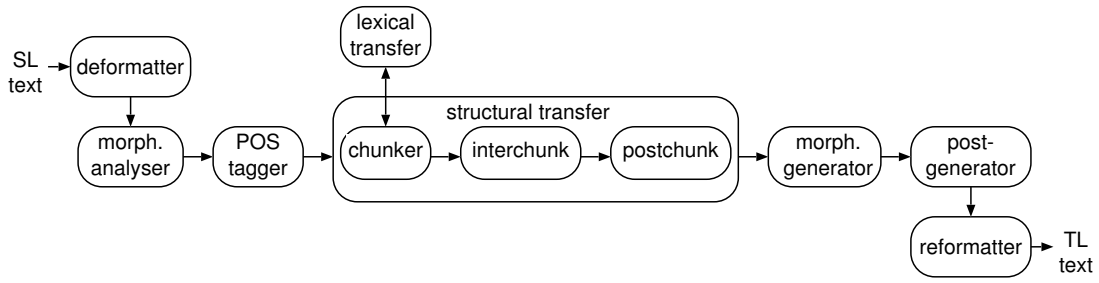

Figure 1: Modular architecture of the Apertium free/open-source MT platform.

remain: whether these improvements result in actual post-editing savings, and—as with human post-editing—how the post-editing operations performed by the SMT systems can possibly be incorporated in order to improve rule-based data.

An alternative to incorporating new data resulting from a laborious analysis of post-editing operations would consist of building a hybrid system which takes advantage of existing bilingual sentence-aligned resources, such as translation memories generated by professional translators when they provide solutions to the translation at hand, but which otherwise uses RBMT as a baseline engine. This can be partially automated, e.g. Eisele et al. [8] use bilingual corpora instead of post-editor output to include lexical entries in an RBMT system. In this way, we retain the predictability of the RBMT system as much as possible, and give it away only where a better alternative is provided by existing corpora.

Trivial hybridisation would work at the sentence level, but its coverage would be low, as whole sentences are quite unlikely to be repeated. *Sub-sentential* hybridisation is therefore needed. One step further is to translate just the difference between the new sentence and the best-matching sentence in a bitext [3]. To that end, one needs to align sentences and extract sub-sentential translation units from them. Other authors have explored a tighter, run-time integration of examples and rules [4].

This paper uses the sub-sentential aligners used in the example-based machine translation (EBMT) system MATREX [9, 18] that segment the source and target sentences into syntactically-motivated *chunks* using the *marker hypothesis*, which states that natural languages contain a small number of surface elements—morphemes and lexemes—that signal the presence of particular syntactic constructions [10]. The resulting chunks are then aligned to produce sub-sentential translation units. These are incorporated into the flow of the RBMT system Apertium [1], which is described in section 2. Integrating marker-based bilingual chunks into an RBMT system seems to be a natural choice because they are syntactically motivated, as compared to linguistically-blind SMT segments.

2 The Apertium free/open-source machine translation platform

Apertium[1] [1] follows the typical *transfer*-based approach to MT [11] that divides the translation into three phases: (1) *analysis* of the SL text into an intermediate representation (IR), (2) *transfer* from the SL IR to a target-language (TL) IR, and (3) *generation* of the TL text from the TL IR. It follows the shallow-transfer approach shown in Figure 1, which may be considered to be somewhere between direct translation and full syntactic transfer, i.e. a *transformer* system [2]. It consists of the following modules:

- A *deformatter* which encapsulates the format information in the input document. For the in-

[1]The MT engine, documentation, and linguistic data for different language pairs can be downloaded from `http://www.apertium.org`.

put HTML text `Francis' <strong>car</strong> is broken`, the output of this module is `Francis'[ <strong>]car[</strong> ]is broken`.

