MT Summit X

Second Workshop on Example-Based Machine Translation

Proceedings of the Workshop
16 September 2005
Phuket, Thailand

Co-chairs

- Michael Carl, IAI, Saarbrücken, Germany
- Andy Way, School of Computing, Dublin City University, Ireland

Programme Committee

- Sivaji Bandyopadhyay, Jadavpur University, Kolkata, India.
- Ralf Brown, Carnegie Mellon University, Pittsburgh, US.
- Ilyas Cicekli, Bilkent University, Ankara, Turkey.
- Walter Daelemans, University of Antwerp, Belgium.
- Mary Hearne, Dublin City University, Ireland.
- Eduard Hovy, ISI, University of Southern California, US.
- Sadao Kurohashi, University of Tokyo, Japan.
- Philippe Langlais, RALI, University of Montreal, Canada.
- Yves Lepage, ATR, Kyoto, Japan.
- Stella Markantonatou, ILSP, Athens, Greece.
- Stephen Richardson, Microsoft, US.
- Paul Schmidt, IAI, Saarbrücken
- Satoshi Shirai, NTT, Kawasaki, Japan.
- Michel Simard, Xerox Research Center Europe, Grenoble, France.
- Harold Somers, University of Manchester, UK.
- Oliver Streiter, National University of Kaohsiung, Taiwan.
- Eiichiro Sumita, ATR, Kyoto, Japan.
- Dekai Wu, HKUST, Hong Kong.

MT Summit X
The 10th Machine Translation Summit
September 12-16, 2005 : Phuket, Thailand

Table of Contents

MT Summit X
The 10th Machine Translation Summit
September 12-16, 2005 : Phuket, Thailand

Intoduction

This is the second workshop on Example-based Machine Translation (EBMT) of its kind to be hosted by the MT Summit X, 2005 in Phuket, Thailand. The 1st EBMT workshop took place in 2001 at the MT Summit VIII in Santiago de Compostela, Spain. While the first workshop resulted in a book "Recent Advances in Example-Based Machine Translation" (2003) which summarises the numerous techniques used in our field and helped bring EBMT to a new audience, we also hope to be able to publish these workshop proceedings such that it is available to a wider audience.

Four years after the first EBMT workshop, the field has considerably matured and evolved. In this second workshop we see a continuation of previous systems and approaches as well as a number of new and innovative methods and applications.

We have interesting papers on semantic and type-driven approaches to EBMT, approaches that look beyond the sentence border, retrieval of fragments, example-based sign-language translation, corpus-based generation, and investigations on data assembly and corpus consistency. Projects such as METIS investigate example-based methods to machine translation that make use of a monolingual TL corpus. In addition to these 'new' horisons, the workshop also presents progression of work and approaches alread known from the the previous workshop. We have excellent papers on tree-based approaches, pure as well as template-driven EBMT, and as in the 2001 workshop, position papers on what actually constitutes an EBMT system. In general we observe a continuing desire for the integration of various techniques and and a strengthening of the statistical underpinning on which EBMT is based.

Given the variety of different topics and methods assembled in this workshop we find that research in EBMT is vibrant and catalyzes an active research community trying to integrate and make sense out of the various corpus-driven approaches to MT. In particular, as co-organisers of this 2nd workshop, we are delighted to welcome a number of researchers who haven' previously published in the area of EBMT. We intend to close the workshop by a panel session where the contributors are asked to envisage what EBMT will be like in the future, and what research directions might be foreseen in the time between now and the 3rd EBMT workshop, and beyond.

We trust that you enjoy this workshop. We have enjoyed putting the programme together, and we wish to offer many heartfelt thanks to our assembled Programme Committee, each of whom did sterling work over and above what might reaonable have been expected from them. The quality of the papers received was very high, and if possible the quality of the reviews even higher!

Andy & Michael.

An n-gram approach to exploiting a monolingual corpus for Machine Translation

Toni Badia, Gemma Boleda, Maite Melero, Antoni Oliver
Universitat Pompeu Fabra
Passeig de Circumval·lació, 8 - 08003 Barcelona
{toni.badia,gemma.boleda,maite.melero,antonio.oliver}@upf.edu

Abstract

In this paper we present an approach to Statistical Machine Translation that uses a bilingual dictionary and a target language model based on n-grams extracted from a monolingual corpus. This approach is still in an experimental stage and is being developed in the context of Metis-II, a UE project that aims at constructing free text translations by retrieving the basic stock for translations from large monolingual corpora. The architecture described in this paper is being applied to translation from Spanish to English and is designed so as to depend as little as possible on complex linguistic processing tools. The only required tools are a POS tagger and lemmatizer for the source language, and another for the target language.

1 Introduction

Corpus-Based Machine Translation (MT), including Statistical Machine Translation (SMT) and Example-Based Machine Translation (EBMT), use bilingual parallel corpora to train translation models. SMT is based on probability theory (Yamada and Knight, 2001); EBMT, on the other hand, is inspired by analogical reasoning: every new translation is computed in analogy to already known translations extracted from a bilingual corpus (Carl and Way, 2003). This approach basically relies on finding translated maximal-length phrases that combine to form a translation.

One basic pre-requisite for Corpus-Based Machine Translation is the existence of adequate bilingual parallel corpora, which may be difficult to acquire, even for widely spoken languages, let alone minority languages. Considering that for statistical systems one of the best ways to improve the results is by using a larger corpus (Banko and Brill, 2001), difficulty to acquire parallel corpora is a major drawback.

Another factor worth taking into consideration is the fact that the existing parallel corpora often belong to a very limited number of domains, such as parliamentary debates like the Hansards (debates of the Canadian Parliament) or Europarl (minutes from the European Parliament; Koehn (2002)).

On the other hand, the availability of monolingual corpora in digital format, belonging to a large variety of domains, keeps growing for all languages.

Given this scenario, the goal of the project Metis-II is to achieve corpus-based translation on the basis of a monolingual target language corpus and a bilingual dictionary only. The project aims at building a translation system for Dutch, German, Greek, and Spanish to English, using the British National Corpus (BNC; Burnard, (1995)) as the monolingual target language corpus.

Metis-II was preceded by Metis-I, which operated on a sentence-level base (Dologlou et al., 2003). Using a bilingual dictionary and some reordering rules, a near word-by-word translation was produced from the source sentence. The target corpus was then searched for the closest match, which was then proposed as the best known translation for the source sentence. Even though the performance of Metis-I was superior to that of a Translation Memory (built using a more expensive resource, namely a parallel corpus), it was clearly limited by the size of its base unit: the sentence.

Metis-II aims at improving on the results of the approach initiated by Metis-I by using segments below the sentence level. Since finding the exact match of a sentence is too strict a requirement, the sentence has to be decomposed into some kind of constituents, in order to perform a partial match. Proposals about how to decompose example sentences abound in the literature on EBMT (Turcato and Popowich, 2001). In most cases, some sort of linguistic analysis is used, from the most low-level to the most deep-level, e.g. clustering methods for

chunking (Brown, 2003), shallow parsing for extraction of translation units (Carl, 2003), use of dependency trees (Watanabe et al., 2003), logical forms or predicate-argument structures in the Microsoft Research MT system (Richardson et al., 2001), etc. The idea behind all these proposals is that examples can be decomposed into smaller constituents to be processed independently. Every approach addresses in one way or other the two main problems of decomposition, namely "boundary definition", i.e. where to segment, and "boundary friction", i.e. how to stitch together the translated pieces.

Different approaches to decomposition and reuse of the material are currently being explored within the Metis-II Consortium. We next explain the approach that is being explored by GLiCom[1], which uses n-grams as base units.

2 The n-gram approach

Statistical MT systems typically consist of a translation model and a target language model (Brown et al., 1993). In our case, the bilingual dictionary functions as a lexical translation model and we only need to compute the target language model, out of the target language corpus.

2.1 Linguistic pre-processing

In our approach n-grams are not built out of words, as it is usually the case in SMT systems, but out of lemmas and/or morphological tags. This implies that both the target corpus and the input sentences have to be lemmatized and tagged. In addition to providing a more generalized representation of the corpus, to avoid data sparseness, this representation has the advantage that it can be directly used in the dictionary lookup: typical machine readable bilingual dictionaries are lemma-to-lemma, so that they need a lemmatized input and provide a lemmatized output.

In order to process the Spanish input, we use a morphological analyzer called KURD (Carl and Schmidt-Wigger, 1998). KURD is a constraint-based formalism that works on the basis of a pattern matching approach that is suitable for shallow or partial linguistic processing. It manipulates morphological analysis in order to kill, unify, replace or delete parts of the structure. The result of the pre-processing with KURD yields a disambiguated morphological analysis that can be fed into the lemma-to-lemma bilingual dictionary.

2.2 The bilingual dictionary

The bilingual lexicon that we use is based on a commercial machine-readable dictionary, the Concise Oxford Spanish (Rollin, 1998), which has 32,653 entries in the Spanish-English direction with an average of 4 translations per headword.

The coverage is being enlarged, using automatic procedures, with entries coming from the reverse direction (English-Spanish) as well as from terminological glossaries. Orthographic and regional variants, such as British and American spellings are also being added, as well as compounds, that appear in the original dictionary as secondary entries under the main headword.

Lemma to lemma translations are automatically extracted from the machine readable dictionary such that mapping from source to target is always one-to-one. Because of simplicity of design, identical headwords with different translations constitute different entries. Likewise, identical headwords with different parts of speech constitute different entries even if the translation is the same. The structure of the resulting entries looks as follows:

- Entry identifier
- Spanish headword (lemma)
- POS of the Spanish headword (PAROLE/EAGLES tag set)[2]
- English translation (lemma)
- POS of the English translation (CLAWS5 tag set)[3]

In order to build our dictionary, we need to calculate the POS of the translation, which is not present in the original Spanish-English dictionary. This POS is automatically assigned on the basis of the POS of the source word and is subsequently validated on the English-Spanish dictionary and other sources, like the target corpus itself. In the few cases where the POS of the target does not coincide with the POS of the source, the validation will overwrite the default. The value of the POS is expressed using the CLAWS5 tag set, which is the same tag set used to tag the BNC.

[1] Computational Linguistics Group of the Universitat Pompeu Fabra

[2] http://www.lsi.upc.es/~nlp/freeling/parole-es.html

[3] http://www.comp.lancs.ac.uk/computing/research/ucrel/

The machine readable dictionary from which our dictionary is extracted provides other types of lexical information as well, such as collocations, sense indicators[4], field labels, examples, etc. In future enhancements of the system, we are considering exploiting part of this information, particularly collocations, in the translation process.

2.3 The language model

The target corpus that we use to validate the translations coming from the dictionary is a lemmatized version of the BNC. In a first step all n-grams are sequences of lemmas. In a second step, one -and just one- of the lemmas of a given n-gram is substituted by its POS tag. This is done for every lemma in the n-gram, one lemma at a time[5]. Here is an example of a 4-gram with tag substitution: *inconvenient on those occasions*:

inconvenient on this occasion
AJ0 on this occasion
inconvenient PRP this occasion
inconvenient on DT0 occasion
inconvenient on this NN1

This is repeated for the tri-grams and bi-grams contained in the 4-gram. The last model to be built is the unigram model. This model, which does not provide contextual information, is nevertheless used as a frequency measure for single words. If no other evidence is found, at least the most frequent word is chosen as translation. The purpose of the target language model within our architecture is twofold:

- Perform lexical selection: i.e. select one translation out of the possible candidates provided by the bilingual lexicon
- Build the sentence structure: i.e. select one of the possible orderings of the tokens within the n-grams, as well as among n-grams

2.4 The translation process

Once the source sentence has been tokenized, tagged and lemmatized, it goes through the following steps:

1. Every lemma in the source sentence is matched against the left side of the bilingual dictionary. Part-of-speech information obtained from the tagger is used to guide lexicon look-up in order to disambiguate between homonymous words, i.e. words with the same lemma but different category. Other morphosyntactic values such as tense or number are not used at this point but are stored and will be consulted at the end of the process in order to generate the right inflected form in the target language. If a source word (i.e. lemma) is not found in the bilingual lexicon, the word is left untranslated.
2. All possible n-grams are built out of the sequence of translated lemmas, starting with the highest value of n (i.e. n=4). A different n-gram is built for each translation possibility. For instance, in the sentence *el niño pequeño come carne* 'the little child eats meat', if *carne* is translated as *meat* or *flesh*, both *the child little eat meat* and *the child little eat flesh* are built.
3. At this point, a reduced set of hand-written mapping rules may need to apply in order to deal with specific phenomena. These rules are an ad-hoc mechanism apt to deal with hard translation problems, such as thematic role inversion (e.g like - gustar) and other structure changing issues. However, for alternatives to the use of mapping rules, see section 3.2.
4. Validation of the translated n-grams proceeds. Based on the frequency in the target language corpus and the length (i.e. the value of n) of the n-gram, a weight is assigned to each candidate. In the case no evidence is found for a given n-gram formed by lemmas, the process is repeated by successively substituting one lemma by its tag. This substitution affects negatively the weight of the resulting n-gram.
5. When all calculations have been done, the n-grams with the highest weights are kept as translation candidates.
6. The n-grams of the portions which have not yet been validated by the model are recalculated, and steps 2-5 are repeated with n = n-1, until all portions are calculated or n = 1. The portions that are validated at a particular stage of the process are not further taken into consideration.

[4] These may be near synonyms or guiding words or explanations.

[5] Except in the case when there is a proper noun or a cardinal number in the n-gram, in which case, we may find more than one tag: the one that is being substituted, plus the tag for the proper noun (NP0) or the tag for the number (CRD)

7. Any POS tag left in the final string, different from *cardinal* or *proper noun*, is replaced by the most frequent translation of the original word according to the unigram model. If none of the proposed translations appear in the target corpus, the first translation provided by the lexicon is then chosen. Tokens tagged as *cardinal* or *proper noun* are replaced by the original word.

3 Dealing with changes of structure

Translations that imply changes of structure, going from source to target, are among the main difficulties of using a bilingual lexicon, and not a true translation model. These changes of structure can be reduced to:

- Insertions.
- Deletions.
- Movements: local and non-local.

Although a small set of hand-written mapping rules can be advisable for some phenomena, and is indeed foreseen in the general Metis architecture, they cannot be the only device to deal with changes in structure, if the system is to be robust and scalable. More generally, we plan to use our target language model to perform these changes.

By allowing reordering of elements, plus deletions and insertions, the combination of possibilities in the search algorithm explodes. In order to limit the search space in a linguistically principled way, we intend to use the information provided by the POS tagger to distinguish between *content words* and *grammatical words*. The idea is to limit (local) movement to content words, and possibility of insertion or deletion to grammatical words.

3.1 Insertions and deletions

The following parts-of-speech are considered to be *grammatical words*: articles, conjunctions, determiners, pronouns, prepositions and, specific to English, the existential *there* and the infinitive marker *to*.

We assume that insertion or deletion affect only grammatical words. These words function as true markers, in the same way as morphological inflection does and therefore, very often, only appear in the source or in the target, but not in both. The following are common examples of this phenomenon:

(1) Dormían en un coche
sleep-PAST-P3-PLR in a car
'THEY slept in a car'

(2) La policía detuvo a
the police arrest-PAST-P3-SNG TO
un sospechoso
a suspect
'The police arrested a suspect'

(3) Los perros ladran
THE dogs bark-PRES-P3-PLR
'Dogs bark'

Example 1 illustrates a case of pro-drop, i.e. absence of explicit subject. This is a common phenomenon in Spanish. The subject pronoun, on the other hand, is obligatory in English and needs to be inserted.

In example 2 the Spanish sentence contains an *a* preposition which functions as a Direct Object marker and must not appear in the English version. Likewise, example 3 illustrates differences in the use of articles in the two languages: generic sentences in English require bare plural nouns while in Spanish the definite article is obligatory.

The way the search algorithm described in 2.4 is intended to deal with insertion and deletion is that the presence or absence of grammatical words does not hinder n-gram matching. Grammatical words are part of the model but they function as if they were effectively *invisible* in the same way that inflection is generally not used when searching for an n-gram candidate.

3.2 Local movements

We distinguish between local and non-local movements. Local movements are changes in the order of individual words that occur inside a linguistic constituent, such as an NP. Non-local movements affect reordering of constituents in the sentence. We address non-local movements in 3.3.

As stated above, only content words are allowed to move. Major categories, such as nouns, verbs, adjectives and adverbs are considered to be *content words*. As a way of example, let us look at the reordering of adjectives inside an NP.

(4) la guerra civil española
the war civil Spanish
'the Spanish Civil war'

Reordering of translated adjectives in example 4 would require in traditional linguistic MT systems information about scope and/or type of the adjectives and position in the source sentence. In a statistical system such as ours, the adjectives, together with the noun, are freely allowed to move, thus expanding the n-gram set. The correct order is eventually validated by the target language model. In contrast, the determiner (being a grammatical word) is not allowed to move.

Certainly, we do not want the adjectives to move *outside* of the boundaries of the NP. How do we achieve such a restriction, considering that we are not using any kind of parser or chunker, but only a POS tagger?

In order to detect linguistically significant constituents, we mirror a chunking procedure in which we pre-define phrase boundary markers. For instance, *Det* is a boundary marker, and so is *Verb-FIN* and *Prep*. Content words are only allowed to move inside two consecutive boundaries.

Another example of what can be achieved by our approach is the translation of noun complements, which in Spanish tend to appear after the head noun, preceded by a *de* preposition, and in English appear as a noun pre-modifying the head. Example 5 is an illustration of this, which includes both reordering and deletion.

(5) un regalo de cumpleaños
a present of birthday
'a birthday present'

3.3 Non-local movements

The procedure described in the previous section is insufficient when changes in order are not local but affect sentence constituents. This happens only occasionally when translating from Spanish to English, e.g. different position of the adverb, subject inversion, etc., but is particularly frequent when going from German to English.

For instance, sentence 6 would not be correctly handled by our system, such as it has been described so far:

(6) In dem Garten isst der
In the-DAT garden eats the-NOM
junge Mann
young man
'The young man eats in the garden'

In German, the finite verb in main clauses must always be in second position, regardless of which kind of constituent occurs in the first position of the clause. In example 6, a locative adjunct (marked in dative case) occurs in first position, and the subject (marked in nominative case) after the verb. This order needs to be reversed in the translation (or at least the subject has to be placed before the verb).

To handle non-local order changes, we propose creating a "second-level language model" apart from the token level language model described in Section 2.3. This is an n-gram model over *sequences of tags*. The tags in this model are complex tags of the type: *DetAdjNoun*. Sequences of tags are limited by the same type of boundaries described in Section 3.2. In this way, the 'second-level' language model gives us a parser-free representation of the syntactic patterns of the target language.

Boundary detection is performed on the output of the lemma-to-lemma dictionary look-up. The result of boundary detection on the lemma-to-lemma translation of the original sentence in example 6 is shown in 7, where * marks the boundaries:

(7) In the garden eats the
*Prep Det Noun *Verb-FIN *Det
young man
Adj Noun

This sequence of complex tags would then be checked against the 'second-level' or 'syntactic model', yielding as a result the most frequent of all possible permutations, in our example, (8c).

(8) a. *PrepDetNoun *Verb-FIN *DetAdjNoun
b. *DetAdjNoun *PrepDetNoun *Verb-FIN
c. *DetAdjNoun *Verb-FIN *PrepDetNoun

As a further enhancement, lexical information could be introduced into this model, most prominently verbal lemmas. In this way, subcategorization information could be taken into account, defining syntactic patterns over verb types. Without this information, interference between different subcategorization frames could bring noise into the model, for example, intransitive structures would give wrong models for transitive verbs. If this solution makes the model too sparse, an alternative would be to build the model with clusters (Resnik, 1993).

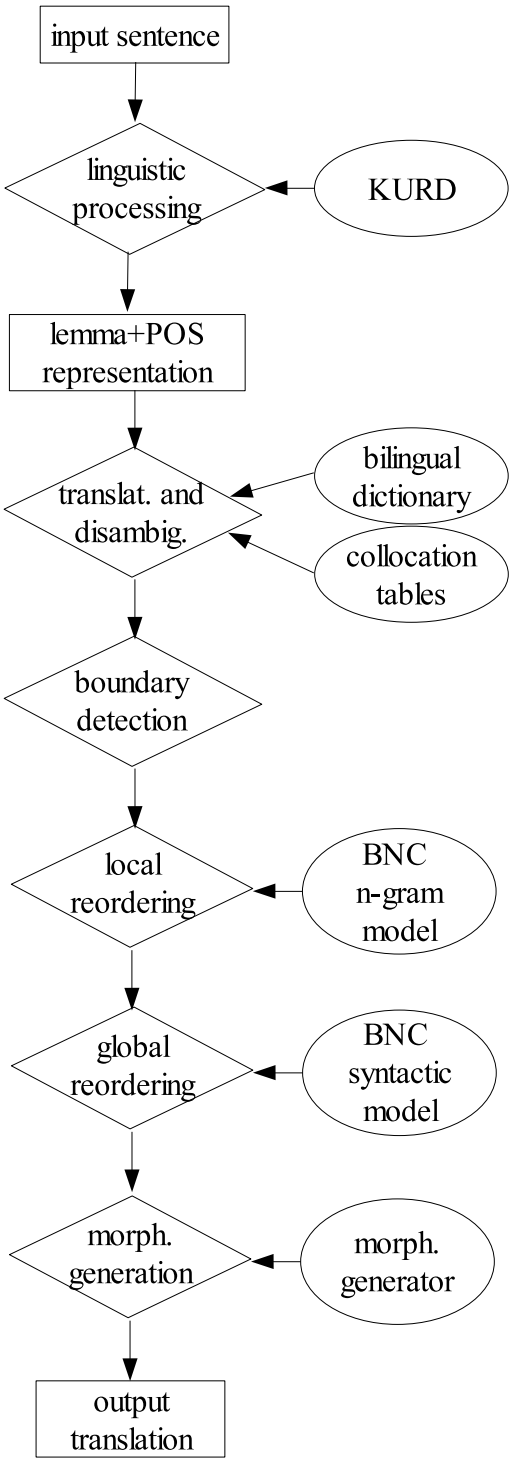


Figure 1: Proposed architecture of the Spanish-English n-gram based translation system

4 Use of a table of collocations to optimize lexical selection

To choose among different translations of a word, or at least to discard many translations, less information than a full language model is arguably needed, if a collocation module is built that exploits the information in the target corpus relative to selectional restrictions. A similar approach, using co-occurrence statistics in a target reference corpus, has been exploited in cross-language information retrieval by (Qu et al., 2002).

We plan to extract from the corpus a table of collocations for certain pairs of POS: Verb+Noun, Adj+Noun, Noun+Noun, and Verb+Adv.[6] The frequencies of the lemma pairs in an n-word window will be collected, associated with one of several possible measures for collocation detection (Evert and Krenn, 2001), and stored in a table.