- A *morphological analyser* which tokenises the text into surface forms and delivers, for each surface form, one or more *lexical forms* consisting of *lemma*, *lexical category* and morphological inflection information. The output of this module for the running example is `^Francis'/Francis<np><ant><m><sg>+'s<gen>$ [ <strong>] ^car/car<n><sg>$[</strong> ]^is/be<vbser><pri><p3><sg>$ ^broken/break<vblex><pp>$`.
- A *part-of-speech tagger* which chooses, using a first-order hidden Markov model [5] (HMM), one of the lexical forms corresponding to an ambiguous surface form. For each word in the input document it chooses the most probable lexical form given the context. The output of this module is `^Francis<np><ant><m><sg>$ ^'s<gen>$[ <strong>] ^car<n><sg>$[</strong> ]^be<vbser><pri><p3><sg>$ ^break<vblex><pp>$`.
- A *lexical transfer* module which reads each SL lexical form and delivers the corresponding TL lexical form by looking it up in a bilingual dictionary. `^Francis<np><ant><m><sg>$` is translated into Spanish as `^Francis<np><ant><m><sg>$`; `^car<n><sg>$` as `^coche<n><m><sg>$`; `^be<vbser><pri><p3><sg>$` as `^ser<vbser><pri><p3><sg>$`; and `^break<vblex><pp>$` as `^romper<vblex><pp>$`.
- A *structural transfer* consisting of three sub-modules:
  - A *chunker* which, after invoking lexical transfer, performs local syntactic operations and segments the sequence of lexical units into chunks. A chunk is defined as a fixed-length sequence of lexical categories that corresponds to some syntactic feature such as a noun phrase or a prepositional phrase. The segmentation into chunks for the running example is `^nom<SN><UNDET><m><sg>{^Francis<np><ant><3><4>$}$ ^pr<GEN>{}$[ <strong>]^nom<SN><UNDET><m><sg>{^coche<n><3><4>$}$[</strong> ]^be_pp<Vcop><vblex><pri><p3><sg><GD>{^estar<vblex><3><4><5>$ ^romper<vblex><pp><6><5>$}$`. Note that in addition to segmenting the output of the tagger and integrating the translations provided by the *lexical transfer* module, some operations have already been applied, such as the replacement of the Spanish verb *ser* by the verb *estar*.
  - An *interchunk* module which performs more global operations with/between the chunks. The output of this module is `[<strong>]^nom<SN><PDET><m><sg>{^coche<n><3><4>$}$ [</strong> ]^pr<PREP>{^de<pr>$}$ ^nom<SN><PDET><m><sg>{^Francis<np><ant><3><4>$}$ ^be_pp<Vcop><vblex><pri><p3><sg><m>{^estar<vblex><3><4><5>$ ^romper<vblex><pp><6><5>$}$`.
  - A *postchunk* module which performs finishing operations on each chunk. After propagating the inflection information accompanying each chunk to the words inside it, the output of this sub-module is `[<strong>]^el<det><def><m><sg>$ ^coche<n><m><sg>$[</strong>]^de<pr>$ ^Francis<np><ant><m><sg>$ ^estar<vblex><pri><p3><sg>$ ^romper<vblex><pp><m><sg>$`.
- A *morphological generator* which delivers a TL surface form for each TL lexical form, by suitably inflecting it. After morphological generation, the translation of the running example is `[<strong>]el coche[</strong> ]de Francis está roto`.
- A *post-generator* which performs orthographic operations such as contractions (e.g. Spanish *del=de+el*) and apostrophations (e.g. Catalan *l'institut=el+institut*). For the running example this module does not perform any change to the input stream.
- A *reformatter* which de-encapsulates any format information. After de-encapsulating the format information we have the output of the MT engine: `<strong>el coche</strong> de`

```
Francis está roto.
```

The Apertium MT engine is completely independent of the linguistic data used to translate for a given language pair. Linguistic data is encoded using XML-based formats,[2] which allows for easy data transformation and maintenance.

3 Integration of bilingual chunks in Apertium

The simplest way to integrate the use of bilingual chunks in an existing MT system would be to detect the SL chunks before performing the translation and replace them with the corresponding TL chunks. However, this simple approach would distort both the context used by the tagger to perform the categorial disambiguation and, more importantly, the recognition of patterns by the structural transfer module, and thus the application of structural transfer rules. Needless to say, this would result in a larger number of ungrammatical translations.

3.1 Translation approach

We propose to integrate the bilingual chunks without affecting the way Apertium translates as follows:[3]

1. Apply a dynamic-programming algorithm to compute the best coverage (see section 3.2) of the input sentence using the source part of the collection of bilingual chunks provided. Once detected, the mechanism used by Apertium to encapsulate formatting information (see section 2) is used to insert delimiters around the SL chunks detected while keeping them intact.
2. Translate the input sentence as usual by Apertium; the detected chunks will also be translated using the rule-based engine, while the delimiters containing the detected SL chunks which have just been introduced remain in place.
3. Chunk delimiters are used to locate appropriate chunks and to replace the translation provided by Apertium with the corresponding one (see below).