Our goal is not to just store collocations, but to more generally model selectional restrictions. In cases such as the example *el niño pequeño come carne* cited above, the pair *eat-meat* will presumably have a higher score than *eat-flesh*, so that *flesh* does not need to enter into the n-gram building process. In this way we expect to help discard some of the lexical combinations resulting from the dictionary look-up, prior to actually doing the n-gram search on the model. In cases where the collocation table does not provide enough evidence, the remaining translations can still be validated with the general translation algorithm.

5 Conclusions

In this paper we have presented an experiment, which is being carried out in the context of Metis-II, to translate from Spanish to English using very basic linguistic resources, namely a POS tagger and lemmatizer for Spanish, a machine readable bilingual dictionary and the tagged and lemmatized version of the British National Corpus. Its architecture, as shown in fig 1, is thus translatable to languages with very little NLP development.

The target corpus is the basis both for lexical selection (selecting among the different translations found in the dictionary) and for structure construction (allowing for both local and global changes in structure). To that end, the building and exploitation of the following models will be explored:

- an n-gram language model over lemma and tag tokens. This model should allow for an efficient treatment of common structural changes in translation, involving insertion,

[6] For simplicity reasons we will only encode binary collocations, at least in a first step.

deletion and local movement. This treatment will make crucial use of the distinction between grammatical and content words, provided by the POS tagger;

- a syntactic model over sequences of tags within sentences, as a representation of the syntactic patterns of the target language, to deal with global movement;
- a collocation table to account for selectional restrictions.

The use of the models as explained in this paper has been designed to dispense with explicit mapping rules, or at least keep them to a really minimal set. If these models can be conveniently exploited, it would be an enormous boost to the scalability and robustness of the system.

The implemented version of the system is still too immature to perform a meaningful evaluation. However, we have discussed promising lines of research to build a full-fledged system which can eventually be evaluated analogously to other MT systems.

References

M. Banko and E. Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of 39th Annual Meeting of ACL*, pages 26–33, Toulouse, France.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter stimation. *Computational Linguistics*, 19(2):263–311.

Ralf Brown. 2003. Clustered transfer rule induction for example-based translation. In Michael Carl and Andy Way, editors, *Recent Advances in Example-Based Machine Translation*, pages 287–306. Kluwer Academic Publishers.

L. Burnard. 1995. *Users reference guide for the British National Corpus.* OUCS.

M. Carl and A. Schmidt-Wigger. 1998. Shallow post morphological processing with kurd. In *Proceedings of NeMLaP'98*, pages 5–11, Sydney, Australia.

M. Carl and A. Way. 2003. *Recent Advances in Example-Based Machine Translation.* Kluwer Academic Publishers.

Michael Carl. 2003. Inducing translation grammars from bracketed alignments. In Michael Carl and Andy Way, editors, *Recent Advances in Example-Based Machine Translation*, pages 339–364. Kluwer Academic Publishers.

Yannis Dologlou, Stella Markantonatou, George Tambouratzis, Olga Yannoutsou, Athanassia Fourla, and Nikos Ioannou. 2003. Using monolingual corpora for statistical machine translation: The metis system. In *Proceedings of the EAMT-CLAW 03:Controlled Language Translation*, pages 61–68, Dublin City University, Dublin, Ireland.

Stefan Evert and Brigitte Krenn. 2001. Methods for the qualitative evaluation of lexical association measures. In *Proceedings of the 39th ACL*, pages 188–195.

P. Koehn. 2002. *Europarl: a multilingual corpus for evaluation of machine translation.* Unpublished.

Y. Qu, G. Gefenstette, and D. A. Evans. 2002. Resolving translation ambiguity using monolingual corpora. In *Working Notes for the CLEF 2002 Workshop*, pages 115–126.

Philip Resnik. 1993. *Selection and Information: A Class-Based Approach to Lexical Relationships.* Ph.D. thesis, University of Pennsylvania.

S. Richardson, W. Dolan, A. Menezes, and J. Pinkham. 2001. Achieving commercial-quality translation with example-based methods. In *Proceedings of the Machine Translation Summit VIII*, pages 293–298, Santiago de Compostela, Spain.

Nicholas Rollin. 1998. *The Concise Oxford Spanish Dictionary.* Oxford University Press.

D. Turcato and F. Popowich. 2001. What is example-based machine translation? In *Proceedings of the Machine Translation Summit VIII*, Santiago de Compostela, Spain.

Hideo Watanabe, Satoshi Kurohshi, and Eiji Aramaki. 2003. Finding translation patterns from dependency structures. In Michael Carl and Andy Way, editors, *Recent Advances in Example-Based Machine Translation*, pages 287–306. Kluwer Academic Publishers.

K. Yamada and K. Knight. 2001. A syntax-based statistical translation model. In *Proceedings of 39th Annual Meeting of ACL*, pages 5–11, Toulouse, France.

Context-Sensitive Retrieval for Example-Based Translation

Ralf D. Brown
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
ralf+@cs.cmu.edu

Abstract

Example-Based Machine Translation (EBMT) systems have typically operated on individual sentences without taking into account prior context. By adding a simple reweighting of retrieved fragments of training examples on the basis of whether the previous translation retrieved any fragments from examples within a small window of the current instance, translation performance is improved. A further improvement is seen by performing a similar reweighting when another fragment of the current input sentence was retrieved from the same training example. Together, a simple, straightforward implementation of these two factors results in an improvement on the order of 1.0–1.6% in the BLEU metric across multiple data sets in multiple languages.

1 Introduction

While context has long been recognized as an important factor in translating texts, it tends to be given lower priority in machine translation system development than improving the quality of isolated translations. Quality can only be improved so far, however, when operating strictly on isolated sentences, and thus further improvements must eventually be sought by taking other sentences into account when performing a translation.

EBMT systems typically treat both training data and the input to be translated as bags of unrelated sentences, though in practice, consecutive sentences are in fact related. Rather than consisting of random sentences, the training data consists of a set of coherent documents, and the input to be translated is one or more documents. In particular, retrieval is done without regard to the results of the prior sentence's translation, and thus differing word senses receive equal weighting. In contrast, by considering whether the previous sentence that was translated used adjacent sentences in the training corpus, the appropriate word sense can be given more import in the final translation, based on the old idea of "one sense per discourse" (Gale et al., 1992). A similar idea of temporal coherence in the use of word senses is used in speech recognition in the form of trigger or cache models for disambiguating homophones.

Figure 1 shows an example of using context to select the appropriate word sense for a translation. The training material includes examples for three senses of the word "bank", two of which produce equally-long matches between the training data and the second sentence of the test input. Without using context, the system can't distinguish between those two matches (which would generate "Ufer" and "Bank" in German, for example). However, by giving a bonus to the match where a nearby training instance was used in generating the first sentence's translation, the hypothesis with the correct "financial institution" sense can be given priority in generating the overall translation.

Similarly, for an EBMT system which uses partial matches of training examples (either explicitly partial matching as in (Brown, 1996; Brown, 2001; Brown, 2004) or complete matches of training instances which may be fragments of the original example sentences as in (Veale and Way, 1997; Gough and Way, 2003)), having multiple matches between the test input and a single training sentence increases confidence in the correctness of *all* matches in that sentence.

The next two sections of this paper describe the implementation of these two simple approaches to taking advantage of context.

2 Local Context

The EBMT system (Brown, 1996; Brown, 2004) used for the experiments described in this paper retrieves contiguous fragments from the training corpus which exactly match portions of the input to be

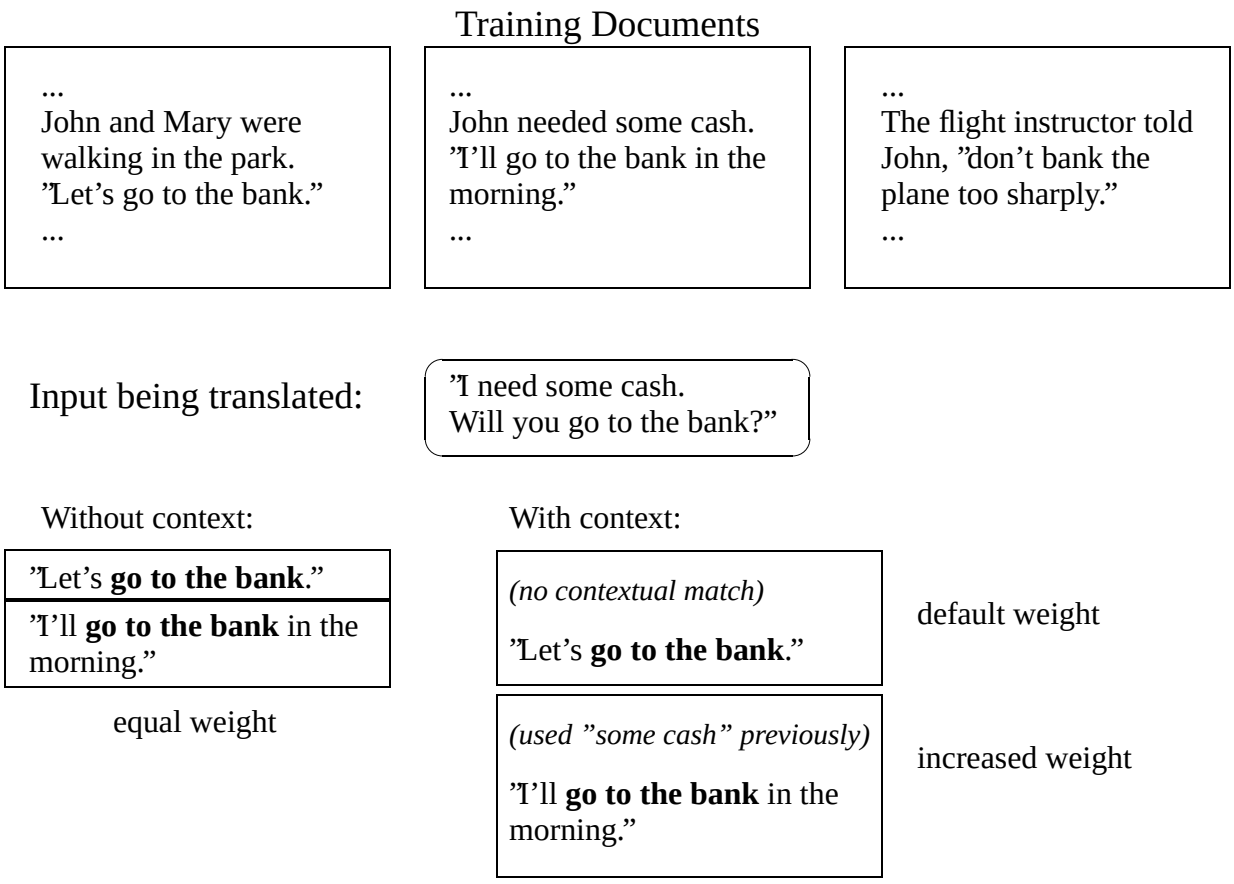


Figure 1: Adjacent sentences affect quality of the retrieved examples

translated[1]. Thus, if a new sentence is largely the same as a training example but contains a section which differs, two (or more) fragments will be retrieved from that example. Clearly, two fragments retrieved from a single example are better than the same fragments retrieved from two different examples (Figure 2). Thus, the translation hypothesis generated by a retrieved partial example should be given greater weight if other fragments of the input text occur in the same training example.

Further, since the system retrieves *every* phrasal match, whenever it finds e.g. a four-gram match, the trigrams and bigrams contained within it contribute to the pool of examples for determining the candidate translations of those trigrams and bigrams. However, the initial implementation did not take advantage of the fact that such contained instances are more reliable because they occur in an appropriate context, while n-gram instances which are not contained within a longer match do not have the same context as the phrase in the test input.

Thus, *local context* can guide the selection of appropriate translation hypotheses by boosting the weight given to a retrieved match whenever other matches of the current input sentence occur within the same training example. For ease of implementation, the initial version of local context weighting uses a greedy one-pass approach rather than separate passes to collect statistics and weight retrieved examples. As a result, some matches receive less of a boost than they should, but the overall impact is expected to be fairly small. By far the most frequent recipients of a bonus are bigrams contained within larger matches, but many of them are never actually processed because (for speed reasons) the EBMT system only examines up to a maximum number of matches for any particular n-gram of the input, typically 1000 or 1500.

Differential weighting based on the local con-

[1]Or exactly match all or a portion of a generalized template formed from the input, but that feature was not used for the experiments described here.

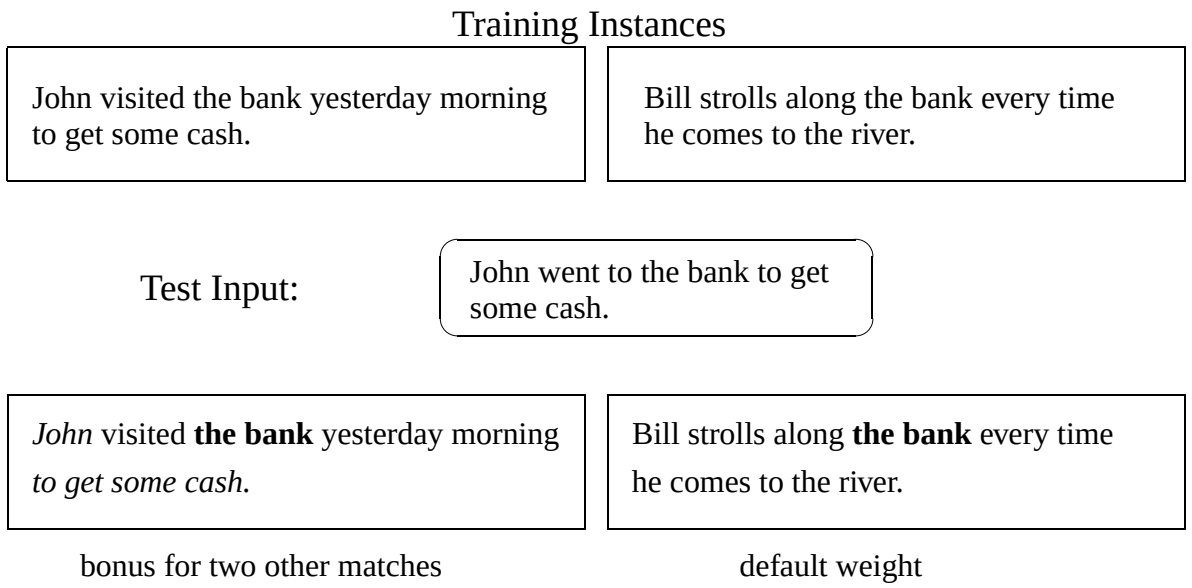


Figure 2: The quality of retrieved fragments varies by relative location.

text was implemented as an extension to an existing differential-weighting mechanism. Each retrieved instance receives a weight based on a combination of the source of the training data and its proportional location in the corpus. For example, when translating newswire texts, newswire training data could receive a weight of 3.0 and parliamentary proceedings a weight of 1.0; and when translating current texts using a corpus gathered over a long period of time, the earliest example could receive a weight of 1.0, linearly increasing to 2.0 for the most recent example in the corpus (all of these weights are configurable). When computing the confidence score for each distinct candidate translation, a weighted sum of all the retrieved instances is used to compute a translation probability, which forms the bulk of the quality score (the highest alignment-confidence score for any instance generating a particular translation forms the remainder of the score). Thus, increasing the weight of a training example increases the translation probability and hence the overall confidence score assigned to the associated translation. This causes a re-ranking of the translation hypotheses for a particular source phrase, and can result in a different set of hypotheses being output whenever there are more distinct translation hypotheses than the system has been configured to produce.

To compute the local context bonus assigned to a retrieved training instance, an array is used to keep counts of all retrievals from each training example in the corpus. The counts are initialized to zero and incremented each time a match from the associated training example is accessed. The base weight of the instance (as described in the previous paragraph) is multiplied by one plus a configurable bonus factor times the total access count. A fairly large bonus factor, typically on the order of 10, is required to counteract the sheer number of other matches which do not receive a bonus and thereby produce a substantive shift in the relative weighting of different translation alternatives. The matches found by examining the index are processed in order from longest to shortest, so a short match contained within a longer one automatically receives a local context bonus. Because a one-pass algorithm was implemented, only the second and subsequent disjoint fragments matching a given training instance will receive a bonus; the first fragment processed will not.

3 Inter-Sentential Context

As mentioned in the introduction, EBMT systems typically treat both training data and the input to be translated as bags of unrelated sentences. But in practice, consecutive sentences are in fact related – the training data consists of a set of coherent documents, and the input to be translated is one or more documents rather than random sentences.

Given the implementation of the local context

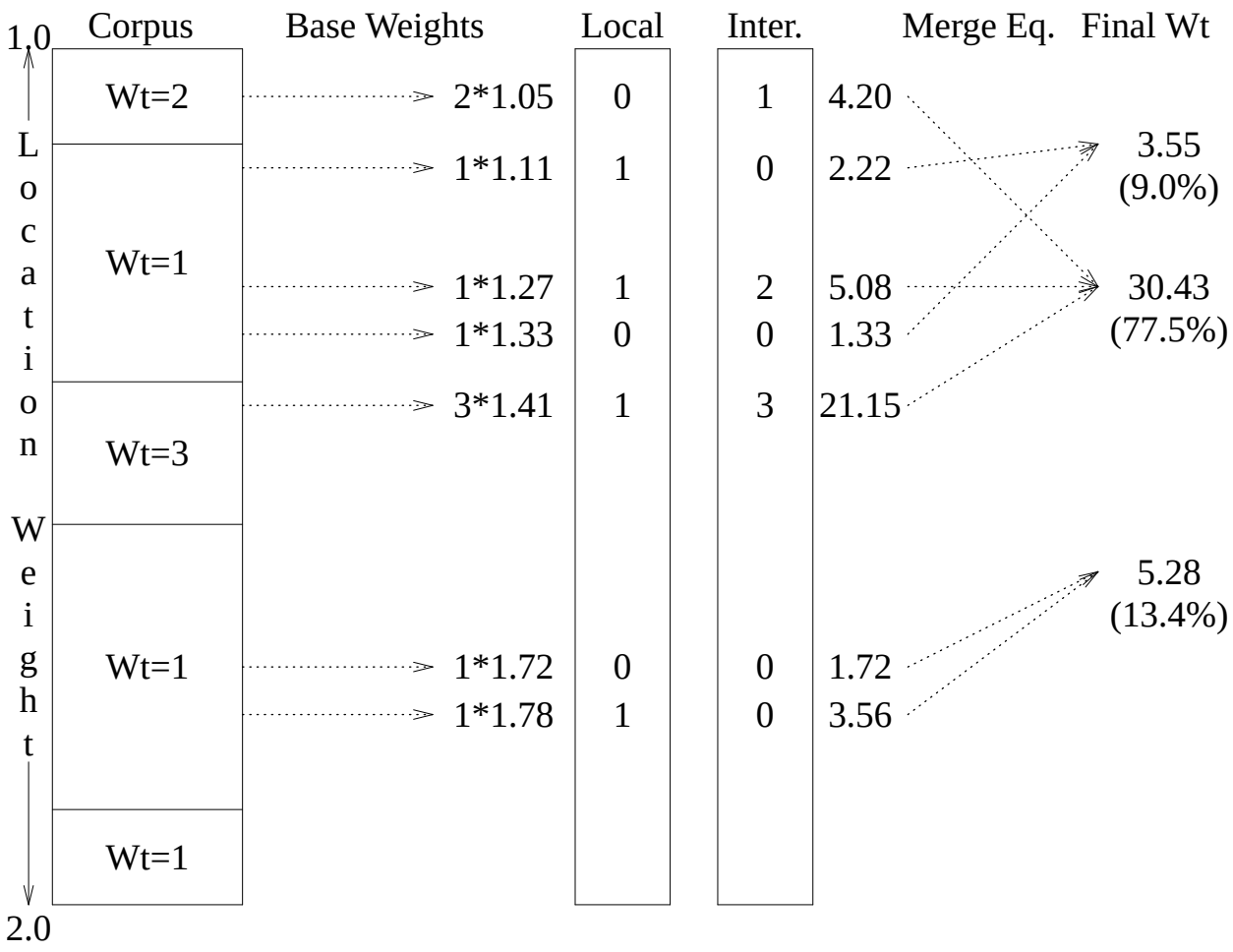


Figure 3: Computing weighted translation probabilites with context bonuses

mechanism described in the previous section, implementation of intersentential context bonuses is very simple: rather than discarding the usage counts after translating an input sentence, they are retained for the following translation, separately from the new local context counts. During the second sentence's translation, the counts within a selected range around each retrieved instance are consulted. The intersentential context bonus is then the weighted sum of the counts within the selected range (in the current implementation, the current example plus the five examples before and after it, though the most distant of those five examples generally receive zero weight).

For example, let the bonus weights be set to 10 for the current training example, 5 for the examples immediately adjacent, and 2 for the examples at distance two, e.g. (2 5 10 5 2). The total bonus for an example where the previous example had one match during the prior translation and the example two sentences later in the corpus had two matches would be $(1 * 5) + (2 * 2)$ or 9.

Intersentential context weights are factored into the base weight of a retrieved instance in the same manner as local context weights, making the final weight of each instance the product of its base weight times one plus the sum of its local context bonus and its intersentential context bonus.

The final weight of a translation alternative is the sum of the individual weights of each of the instances which generate that alternative, computed as just described. See Figure 3 for a visual representation of this process.

4 Experimental Design

To determine the efficacy of the two context bonuses, multiple test sets were translated and scored using the BLEU metric under each of four

conditions:

- **baseline**: no context bonuses
- **local**: only local context bonus applied
- **intersent.**: only intersentential context bonus
- **both**: both bonuses applied

Each of the four conditions was separately tuned to determine the best values for several key parameters of the EBMT system (maximum number of hypotheses for a given source phrase, alignment confidence threshold, proportion of confidence score from translation probability, and relative importance of target-language trigram language model). The intent was to show the maximum performance possible for each context bonus and for the combination of the two bonuses to evaluate their potential benefit.

Four language pairs were used: French-English, Spanish-English, Chinese-English, and Romanian-English. For each language pair, two test sets were selected, one on which to tune (producing peak-to-peak comparisons between the experimental conditions), and one as held-out data to estimate real-world performance on unseen test data.

The French-English EBMT system was trained on 20,000 sentence pairs from files 000 and 001 of the IBM Hansard corpus (Linguistic Data Consortium, 1997). The test sets were 100 sentence pairs drawn from file 020 for tuning and 1000 sentence pairs drawn from file 060 for evaluation.

The Spanish-English system was trained on some 700,000 sentence pairs (approximately 22 million words) from the UN Multilingual Corpus, about one-tenth that amount of text from European Parliament proceedings, and a small amount of text from the Pan-American Health Organization. The test sets were 280 and 1389 sentences, respectively, held out from the European Parliament texts.

The Chinese-English system was trained on slightly less than two million sentence pairs drawn primarily from the UN Chinese-English corpus available from the Linguistic Data Consortium. The test sets were the 993-sentence test set from the 2002 DARPA TIDES Machine Translation Evaluation for tuning and the 919-sentence test set from the 2003 MT Evaluation as unseen data, both primarily newswire text.

The Romanian-English system was trained on the parallel corpus provided to participants in the shared word-alignment task for the 2003 and 2005 Workshops on Parallel Text (Mihalcea and Pedersen, 2003), approximately one million words per language. The 2003 test set of 248 sentences was used as the tuning set, and the 2005 test set of 203 sentences was intended for use as the unseen test data. Unfortunately, the latter set proved to consist of sentences drawn from the training corpus, which thus made it unusable without first modifying the training data to remove those sentence pairs (as the EBMT system produced perfect matches for the reference translations regardless of settings). Therefore, only one test set was used for Romanian-English experiments.

We performed significance tests on the experiments using the four test sets of around 1000 sentences (the other three test sets were too small to produce reliable results). To compute the statistical significance of changes in performance, the test set was split into ten approximately equal-sized parts and BLEU scores computed for each part. The two-tailed version of Student's paired t-test was applied to the sets of scores to compute p-values.

The BLEU metric uses a global brevity penalty to partially compensate for its lack of direct recall measurement. Because this penalty more easily becomes substantial with smaller test sets, the average score obtained on a set of smaller files tends to be somewhat lower than the score obtained on the concatenation of those files (where the natural variability in translation lengths tends to be smoothed out). The reduction averaged slightly more than 2 percent over the various combinations of test condition and test set on which the ten-way split was used.

5 Results

For all four language pairs, each of the two classes of context alone and in combination resulted in improved performance when pitted against the original implementation without context awareness (Table 1). The "real-world" performance on previously-unseen data using the optimal parameters determined on the tuning set was rather mixed (Table 2) for intersentential context and the combination of local and intersentential, but local context still provided a statistically significant improvement in two of three cases (statistically-significant differences are shown in boldface in Tables 1 and 2).

Three of the four larger test sets for which significance could be computed achieved statistically significant improvements in BLEU scores. For Spanish-English, there was extremely high variance between the ten slices of the test set (in particular, one slice scored less than half the average, possibly

Language	Test Size	Local	Intersent.	Both
French	100	+0.71%	+0.97%	+1.03%
Chinese	993	+1.36%	+0.58%	**+1.69%**
Romanian	248	+0.86%	+0.79%	+1.44%
Spanish	280	+1.36%	+0.63%	+1.36%

Table 1: Relative Improvements from Using Context (Peak-to-Peak)

Language	Test Size	Local	Intersent.	Both
French	1000	**+1.51%**	+0.33%	-0.26%
Chinese	919	**+0.83%**	-0.33%	+1.08%
Spanish	1389	+1.22%	-0.60%	-0.28%

Table 2: Relative Improvements from Using Context (Unseen Test Data)

due to errors or divergences[2] in the available translation), and thus resulted in a non-significant p-value of 0.20 even for local context.

6 Conclusions

Although very simple, the implementation of local context described in this paper proves to be beneficial in all cases, while the simple implementation of intersentential context is more of a mixed bag in terms of performance. The computation of intersentential context bonuses is probably being affected by document boundaries, which are not being taken into account. Particularly where the original documents are short, such as newswire stories, even a three-sentence window on either side of the current instance has a good chance of including text from another document.

Because the contextual bonuses result in a reranking of hypotheses, it is possible for the local and intersentential bonuses to act against each other. This is likely what happened on the larger French test set, where the two bonuses individually produced improvements in the BLEU score while the combination was actually detrimental.

It is interesting to note that the only language pair on which the combination of local and intersentential contexts improved performance on the unseen data is also the only language pair where the tuning set was itself large enough to perform statistical significance tests. The failure to produce an improvement may therefore simply be a result of tuning sets which were too small to find appropriate parameter settings for the general case, rather than just the limited number of sentences used for tuning.

7 Future Work

As a first, very quick implementation, many enhancements still await implementation and investigation. Two enhancements which have already been mentioned are two-pass calculation of bonuses and consideration of document boundaries. Other, more global, matching is also likely to improve performance.

Two-pass calculation of contextual bonuses will eliminate the cases where the existing one-pass calculation does not give a retrieved instance as much of a context bonus as it should receive, because not all of the contextual instances which contribute to the bonus have been processed yet. For intersentential context, using two passes in a batch mode will also permit the assignment of a bonus based on following sentences in the input, e.g. if the input sentences in Figure 1 were reversed, the appropriate sense of "bank" would still receive a bonus. Naturally, some applications of machine translation require production of a translation immediately upon receipt of a sentence, and in those applications such batching will not be possible (but a two-pass calculation can still be used for local context).

Consideration of document boundaries will eliminate the cases where a sentence from another document contributes to the intersentential context bonus merely because it lies within the window being considered.

Finally, where the fine-grained document boundaries are available, the base weights assigned to re-

[2] In at least one case, two consecutive sentences were translated with some of the information from one moved to the other in the translation.

trieved matches can be dynamically adjusted. When performing a batch translation of a document, a global similarity can be computed between the input document and each of the training documents, and base weights adjusted upwards for the most similar documents. This then automatically biases the translations towards those used in the documents which are most similar in subject matter, style, and genre to the input text, much as the current code permits a static adjustment of weights by the user to match the anticipated domain of the text to be translated.

Orthogonal to all of the above enhancements, more investigation is needed to ensure that improved scores on the tuning data reliably result in improved scores on unseen texts.

References

Ralf D. Brown. 1996. Example-Based Machine Translation in the PANGLOSS System. In *Proceedings of the Sixteenth International Conference on Computational Linguistics*, pages 169–174, Copenhagen, Denmark. http://www.cs.cmu.edu/~ralf/papers.html.

Ralf D. Brown. 2001. Transfer-Rule Induction for Example-Based Translation. In *Proceedings of the Workshop on Example-Based Machine Translation*, September. http://www.cs.cmu.edu/~ralf/papers.html.

Ralf D. Brown. 2004. A Modified Burrows-Wheeler Transform for Highly-Scalable Example-Based Translation. In *Machine Translation: From Real Users to Research, Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA-2004)*, volume 3265 of *Lecture Notes in Artificial Intelligence*, pages 27–36. Springer Verlag, September-October. http://www.cs.cmu.edu/~ralf/papers.html.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. One Sense Per Discourse. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York*, pages 233–237, February. http://www.ldc.upenn.edu/H/H92/.

Nano Gough and Andy Way. 2003. Controlled Generation in Example-Based Machine Translation. In *Proceedings of the Ninth Machine Translation Summit (MT Summit IX)*, pages 133–140.

Linguistic Data Consortium. 1997. *Hansard Corpus of Parallel English and French*. Linguistic Data Consortium, December. http://www.ldc.upenn.edu/.

Rada Mihalcea and Ted Pedersen. 2003. An Evaluation Exercise for Word Alignment. In *Proceedings of the HLT-NAACL 2003 Workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 1–10. Association for Computational Linguistics, May.

Tony Veale and Andy Way. 1997. Gaijin: A Template-Driven Bootstrapping Approach to Example-Based Machine Translation. In *Proceedings of the NeMNLP'97, New Methods in Natural Language Processessing*, Sofia, Bulgaria, September. http://www.compapp.dcu.ie/~tonyv/papers/gaijin.html.

Reversible Template-based Shake & Bake Generation

Michel Carl, Paul Schmidt and Jörg Schütz
Institut für Angewandte Informationsforschung
66111 Saarbrücken
Germany
{carl,paul,joerg}@iai.uni-sb.de

Abstract

Corpus-based MT systems that analyse and generalise texts beyond the surface forms of words require generation tools to re-generate the various internal representations into valid target language (TL) sentences. While the generation of word-forms from lemmas is probably the last step in every text generation process at its very bottom end, token-generation cannot be accomplished without structural and morpho-syntactic knowledge of the sentence to be generated. As in many other MT models, this knowledge is composed of a target language model and a bag of information transferred from the source language.

In this paper we establish an abstracted, linguistically informed, target language model. We use a tagger, a lemmatiser and a parser to infer a template grammar from the TL corpus. Given a linguistically informed TL model, the aim is to see what need be provided from the transfer module for generation.

During computation of the template grammar, we simultaneously build up for each TL sentence the content of the bag such that the sentence can be deterministically reproduced. In this way we control the completeness of the approach and will have an idea of what pieces of information we need to code in the TL bag.

1 Introduction

METIS-II[1] investigates the possibilities to develop a data-driven MT system using a huge monolingual target language (TL) corpus and a bilingual dictionary. While the dictionary is used to map SL items onto the TL, the corpus serves as a model to generate the TL sentences. This translation strategy parallels with shake & bake (S&B)(Whitelock, 1991; Whitelock, 1992). In S&B the bilingual knowledge is exhausted by the equivalence of basic expressions and TL generation as parsing is under direct control of the TL grammar.

Shake & bake generation starts from a bag of TL items. The order of the items in the bag is irrelevant. Generation freely combines the items to produce all sentences that are compatible with the constraints in the bag and in the TL grammar. While the content of the bag is obtained from the analysis of the source language (SL) and a dictionary lookup, the main challenge in S&B is the generation of TL sentences from a bag of TL items.

In this paper we investigate a corpus-based approach to S&B generation. In contrast to S&B, where the free combination of items in the bag is restricted by constraints of a hand-made TL grammar (Brew, 1992), we automatically induce a TL grammar from a corpus of TL sentences. The TL grammar serves as a model to select and serialise items in the bag according to the TL syntax.

A similar strategy is also proposed by (Cao and Li, 2000) who translate base noun phrases using a dictionary and the web. As in (Habash and Dorr, 2002) we view machine translation as a 'generation heavy' process. We assume a large number of resources in the TL, first of all a huge corpus of TL sentences, so as to shift most of the processing from SL analysis to the TL generation.

Current language modeling in corpus-based machine translation relies on n-grams (Stolcke, 2002; Goodman, 2002; Badia, 2005). Probabilities of overlapping word n-grams are an excellent means to generate and weight coherent sequences of words. However, long distance dependencies cannot easily be handled with n-gram models. In addition, n-grams are usually obtained from inflected words. In METIS, however, we assume lemmatised words in the TL bags. There is thus a gap between lemmatised forms in the input bag and n-gram models based on full word forms.

We present a corpus-derived language model that overcomes these shortcomings. A corpus of English sentences is tagged, lemmatised and parsed. The parsed structures are converted into a normalised context-free grammar and stored in a database. Due to the shape of the representations we call the resulting database a template grammar. The template grammar is the basis of our language model that contains the basic information required to generate

[1]METIS-II is sponsored by EU under the FET-STREP scheme of FP6 (METIS-II, IST-FP6-003768).

English sentences.

Sentence templates have been studied and used for some time. Templates consist of sequences of constant and variable elements which emerge with the identification of similarities and differences with forms in memory. (Cicekli and Guvenir, 2003) give a formalisation of this process while (Malavazos and Piperidis, 2000) establishes a link between templates and analogical modelling.

Recently (Cicekli, 2005; Carl, 2003) extend translation templates with type constraints. (Gough and Way, 2004) produces a set of marker templates by replacing the marker word by its relevant tag. Similarly, we generalise templates from monolingual sentences by replacing constituents by their relevant tag.

To produce a sentence (or a text) from a template grammar, we need additional information from the TL bag. The items and constraints in the bag select and activate a subset of rules in the grammar which then produces a TL sentence. By mapping the bag on the template grammar, word order is determined and features for morpho-syntactic generation are fixed. Thus, the content of the bag should interact with the template grammar such that information is complete to resolve all major morphological and syntactic ambiguities for generation. In the same time the model should be flexible enough to produce all desirable sentences in the TL.

Obviously, the content of the bags depend on the information available in the template grammar and vice versa. In many MT systems, TL generation is seen a consequence of SL analysis. Thus, almost all (symbolic) approaches to MT start from SL sentences and design TL generation according to the information available after transfer. However, statistical (IBM) approaches have shown that a reversed method is not only possible but also leads to a reasonable decomposition of the translation task: To find the most likely translation SL $\rightarrow$ TL, Bayes' theorem allows to train probabilities TL $\rightarrow$ SL and an (independent) target language model. Thus parameters are trained in the inverted order of the intended translation direction while for translation the reversed model is used.

In this paper we follow this intuition for the generation of a language model from a TL corpus. Simultaneously to the language model, we generate for each sentence a bag of items and constraints that complements the language model such that the original sentences can be reproduced. That is, for each step in the construction of an abstracted TL model, we compute and assemble the bits of information that enable the reproduction of the original sentence. In this way we obtain a template grammar and a set of bags for the English sentences. The bags contain lemmas, structural and morpho-syntactic information such that the original text can be reproduced.

Only if we know how a bag looks like in order to generate a particular sentence with a given (template) grammar, we can try to obtain similar bags as a result of transfer and through a bilingual dictionary from a SL sentence. Further research will show whether and to what extent this is an appropriate basis for S&B translation.

We incrementally build a target language model on four levels:

- First we have trained the TnT tagger(Brants, 2000) with the BNC data to obtain tagged sentences.
- Section 2 describes a reversible lemmatisation/token generation tool that takes as its input the tagged text[2].
- Section 3 describes a number of experiments to generate word forms from lemmas with partial information.
- Section 4 describes reversible parsing and morphological processing

It turns out that constraints are essentially determined by the way we implement parsing and morphological processing. To make the process reversible, the bags need to be extended with additional structural and morpho-syntactic information, while near perfect token generation can be obtained even with restricted information.

2 Reversible Lemmatisation

This section describes a reversible lemmatiser/token-generator for English. The lemmatiser produces a normalized form for word-tokens in the following sense:

1. convert the lemma into lower-case alphabetical characters
2. apply rules or a token-lemma dictionary to generate the lemma

Lemmatisation rules are used to strip off or modify regular inflection suffixes from the tokens. A lemmatisation lexicon is used for the irregular cases.

The lemmatiser reads a CLAWS5-tagged[3] file, generates a lemma together with two additional features indicating the orthographic properties (O) and the inflection rule (IR) that applies to the word. The token-generator reads a lemma together with a CLAWS5-tag (henceforth CTAG) and the O and the IR feature. Token generation is to a 100% reversible, that is: a token set {token,CTAG} is equivalent to a lemma set {lemma,CTAG,O,IR} and both sets can

[2]The lemmatiser can be obtained from the authors

[3]`http://www.comp.lancs.ac.uk/computing/research/ucrel/claws/`

be transformed without loss of information into each other.

In section 2.1 we give a small introduction to the CLAWS5 tag set. The material is essentially copied from their web-site at `http://www.comp.lancs.ac.uk/computing/research/ucrel/claws/`.

Orthographic normalisation is described in section 2.2. The lemmatiser makes use of a lemmatisation lexicon and lemmatisation rules as described in section 2.3 and 2.4. Section 2.5 explains token-generation under the assumption that all required information is available.

NN0	Common noun, neutral for number (e.g. aircraft, data, committee)
NN1	Singular common noun (e.g. pencil, goose, time, revelation)
NN2	Plural common noun (e.g. pencils, geese, times, revelations)
NP0	Proper noun (e.g. London, Michael, Mars, IBM)
VVB	The finite base form of lexical verbs (e.g. forget, send, live, return) [Including the imperative and present subjunctive]
VVD	The past tense form of lexical verbs (e.g. forgot, sent, lived, returned)
VVG	The -ing form of lexical verbs (e.g. forgetting, sending, living, returning)
VVI	The infinitive form of lexical verbs (e.g. forget, send, live, return)
VVN	The past participle form of lexical verbs (e.g. forgotten, sent, lived, returned)
VVZ	The -s form of lexical verbs (e.g. forgets, sends, lives, returns)

Table 1: Subset of the CLAWS5 tag set

2.1 The CLAWS Tag set

The POS tagging software for English text, CLAWS (the Constituent Likelihood Automatic Word-tagging System), has been continuously developed since the early 1980s (see `http://www.comp.lancs.ac.uk/computing/research/ucrel/claws/`).

Accuracy CLAWS has consistently achieved 96-97% accuracy (the precise degree of accuracy varying according to the type of text). Judged in terms of major categories, the system has an error-rate of only 1.5%, with c.3.3% ambiguities unresolved, within the BNC. The amount of error in the tagging of the corpus varies greatly from one tag to another. The most error prone-tag, by a large margin, is VVB, with more than 17 per cent error, while many of the tags are associated with no errors at all, and well over half the tags have less than a 1 per cent error.

The CLAWS5 tagset for the BNC has just over 60 tags. This tagset was kept small because it was designed for handling much larger quantities of data than were dealt with up to that point. For instance there are four different tags for nouns and six for verbs as shown in table 1.

In addition, there are 30 "Ambiguity Tags". These are applied wherever the probabilities assigned by the CLAWS automatic tagger to its first and second choice tags were considered too low for reliable disambiguation. So, for example, the ambiguity tag AJ0-AV0 indicates that the choice between adjective (AJ0) and adverb (AV0) is left open, although the tagger has a preference for an adjective reading. The mirror tag, AV0-AJ0, again shows adjective-adverb ambiguity, but this time the more likely reading is the adverb.

The term 'multiwords' denotes multiple-word combinations which function as one wordclass - for example, a complex preposition, an adverbial, or a foreign expression naturalised into English as a compound noun.

AV0	of course	(adverb)
PRP	according to	(preposition)
NN1	persona non grata	('naturalised' compound noun)

2.2 Orthographic Normalisation

The lemmatiser converts characters into lower case. The O feature keeps track of the orthographic properties of the original word. The O feature has the following values:

- **n** the token consists only of digits [0-9]
- **s** the token does not contain alphabetical characters
- **l** the token consists only of lower-case alphabetical characters, and may contain digits and the special characters `-\'`
- **c** the token consists only of upper-case alphabetical characters, and may contain digits and the special characters `-\'`
- **f** the first character is upper-case and all others are lower-case, digits or the special characters `-\'`
- **m** for all other tokens.

The lemma is identical to the word form for the cases **n**, **s**, and **m**. That is, no explicit lemma conversion takes place if the token is not a proper word. For **l**, **c** and **f**, the lemma is converted into lower-case characters and inflection is checked[4]

[4]We allow digits to occur in proper words because some special symbols (e.g. blanks) in (compound) words can be escaped with a backslash (\) followed by their ASCII code.

Lemmatisation Rules

{CTAG,	IR}	token-suffix	⇒	lemma-suffix
VVG	1	`ffing`	⇒	`ff`
VVG	2	`^(.{1,3}ll)ing`	⇒	`\$1`
VVG	3	`ssing`	⇒	`ss`
VVG	4	`([bcdfghjklmnpqrstvwxz])\1ing`	⇒	`\$1`

Token-generation Rules

{CTAG,	IR}	lemma-suffix	⇒	token-suffix
VVG	1	`ff`	⇒	`ffing`
VVG	2	`(.{1,3}ll)`	⇒	`$1ing`
VVG	3	`ss`	⇒	`ssing`
VVG	4	`([bcdfghjklmnpqrstvwxz])`	⇒	`$1$1ing`

Table 3: First four rules of the VVG paradigm: rules are reversed for token-generation

CTAG	IR	token	lemma
NN2	L8	analyses	analysis
VVN	L28	gone	go
VVD	L29	went	go
VVZ	L6	goes	go
PNQ	whom	whom	who
PNQ	whose	whose	who
AJC	1	better	good
AJS	17	best	good

Table 2: Excerpt from the Lemmatisation Lexicon

2.3 The Lemmatisation Lexicon

The lemmatisation lexicon encodes a word token together with a CTAG, its lemma and an IR as shown in table 2. Each {token,CTAG} combination is associated with one {lemma,IR} combination. To ensure reversibility of lemmatisation, the IR must be chosen such that each {lemma,CTAG,IR} is unique. In this way every {token,CTAG} combination is equivalent to exactly one {lemma,CTAG,IR}.

Lexical lemmatisation looks up a {token,CTAG} in the dictionary and retrieves a {lemma,CTAG,IR}.

The IR can encode morpho-syntactic and even semantic information in a systematic way such that it can be used during processing in subsequent processes. For instance, a finer grained distinction can be modeled between "us" and "we" or "whom" and "whose" while both forms can be reduced to the same lemma. Otherwise the IR can also consist of any distinguishing string or number as shown in the case of "best" and "better" in table 2. Lemmatisation should be lexicalised only one of the following conditions apply:

1. the word belongs to a closed class
2. the word is an inflectional exception or irregular form such as "better" and "best"
3. further morphological information is required that can be coded in the IR

2.4 Lemmatisation Rules

Lemmatisation rules map a word on its lemma by modifying the suffix of the word. This is particularly important for regular inflection of open class words.

A CTAG represents an inflection paradigm that is covered by a number of lemmatisation rules. For each {token,CTAG} — if it is not lexicalised — the lemmatiser applies a number of lemmatisation rules in a predefined order. If a lemmatisation rule matches the token, it is modified and thereby transformed into a lemma. The lemmatiser returns the lemma together with a CTAG, the O feature and the IR.

For instance, the "VVG" paradigm is associated with a list of 28 lemmatisation rules. The first 4 lemmatisation rules are shown in table 3.

The body of the rules are regular expressions that are mapped on the word tokens. A matching token suffix is substituted be a lemma suffix. Parts of in the token-suffix can be enclosed in brackets, as in rule 2. The variable `$1` in the lemma-suffix will be instantiated with the bracketed sequence of the token-suffix such that sequences are copied from the token-suffix to the lemma-suffix.

2.5 Token Generation

The lemmatisation process is reversed for token generation. Reversing the lemmatisation lexicon (see section 2.3) becomes a tokenisation lexicon: For every lemma set {lemma,CTAG,IR} that is found in the lexicon, the associated {tag,CTAG} is returned. Reversing the lemmatisation rules (see section 2.4) becomes token-generation rules: The token-generator looks up the lemmatisation rule indicated by IR in the CTAG paradigm and applies the retrieved lemmatisation rule in the reversed order as shown in table 3

As outlined above, token generation is to a 100% reversible if the lemma set is complete. That is: a token set {token,CTAG} is equivalent to a lemma set

token	CTAG	⇔	lemma	CTAG_O_IR
sniffing	VVG	⇔	sniff	VVG_l_1
dialling	VVG	⇔	dial	VVG_l_2
DRESSING	VVG	⇔	dress	VVG_c_3
Setting	VVG	⇔	set	VVG_f_4

Table 4: Input and Output of Lemmatisation and Token-generation

{lemma,CTAG,O,IR} and both sets can be transformed without loss of information into each other. In the remainder of this paper we abstract from orthographic properties (upper/lower case characters) of the word forms as coded in the O feature. That is, we restrict the lemma set to {lemma,CTAG,IR} and consider it equivalent to a token set.

3 Generating Incomplete Lemma Sets

However, we cannot always assume to have all the bits of information even in a reduced lemma set available. Assume, for instance, a verb has to be regenerated in present tense, or a singular noun should be transformed into a plural noun to adjust a stored sentence fragment to a new context. In these cases we still know the lemma of the word and the CTAG. It is unclear, however, what inflection rule should apply to generate the correct word-form.

In this section we report on some experiments to "guess" an appropriate IR for an incomplete lemma set {token,CTAG}. We show that lemmas can be re-converted into word tokens with a very high degree of accuracy even if only partial information is available. We investigate several methods to infer an appropriate inflection rule for generation from corpora and achieve accuracy of more than 99.5%.