The introduction of chunk delimiters as formatting information may have some side-effects. As a result of the application of some structural transfer rules, the formatting information (treated by the Apertium modules as blank space between words) may have been placed in a different position, e.g. by rules that reorder the words in a chunk, or otherwise insert or delete lexical items.

In section 4 we evaluate the performance of our approach when using bilingual chunks automatically obtained from parallel corpora. As a result of this, a SL chunk may have different TL translations. To select the best translation for each SL chunk, we use a language model to score each possible translation in the context in which it appears; the Apertium translations are also considered to counteract the possible side-effects of chunk delimiters being moved around.

3.2 Best coverage

We try to cover the maximum number of words of each SL sentence by using the longest possible chunks; the longer the chunks, the more accurate their translations may be expected to be, because they integrate more context. The best coverage can be computed efficiently by storing the set S of SL chunks in a prefix tree (trie) of strings. This trie—a deterministic finite-state automaton—is defined as $\langle Q, V, \delta, q_0, F \rangle$,

[2] The XML formats (`http://www.w3.org/XML/`) for each type of linguistic data are defined through conveniently-designed XML document-type definitions (DTDs) which may be found inside the `apertium` package.

[3] The method described here is implemented inside the package `apertium-chunks-mixer` and released under the GNU GPL license; it can be freely downloaded from `http://sf.net/projects/apertium/files/`.

where $Q = \Pr(S)$ is the set of states, each one corresponding to a prefix of the SL chunks; V is the vocabulary of the words that can be found in the SL chunks; $\delta : Q \times V \rightarrow Q$ is the transition function, such that $\delta(X,w) = Xw$ for $w \in V$ if $X, Xw \in \Pr(S)$; $q_0 = \varepsilon$ is the initial state (the empty string); and $F = S$ is the set of final states recognising the SL chunks stored.

To compute the best coverage a dynamic-programming algorithm (Alg. 1) is applied, which starts a new search in the trie at every new word in the sentence to translate, and uses a set of alive states A in the trie and a map M that, for each word in the sentence, returns the best coverage up to that word together with its score. Actually, only the best coverage up to the last l words, where l is the maximum length of the SL chunks, needs to be stored.

Algorithm 1 BESTCOVERAGE: Algorithm to compute the best coverage of an input sentence.

Require: *sent*: SL sentence to translate; *i*: word index
  while $(A \neq \emptyset) \wedge (i \leq \text{WORDCOUNT}(sent))$ **do**
    $M[i] \leftarrow \emptyset$
    $A \leftarrow A \cup \{q_0\}$ /* To start a new search from this word */
    for all $s \in A$ **do**
      if $\exists c \in Q : \delta(s, sent[i]) = c$ **then**
        $A \leftarrow A \cup \{c\}$
        if $c \in F$ **then**
          $M[i] \leftarrow \text{BESTCOVERAGE}(M[i], \text{NEWCOVERAGE}(M[i - \text{CHUNKLENGTH}(c)], c))$
        end if
      end if
      $A \leftarrow A \setminus \{s\}$
    end for
    $M[i] \leftarrow \text{BESTCOVERAGE}(M[i], \text{NEWCOVERAGE}(M[i-1], \emptyset))$ /*$\emptyset$ means word not covered*/
    $i \leftarrow i + 1$
  end while
  return M[i-1]

Algorithm 1 uses four external procedures: $\text{WORDCOUNT}(sent)$ returns the number of words in the string *sent*; $\text{CHUNKLENGTH}(c)$ returns the number of words of the SL chunk recognised by state c; $\text{NEWCOVERAGE}(cov, c)$ computes a new coverage by adding to coverage *cov* the chunk recognised by state c; finally, $\text{BESTCOVERAGE}(a, b)$ receives two coverages and returns the one with the best score.

The best coverage is the one using the least possible number of chunks, i.e. the longest possible chunks; each word which has not been covered counts as a single chunk. If two different coverages use the same number of chunks, the chunk having the most frequent SL part is preferred.