Depending on what information is available we distinguish three cases:

1. if the full lemma set is available proceed as described in section 2.5.
2. else if IR is missing, look up the lemmatised BNC whether it contains a form {lemma,CTAG$_{new}$} and retrieve the associated IR. This approach is described in section 3.1
3. else if the BNC does not contain a suitable lemmatised form, "guess" an IR by comparing suffixes of the lemmas. This is described in section 3.2

3.1 Re-generating known Wordforms

In this first model we retrieve an IR of an incomplete lemma set {lemma,CTAG$_{new}$} from the lemmatised BNC. The word-form is re-generated that corresponds to the most frequent IR associated to a {lemma,CTAG} in the BNC. We call this model the F model since the sought IR is available in the BNC it has access to their frequency distribution.

token	lemma	CTAG	IR	freq	generated
burned	burn	VVD	29	542	burned
burnt	burn	VVD	L29	150	burned
focussed	focus	VVD	L29	34	focused
focused	focus	VVD	L29a	411	focused
brothers	brother	NN2	10	3511	brothers
brethren	brother	NN2	L8	157	brothers
aquariums	aquarium	NN2	10	48	aquaria
aquaria	aquarium	NN2	L8	82	aquaria
cookin'	cook	VVG	29	303	cooking
cooking	cook	VVG	28	1043	cooking
coming	come	VVG	27	17726	coming
comeing	come	VVG	28	2	coming
comin'	com	VVG	29	89	coming
comming	com	VVG	4	5	coming

Table 5: Regenerating word tokens in the Frequency model

As plotted in table 8, from a set of 244,500 different words, this produces 0.3648% 'noise'. That is, 892 re-generated words differ from their original form.

In some cases a given {lemma,CTAG} combination occurs with several IR in the BNC. Some examples are given in table 5. For many of these cases several writing variants are possible as e.g. British vs. American writing. When regenerating the word the more frequent variant is chosen. This caused 'noise' in the case of burnt ⇒ burned and focussed ⇒ focused where both variants are correctly reduced to the same {lemma,CTAG} but the more frequently occurring variant is re-generated. Note that the original variant could have been re-generated with the appropriate IR. Thus, {focus,VVD,L29} would generate "focussed" and {burn,VVD,L29} would generate "burnt".

In some cases an erroneous regular form is detected by an inflection rule but the irregular, correct form is re-generated (e.g. aquariums ⇒ aquaria, comeing ⇒ coming). Note also here that the incorrect forms would be re-generated with the appropriate IR.

Most of the 'noise' is, however, due to speech subscription which is part of the BNC (e.g. cookin', comin'). These spoken forms are regenerated in their correct written form (cooking, coming) as shown in table 5. It is 324 -in' forms out of the 892 noisy regenerated words that are reproduced as -ing which accounts for more than 1/3 of the 'noise'.

3.2 Guessing a new IR

In case a {lemma,CTAG$_{new}$} does not occur in the BNC and in the tokenisation lexicon, we have to find some other means to infer an appropriate IR.

As a first method we have applied the token generation rules in their pre-defined order. When a lemma suffix matches a generation rule, a word token would be produced. When no token-generation

token	lemma	CTAG	IR	re-generated
surfing	surf	VVG	4	surffing
boiling	boil	VVG	4	boilling
aborting	abort	VVG	4	abortting

Table 6: Erroneous token generated in the base-line model

lemma-suffix	CTAG	IR	rel.freq
Suffix Model S_1			
t	VVG	28	0.6092
t	VVG	4	0.3022
t	VVG	18	0.0800
t	VVG	L28	0.0072
t	VVG	29	0.0013
Suffix Model S_2			
rt	VVG	28	0.9989
rt	VVG	29	0.0011
Suffix Model S_3			
ort	VVG	28	0.9998
ort	VVG	29	0.0001
Suffix Model S_4			
bort	VVG	28	1

Table 7: Lemma suffixes from the BNC with CTAG, IR and relative frequencies of IR

model	# noise	% noise
F	892	0.365%
base	17357	7.099%
S_1	5220	2.135%
S_2	3095	1.266%
S_3	1798	0.735%
S_4	1756	0.718%
S_{dyn}	1023	0.418%

Table 8: Comparing noise of different IR estimation models from a set of 244,500 different words.

rule matches, the token is assumed to be identical to the lemma. The method can be seen as a base-line since it just inverses the lemmatisation process.

This method performs quite poorly producing 7.099% noise from the 244,500 word tokens (see table 8). That is, 17,357 words were re-produced differently from how they appear in the original list.

Most error prone were (endings of) plural noun and some verb forms. In the VVG paradigm, for instance, the first matching rule was in many instances IR 4. This rule transforms a double consonant into a single consonant for token ⇒ lemma transformation. However, for lemma ⇒ token transformation this produces many erroneous tokens as shown in table 6.

In another approach, we have indexed the suffixes of the lemmas from the BNC together with their CTAG and IR. The idea was to match the suffix of the lemmas to be re-generated together with its $CTAG_{new}$ on the indexed lemma suffixes and retrieve the associated IR.

Thus, to retrieve an IR for the incomplete lemma set {abort,VVG} we would look into a list suffixes as in table 7. By checking the last character "t" we have a choice of 5 rules with their relative distribution in the BNC. Similar to the base-line model, we apply the rules in the order of their relative frequencies and generate the token with the first applying rule. Thus, IR 28 is the most frequent inflection rule for the VVG paradigm that occurs with lemmas ending on "t". The inferred set {abort,VVG,28} generates also the correct form "aborting".

This model S_1 can be seen as an extention of the base-line model. It reorders the inflection rules according to frequencies in the BNC. As shown in table 8, the suffix model S_1 reduces noise to 2.135%.

In further experiments we have extended the length of the suffixes to 2, 3 and up to 5, where each model S_i includes the suffixes of the models S_{i-1}. The IR of the lemma to be generated would be chosen from the longest possible suffix. As can be seen in table 7, longer suffixes tend to be associated with fewer IR and show a stronger discrimination between different choices. With a suffix length of 4, inflection rule 28 can be deterministically applied for "abort".

A further enhancement of the method consists in keeping suffixes dynamically up to the length where only one inflection rule applies. There is, for instance, no point in storing {abort,VVG,29} in the suffix lexicon when {bort,VVG,29} is already unambiguous. This not only reduces the number of stored suffixes to slightly more than 20,000 compared to more than 30,000 for the S_3 model, but also increases accuracy considerably. Table 8 shows that the model S_{dyn} is only marginally worse than the frequency model F. With 0.05% more noise we can assume to re-generate word tokens from incomplete lemma sets with reasonable precision.

This also means that word tokens can be represented as lemma sets {lemma,CTAG} little loss of information. Lemma sets are the basic entities from which the original word tokens can be re-generated with high accuracy.

4 Reversible Parsing/Morphological Generation

This section describes the morpho-syntactic level of the language model. It builds up on the lemma sets and formalises morpho-syntactic properties of the BNC in a reversible manner.

First we parse the lemmatised English sentences.

We use a flat parser that is implemented in KURD (Carl, 2005). The parser consists of three sets of rules which incrementally produce larger brackets: the LEX set marks only the lexical items: nouns, adjectives, adverbs and numbers. The PHRASE set marks adjective phrases, noun phrases, conjunctions of noun phrases and prepositional phrases. The CLAUSE set marks subordinate clauses and sentences.

The parser generates 'internal' nodes that express relations between terminal lemma sets similar to a constituent tree. It uses a unique set of parsing tags (PTAGs) that characterise the properties of the subsumed nodes.

From the parses we extract two distinct sets: a normalised context free grammar and a set of "constraints". The set of "constraints" and the grammar complement each other such that the original lemmatised English sentences can be reproduced.

For parsing we consider the lemma sets {lemma,CTAG} the leaves of a phrase-structure tree. For internal nodes, the parser uses a distinct set of 'internal' tags and features.

4.1 Parsing

Partial parsing yields a bracketed structure, as shown in the table 9. The proper noun "john", the noun "apple" and the noun phrase "an apple" are bracketed.

{john,NP0}, {eat,VVZ}, {a,AT0}, {apple,NN1}

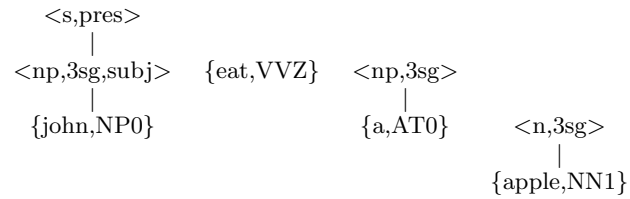


Table 9: Lemma sets and parsed sentence "John eats an apple"

We do not allow overlapping and/or ambiguous segmentation but enable recursive bracketing. Thus, a noun can be bracketed within a larger noun phrase which can be part of a prepositional phrase etc. For instance, the bracketed noun "apple" is contained in the larger noun phrase (an apple)$_{\texttt{np}}$.

In addition we percolate agreement and other information into the internal nodes. Currently we use three features <fcase>, <agr> and <tns>. The <fcase> feature can take the values `subject`, `objective` or `genitive`. The <agr> feature can take (among others) the value `3sg`, and the and <tns> feature has the values `pres` and `past`.

4.2 Reversible Morphological Generation

Lemmatisation abstracts away from number in nouns and number, person and tense in verbs. That is, the PTAG features <agr> and <tns> exhaustively describe the inflectional properties of the subsumed terminal lemma sets. The structure of the parse is designed such that all relevant inflection information for every lemma set is assembled and mirrored in the immediate dominating internal node. Thus, a singular noun, coded as NN1 in the lemma set, is represented as `3sg` in the dominating node, a VVZ verb is coded as `3sg`.

To transform a singular noun into plural we need to replace NN1 with NN2; to transform a past tense verb into present we transform VVD into VVB or VVZ for 3rd person singular.

In this way, by knowing the <agr> and <tns> values of the internal nodes we can re-produce the original terminal CTAGs with 100% accuracy. Knowing the lemma and the CTAG for each lemma set guarantee reversible deterministic generation of the word token as shown in section 3.

CTAG information in the parse tree that is independent from the PTAGs remains untouched and serves as a default for the token-generation.

We have verified reversibility of the parsed structure on a set of 1.000.000 sentences take from the BNC. In future we also intend to tackle closed class words such as articles, pronouns and prepositions in the same way.

4.3 Grammar Inference

We extract a CFG grammar from the parse in the following way. On the one hand, we extract rules from the bracketed structures by transforming the tag into the left-hand side (LHS) of the rules and the content into the right-hand side RHS. Thus the tag <noun> appears on the LHS in rule (4) while the content of the bracketed expression {apple,NN1} occurs in the RHS. On the other hand, templates are generated by replacing the bracketed constituents with their tags. A template consists of terminal symbols and nonterminal symbols. Template (1) consists of one leaf {eat,VVZ} and two non-terminals <np,3sg,subj> and <np,3sg>.

LHS		RHS
<snt>	→	<np,subj> , {eat,VVZ} , <np>
<np>	→	{a,AT0} , <np>
<np>	→	{john,NP0}
<n>	→	{apple,NN1}

Table 10: A sentence template grammar extracted from the parse

The grammar extracted from the parse in table 9 consists of 4 context-free rules, where <snt> is the top-level symbol and <np>, <n> are non-terminals.

Note that at least one terminal symbol must occur in the RHS of the rules.

To reduce the number of different rules in the grammar and to make them consistent amongst each other, some CTAGS are normalised. Thus, all plural nouns are converted into singular (NN2 ⇒ NN1) and all finite verbs VVD and VVB are converted into VVZ (3rd person singular). Internal features in non-terminal nodes are set to 3sg and pres. In the current example, all features correspond already to the default setting.

4.4 Extraction of Constraints

In addition to the template grammar, a set of constraints is extracted from the parse. The constraints contain features and structural information of the internal nodes of the parse. Feature information includes the tags <fcase>, <agr> and <tns> as outlined in section 4.1. Examples of the extracted constraints are given in table 11.

The structural information is represented by the numbers of the words that are matched in the structure. Each structural constraint consists of the word numbers matched on the top-level template followed by the sets of word numbers matched in the daughter nodes.

For instance the <snt> node has three daughters from which the first and the third nodes are non-terminals. The terminal {eat,VVZ} is the second word in the sentence. The subtrees of the first and the third daughter nodes are instantiated by word 1 and the set of words 3 and 4 respectively. This information is represented as "2_1_3|4". That is, the first set of number(s) (i.e. 2) represents the words matched by the top-level template, while the words matched in the successive daughter nodes are separated by an underscore "_". This information is extracted for every internal node in the parse. Thus, the <np> node covering "an apple" is linked to the partial tree 3_4, where the 4th wort in the sentence (apple) is a subi-structure of the 3rd word "the".

wnr	PTAG	agr	fcase	tns
2_1_3|4	snt	3sg	—	pres
1	np	3sg	subj	—
3_4	np	3sg	—	—
4	n	3sg	—	—

Table 11: Constraints extracted from the parse

4.5 Reversible Syntactic Generation

In this section we show how the original parse tree (as e.g. in table 9) can be reproduced from the bag of TL lemmas (e.g. as in table 12), a sentence grammar (as in table 10) and a set of constraints (as in table 11).

Syntactic generation starts from a bag of TL lemmas. Each lemma in the bag is associated with a unique index as shown in table 12. The bag is mapped on the sentence grammar and the word indexes are copied into the matching nodes (see table 13). Thereafter the rules are stitched together to form a parse tree taking into account structural constrains.

wnr	lemma
1	john
2	eat
3	a
4	apple

Table 12: Bag of TL lemmas

Since in the reversible setting, we have for each sentence grammar a consistent set of structural constraints, an optimal combination of the grammar rules can be found from any starting point. That is, irrespectively with which grammar rule we start, the constraints will always lead to the initial best parse tree.

For instance, there is only one structural constraint (i.e. 3_4) that applies to rule #3 in table 13. This constraint requires word number 3 (i.e. "a") to be linked to the template where a subtree is linked to word number 4. The constraint thus favors the partial structure (a (apple)) which combines rules #3 and #4. This partial tree can be stored and inserted in the second slot of rule 2 as required be the constraint "2_1_3|4". Once an optimal parse tree is generated, the internal nodes are instantiated with PTAG features. The results is then input to morphological generation as described in section 4.

#	LHS		RHS
1	<np>	→	1: {john,NP0}
2	<snt>	→	<np> , 2: {eat,VVZ} , <np> .
3	<np>	→	3: {a,AT0} , <np>
4	<n>	→	4: {apple,NN1}

Table 13: Instantiated generation grammar

5 Conclusion and Outlook

In this paper we have presented a method to decompose sentences into three disjoint sets: a bag of lemmas, a set of structural and morpho-syntactic constraints and a template grammar. Several steps of analysis are involved in the construction of these sets: tagging, lemmatisation, and parsing. We have shown that the decomposition is reversible to a very high degree, i.e. the original sentence can be deterministically re-composed from these sets with only the token-generation remains with 0.05% noise below an optimal result.

While the template grammar serves as an abstracted language model, the bag of items and con-

straints select their preferred combination combinations and fix morphological properties of the senence to be generated.

In the future we want to extend the approach in several ways. We need to investigate in how far and with what precision a particular sentence grammar can be retrieved from a large set of templates and grammar rules. This investigation will follow the approach of a previous study in (Carl et al., 2005). Sophisticated weighing and selecting strategies are required. For instance, in many cases more than 200,000 rules are extracted from a templates grammar with 1.8 million entries. The extracted rules share one or more tokens with the lemmas in the bag. Since exhaustive combination of all rules is infeasible, the matched rules have to be weighted and graded using different knowledge resources.

Once we know what item and constraints are required in the TL bag to extract a particular sentence grammar, we will try to generate new sentences that are not in the original TL corpus.

Within the METIS-II consortium we plan to run an experiment where TL bags obtained from a bilingual dictionary are to be generated in the TL. These bags can be expected to contain ambiguities and noise and constraints be partially inconsistent with the retrieved sentence grammar.

These experiments will shed more light on the usefulness of the approach proposed in this paper and will show whether further constraints and mechanisms that certainly will turn out to become necessary can be implemented consistently in the proposed framework.

References

Toni Badia. 2005. An n-gram approach to exploiting monolingual corpus for MT. In *Second EBMT Workshop.*

Thorsten Brants. 2000. A Statistical Part-of-Speech Tagger? In *Proceedings of TUANLP*, pages 224–231.

Chris Brew. 1992. Letting the Cat out of the Bag: Generation for Shake & Bake MT. In *Proceedings of COLING92.*

Yunbo Cao and Hang Li. 2000. Base Noun Phrase Translation: Using Web Data and the EM Algorithm. In *COLING.*

Michael Carl, Ecaterina Rascu, and Paul Schmidt. 2005. Using template grammars for shake & bake paraphrasing. In *EAMT.*

Michael Carl. 2003. Inducing Translation Grammars from Bracket Alignments. In *Recent Advances in Example-Based Machine Translation.*

Michael Carl. 2005. *KURD.* IAI, electronic working paper 38. forthcoming.

Ilyas Cicekli and H. A Guvenir. 2003. Learning Translation Templates from Bilingual Translation Examples. In *Recent Advances in Example-Based Machine Translation.*

Ilyas Cicekli. 2005. Learning Translation Templates with Type Constraints. In *Workshop on EBMT.*

Joshua Goodman. 2002. The State of the Art in Language Modeling. In *Tutorial Presented at AMTA.*

Nano Gough and Andy Way. 2004. Example-Based Controlled Translation. In *EAMT.*

Nizar Habash and Bonnie Dorr. 2002. Handling Translation Divergences: Combining Statistical and Symbolic Techniques in Generation-Heavy Machine Translation. In *AMTA.*

Christos Malavazos and Stelios Piperidis. 2000. Application of Analogical Modelling to Example Based Machine Translation. In *COLING.*

A. Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *ICSLP.*

P. Whitelock, 1991. *Shake-and-Bake Translation.* Unpublished Draft.

P. Whitelock. 1992. Shake-and-Bake Translation. In *Proceedings of the COLING92.*

Learning Translation Templates with Type Constraints

Ilyas Cicekli

Department of Computer Engineering, Bilkent University
Bilkent 06800, Ankara, TURKEY

ilyas@cs.bilkent.edu.tr

Abstract

This paper presents a generalization technique that induces translation templates from given translation examples by replacing differing parts in these examples with typed variables. Since the type of each variable is also inferred during the learning process, each induced template is associated with a set of type constraints. The type constraints that are associated with a translation template restrict the usage of that translation template in certain contexts in order to avoid some of wrong translations. The types of variables are induced using the type lattices designed for both source language and target language. The proposed generalization technique has been implemented as a part of an EBMT system.

KeyWords: EBMT, Machine Learning

1 Introduction

An example-based machine translation [8] (EBMT) system uses a bilingual corpus to translate a given sentence in a source language into a target language. Some EBMT systems use a bilingual corpus to find translations of the parts of a given sentence, and combine these partial solutions to get the translation of the whole sentence. Some EBMT systems [1,2,3,4,5,6] extract translation templates from example sentences in a given bilingual corpus and use these translation templates in the translation of other sentences. The main differences between these EBMT systems are the assumptions that they made on the structure of the bilingual corpus and their generalization techniques. The EBMT translation system which uses the generalization technique described in this paper also extracts translation templates from a set of translation examples.

In the EBMT system presented in [3,4], a translation template is induced from given two translation examples by replacing differing parts in these examples by variables. A variable replacing a difference that consists of two differing parts (one from the first example, and the other one from the second example) is a generalization of those two differing parts. Later, that variable can be replaced by any string during the translation process without putting any restriction on the possible replacements. Although the learned translation template works correctly in certain environments, it can lead wrong translations in some other unrelated environments because that variable replacement cannot be appropriate in the unrelated environment. In this paper, we propose a generalization heuristic that replaces the differences with variables and it also induces the types of these variables from the differences. Since the types of variables disallow some possible replacements for the variables, the generation of wrong translation results in the unrelated contexts can be avoided.

The type of a variable which replaces a difference is found by using a type lattice for the language of the symbols appearing in the difference. Since the generalization technique described in this paper is used as a part of an EBMT system between English and Turkish, the type lattices for English and Turkish have been developed by hand and they are used in the EBMT system. The quality of the induced translation templates also depends on the quality of the type lattices.

The rest of the paper is organized as follows. The structure of translation templates without type constraints is discussed in Section 2. Section 3 introduces the structure of translation templates with type constraints. The generalization process

that learns the translation templates with type constraints is presented in Section 4. We give the concluding remarks and possible future extensions in Section 5.

2 Translation Templates Without Type Constraints

A *language* is a set of strings in the alphabet of that language, and the *alphabet of a language* is a finite set of symbols. For example, a string in a natural language, such as English or Turkish, is a sequence of tokens in that natural language. Each token in a natural language can be a root word or a morpheme. In other words, the set of all root words and morphemes in a natural language will be treated as its alphabet in our discussions. We also associate each language with a finite set of variables. A *generalized string* is a string of the symbols of the alphabet of the language and the variables in the set of variables associated with the language. This means that a generalized string is a string that contains at least one variable. We will assume that each language will be associated with a different set of variables. A string without variables is called as a *ground string*.

A translation template can be an *atomic* or *general translation template*. An *atomic translation template* $T_a \leftrightarrow T_b$ between languages L_a and L_b is a pair of two nonempty strings T_a and T_b where T_a is a ground string in L_a and T_b is a ground string in L_b. An atomic translation template $T_a \leftrightarrow T_b$ means that the strings T_a and T_b correspond to each other. A given *translation example* will be an atomic translation template.

A *general translation template* between languages L_a and L_b is an if-then rule in the following form:

$$T_a \leftrightarrow T_b \quad \textbf{if} \quad X_1 \leftrightarrow Y_1 \ \textbf{and} \ \ldots \ \textbf{and} \ X_n \leftrightarrow Y_n$$

where $n \geq 1$, T_a is a generalized string of the language L_a, and T_b is a generalized string of the language L_b. Both T_a and T_b must contain n variables. The variables in T_a are $X_1 \ldots X_n$, and the variables in T_b are $Y_1 \ldots Y_n$. Each generalized string (T_a and T_b) in a general translation template should contain at least one token from the alphabet of the language of that string.

For example, if the alphabet of L_a is $A = \{a,b,c,d,e,f,g,h\}$ and the alphabet of L_b is $B = \{t,u,v,w,x,y,z\}$, the following are some examples of translation templates between L_a and L_b.

- $de \leftrightarrow vyz$
- $abX_1c \leftrightarrow uY_1$ **if** $X_1 \leftrightarrow Y_1$
- $aX_1X_2b \leftrightarrow Y_2vY_1$ **if** $X_1 \leftrightarrow Y_1$ **and** $X_2 \leftrightarrow Y_2$

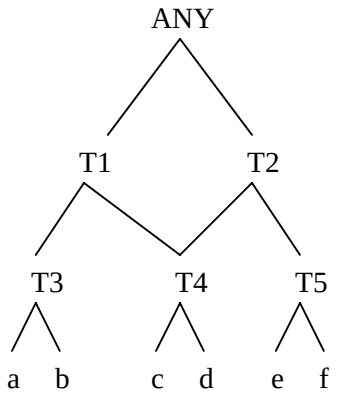
The first translation template is an atomic translation template, and last two are general translation templates. The first atomic translation template means that *de* in the language L_a and *vyz* in the language L_b correspond to each other. A general translation template is a generalization of translation examples, where certain components are generalized by replacing them with variables and establishing bindings between these variables. For example, in the second example above, the generalized string abX_1c represents all sentences of L_a starting with *ab* and ending with *c* where X_1 represents a non-empty string on *A*, and the generalized string uY_1 represents all sentences of L_b starting with *u* where Y_1 represents a non-empty string on *B*. That general template says that a sentence of L_a in the form of abX_1c corresponds to a sentence of L_b in the form of uY_1 given that X_1 corresponds to Y_1. If we know the correspondence $de \leftrightarrow vyz$, the correspondence $abdec \leftrightarrow uvyz$ can be inferred from that general template.

3 Translation Templates With Type Constraints

3.1 Type Expressions

All symbols in the alphabet of a language are organized as a *type lattice*. The symbols in the alphabet of the language appear at the bottom of the type lattice. In fact, each symbol is treated as a *ground type name* that represents itself in the type lattice. Inner nodes in the lattice are *type names* that are used for the language, and each type name represents a set of ground type names. Thus, a ground type name represents a singleton set containing that ground type name. At the top of the lattice, there is a special type name, called *ANY*. The type name *ANY* represents the set of all ground type names in the language. If *t* is a type name, we will say that GT_t is the set of the ground type names that are covered by *t*. Each node in the lattice, except *ANY*, can have one or more parents. If node *P* is a parent of node *C* in the type lattice, $GT_P \supset GT_C$ holds. Figure 1 gives a type lattice for a simple language. Since type name *T1* is the parent of type name *T3*, $GT_{T1} \supset GT_{T3}$ will be true for that type lattice.

Each variable of a generalized string in a general translation template with type constraints is associated with a type expression, and the type expression is called *the type of the variable*. The type of a variable indicates the possible ground strings which can replace that variable during the



- Ground Type Names = {a,b,c,d,e,f}
- The set of ground type names is also the alphabet of this simple language.
- The sets of ground type names represented by some type names.
  $GT_a = \{a\}$
  $GT_{T3} = \{a,b\}$
  $GT_{T1} = \{a,b,c,d\}$
  $GT_{T2} = \{c,d,e,f\}$
  $GT_{ANY} = \{a,b,c,d,e,f\}$

Figure 1. A Type Lattice for A Simple Language

translation process. A *type expression* is a non-empty sequence of atomic type expressions. An *atomic type expression* can be either T or nullor(T) where T is a type name from the type lattice. If the type of a variable is a type name T, this means that the variable can be replaced by a ground type name from GT_T. In the second case where the type of a variable is nullor(T), the variable is replaceable with an empty string in addition to a ground type name from GT_T. In other words, $GT_{nullor(T)}$ is equal to $GT_T \cup \{\varepsilon\}$.

The definition of GT can be extended for the type expressions that consist of more than one atomic type expression. If a type expression T is an atomic type sequence T1...Tn, GT_T is equal to the concatenation of the sets GT_{T1} through GT_{Tn}. In general, a variable of type T is replaceable with a ground string from GT_T. For example, let us consider the simple language and its type lattice in Figure 1. If the type of a variable is type T3, this means that it can be replaced with a ground string from GT_{T3}={a,b}. When the type of a variable is nullor(T3), it can be replaced with an empty string or a string from GT_{T3}. A variable of the type ANY can be replaced with any ground type name. If a type expression T is an atomic type sequence "T3 T4", GT_T is equal to {ac,ad,bc,bd}.

Type lattices for English and Turkish are partially created by hand in order to be used in the developed EBMT system. Simplified partial type lattices for these languages can be seen in Figure 2. The details of those type lattices are not given in the figure. Major type names in each type lattice are the part of speech tags used for that language. The affixes used in a language are also considered as major type names. For example, the major part of speech tags such as noun, verb, pronoun and adjective are major type names in English type lattice, and they appear as children of ANY. The type names between major type names and ground type names generally represents the subgroups of part of speech tags. The affixes are grouped according to where they can be used. For example, all suffixes can be added to verbs is considered as a major type name.

3.2 Translation Templates With Type Constraints

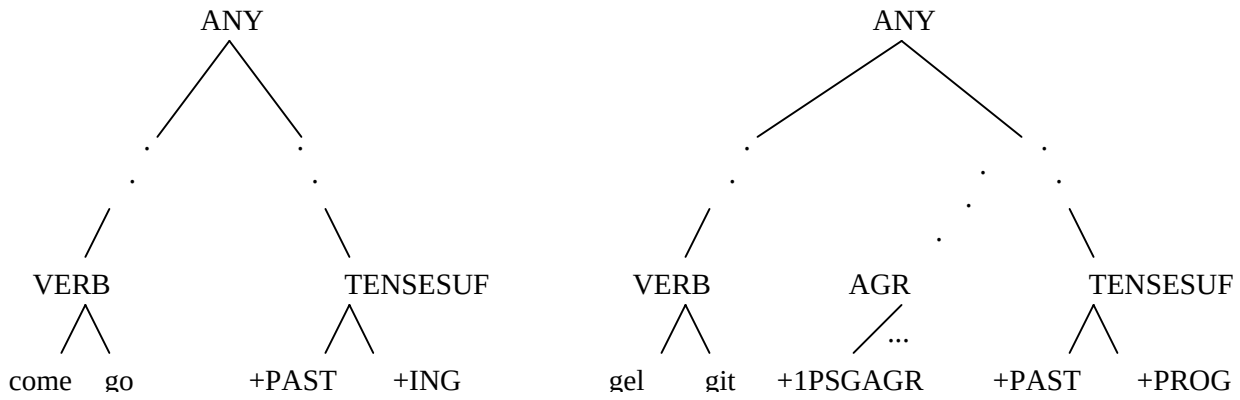
A *translation template with type constraints* is a general translation template where all variables are associated with type expressions. A translation template with type constraints will be a translation template in the following form:

$$T_a \leftrightarrow T_b$$
$$\textbf{if } X_1^{TA1} \leftrightarrow Y_1^{TB1} \textbf{ and}...\textbf{and } X_n^{TAn} \leftrightarrow Y_n^{TBn}$$

where each of TA1,...,TAn and TB1,...,TBn is a type expression. A translation template with type constraints also puts a restrictriction on the possible replacements of variables during the translation process. For example, the following is a translation template with type constraints

$$I\ X^{VERB}\ +PAST \leftrightarrow Y^{VERB}\ +PAST\ +1PSAGR$$
$$\textbf{if } X^{VERB} \leftrightarrow Y^{VERB}$$

This general template represents that an English sentence in the form of "I X^{VERB} +PAST" corresponds to a Turkish sentence in the form of "Y^{VERB} +PAST +1PSAGR" given that X corresponds to Y. This template also specifies that X can only be replaced by a verb at English side, and Y can only be replaced by a verb at Turkish side. In this example, "+PAST" means the past tense suffix at English side, and "+PAST" and "+1PSAGR" at Turkish side mean that the past tense suffix and the first person singular agreement suffix, respectively. This translation template can



a) Simplified Type Lattice for English

b) Simplified Type Lattice for Turkish

Figure 2. Simplified Type Lattices for English and Turkish

be used in the translation of the following Turkish sentence

geldim
gel+PAST+1PSAGR

into the following English sentence

I came
I come+PAST

given that the correspondence "gel↔come" is available. During the translation process, both variables are replaced by English and Turkish verbs without violating type constraints in the translation template.

Type constraints in the translation templates restrict wrong usages of templates in certain circumstances. For example, if we try to use the previous translation template without type constraints, it may lead to wrong translation results. Let us assume that we want to translate the following Turkish sentence into English using this translation template without type constraints.

utangaçtım (I was shy)
utangaç+PAST+1PSGAGR

Without using the type restrictrictions, variable Y at Turkish side can match with "utangaç" which is an adjective (not a verb). If the correspondence "shy↔utangaç" is available, variable X at English side can match with "shy" (not a verb). Thus, it can lead to the meaningless translation result "I shy +PAST" at the lexical level. Type constraints in the translation template will avoid this wrong translation by rejecting to bind Y with "utangaç" which is an adjective.

4 Learning Translation Templates

In the EBMT system described in [3,4], translation templates are inferred without type constraints from given translation examples. Each translation example consists of an English sentence and a Turkish sentence and their lexical level representations are used for the sentences. A translation template is a generalization of two translation examples where some differing parts of the sentences are generalized by replacing them with variables, and establishing bindings between these variables.

In order to induce a translation template from given two translation examples $E^1_a \leftrightarrow E^1_b$ and $E^2_a \leftrightarrow E^2_b$, we first find the match sequence $M_a \leftrightarrow M_b$ where the match sequence M_a is a match sequence between E^1_a and E^2_a, and the match sequence M_b is a match sequence between E^1_b and E^2_b. A match sequence between two sentences is a sequence of similarities and differences between those sentences. A similarity between two sentences is a non-empty sequence of common items in both sentences. A difference between two sentences is a pair of two sequences (D_1,D_2) where D_1 is a sub-sequence of the first sentence and D_2 is a sub-sequence of the second sentence, and D_1 and D_2 do not contain any common item.

For example, let us assume that the lexical representations of the following two translation examples between English and Turkish are given.

I come +PAST ↔ gel +PAST +1PSAGR
I go +PAST ↔ git +PAST +1PSAGR

where common parts in the sentences are underlined. From these two examples, the following match sequence is found.

I (come,go) +PAST ↔
(gel,git) +PAST +1PSAGR

where (come,go) is a difference at English side, (gel,git) is a difference at Turkish side, other parts of the match sequence are similarities.

One of the learning heuristics described in [3,4], infers a translation template by replacing differences by variables and establishing bindings between these variables. This learning heuristic can create a translation template if both sides of the match sequences contain n differences where n≥1, and the correspondences of n-1 difference pairs have been already learned. For example, for the match sequence above, this learning heuristics infers the following translation templates.

I X +PAST ↔ Y +PAST +1PSAGR
if X ↔ Y
come ↔ gel
go ↔ git

The first translation template is a general translation template created by replacing differences with variables X and Y. The last two translation templates are atomic translation templates and they are inferred from the correspondence of the differences (come,go) and (gel,git).

Variables X and Y in this translation template do not have any type constraints, and they are replaceable with any ground strings as long as they are translations of each other during translation process. As we discussed in Section 3.1, this can lead to wrong translation results in unrelated environments. In order to reduce the amount of wrong translation results, translation templates will be associated with type constraints. In the rest of this section, we describe how translation templates with type constraints are inferred from the given translation examples.

4.1 Inferring A Type Expression for Two Symbols

When we replace a difference with a variable, we should also find a type expression for that variable. If both constituents of a difference are symbols (strings with length 1), the type expression for those symbols is found using the type lattice of that language, and the found type expression will be used as a type constraint for the variable replacing that difference. For example, when we infer a translation template from the match sequence "I (come,go) +PAST ↔ (gel,git) +PAST +1PSAGR", we also infer types of the variables replacing the differences (come,go) and (gel,git). Of course, we use English type lattice for the difference (come,go), and Turkish type lattice for the difference (gel,git).

If we have two symbols, they are also ground type names in the type lattice of the language of those symbols. For example, *come* and *go* are ground type names in English type lattice. Since the variable replacing the difference (come,go) represents the symbols *come* and *go*, the type of this variable should cover both of those symbols. We say that a ground type *gt* is covered by a type *t*, if $gt \in GT_t$. So, if type T covers both symbols *come* and *go*, both $come \in GT_T$ and $go \in GT_T$. At the worst case, type *ANY* will cover any given two ground type names in a language.

In general, there can be more than one type covering any given two type names. Since we do not want to over-generalize, we select the most specific type covering both of them. We say that type T_2 is more specific than type T_1, if $GT_{T1} \supset GT_{T2}$ holds. This means that T_1 is one of the ancestors of T_2. So, if both T_1 and T_2 covers given type names and T_2 is more specific than T_1, T_2 is selected as a type expression for the given type names.

In some cases, there can be two ancestors T_1 and T_2 of a given pair of type names, and the ancestors may not hold any specifity relation between them. That is, neither $GT_{T1} \supset GT_{T2}$ nor $GT_{T2} \supset GT_{T1}$ holds. So, the youngest ancestor of the two given types is selected to represent them.

In order to find a youngest ancestor of two given types, the shortest path containing one of their ancestors is found and the ancestor on that shortest path is the youngest ancestor of them. A type is also considered as an ancestor of itself. Thus, the youngest ancestor of types T_1 and T_2 will be T_1 if T_1 is an ancestor of T_2.

According to English type lattice, the youngest ancestor of *come* and *go* is type VERB, and the youngest ancestor of *gel* and *git* is type VERB according to Turkish type lattice in Figure 2. So, the following translation template with type constraints is induced from the match sequence "I (come,go) +PAST ↔ (gel,git) +PAST +1PSAGR":

I X^{VERB} +PAST ↔
Y^{VERB} +PAST +1PSAGR
if $X^{VERB} \leftrightarrow Y^{VERB}$

When we replace a difference (t1,t2) where t1 and t2 are two different type names in their type lattice with a type name t3 which is the youngest ancestor of t1 and t2, we generalize (t1,t2) as t3. Each generalization has a generalization score to indicate the amount of that generalization. We use the length of the shortest path between t1 and t2 as a generalization score. For example, the score for the generalization of (come,go) as VERB is 2, because the length of the shortest path between *come* and *go* is 2. In fact, when a difference is generalized, the generalization with the smallest generalization score is used. We will say that *gen(t1,t2)* is t3, and *genscore(t1,t2)* is 2.

4.2 Inferring A Type Expression for Two Strings

If a difference has a constituent whose length is greater than one, the generalization of that difference cannot be an atomic type expression. If *n* is the length of the longest constituent of a difference, its generalization will be a type expression consisting of *n* atomic type expressions. If a difference is (a1...an,b1...bn) where the lengths of the constituents are equal, the generalization gen(a1...an,b1...b4) will be

gen(a1,b1) gen(a2,b2) ... gen(an,bn).

The generalization score genscore(a1...an,b1...bn) for this generalization will be equal to

genscore(a1,b1) + genscore(a2,b2) + ...
+ genscore(an,bn).

If the lengths of constituents are different, we have to consider different possibilities and some symbols have to be generalized with empty strings. For example, we have to consider the following three generalizations for the difference (abc,de):

gen(a,d) gen(b,e) gen(c,ε)
gen(a,d) gen(b,ε) gen(c,e)
gen(a,ε) gen(b,d) gen(c,e)

When there are more than one possible generalization for a difference, we select the one with the smallest generalization score. Since we assume that we have an imaginary type for each ground type name in the type lattice such that it is a parent of that ground type name and the empty string, the score of the generalization of a symbol with the empty string is assumed to be 2. The generalization of a symbol *a* and the empty string is represented by *nullor(a)*.

Let us consider the following two translation examples.

I come +PAST ↔ gel +PAST +1PSAGR
I am go +ING ↔ git +PROG +1PSAGR

For these examples, the following match sequence is found.

I (come +PAST, am go +ING) ↔
(gel +PAST, git +PROG) +1PSAGR

In order to select the generalization for the difference (come +PAST, am go +ING), we have to consider the following three generalizations:

gen(come,am) gen(+PAST,go) gen(ε,+ING)
gen(come,am) gen(ε,go) gen(+PAST,+ING)
gen(ε,am) gen(come,go) gen(+PAST,+ING)

Since the last generalization has the smallest generalization score, it will be selected as the generalization for this difference. So, the generalization for this difference will be the following type expression:

nullor(am) VERB TENSESUF

Similarly, the difference (gel +PAST, git +PROG) has only one possible generalization:

gen(gel,git) gen(+PAST,+PROG)

Thus, the generalization for the difference (gel +PAST, git +PROG) will be the following type expression:

VERB TENSESUF

As a result, the following translation template with type constraints will be inferred from these two translation examples.

I $X^{\text{nullor(am) VERB TENSESUF}}$ ↔
$Y^{\text{VERB TENSESUF}}$ +1PSAGR
if $X^{\text{nullor(am) VERB TENSESUF}}$ ↔ $Y^{\text{VERB TENSESUF}}$

5 Conclusion

In this paper, we have presented a learning technique that induces translation templates from given translation examples, by replacing the differing parts with variables. Types of variables are also learned during the learning phase from the replaced differing parts. The types of variables help to reduce the amount of wrong translation results by restricting the usage of the translation templates in unrelated contexts.

The learning heuristic described in this paper has been implemented as a part of an EBMT system between English and Turkish. When the

translation results of the EBMT system using translation templates with type constraints were compared with the translation results of the EBMT system using translation templates without type constraints, the type constraints have eliminated more wrong translations from the translation results.

The type expression that is inferred for a variable replacing a difference with two symbols depends on the shortest path between those two symbols in their type lattice. The youngest ancestor of those symbols is the generalization of that difference. By selecting the youngest ancestor for those symbols, we hope that we get the most specific generalization for those symbols. The youngest ancestor may not be most specific generalization depending on those symbols and the structure of the type lattice. Although there can be another techniques to find the most specific generalization, the shortest path is one of the good techniques.

The inferred type expression by the generalization technique presented here is a most specific generalization. If we do not use any type constraint for a variable, it will be most general generalization. Other generalizations may be preferred by using certain generalization metrics. In this case, the regular expressions can be a better choice to represent type expressions. We are currently investigating these alternatives.

In this paper, the constraints for the variables are type constraints. The generalization technique described here can be also used in the inference of the semantic constraints if the semantic lattices (similar to Wordnet) are available for source and target languages. The quality of translation templates will depend on the quality of the used semantic lattices. The EBMT system in [7] also tries to generalize semantic features.

References

[1] Brown, R. D., Clustered Transfer Rule Induction for Example-Based Translation, in: *Recent Advances in Example-Based Machine Translation*, Carl, M., and Way, A. (eds.), The Kluwer Academic Publishers, Boston, 2003, pp: 287-306.

[2] Carl, M., Inducing Translation Grammars from Bracket Alignments, in: *Recent Advances in Example-Based Machine Translation*, Carl, M., and Way, A. (eds.), The Kluwer Academic Publishers, Boston, 2003, pp:339-361.

[3] Cicekli, I., and Guvenir, H. A., Learning Translation Templates from Bilingual Translation Examples, in: *Recent Advances in Example-Based Machine Translation*, Carl, M., and Way, A. (eds.), The Kluwer Academic Publishers, Boston, 2003, pp: 255-286.

[4] Cicekli, I., and Guvenir, H. A., Learning Translation Templates from Bilingual Translation Examples, *Applied Intelligence*, Vol. 15, No. 1, 2001, pp: 57-76.

[5] Kaji H., Kida Y., and Morimoto Y., Learning Translation Templates from Bilingual Text, in: *Coling* (1992), pp: 672-678.

[6] McTait, K., Translation Patterns, Linguistic Knowledge and Complexity in EBMT, in: *Recent Advances in Example-Based Machine Translation*, Carl, M., and Way, A. (eds.), The Kluwer Academic Publishers, Boston, 2003, pp: 307-338.

[7] Matsumoto, Y., and Kitamura M., Acquisition of Translation Rules from Parallel Corpora, in: Recent Advances in Natural Language Processing, Amsterdam, John Benjamins, 1995, pp: 405-416.

[8] Nagao, M.A., Framework of a Mechanical Translation between Japanese and English by Analogy Principle, in: *Artificial and Human Intelligence*, Elithorn, A., and Banerji, R. (eds.), North Holland, Amsterdam, 1984, pp: 173-180.

The influence of example-data homogeneity on EBMT quality

Etienne Denoual
ATR Spoken Language Translation Research Labs,
2-2-2 Keihanna Science City, Kyoto 619-0288, Japan
Laboratoire CLIPS - GETA - IMAG, Université Joseph Fourier, Grenoble, France
etienne.denoual@atr.jp

1 Introduction

Homogeneity of large corpora is still a largely unclear notion. In this study we first make a link between the notions of similarity and homogeneity : a large corpus is made of sets of documents to which may be assigned a score in similarity defined by cross-entropic measures, such similarity being implicitly expressed in the data. The distribution of the similarity scores of such subcorpora may then be interpreted as a representation of the homogeneity of the main corpus. A blatant fact is that the quality of an example-based machine translation (EBMT) system will depend heavily on the training examples it is fed. Being able to tune an MT system to a specific application through a wise selection of training data is therefore a critical issue. From this viewpoint, such a representation of homogeneity may be used to perform corpus adaptation to tune an EBMT system to the particular domain, or sublanguage, of an expected task. In the following study we further describe this framework and compare it with existing methods based on computing linguistic feature frequencies.

(Cavaglià 2002) made the general assumption that a corpus-based NLP system generally yields better results with homogeneous rather than heterogeneous training data, and experimented on a text classifier system (Rainbow[1]), with mixed conclusions. Not finding such an assumption completely straightforward, we reassess it by experimenting on language model perplexity, and on a grammar-based EBMT system translating from Japanese to English, in order to see if there is a real correlation between EBMT system performance and the homogeneity of the corpus of examples.

[1]See http://www.cs.cmu.edu/mccallum/bow .

2 A framework for corpus homogeneity

2.1 Previous work on corpus similarity and homogeneity

Corpus similarity has been extensively studied in past literature, and a wide range of measures have been put forward : (Kilgarriff and Rose 98; Kilgarriff 2001) investigated the similarity and homogeneity of corpora and proceeded to compare "Known Similarity Corpora" (KSC) using perplexity and cross-entropy on words, word frequency measures, and a χ^2-test which they found to be the most robust. However (as acknowledged in (Kilgarriff and Rose 98)), such a comparison methodology requires that the two corpora chosen for comparison are sufficiently similar that the most frequent lexemes in them almost perfectly overlap. Whereas intuition would hint at this being true for very large corpora, (Liebscher 2003) showed by comparing frequency counts of different Google Group corpora that it is generally not the case. Furthermore, measuring homogeneity by counting word / lexeme frequencies introduces another additional difficulty : this assumes that the word is a clearly defined unit, which is not the case in the Chinese (Sproat and Emerson 2003) or Japanese language (Matsumoto et al., 2002), for instance, where there is no word segmentation.

We claim that similarity between corpora can be adequatly quantified with a coefficient based on the cross-entropies of probabilistic models, built upon reference data. The approach needs no explicit selection of features and is language independent, as it relies on character-based models (as opposed to word-based models) thus bypassing the word segmentation issue and making it applicable on any electronic data.

The cross-entropy $H_T(A)$ of an N-gram model p constructed on a training corpus T, on a test

corpus $A = \{s_1, .., s_Q\}$ of Q sentences with $s_i = \{c_1^i..c_{|s_i|}^i\}$ a sentence of $|s_i|$ characters is:

$$H_T(A) = \frac{\sum_{i=1}^{Q}[\sum_{j=1}^{|s_i|} -logp_j^i]}{\sum_{i=1}^{Q}|s_i|} \quad (1)$$

where $p_j^i = p(c_j^i|c_{j-N+1}^i..c_{j-1}^i)$.