4 Experiments

Corpora and tools. We have tested our approach on Spanish↔English translation[4] when integrating bilingual chunks automatically obtained from the Spanish–English parallel corpus distributed for the EACL 4^{th} workshop on SMT.[5] Before the extraction of bilingual chunks, sentences longer than 45 words or sentence pairs having TL/SL or SL/TL word ratios greater than 1.5 were discarded.[6] Table 1 gives

[4]The Apertium data used to run the experiments can be obtained from the Subversion repository in the SourceForge project `http://sf.net/projects/apertium/`: package `apertium-en-es`, revision 9284.

[5]`http://www.statmt.org/wmt09/`

[6]The word-ratio threshold is the result of adding one standard deviation to the observed average in the corpus.

Corpus	Sentences	English words	Spanish words
Training	1,187,905	26,983,025	27,951,388
Development	2,050	49,884	52,719
Test	3,027	77,438	80,580

Table 1: Number of sentences and words in the corpora used in the experiments.

Translation	No chunks		Marker-based			No. of chunks used (% words covered)		
	dev	test	θ	dev	test	Detected	Translated	
							All	Apertium
English→Spanish	17.10	18.51	11	17.41	18.94	6,812 (18%)	5,546 (15%)	2,662 (7%)
Spanish→English	17.71	18.81	28	17.91	19.14	6,321 (17%)	5,488 (14%)	2,929 (8%)

Table 2: Frequency threshold θ and percent BLEU score when translating the development and test corpora. The number of chunks detected and the number of chunks finally used in the translation of the test corpus are also reported together with the approximate percentage of words covered by chunks.

details on the corpus finally used for training, the development corpus used to filter the bilingual chunks obtained (see below) and the corpus used for testing.

Before the extraction of bilingual chunks, word alignments were computed by means of Giza++ [14], symmetrised using the "grow-diag-final-and" heuristic implemented in Moses [13], and used to discard bilingual chunks whose words are not aligned; bilingual chunks containing punctuation marks or numbers were also discarded.

To decide among all the possible translations of each SL chunk, we used a 5-gram target language model trained by means of the SRILM toolkit [17] over the TL side of the same parallel corpus used to obtain the bilingual chunks.

The marker-based bilingual chunks used were obtained by segmenting both sides of the parallel corpus with the marker-based chunkers [9] provided by the MATREX [6, 18] EBMT platform, and then by aligning the chunks through the chunk alignment algorithm implemented in MATREX [18, sec. 2.2].

Results. To decide which bilingual chunks to take into account in the translation, we translated the development corpus with Apertium when integrating bilingual chunks that were seen at least θ times; tested values for θ are in the interval $[5, 80]$. The set of bilingual chunks providing the best BLEU score [15] was used to translate the test corpus.

Table 2 reports the approximate translation performance, as indicated by the BLEU metric, when the translation is performed by Apertium without using bilingual chunks, and when using marker-based bilingual chunks. Note the small improvement achieved in the translation of both the development corpus and the test corpus; statistical significance tests using bootstrap resampling [12] show that this improvement is not significant. Also note that, contrary to what one might expect, the improvement is larger when translating the test corpus than when translating the development corpus.

The table also reports the number of chunks detected, together with the approximate percentage of words covered, in the first stage of our translation approach when translating the test corpus. The number of chunks finally used because the language model chose a translation in the collection of bilingual chunks is also shown. The last column in the table reports the number of chunks that are translated exactly the same way as Apertium does; note that in the final analysis, around half of the chunks used in the translation are translated in the same way as Apertium.

The differences between the chunks detected and those actually used in the translation may be due

to chunk delimiters moved and placed in the wrong position by the structural transfer component. This remains for further study.

5 Discussion

We presented a novel approach to the integration of sub-sentential translation units (bilingual chunks) into the Apertium free/open-source rule-based MT platform. Our approach uses the longest possible bilingual chunks available, in the hope that they will be more accurate since they integrate more context, together with a language model to decide which translation should be used when an SL chunk has more than one TL equivalent.

To test our approach we automatically extracted bilingual chunks based on the marker hypothesis from parallel corpora. The results show a small, but not statistically significant improvement in translation performance as measured by BLEU. The results also show that the percentage of SL words finally covered by chunks producing a translation different from the one provided by Apertium is relatively small.