We therefore define a scale of similarity between two corpora on which to rank any third given one. Two reference corpora T_1 and T_2 are selected by the user, and used as training sets to compute N-gram character models. The cross-entropies of these two reference models are estimated on a third test set T_3, and respectively named $H_{T_1}(T_3)$ and $H_{T_2}(T_3)$ as in the notation in Eq. 1. Both model cross-entropies are estimated according to the other reference , i.e., $H_{T_1}(T_2)$ and $H_{T_1}(T_1)$, $H_{T_2}(T_1)$ and $H_{T_2}(T_2)$ so as to obtain the weights W_1 and W_2 of references T_1 and T_2 :

$$W_1 = \frac{H_{T_1}(T_3) - H_{T_1}(T_1)}{H_{T_1}(T_2) - H_{T_1}(T_1)} \quad (2)$$

and :

$$W_2 = \frac{H_{T_2}(T_3) - H_{T_2}(T_2)}{H_{T_2}(T_1) - H_{T_2}(T_2)} \quad (3)$$

after which W_1 and W_2 are assumed to be the weights of the barycenter between the user-chosen references. Thus

$$I(T_3) = \frac{W_1}{W_1 + W_2} \quad (4)$$

is defined to be the similarity coefficient between reference sets 1 and 2, which are respectively corpus T_1 and corpus T_2 . Let us point out that given the previous assumptions, $I(T_1) = 0$ and $I(T_2) = 1$; furthermore, any given corpus T_3 is then awarded a score between the extrema $I(T_1) = 0$ and $I(T_2) = 1$

This framework may be applied to the quantification of the similarity of large corpora, by projecting them to a scale defined implicitly via the reference data selection. In this study we specifically focus on a scale of similarity bounded by a sublanguage of spoken conversation on the one hand, and a sublanguage of written style media on the other.

2.2 Experimental data used

To set up a scale of similarity between spoken conversation style data and written style documents, we need to select reference data which shall implicitly bound the scale.

For the sublanguage of spoken conversation we used for both English and Japanese languages the SLDB (Spontaneous Speech Database) corpus, a multilingual corpus of raw transcripts of dialogues described in (Nakamura et al., 1996).

For the sublanguage of written style media, we used for the English language a part of the Calgary[2] corpus, familiar in the data-compression field, containing several contemporary English literature pieces[3], and for the Japanese language a corpus of collected articles from the Nikkei Shinbun newspaper[4].

The large multilingual corpus that is used in our study is the C-STAR[5] Japanese / English part of an aligned multilingual corpus, the Basic Traveller's Expressions Corpus (BTEC).

Statistical aspects for each corpus are shown in Tables 1 and 2 for English and Japanese.

A prerequisite of the method is that levels of data transcriptions are strictly normalized, so that the comparison is not made on the transcription method but on the underlying signal data itself.

2.3 A comparison with other existing similarity measures

As mentioned in Section 2.1, a number of similarity measures have been investigated, which make use of linguistic feature counts such as the frequency lists of words or lexemes. Such methods assume that the word is a well-defined unit, or rely on the use of segmenters when dealing with languages in which text is not segmented into words. We wish to compare our proposed method to two measures based on feature frequency computation, which have been previously applied to English corpora in past literature : Chi Square (χ^2) and Log-likelihood (G^2). Both measures are symmetric, and compare one document to another via their feature frequency lists. The ouput number is interpreted as an

[2]The Calgary Corpus is available via anonymous ftp at ftp.cpcs.ucalgary.ca/pub/projects-/text.compression.corpus .

[3]Parts are entitled book1, book2 and book3.

[4]The use of classical Japanese literature is not appropriate as (older) copyright free works make use of a considerably different language. In order to maintain a certain homogeneity, we limit our study to contemporary language.

[5]See http://www.c-star.org .

English corpora	SLDB	BTEC	Calgary
Word/Sent.	11.27±6.85	5.94±3.25	20.21±15.18
Char./Sent.	64.51±35.95	31.15±17.02	107.70±84.69
Char./Word	5.72	5.24	5.33
Total Char.	1,037K	5,026K	757K
Total Words	181.2K	964.2K	142.2K
Total Sent.	16,078	162,318	7,035

Table 1: Statistical aspects of several English corpora. (Mean ± std. dev)

Japanese corpora	SLDB	BTEC	Nikkei
Char./Stce (Mean)	32.61±22.22	14.45±7.12	44.21±28.34
Total Char.	20,806K	2,426K	2,772K
Total Sent.	84,751	162,318	253,016

Table 2: Statistical aspects of several Japanese corpora. (Mean ± std. dev)

inter-document distance.

2.3.1 Similarity measures in previous works

The Chi Square measure (χ^2), as in (Kilgarriff 2001) : the number of occurrences of a feature that would be expected in each document is calculated from the frequency lists. If the sizes of documents A and B are respectively N_A and N_B, and feature w has been observed with a frequency of $o_{w,A}$ in A and $o_{w,B}$ in B, then the expected value $e_{w,A}$ is :

$$e_{w,A} = \frac{N_A(o_{w,A} + o_{w,B})}{N_A + N_B} \quad (5)$$

and likewise for $e_{w,B}$ for document B. The χ^2 value for the document pair A and B is then computed as follows :

$$\chi^2 = \sum_{i=1}^{n} \frac{(o_i - e_i)^2}{e_i} \quad (6)$$

with the sum over the n features.

The Log-likelihood measure (G^2) : (Dunning 1993) showed that G^2 is a better approximation of the binomial distribution than χ^2, especially for less frequent events. It was shown to work well with documents of various sizes and to allow the comparison of both frequent and rare events. G^2 is the sum of the log-likelihoods G_w^2 of all n features w :

$$\begin{aligned} G_w^2 = 2(&a\log(a) + b\log(b) + c\log(c) + d\log(d) \\ &- (a+b)\log(a+b) - (a+c)\log(a+c) \\ &- (b+d)\log(b+d) - (c+d)\log(c+d) \\ &+ (a+b+c+d)\log(a+b+c+d)) \end{aligned} \quad (7)$$

	Doc.A	Doc.B
w	a	b
$\neg w$	c	d

Table 3: Contingency table for feature w in documents A and B.

a, b, c and d being defined for each feature by the contingency table given in Table 3, so that in the end :

$$G^2 = \sum_{i=1}^{n} G_i^2 \quad (8)$$

Both measures yield a value which is interpreted as the inter-document distance between two documents. Such distances can in turn be transposed in the view of our framework, so as to define similarity coefficients based on G^2 and χ^2 (i.e., character cross-entropy $H_T(A)$ is replaced in our framework by χ^2 or G^2 measures).

2.3.2 Evaluation

In order to compare our method with the alternative similarity coefficients based on G^2 and χ^2, we use the method of Known Similarity Corpora (KSC) as in (Kilgarriff 2001). The comparison will be performed on Japanese, a language without clear word segmentation, so that

text data will first have to be run through an analyser when using G^2 and χ^2 distances. To allow a fair comparison, our method will be applied on raw unsegmented data. We construct three sets of KSCs with the previously described SLDB, BTEC and Nikkei corpora (See Section 2.2) : slices of 10,000 words (or their equivalent in unsegmented data) are taken from each corpus and randomly rearranged so that each KSC set includes different mixes of one pair of corpora. For instance, the KSC set of SLDB and BTEC includes a subset $s10b0$ containing ten slices of SLDB and zero slices of BTEC (100% SLDB, 0% BTEC), a subset $s9b1$ of nine slices of SLDB and one slice of BTEC (90% SLDB, 10% BTEC), and so on. Each subset is made of ten slices and is therefore the equivalent of 100,000 words of data, on which we can produce a number of Gold Standard assertions, such as "$s10b0$ should be ranked with a lower coefficient than $s9b1$ because all its data comes from the corpus SLDB" (if we assume that corpora more similar to SLDB get low coefficients, and more similar to BTEC, high coefficients). Each KSC set is made of 11 subsets of 100 000 words of data. The equivalent of 500,000 words of data is left out to be used as references for distance/entropy estimation in our framework. As in (Cavaglià 2002), frequency lists include the 500 most frequent features in each document (preliminary experiments having shown that best results were achieved for 320 to 640 features).

Once KSC sets have been prepared they are scored on the three coefficients and ranked accordingly. The ranks are then compared to the Gold Standard rankings through the computation of Kappa coefficients, and Spearman rank order correlations. Results are shown in Table 4.

The KSC method has the following limitations to its validity : firstly, it does not compare different language varieties but rather mixes of the same varieties. Secondly, the size of slices may be too small to allow a fair comparison, as one corpus used in a KSC set might include highly heterogeneous parts. All three measures display very high correlations with the Gold Standard rankings. This only tends to confirm their validity as similarity indicators, at least when dealing with mixes of the same varieties of language. The best scores differ depending on the KSC sets, showing no superiority of one measure over the other two. However, our method could be applied to Japanese data with no prior preprocessing, such as word segmentation, which makes its range of application wider than any measure relying on counting linguistic features such as words or lexemes.

2.4 Representing corpus homogeneity

Corpora are collected sets of documents usually originating from various sources. Whether a corpus is homogeneous in content or not is scarcely known besides the knowledge of the nature of the sources. As homogeneity is multidimensional (see (Biber 1988) and (Biber 1995) for considerations on the dimensions in register variation for instance), one cannot trivially say that a corpus is homogeneous or heterogeneous : different sublanguages show variations that are lexical, semantic, syntactic, and structural (Kittredge and Lehrberger 1982).

In this study we wish to implicitly capture such variations by applying the previously described similarity framework to the representation of homogeneity. Coefficients of similarity may be computed for all smaller sets in a corpus, the distribution of which shall depict the homogeneity of the corpus relatively to the scale defined implicitly by the choice of the reference data.

Homogeneity as depicted here is relative to the choice of reference training data, which implicitly embrace lexical and syntactic variations in a sublanguage (which are by any means not unidimensional, as argued previously). We focus on a scale of similarity bounded by a sublanguage of spoken conversation on the one hand, and a sublanguage of written style media on the other.

3 A study of the homogeneity of a large bicorpus: the BTEC

The BTEC is a collection of sentences originating from 197 sets (one set originating from one phrasebook) of basic travel expressions. Here we examine the distribution of the similarity coefficients assigned to its subsets.

Whereas the corpus may be segmented in a variety of manners, we wish to proceed in two intuitive ways : firstly, by keeping the original subdivision, i.e., one phrasebook per subset ; secondly, at the level of the sentence, i.e., one sentence per subset .

Figure 1 shows the similarity coefficient distributions for Japanese and English at the sen-

Kappa	$I_{Entropy}$	I_{χ^2}	I_{G^2}
SLDB-BTEC	0.5	0.7	0.8
SLDB-Nikkei	0.9	0.7	0.7
BTEC-Nikkei	0.6	0.9	0.9

Spearman	$I_{Entropy}$	I_{χ^2}	I_{G^2}
SLDB-BTEC	0.918	0.973	0.990
SLDB-Nikkei	1.000	0.936	0.990
BTEC-Nikkei	0.982	1.000	1.000

Table 4: Kappa coefficients (ten intervals) and Spearman correlation scores of rank orders produced by similarity coefficients based on entropy, χ^2 and G^2 compared to the Gold Standard ranks.

tence and subset level, and Table 5 shows their means and standard deviations.

Coefficient	Japanese	English
Phrasebook	0.330±0.020	0.288±0.027
Line	0.315±0.118	0.313±0.156

Table 5: Means ± standard deviations of the similarity coefficient distributions in Japanese and English.

The difference in means and standard deviation values is explained by the fact that all phrasebooks do not have the same size in lines[6]. The distribution of similarity coefficients at the line level, however similar to the distribution at the phrasebook level, suggests in its irregularities that it is indeed safer to use a larger unit to estimate cross-entropies. Moreover, we wish not to tamper with the integrity of the original subsets, that is to keep the integrity of phrasebook contents as much as possible.

Let us point out that on the phrasebook level, the similarity coefficient has a low correlation on both the average phrasebook length (0.178) and the average line length (0.278) (which does not make it a too "shallow" profiling method). On the other hand, correlation is high between the coefficients in Japanese and English (0.781), which is only to be expected intuitively.

4 Experiments

4.1 Method

This work wishes to reassess the assumption that, for a similar amount of training data, an example-based NLP system performs better when its data tends to be homogeneous. Here we use the representation of homogeneity defined by the similarity coefficient scale to select data that tends to be homogeneous to an expected task. Experiments are performed both on randomly selected data, and on data selected according to their similarity coefficient. The closer the coefficient of the training data is to the coefficient of the expected task, the more appropriate.

We assume that the task is sufficiently represented by a set of data from the same domain as the large bicorpus used, the BTEC. Experiments are performed on a test set of 510 Japanese sentences which are randomly taken from the resource (and excluded from the training set). These sentences are first used for language model perplexity estimation, then as input sentences for the EBMT system. The task is found to have a coefficient of $I_0 = 0.331$. The average coefficient for a BTEC phrasebook being 0.330, the random selection of the test set making sure that the task is particularly in the domain of the overall resource. We examine the influence of training data size first on language model perplexity, then on the quality of translation from Japanese to English by an example-based MT system.

4.1.1 Language model perplexity

Even if perplexity does not always have a high correlation with NLP system performance, it is still a valuable indicator of language model complexity as it gives an estimate of the average branching factor in a language model. The measure is popular in the NLP community because admittedly, when perplexity decreases, the performance of systems based on stochastic models

[6]The BTEC phrasebooks have an average size of 824 lines with a standard deviation in size of 594 lines.

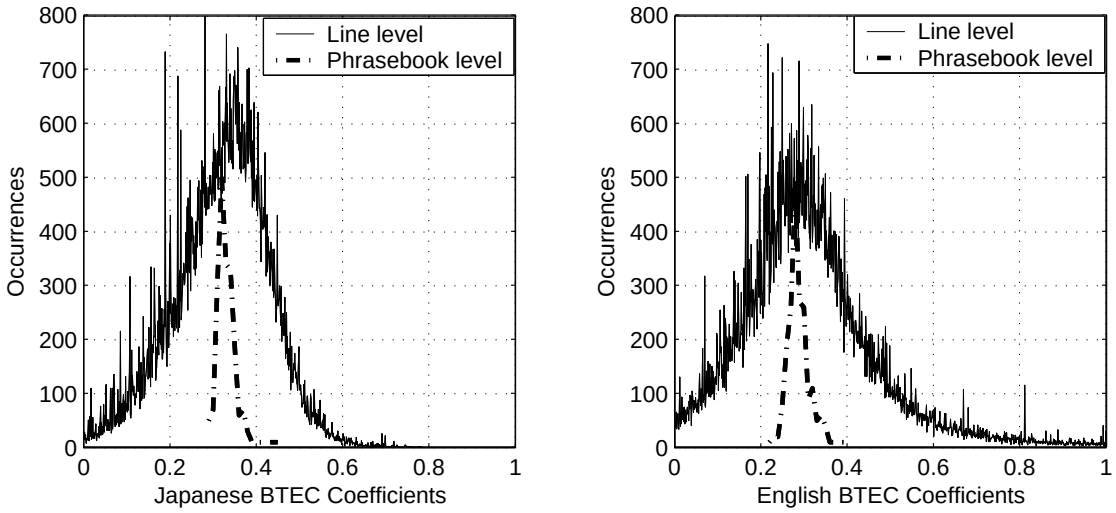


Figure 1: Distributions of similarity coefficients at the sentence level (thin line) and at the phrasebook level (thick line), respectively for Japanese and English.

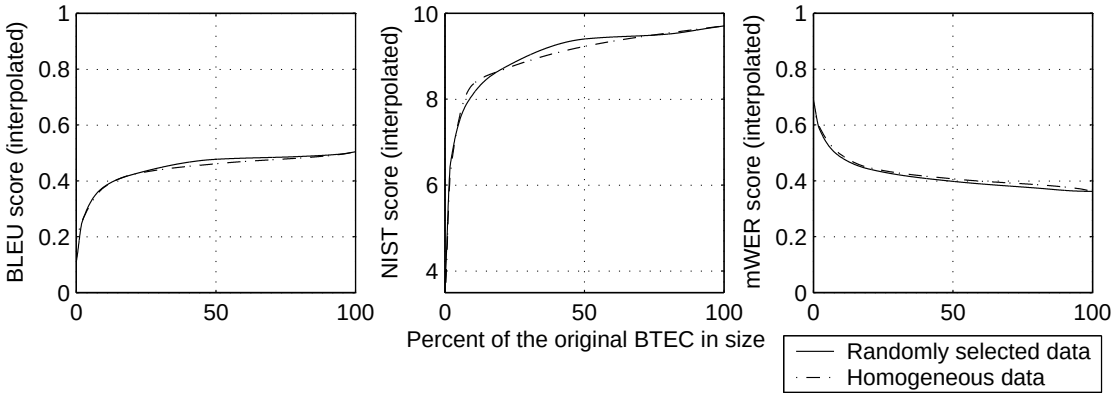


Figure 2: BLEU, NIST and mWER scores for EBMT systems built on increasing amounts of randomly chosen and homogeneous BTEC data.

tends to increase.

We compute perplexities of character language models built on variable amounts of training data first randomly taken from the Japanese part of the BTEC, and then selected around the expected task coefficient I_0 (thresholds are determined by the amount of training data to be kept). Cross-entropies are estimated on the 510 sentence test set, and all estimations are performed five times for the random data selections and averaged. Figure 3 shows the character perplexity values for increasing amounts of data from 0.5% to 100% of the BTEC and interpolated. As was to be expected, perplexity decreases as the amount of training data increases and tends to have an asymptotic behaviour when more data is being used as training.

While homogeneous data yield lower perplexity scores for small amounts of training data (up to 15% of the resource - roughly 1.5 Megabytes of data), beyond this value perplexity is slightly higher than for a model trained on randomly selected data. Except for the smaller amounts of data, there indeed seems to be no benefit in using homogeneous rather than random heterogeneous training data for model perplexity. On the contrary, excessively restricting the domain seems to yield higher model perplexities.

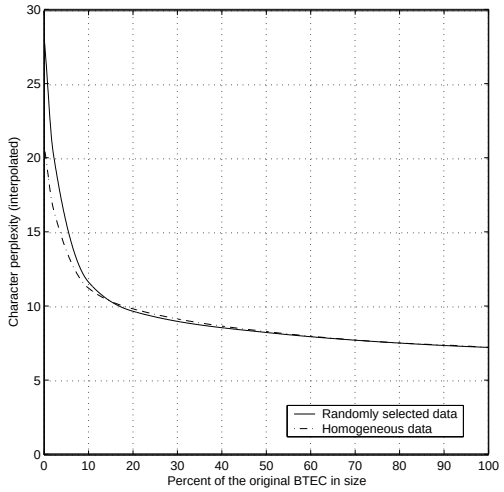


Figure 3: Perplexity of character language models built on increasing amounts of randomly chosen BTEC and homogeneous Japanese data.

4.1.2 Automatic evaluation of the translation quality

In this section we experiment on a Japanese to English grammar-based EBMT system, HPATR (described in (Imamura 2001)), which parses a bicorpus with grammars for both source and target language. Translation is done by automatically generating transfer patterns from bilingual trees constructed on the parsed data. Not being an MT system based on stochastic methods, it is conveniently used here as a task evaluation criterion complementary to language model perplexity.

Systems are likewise constructed on variable amounts of training data, and evaluated on the same previous task of 510 Japanese sentences, to be translated from Japanese to English.

Because it is not feasible here to have humans judge the quality of many sets of translated data, we rely on an array of well known automatic evaluation measures to estimate translation quality :

> BLEU (Papineni et al. 2002) is the geometric mean of the N-gram precisions in the output with respect to a set of reference translations. It is bounded between 0 and 1, higher scores indicate better translations, and it tends to be highly correlated with the fluency of outputs ;
>
> NIST (Doddington 2002) is a variant of BLEU based on the arithmetic mean of weighted N-gram precisions in the output with respect to a set of reference translations. It has a lower bound of 0, no upper bound, higher scores indicate better translations, and it tends to be highly correlated with the adequacy of outputs ;
>
> mWER (Och 2003) or Multiple Word Error Rate is the edit distance in words between the system output and the closest reference translation in a set. It is bounded between 0 and 1, and lower scores indicate better translations.

Figure 2 shows BLEU, NIST and mWER scores for increasing amounts of data from 0.5% to 100% of the BTEC and interpolated. As was expected, MT quality increases as training data increases and tends to have an asymptotic behaviour when more data is being used in training.

Here again except for the smaller amounts of data (up to 3% of the BTEC in BLEU, up to 18% in NIST and up to 2% in mWER), using the three evaluation methods, translation quality when using random heterogenous data is found to be equal or higher than when using homogeneous data. If we perform a mean comparison of the 510 paired score values assigned to sentences, for instance at 50% of training data, this difference is found to be statistically significant between BLEU, NIST, and mWER scores with confidence levels of 88.49%, 99.9%, and 73.24% respectively.

5 Discussion and future work

The contribution of this work is twofold :

We describe a method of representing similarity to reference sublanguages through a cross-entropic measure, that can be used to profile the homogeneity of language resources. Comparing our approach to other existing similarity measures shows similar performance, while extending widely their range of application to electronic data written in languages with no clear word segmentation. A corpus may be represented by the distribution of the similarity coefficients of the smaller subsets it contains, and atypical therefore heterogeneous data may be characterized by the lower occurrences of their values.

We further observe that marginalizing such atypical data in order to restrict the domain on

which a corpus-based NLP system operates does not yield better performance, either in terms of perplexity when the system is based on stochastic language models, or in terms of objective translation quality with an EBMT system.

Having observed that heterogeneous data in a resource may indeed contribute to better NLP system performance, one of our objectives for future work is to study corpus adaptation with Out-of-Domain data. While (Cavaglià 2002) also acknowledged that for minimal sizes of training data, the best NLP system performance is reached with homogeneous resources, we would like to know more precisely why and to what extent mixing In-Domain and Out-of-Domain data could yield better accuracy.

As far as the representation of homogeneity is concerned, other experiments are needed to tackle the multidimensionality of sublanguage varieties less implicitly. We would like to consider multiple sublanguage references to untangle the dimensions of register variation in spoken and written language.

6 Acknowledgements

This research was supported in part by the National Institute of Information and Communications Technology.

References

Douglas Biber. 1988. *Variation across speech and writing.* Cambridge University Press.

Douglas Biber. 1995. *Dimensions in Register Variation.* Cambridge University Press.

Gabriela Cavaglià. 2002. *Measuring corpus homogeneity using a range of measures for inter-document distance.* Proceedings of LREC, pp. 426-431.

George Doddington. 2002. *Automatic evaluation of machine translation quality using N-gram co-occurrence statistics.* Proceedings of Human Lang. Technol. Conf. (HLT-02), pp.138-145.

Ted Dunning. 1993. *Accurate methods for the statistics of surprise and coincidence.* Computational Linguistics, 19(2):219-41.

Kenji Imamura. 2001. *Hierarchical Phrase Alignment Harmonized with Parsing.* Proceedings of NLPRS, pp.377-384.

Adam Kilgarriff and Tony Rose. 1998. *Measures for corpus similarity and homogeneity.* Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing, Granada, Spain, pp. 46 - 52.

Adam Kilgarriff. 2001. *Comparing corpora.* International Journal of Corpus Linguistics 6:1, pp. 1-37.

Richard Kittredge and John Lehrberger. 1982. *Sublanguage. Studies of language in restricted semantic domains* Walter de Gruyter, editor.

Robert A. Liebscher. 2003. *New corpora, new tests, and new data for frequency-based corpus comparisons.* Center for Research in Language Newsletter, 15:2

Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, Yoshitaka Hirano, Hiroshi Matsuda, Kazuma Takaoka and Masayuki Asahara. 2002 *Morphological Analysis System ChaSen version 2.2.9 Manual.* Nara Institute of Science and Technology.

Atsushi Nakamura, Shoichi Matsunaga, Tohru Shimizu, Masahiro Tonomura and Yoshinori Sagisaka 1996.
Japanese speech databases for robust speech recognition.
Proceedings of the ICSLP'96, Philadelphia, PA, pp.2199-2202, Volume 4

Franz Josef Och. 2003.
Minimum Error Rate Training in Statistical Machine Translation.
Proceedings of ACL 2003, pp.160-167.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002.
Bleu: a Method for Automatic Evaluation of Machine Translation.
Proceedings of ACL 2002, pp.311-318.

Richard Sproat and Thomas Emerson. 2003
The First International Chinese Word Segmentation Bakeoff.
The Second SIGHAN Workshop on Chinese Language Processing, Sapporo, Japan.

METIS-II: Example-based machine translation using monolingual corpora - System description

Peter Dirix, **Ineke Schuurman**, and **Vincent Vandeghinste**
Centre for Computational Linguistics
Katholieke Universiteit Leuven
Maria Theresiastraat 21
B-3000 Leuven
Belgium
firstname.lastname@ccl.kuleuven.be

Abstract

The METIS-II project[1] is an example-based machine translation system, making use of minimal resources and tools for both source and target language, making use of a target-language (TL) corpus, but not of any parallel corpora.

In the current paper, we discuss the view of our team on the general philosophy and outline of the METIS-II system.

1 Introduction: Background of METIS-II

The METIS-II project is an example-based machine translation project, which in principle does not make use of parallel corpora. As most other known example-based machine translation (and statistical) systems make use of parallel corpora or bitexts, our system is a new approach towards the automated translation problem (Dologlou et al., 2003), although e.g. Grefenstette (1999) made use of the world wide web in combination with a bilingual lexicon to translate compounds from Spanish and German to English.

We deviced our system to be used in those circumstances where other machine translation systems are not available or of insufficient quality, because of lack of sufficiently large parallel corpora, in general or for the given domain, or because of the unavailability of the desired language pair. This is often the case in the European context as there is a high number of smaller languages.

Building a rule-based system for language pairs involving smaller languages is too costly and time consuming. By building a hybrid system[2], which does not rely on parallel corpora and which does not use an extensive rule set, the METIS-II consortium provides an alternative solution.

For a system like METIS it is therefore not necessary to invest scores of man years into developing a rule-based MT system or several man years into collecting and preparing bilingual corpora. METIS should work just using basic resources. The way the system is designed, however, should allow for the use of more advanced resources as well. It should for example allow the use of a source-language (SL) corpus plus the data that can be distilled from it. It should also allow for integration with a translation memory (TM). Once enough material has been translated and post-edited, such a TM is to be considered a very valuable part of the workflow. Therefore, such aspects should be taken into account when developing the framework. This (automated) TM is not going to be used the traditional way, but during translation itself to build up a parallel corpus containing all SL sentences and their translations (after approval by the user). This will be used as an extra bilingual set of preferred translations that can be selected by the METIS engine. This way the performance of METIS-II when dealing with phenomena like light verbs or prepositional objects may improve quite seriously. The real challenge however is to develop a system to start with for a given language pair or a given domain when little or no other resources are available but a bilingual dictionary and a TL corpus: it should be good enough that people are willing to use it because otherwise there will in the end be no 'parallel' corpus derived from TM to improve the quality of the translations! Therefore, within the current project we are concentrating on developing the main translation tool.

The rationale behind the METIS projects is that a monolingual corpus in the TL, together with a bilingual dictionary guiding the raw lemma-to-lemma translation, should in principle suffice to generate good translations using a combination of statistics and linguistic rules, i.e. a hybrid approach. This monolingual TL corpus is likely to contain (parts of) sentences with the target words in them. Finding and recombining these is in fact what METIS-II is about. Successful development of such a simple tool for a rather complex task could give NLP a real boost in circumstances in which little resources are

[1]Project FP6-IST-003768 funded by the IST in the 6th Framework.

[2]EBMT systems are often hybrid, incorporating some rule-based and statistical methods (Somers, 2003). In this case, e.g. the chunker is rule-based.

available : tasks for which parallel corpora and other expensive resources were thought to be indispensable, are then proved to be feasible without them.

Although the languages involved in METIS-II (Dutch, German, Greek, and Spanish as SL, English as TL) do not really belong to the smaller languages referred to above, we refrain from using such resources that are usually only available for the larger languages. The system therefore needs to be designed in such a way that it can be used for other (Indo-European) languages by plugging in the appropriate language-dependent modules. Therefore, we make use of resources that either will already be available for most languages, smaller ones included, or can be developed rather easily and at low cost.

Next to the bilingual dictionary and a TL corpus we also make use of

1. a tokeniser,
2. a part-of-speech tagger,
3. a chunker,
4. a lemmatiser/morphological generator (Carl et al., 2005).

In case the TL corpus is not yet tagged, chunked and lemmatised, this should be done as well, meaning that tools for doing so (1 - 4) should also be available for the TL. We are using the BNC as TL corpus, which is already tagged but not yet chunked and lemmatised. So we need a chunker and lemmatiser for English as well.

The approach described below differs from the one adopted in METIS-I in that

- sentences are cut up in smaller chunks;
- linguistic information is also used outside the mapping rules;
- the TL corpus is indexed in different ways in order to increase the time efficiency;
- a general-purpose working prototype is built.

In a first stage, the consortium partners conduct separate experiments on different ways of chunking (no chunking, grammatical chunking, n-grams), indexing, and creating a search engine. Other approaches can be found in (Markantonatou et al., 2005) and (Badia et al., 2005).

METIS-II (like METIS-I) targets the construction of free text translations making use of pattern-matching techniques and target-language retrieval from a large monolingual TL corpus. The system's performance and adaptability is enhanced by:

- breaking sentence-internal barriers: the system retrieves pieces of sentences (chunks) and recombines them to produce a final translation;
- extending the resources and integrating new languages;
- using post-editing facilities;
- adopting semi-automated techniques for adapting the system to different translation needs;
- taking into account real user needs, especially as far as the post-editing facilities mentioned before are concerned.

2 Global description of the METIS-II system

When translating a word by meas of the bilingual dictionary, translations one gets are often inaccurate, as it is often the case that one and the same lemma, even when the tag is taken into account as well, may be translated in several ways. In such a case the right choice often depends on its context: the choice of an adjective may depend on the noun it is combined with, and the same holds for the relation between the verb and its object noun or the presence of a determiner before a noun, e.g.:

(1) Ik beschouw Churchill als een groot politicus.
I consider Churchill as a tall/great politician.
I consider Churchill to be a great politician.

In a first step the sentence to be translated is tokenised, tagged, lemmatised and chunked. When all lemmas in the SL sentence have got one or more translations in the TL, one may try to find this 'sentence' as such in the target language. The order of words in the TL often differs from that in the SL. Therefore all translated lemmas are offered chunk by chunk in a bag, i.e. unordered. It is clear that finding the literal translation of the SL sentence in the TL corpus is not very likely to succeed, except for fixed expressions and the like. Therefore, our procedure is implemented in a bottom-up way. First the lowest-level chunks are handed to the search engine to find a match in the SL corpus. One of the tasks in searching the TL corpus is finding the right translation of words (rather: lemmas) on basis of the context, next to the correct order. That is why co-occurence in NPs is so important. In order to translate clauses and whole sentences, the same procedure is applied to combinations of verbs and heads of NPs and PPs (always using the bag-of-lemmas approach), until every level of the

shallow parse tree has been checked with the TL corpus.

To translate expressions, they have to be chunked as such in the SL analysis. The expression needs to be the lowest level of the shallow parse tree and is translated immediately using the expressions section of the bilingual dictionary.

When these are found, the various translations are assigned probability scores, and it depends on these scores which translation is favoured. These scores also determine how the translated string is presented to the end user for treatment during post-editing; unreliable or doubtful translations are marked as such. Before post-editing takes place, postprocessing has been taken care of by the system itself (automatic 'adjustment' of agreement, morphological generation of terms and the like).

In the next sections, we will describe of which modules METIS-II consists, and the requirements that are already clear (as we are still experimenting [cf. (Vandeghinste et al., 2005)], several things are still unclear).

3 General concepts

Before the various modules are described, some more general concepts should be described as these play an important role in our system.

3.1 Universal data format

The idea is to have one universal data format for all the data that go through the system. It is an XML format that can be read and produced by all the modules and tools involved. Each single module picks the parts it is interested in and adds further information when needed. The representation can be piped through the different processes and visualised in the GUI of the user environment. The proposed format is not definitive yet, since the research on the search engine might force us to add additional features.

The representation needs to

1. represent all information added and needed by the different processing modules and tools, e.g
   - reading in the (tagged) source sentence
   - morphological analysis and lemmatisation
   - chunking
   - dictionary lookup
   - add synonyms from other sources, e.g. WordNet[3]
   - apply mapping rules
   - perform syntactical and morphological generation
   - output target-language sentence
2. allow and deal with ambiguities on several levels
   - ambiguity of tags (more possible tags attached to one token)
   - ambiguity of lemmas (more possible lemmas for one token)
   - ambiguity of translations (more possible translations)
   - ambiguity introduced by the tag-mapping rules (the rules have more than one right-hand side)
   - ambiguity of chunks/bags (more possibilities because of tag, lemma, translation and tag-mapping ambiguities)

Each step in the overall process adds or changes a section delimited by XML tags. We use three types of representations, the <s> tags (sequences, i.e. ordered sets of tokens or bags, thus chunks, clauses, sentences), <b> tags (bags, i.e. unordered sets of tokens or chunks) and <t> (tokens). The lowest level of the representations (leaves in the tree) is called a 'token'. The sequences and bags are roots of (embedded) graphs. The type of root tells how the nodes are connected in the subgraphs (ordered or unordered sets). We do not allow cyclic graphs. Tokens do only occur as leaves of the tree and or the lowest-level representation.

3.2 Dictionary format

Every tab-separated dictionary is easily converted to the XML dictionary format by a simple script. We need at least four columns: source-language lemma, source-language PoS, target-language lemma and target-language PoS. The source-language lemma and PoS are represented by <sll> and <slt> tags. The translations are represented by lemma-tag pairs (<tll> and <tlt> tags). Adding additional tags allow for discontinuous units to be represented.

The tags in the dictionary are those of the lemma (i.e. abstracting away from plural etc), unless some tokens are to show up in a particular form (i.e. in fixed expressions). Note that we cannot do with only one column of PoS tags 'because a noun in the SL will become a noun in the TL as well'. Note that the situation is not always that straightforward, for example when one word in the SL is to be

[3] Languages that have not got their own implementation of WordNet, could use bilingual dictionaries to English and the English WordNet to find synonyms and other relations.

translated in several in the TL. But especially when the tag sets of SL and TL are designed in different ways (i.e. form-oriented and function-oriented, resp.) there are many inconsistencies.

The Dutch-English dictionary was compiled from the free Ergane[4] dictionary and the Dutch part of EuroWordNet[5] (Dirix, 2002a). The entries and PoS tags are checked manually. It contains about 110 000 lemma-to-lemma translations.

Example:

(2) zijn oog laten vallen op
one 's eye let fall on
have one's eye on

```
<lx>
    <sll>
        <u i="1"><abstr level="token"></u>
        <u i="2">oog</u>
        <u i="3">laten</u>
        <u i="4">vallen</u>
        <u i="5">op</u>
    </sll>
    <slt>
        <u i="1">VNW</u>
        <u i="2">N</u>
        <u i="3">WW</u>
        <u i="4">WW</u>
        <u i="5">VZ</u>
    </slt>
    <tll>
        <u i="1">have</u>
        <u i="2"><abstr level="token"></u>
        <u i="3">eye</u>
        <u i="4">on</u>
    </tll>
    <tlt>
        <u i="1">VV?</u>
        <u i="2">DTS</u>
        <u i="3">NN1</u>
        <u i="4">PRP</u>
    </tlt>
</lx>
```

The <u> tags represent continuous units. In this case, all Dutch words can permute and have to be in separate units. We use <abstr> in order to abstract the possessive pronoun in Dutch and in the English, since the expression can be used for all persons and both numbers. Abstraction cannot only be done on the token level, but also on the phrase or clause level.

3.3 Weights

Every step in the translation process which leads to ambiguities takes all the alternatives into account and applies a weight to each of these solutions. Introducing weights allows for disambiguation and choosing the most likely translation.

When tagging is performed, TnT provides us with probabilities of alternative tags, which we will use as weights. To get the weights of the different shallow parse trees, we multiply the weights of the tagged tokens for that shallow parse tree, and assign this product to the shallow parse tree. The weights of the tagged tokens are then set to 1.

Lemmatisation occasionally leads to different alternatives. When this is the case, the same type of weight assignment is applied as above.

When a lemma is looked up in the bilingual dictionary, this can result in several alternative translations. For now, we assign an equal weight to these alternatives, but in a later stage we might apply weights based on the frequency of the alternative. Experiments will have to show if this improves the average translation quality.

When matching a bag with a corpus entry, we use the frequency of that corpus entry divided by the total frequency of all the matching corpus entries.

The total weight of a translation alternative will be the product of all the above mentioned weights. The user will also be able to tune the weights of different PoS and sub-PoS categories, e.g. assign a lower penalty for not translating articles or light verbs. The end user can also set the weight assigned to using phrases stored in the TM.

3.4 Mapping rules

Mapping rules are used to perform changes between SL and TL tokens and strings, or to relate such tokens and strings. An example of the latter are the tag-mapping rules. Other mapping rules may insert, delete, modify or permute tokens and strings. An example of insertion is *do-support*, which as a consequence also modifies the appearance of other tokens (him, see , John, ? $\rightarrow$ do, him, see, John, ?). The tag sets used in SL and TL are likely to be different. As we are to know which tag in the SL tag set corresponds to which tag in the TL tag set, we are to draw a table in which equivalent tags are related (one-to-one, many-to-one or one-to-many). When translating between Dutch and English, using the CGN tag set (Van Eynde, 2004) for Dutch and the CLAWS5 tag set[6], over 300 CGN tags are to be related to some 70 CLAWS5 tags. This means that in quite a number of cases several CGN tags are to be mapped onto one and the same CLAWS5 tag, although there are also a number of cases in which it is the other way around. This is also because of

[4] http://www.travlang.com/Ergane/

[5] (Vossen et al., 1999)

[6] Cf. http://www.comp.lancs.ac.uk/ucrel/claws5tags.html.

the fact that the CGN tag set is form-based and the BNC tag set is function-based.

The attributes of the Dutch tag for possessive pronouns (`VNW(bez)`) cover distinctions in person, number and form reduction (63 different combinations in total), while CLAWS5 make no subdivisions and has only one tag (`DPS`). On the other hand, a simplex tag for singular present tense (1st person) `WW(pv,tgw,ev)` is related to a series of tags via the tag-mapping rule

`[V-1] WW(pv,tgw,ev)` → `VBB,VDB,VHB,VVB,...`

since the BNC makes a distinction between auxiliary verbs (*to be*, *to have*, and *to do*), modal auxiliary verbs and other verbs.

Which of these CLAWS5 tags turns out to be the correct one for a given verb, is deduced from the lexicon. For *ben*, for example, the correct tag to be related to the Dutch one will be `VBB`, as the lemma of *am* has a `VB?`[7] tag associated with it in the lexicon.

4 Global translation flow

A sentence to be translated is to go through the following modules (cf. Figure 1):

4.1 SL analysis

4.1.1 Tokeniser

All language resources (source and target) should be in UTF-8. If a language is using non-Unicode compliant tools or resources (tokeniser, tagger/lemmatiser) the METIS main engine will pipe the tool input and output text through a converter before and after the process, so that no information is lost. Input information should be in text format, if desired with XML-compliant markup. Database servers storing the lexicon and corpus tables should also have UTF-8 as their default character encoding.

A source-language corpus may be preprocessed in order to get additional linguistic information about the source language, e.g. about frequencies of collocations.

The tokeniser takes a SL sentence as input. One of its tasks is the separation of words and punctuation. The tokeniser adds tags marking words and sentences.

Another task is the identification of continuous multiword units (MWUs). These may be compound prepositions (such as the English *in line with*), conjunctions (*as far as*), adverbs (*time and again*), determiners (*a lot of*), named entities (*Lernout & Hauspie*), or expressions in a foreign language (*a priori*). Furthermore, for Continental Germanic languages, the tokeniser must be able to recognise the constituting full words in cases like *in-en uitvoer*, short for *invoer en uitvoer* (import and export), *treinbegeleiders en -bestuurders* short for *treinbegeleiders en treinbestuurders* (train guards and train drivers), *LehrerInnen*, short for *Lehrer und/oder Lehrerinnen* (male and/or female teachers).

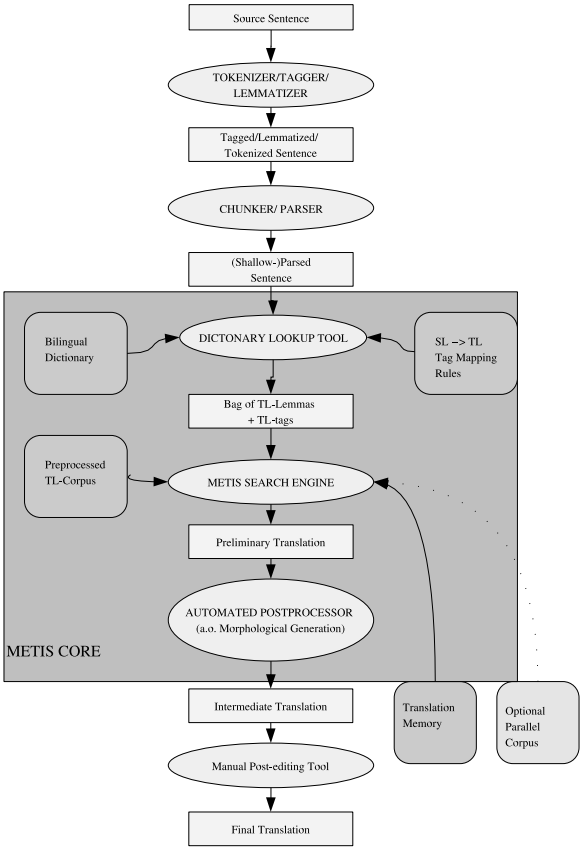


Figure 1: Data flow

4.1.2 PoS tagger

Any tagger can be used, the same holds for the tag set. In case the tagger used provides probabilities concerning the tags to be selected, these can be used to adjust the weights. The tagger we use, is the TnT tagger (Brants, 2001), trained on the Corpus Spoken Dutch.

4.1.3 Lemmatiser

Tokens are related to lemmas in order to facilitate searching in the bilingual dictionary.

One of the tasks of the lemmatiser is to relate discontinuous parts of tokens. In Dutch and German there are verbs like *openmaken, aufmachen* (to open), i.e. verbs with separable particles, which

[7] We use question marks for generalisation of BNC tags. `VB?`is a generalisation of all combinations starting with `VB`: `VBB`, `VBD`, `VBG`, `VBI`, `VBN`, and `VBZ`.

may be realised with other words intervening:

(3) a. Hij *maakt* zijn cadeautje *open*.
He makes his present open
He opens his present.
b. Er *macht* sein Geschenk *auf*.

The lemmatiser may come up with more than one lemma for a given token, thus implying that it is ambiguous. Sometimes such a string may be ambiguous from a human perspective as well, sometimes only for the machine (because of lacking world knowledge, for example). An example of the latter:

(4) a. Hij stond te bedelen.
b. He was begging/endowing.

bedelen	8566	WW(inf,vrij,zonder)
bedelen	8557	WW(inf,vrij,zonder)

These are two homonyms (also with different pronunciation), whose inflected forms also differ. Their LemmaID can be used in order to distinguish them, e.g. in the past tense.

bedelde	bedelen	8566
bedeelde	bedelen	8557

4.1.4 Chunker / Shallow parser

In this component the separate parts are identified which will be searched for in the TL corpus. There are in principle two ways to identify such parts:

1. making use of grammatical units (NPs, clauses, ...)
2. making use of statistical units (n-grams)

We are experimenting with grammatical units and use the in-house developed ShaRPa chunker[8].

There are also several ways to enrich these bare chunks: adding information with respect to its head, identify the subject NP, ... Fairly trivial tools can account for this, stating for example for Dutch that the head of an NP is always the last element; or that the subject is always the leftmost NP except when agreement tells otherwise, although we are aware of the fact that this is not always true. The basic idea is that even with such trivial tools the result will be better as without them.

[8]Described in (Vandeghinste, 2005). An evaluation for Dutch is available in (Vandeghinste and Tjong Kim Sang, 2004).

4.2 SL to TL mapping

METIS makes use of flat bilingual dictionaries, i.e. dictionaries consisting of at least a pair of lemmas and their part-of-speech tag. The tokens in the source and/or target language may be complex, for example when a verb comes with a fixed preposition. Even more complex expressions (like complete phrases) may be contained in it, or they may be stored in a separate dictionary, depending on existing resources. The tags can be used as they are, i.e. with the original PoS tags. These are mapped onto the one used in the target corpus, i.e. the BNC. This is done making use of the mapping rules.

zijn	WW	→	*be*	VB
zijn	VNW	→	*his*	DP

In case of homonyms with different LemmaIDs one has to add this additional information in order to link the proper lemmas.

portier	N	→	79808	*door*	NN
portier	N	→	79809	*doorkeeper*	NN

One could also make use of some of the information stored in the tags to distinguish such lemmas. In this case we could have used

portier	N(onz)	→	*door*	NN
portier	N(zijd)	→	*doorkeeper*	NN

In many cases one token in the SL can be translated in several ways in the TL. All these translations are to be taken into account. Furthermore, sometimes a series of tokens get a special translation as an idiomatic expression. One of the problems in this respect is that these tokens can be realised in a discontinuous way, and some of them are subject to variation (for example when a reflexive pronoun is involved).

The number of possible translations is reflected in the probability scores. These can be determined, for example by the number of translations in the bilingual dictionary, a score in the dictionary or with respect to their use in the TL corpus.

At this point, tag-mapping rules could apply. In the METIS-I project[9], we developed a set of tag-mapping rules to transform PoS categories and verbal tenses from Dutch CGN format to English BNC format (Dirix, 2002b).