We plan to improve the way we select the bilingual chunks to use from all the chunks obtained from the training corpus. The current approach is based only on the frequency of the chunks, so longer, perhaps more accurate, bilingual chunks are penalised in favour of shorter, less accurate chunks.

Acknowledgements

We thank Sergio Penkale and John Tinsley for their help with MATREX. Felipe Sánchez-Martínez's stay at Dublin City University, where this research was conducted, was funded by Generalitat Valenciana through grant BEST/2009/057. Support from the Spanish Ministry of Science and Innovation through project TIN2009-14009-C02-01 is also acknowledged. Mikel Forcada is supported by Science Foundation Ireland (SFI) through ETS Walton Award 07/W.1/I1802. Andy Way acknowledges support from SFI through grants 05/IN/1732 and 06/RF/CMS064.

References

[1] C. Armentano-Oller, R. C. Carrasco, A. M. Corbí-Bellot, M. L. Forcada, M. Ginestí-Rosell, S. Ortiz-Rojas, J. A. Pérez-Ortiz, G. Ramírez-Sánchez, F. Sánchez-Martínez, and M. A. Scalco. Open-source Portuguese-Spanish machine translation. In *Computational Processing of the Portuguese Language, Proceedings of the 7th International Workshop on Computational Processing of Written and Spoken Portuguese*, volume 3960 of *Lecture Notes in Computer Science*, pages 50–59. Springer-Verlag, Itatiaia, Brazil, May 2006.

[2] D. Arnold, L. Balkan, S. Meijer, R. Humphreys, and L. Sadler. *Machine translation: An introductory guide*. NCC Blackwell, Oxford, 1994.

[3] F. Bond and S. Shirai. *Recent Advances in Example-Based Machine Translation*, volume 21 of *Text, Speech and Language Technology*, chapter A hybrid rule and example-based method for machine translation. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[4] M. Carl, C. Pease, L. L. Iomdin, and O. Streiter. Towards a dynamic linkage of example-based and rule-based machine translation. *Machine Translation*, 15(3):223–257, 2000.

[5] D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. A practical part-of-speech tagger. In *Third Conference on Applied Natural Language Processing. Proceedings of the Conference*, pages 133–140, Trento, Italy, 1992.

[6] J. Du, Y. He, S. Penkale, and A. Way. MATREX: the DCU MT system for WMT 2009. In *Proceedings of the Fourth Workshop on Statistical Machine Translation, EACL 2009*, pages 95–99, Athens, Greece, 2009.

[7] L. Dugast, J. Senellart, and P. Koehn. Statistical post-editing on SYSTRAN's rule-based translation system. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 220–223, Prague, Czech Republic, 2007.

[8] A. Eisele, C. Federmann, H. Uszkoreit, H. Saint-Amand, M. Kay, M. Jellinghaus, S. Hunsicker, T. Herrmann, and Y. Chen. Hybrid machine translation architectures within and beyond the EuroMatrix project. In *Proceedings of the 12th annual conference of the European Association for Machine Translation (EAMT 2008)*, pages 27–34, Hamburg, Germany, September 2008.

[9] N. Gough and A. Way. Robust large-scale EBMT with marker-based segmentation. In *Proceedings of the Tenth Conference on Theoretical and Methodological Issues in Machine Translation (TMI-04)*, pages 95–104, Baltimore, MD., 2004.

[10] T.R.G. Green. The necessity of syntax markers: Two experiments with artificial languages. *Journal of Verbal Learning and Verbal Behaviour*, 18(4):481–496, 1979.

[11] W. J. Hutchins and H. L. Somers. *An Introduction to Machine Translation*. Academic Press, London, 1992.

[12] P. Koehn. Statistical significance tests for machine translation. In *2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 388–395, Barcelona, Spain, 2004.

[13] 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 the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, 2007.

[14] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[15] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA., July 2002.

[16] M. Simard, N. Ueffing, P. Isabelle, and R. Kuhn. Rule-based translation with statistical phrase-based post-editing. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 203–206, Prague, Czech Republic, 2007.

[17] A. Stolcke. SRILM – an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904, Denver, CO., June 2002.

[18] J. Tinsley, Y. Ma, S. Ozdowska, and A. Way. MATREX: the DCU MT system for WMT 2008. In *Proceedings of the Third Workshop on Statistical Machine Translation, ACL 2008*, pages 171–174, Columbus, OH., 2008.