[9]Predecessor of METIS-II, IST-2001-32775.

4.3 TL generation

4.3.1 Preprocessing of the TL corpus

Thus far the developer could make use of his own in-house resources (tagger, chunker, bilingual dictionary etc.). From here on he has to make use of the tools developed by the METIS-II consortium, in order to arrive at the proper translations, the most important tool of these being the METIS Search Engine. First we will say something about the way the TL corpus is to be preprocessed.

The TL corpus should be tokenised, tagged, lemmatised and chunked. The corpus should be preprocessed at the same level as the input sentence. One could use the same tools as for the SL. Of course the tools should be adapted to deal with the TL if necessary. When the corpus was already prepared in all these respects, one has to verify whether the results are compatible with what has been done for the SL. If not, one may have to write some wrappers, mapping rules or the like. In order to be able to perform a fast search in the TL corpus, it has to be preprocessed in other ways as well, using indexing and drawing frequency tables out of the corpus. Many statistics can be made based on the TL corpus. They can be used at several points during the translation. The same preprocessing steps are to be executed for the TM and the parallel corpus, if available and used in the system.

The consortium will make use of collocation statistics in order to find out which tokens frequently come together. Tables with often co-occuring lemmas are being derived. These may help to weed out the most unlikely translations of tokens in a sentence (or rather: to give them the most appropriate weights) when several translations are possible. This way one may reduce the number of possible translations offered to the search engine

A fast way of searching in the TL corpus is necessary. One way to do so is to convert the preprocessed corpus into a database. This can be done in several ways: all NPs are indexed on the head noun, all sentences are indexed on the main verb, and so on.

In order to determine the order in which the chunks found are to be combined to derive a correct sentence, one could make use of templates. These are derived from the TL corpus, for example by replacing all NP chunks by the label NP (and possibly some information about the missing NP, e.g. its lexical head), and the same for other types of chunks.

4.3.2 Search engine

The METIS-II translation engine as such is to be a very modular one. The kernel and the language-specific modules can be written in any programming language (Java, Perl, C, ...), as long as input and output conform to the universal data format. The METIS Search engine should be able to take a bag of TL lemmas and TL tags as its input, and look it up in the preprocessed TL corpus.

As we also want METIS-II to cope with several kinds of texts, we have to anticipate several switches and slots: for the domain-specific translations, specialised term databases have to be connected, grammar checkers (source and target), etc. Thus the system needs to be open to a certain extent. What the language-specific modules will exactly look like, depends on the outcome of the current experiments.

After SL analysis, the resulting shallow parse tree is processed depth-first.

In order to translate the first node in the shallow parse tree, each of the daughters of this node is translated first. When this concerns a lemma, this lemma is looked up in the bilingual dictionary.

All daughters of a node are put in a bag and this bag is matched with the TL corpus which is preprocessed up to the same level as the original SL node (for instance up to the basic NP level).

When all daughters of the shallow parse tree have been translated, all these translations are put in a bag. We try to find a match with the TL corpus, where we use the heads of each node to find the best match.

Syntactic generation could be considered a subpart of the translation engine, i.e. the part in which all the parts and pieces found by the search engine will be combined in order to yield correct translations (serialisation).

At this stage, we have an engine for NP translations (cf. Vandeghinste et al., 2005). It uses an indexed database of NPs drawn from the BNC, and tries to match the bags of translated words with the database. The same procedure will be used on a more abstract level (lemmas substituted with tags or generalised expressions) to find whole clauses and sentences.

4.3.3 Automated postprocessing

Although the parts of the translation are already in a correct order, some other phenomena still need to be taken care of: the combinations of lemma and tag are to be realised as tokens (Carl and Schütz, 2005), agreement (adjusting number, person, ...). Which phenomena exactly are dealt with here depends on the TL.

Up to here, we have looked up the bags of lemmas in the target language corpus, and retrieved the sentence/clause/chunk structures. This means we need morphological generation. Based on the lemmas

(coming from the target side of the bilingual dictionary) and the tags (coming from the target side of the tag-mapping rules, or the target side of the dictionary), we can generate the target token. In order to be sure that a lemma-tag combination leads to a unique token, we added some features to the CLAWS5 tagset, where we noticed that several tokens could be generated from one lemma-tag combination (see for example the CLAWS5 tag for the past tense forms of the verb *to be*, which is `VBD` for both singular past and plural past). As a result of all these steps, we now have reached the stage of 'intermediate translation'. This should be of a pretty good quality. After postprocessing, the end user is supposed to do some post-editing in order to get a final, proper translation. These translations are to be fed to the translation memory.

5 Conclusions

A first evaluation (see Vandeghinste et al., 2005) concerning the translation of NPs along the lines described in this paper, shows that we are on the right track. In this experiment, a set of 685 NPs (2/3 fiction, 1/3 newspaper) were translated. In almost 58% the translation ranked by the system was a correct one, in another 14% the correct translation was among the other translation alternatives. In quite a number of cases no translation or a wrong one were given due to the coverage of the lexicon (over 37 000 lemmas and over 110 000 entries, which is still too small, and it turns out that many Belgian Dutch words are still lacking). Extending the lexicon is therefore likely to improve our results.
In our approach, in which we are essentially trying to build up the translation of a sentence out of the combination of translated chunks (based on the BNC), the translation of smaller units, like NPs is one of the building blocks.

We are aware of the fact that when translating sentences for example the verb may also influence which translation of an NP (cf. section 2) is to be considered the best one: searching the TL corpus is to guide this.

Breaking up the sentence, one of the major differences with METIS-I, seems to be a successful approach.

6 References

T. Badia, G. Boleda, M. Melero, and A. Oliver. 2005. An *n*-gram approach to exploiting a monolingual corpus for Machine Translation. In *Proceedings of the EBMT Workshop 2005*, Phuket (this volume).

T. Brants. 2001. *TnT - A Statistical Part-of-Speech Tagger*. Published online at http://www.coli.uni-sb.de/thorsten/tnt.

M. Carl, P. Schmidt, and J. Schütz. 2005. Reversible Template-based Shake & Bake Generation. In *Proceedings of the EMBT Workshop 2005*, this volume.

P. Dirix. 2002a. *The METIS Project: Lexical Resources.* Internship Report, KULeuven.

P. Dirix. 2002b. *The METIS Project: Tag-mapping rules.* Paper, KULeuven.

Y. Dologlou, S. Markantonatou, G. Tambouratzis, O. Yannoutsou, A. Fourla, and N. Ioannou, 2003. Using Monolingual Corpora for Statistical Machine Translation: The METIS System. In *Proceedings of EAMT - CLAW 2003*, Dublin, pp. 61-68.

G. Grefenstette. 1999. The World Wide Web as a Resource for Example-Based Machine Translation Tasks. *ASLIB, Translating and the Computer 21.* London.

S. Markantonatou, S. Sofianopoulos, V. Spilioti, Y. Tambouratzis, M. Vassiliou, O. Yannoutsou, and N. Ioannou, 2005. Monolingual Corpus-based MT using Chunks. In *Proceedings of the EBMT Workshop 2005*, Phuket (this volume).

H. Somers. 2003. An overview of EBMT. In M. Carl and A. Way (ed.), *Recent advances in example-based machine translation*, Kluwer Academic Publishers, Dordrecht.

V. Vandeghinste. 2005. *Manual for ShaRPa 2.0.* Internal Document. Centre for Computational Linguistics, Leuven.

V. Vandeghinste, P. Dirix, and I. Schuurman. 2005. Example-based Translation without Parallel Corpora: First experiments on a prototype. In *Proceedings of the EBMT Workshop 2005*, Phuket (this volume).

V. Vandeghinste and E. Tjong Kim Sang. 2004. Using a Parallel Transcript/Subtitle Corpus for Sentence Compression. In *Proceedings of LREC2004*. ELRA. Paris.

F. Van Eynde. 2004. *Tagging and Lemmatisation for the Spoken Dutch Corpus*. Internal report.

P. Vossen, L. Bloksma, and P. Boersma. 1999. *The Dutch WordNet*. University of Amsterdam.

Graph-based Retrieval for Example-based Machine Translation Using Edit-distance

Takao Doi, Hirofumi Yamamoto and Eiichiro Sumita
ATR Spoken Language Communication Research Laboratories
2-2-2 Hikaridai, Kansai Science City, Kyoto, 619-0288 Japan
{takao.doi, hirofumi.yamamoto, eiichiro.sumita}@atr.jp

Abstract

An EBMT system whose translation example unit is a sentence, can produce an accurate and natural translation if translation examples similar enough to an input sentence are retrieved. Such a system, however, suffers from the problem of narrow coverage. To reduce this disadvantage, a large-scale parallel corpus is required, which calls for an efficient retrieval method. The authors propose a method for a sentence-wise EBMT system to efficiently retrieve the most similar sentences using the measure of edit-distance without omissions. The proposed method uses search space division, word graphs and an A* search algorithm. The performance of the EBMT system implemented with the method was evaluated through Japanese-to-English translation experiments using a bilingual corpus comprising hundreds of thousands of sentences from a travel conversation domain. The EBMT system achieved a high-quality translation ability by using a large corpus, and also achieved efficient processing by using the proposed retrieval method.

1 Introduction

An Example-Based Machine Translation (EBMT) system retrieves the translation examples that are most similar to an input expression and adjusts the examples to obtain the translation. The translation example unit is usually a phrase, and the translations of phrases are combined into a translation of the input sentence. Although a phrase looks like a suitable translation example unit because of its generality for covering various sentences, there is a risk of mixing errors or producing unnatural translations while combining phrases into a sentence (Somers, 2003). Such risk, however, can be reduced if the translation example unit is a sentence. A translation example similar enough to an input sentence as a whole results in an accurate and natural translation of the input sentence. Of course, an EBMT system whose translation example unit is a sentence (hereafter, we call this a sentence-wise EBMT system), suffers from the problem of narrow coverage because a sentence is a longer unit and its generality is lower. To achieve sufficiently broad translation coverage by using sentence-wise EBMT, we must prepare a large-scale parallel corpus and, therefore, an efficient method is needed to retrieve translation examples from a large-scale corpus.

In this paper, we propose a retrieval method for a sentence-wise EBMT system, whose measure of similarity is edit-distance. An efficient retrieval method for EBMT has much in common with Translation Memory (TM). Research efforts on TM (Sato, 1992; Cranias et al., 1997; Planas and Furuse, 1999; Baldwin and Tanaka, 2001) focus on filtering algorithms for sentence sets and/or matching algorithms between two sentences. The methods adopting filtering algorithms, first, filter sentences out by using, for example, a clustering technique that applies word vectors. Then, for each of the sentences left as candidates, they repeat the matching procedure between the two sentences, a candidate and the input. Unfortunately, these methods sometimes omit the most similar sentences in the filtering process. On the other hand, this paper proposes an efficient retrieval method without omissions, as a solution for the problem of searching for the sentences with the least edit-distance among a corpus. This method does not repeat the matching procedure between two sentences, but proceeds to match the input sentence and sentences in a corpus concurrently, where the sentences are expressed as a graph.

The following sections give an overview of the target EBMT system, an overview of the proposed retrieval method, a description of the search algorithm of the method, and a performance evaluation.

2 Target EBMT System

2.1 Overview

(Sumita, 2003) proposed the Dp-match Driven transDucer (D^3), a sentence-wise EBMT method, in which translation examples are sentence pairs of source and target languages. When translating an input sentence, the system retrieves the translation examples whose source sentences are the most similar to the input sentence using the measure of edit-distance. Translation patterns are then dynamically generated with consideration of differences between the input sentence and the translation examples. D^3 keeps and retrieves translation examples that are not abstracted more than the word sequences given in a corpus. Furthermore, the changes in the target sentences of translation examples are kept as small as possible while translations are generated. Therefore, natural translations occur if there are examples similar enough to given input sentences.

2.2 Example Retrieval

Among the processing phases of D^3, the example retrieval phase tends to take the greatest portion of the translation processing time. The function of example retrieval is to find the examples with the minimum distance in the bilingual corpus. This distance is measured between word sequences of the input sentence and the source sentence of an example.

The distance between word sequences is defined as *dist* in Eq. (1), which is the edit-distance including a semantic factor. In this equation, L_{input} and $L_{example}$ respectively indicate the number of words in the input sentence and that in the source sentence of the example, while I and D denote the numbers of insertions and deletions, respectively. Substitution is considered as the semantic distance between two substituted words, described as $SEMDIST$. Substitutions are permitted only between the content words of a common part of speech. Following this equation, *dist* is the total value of insertions, deletions and substitutions normalized by the lengths of the word sequences. $SEMDIST$ is defined using a thesaurus and ranges from 0 to 1. Furthermore, $SEMDIST$ is the division of K (the level of the least common abstraction of two words in the thesaurus) by N (the height of the thesaurus) according to Eq. (2) (Sumita and Iida, 1991).

$$dist = \frac{I + D + 2\sum SEMDIST}{L_{input} + L_{example}} \quad (1)$$

$$SEMDIST = \frac{K}{N} \quad (2)$$

If the minimum distance is not small enough, the examples are not useful for translation. Therefore, we use a threshold for the distance. If there are no examples within the given threshold, the retrieval, and furthermore, the whole translation process fails with no output.

3 Proposed Retrieval Method

The function of our target retrieval process is to retrieve every sentence whose edit-distance *dist* against a given input sentence is the least and falls within a given threshold, from among the candidate sentences, which are all of the source sentences in translation examples. This distance is defined by the relation of two sentences and can be calculated using DP-matching (Cormen et al., 1989) between the two sentences. Therefore, the candidate sentences with the least distance can be found by repeating DP-matching between a candidate and the input. However, because the procedure of such a naive algorithm takes time proportional to the number of translation examples, it is difficult to implement real-time processing for translation using a large-scale corpus on ordinary computers. Consequently, we propose an efficient retrieval method using the classification of candidates, word graphs and an A* search algorithm. This method does not make omissions; that is, in the definition of the distance *dist* and a given threshold, the retrieval result of this method is the same set of sentences as the case of using DP-matching sequentially.

3.1 Candidate Set Classification

Candidate sentences are classified by the number of content words and the number of functional words. This makes it possible to filter candidates according to the numbers of content words and functional words in the input sentence and the distance threshold. That is, the least possible distance can be calculated on the assumption that all content words are the same as each other, and all functional words are also the same as each other. The classes of the least possible distance greater than the threshold are filtered out. The smaller the least possible distance of the class, the sooner the search process is applied to the class. If a candidate of distance

smaller than the threshold is found in a class, the threshold is updated with the distance of that candidate. The smaller threshold can filter out more classes. Furthermore, the search algorithm in a class, which is described in Sect. 4, utilizes the precondition that all sentences in a class have the same number of content words and the same number of functional words, and therefore, the same number of words.

3.2 Word Graph

For each group classified by the numbers of content words and functional words, a word graph is composed of all candidate sentences in the class. Figure 1 illustrates an example of a word graph. The word graph is a directed graph and has a start node and a goal node. Each possible path from the start node to the goal node corresponds to a candidate sentence. Common word sequences in multiple sentences share the same edges. The word graph is compressed so that the number of nodes is the minimum by the method of converting finite state automata (Brzozowski, 1962). By using the word graph, the search process scans all the sentences in a class concurrently.

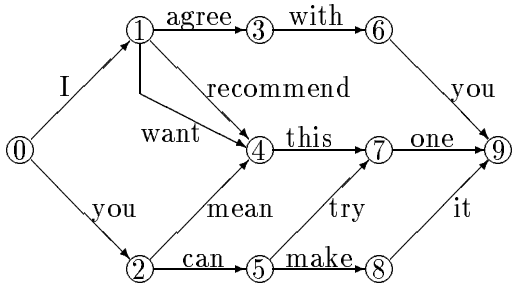


Figure 1: Example of Word Graph

3.3 A* Search Algorithm

The result of matching between two word sequences is represented as a sequence of substitutions, insertions and deletions. We call the result a matching-sequence. The search process in a class is to search for the matching-sequences of the least distance among all possible matching-sequences between the input sentence and each sentence of the class. We use an A* search algorithm (Nilsson, 1971) to solve the search problem.

Generally in an A* search algorithm, the state of the least estimated cost is selected and extended into successor states. In our search problem, the state means an incomplete matching-sequence between the input sentence and a path from the start node to the goal node in a word graph.

4 Search

In this section, we focus on the search process using a word graph. A word graph consists of nodes and edges, and it has a start node and a goal node. An edge consists of a word as label, a source node and a destination node.

4.1 State Space Representation

We represent the search problem using the state structure, operators and the definitions of initial state and goal state.

4.1.1 State

A state has four attributes: paths, node, input and trans, whose contents are as follows.

- `paths` : List of partial matching-sequences.
- `node` : Node in the word graph indicating that matching has proceeded until this node.
- `input` : Partial word sequence of the input sentence not used for matching yet.
- `trans` : Indicator of operators available to this state.

An exact match, a substitution, an insertion and a deletion in matching-sequences of the paths are represented as records including a label and a word or words: (E, *word*), (S, *graph word*, *input word*), (I, *input word*) and (D, *graph word*) respectively. The cost of a state is the cost of an arbitrary matching-sequence in the paths, where matching-sequences have the same cost. The cost of a matching-sequence is the sum of costs of the records in it. The costs are defined as 0 for an E record, 1 for an I record, and 1 for a D record. The cost for an S record is defined as twice the semantic distance between the words in the record, except that it is a small positive value when the distance is 0. [1] The small value gives us the minimum cost of an S record.

4.1.2 Operators

A state is expanded into a successor state by an operator. Five operators are defined as follows. While each T-operator and I-operator is applied to a state, each of the operators, E, S and D is applied to a combination of a state and

[1] This exception is set up in order to distinguish synonymy from identity.

an edge. In the following description, each operator is applied to a state s, if necessary, with an edge e, and s is expanded into a successor state s'. For each operator, we describe the condition where the operator can be applied, and the successor state to be generated as the result of the application.

- `T-operator` :
  - `condition` : s.trans is E-operator or S-operator.
  - `result` : s'.trans = selection from S-operator and NIL (described below) if s.trans is E-operator, NIL if s.trans is S-operator.
    Other attributes of s' are the same as those of s.
- `E-operator`— :
  - `condition` : s.trans is E-operator,
    and s.input is not empty,
    and e.source is s.node,
    and e.label and the head of s.input are identical.
  - `result` : s'.paths is generated by adding an E record to each element of s.paths.
    s'.node = e.destination.
    s'.input is generated by deleting the head of s.
    s'.trans = selection from E-operator, S-operator and NIL (described below.)
- `S-operator` :
  - `condition` : s.trans is S-operator,
    and s.input is not empty,
    and e.source is s.node,
    and e.label and the head of s.input are content words of the same part of speech but not identical,
    and the semantic distance between these two words is smaller than 1.
  - `result` : s'.paths is generated by adding an S record to each element of s.paths.
    s'.node = e.destination.
    s'.input is generated by deleting the head of s.
    s'.trans = selection from E-operator, S-operator and NIL.
- `I-operator` :
  - `condition` : s.trans is NIL,
    and s.input is not empty.
  - `result` : s'.paths is generated by adding an I record to each element of s.paths.
    s'.node = s.node
    s'.input is generated by deleting the head of s.
    s'.trans = selection from E-operator, S-operator and NIL.
- `D-operator` :
  - `condition` : s.trans is NIL,
    and s.paths includes such an matching-sequence whose last record is not an I record,
    and e.source is s.node.
  - `result` : s'.paths is generated from s.paths: first such matching-sequences, whose last records are I records, are deleted; second a D record is added to each matching-sequence left.
    s'.node = e.destination.
    s'.input = s.input.
    s'.trans = selection from E-operator, S-operator and NIL.

In the definitions above, the selection from S-operator and NIL means S-operator if there is a possibility that S-operator can be applied to s', and NIL otherwise. We judge that the possibility exists if the head of s'.input is a content word and there is an edge whose source is s'.node and whose label has the same part of speech but is not identical to the head of s'.input. The selection from E-operator, S-operator and NIL means E-operator if there is an edge whose source is s'.node and whose label is the head of s'.input, and otherwise, the same as the selection from S-operator and NIL. T-operator does not proceed to the actual matching process, but controls the application order of other operators through the trans attribute.

The second condition of D-operator prohibits a D record after an I record. That is, we make it a rule to put a D record before an I record in a sequence of I and D records in order to avoid the redundancy of the multiple appearance of substantially the same matching-sequences.

4.1.3 Initial State and Goal State

In the initial state, the paths attribute is a list of an empty matching-sequence, the node attribute is the start node, the input attribute is

the whole word sequence of the input sentence, and the trans attribute is the E-operator. A goal state is such a state whose node attribute is the goal node and whose input attribute is empty.

4.2 Search Algorithm

From the state space formed using the definitions of the initial state, the operators and the goal states, we search for the goal states of the minimum cost. As an initial condition, an upper limit of cost is given, which is a given distance threshold multiplied by the sum of the lengths of the input sentence and a candidate sentence in the word graph.

4.2.1 Evaluation Function

We define the evaluation function f^* as follows.

$$f^*(s) = g(s) + h^*(s)$$

$g(s)$ is the cost from the initial state to the state s, which equals the cost of state and can be calculated from s.paths. If s is a goal state, $f^*(s) = g(s)$. $h^*(s)$ is the lower limit of cost that is taken from the state s to a goal state.

All sentences in a word graph have the same number of content words and the same number of functional words. Therefore, we can tell the numbers of content words in the input sentence, content words in the word graph, functional words in the input sentence, and functional words in the word graph, which are untreated in the state s. Here, these numbers are represented as C_{input}, C_{graph}, F_{input} and F_{graph} respectively. The lower limit of cost based on the numbers of untreated words is expressed as $h'(s)$ below.

$$h'(s) = |C_{input} - C_{graph}| + |F_{input} - F_{graph}|$$

Furthermore, on the assumption that one of the operators E, S, I and D is applied to the state s where the application of T-operator precedes if necessary, the lower limit of the cost from s to a goal state is expressed as $h''(s, o)$ where o indicates an operator. The value of $h''(s, o)$ is as follows for each operator of o.

- `E-operator` : $h'(s)$.
- `S-operator` : $h'(s)$ plus the minimum cost of an S record.
- `I-operator` : $|(C_{input} - 1) - C_{graph}| + |F_{input} - F_{graph}| + 1$ if the head of s.input is a content word, $|C_{input} - C_{graph}| + |(F_{input} - 1) - F_{graph}| + 1$ otherwise.
- `D-operator` :
  H_c+1 if there is no edge whose source is s.node and whose label is a functional word,
  H_f+1 if there is no edge whose source is s.node and whose label is a content word,
  1 plus the minimum value between H_c and H_f otherwise, where
  $H_c = |C_{input} - (C_{graph} - 1)| + |F_{input} - F_{graph}|$,
  $H_f = |C_{input} - C_{graph}| + |F_{input} - (F_{graph} - 1)|$.

By using these values, $h^*(s)$ is defined as: 1) $h''(s$,E-operator) if s.trans is E-operator; 2) the minimum value among $h''(s$,S-operator), $h''(s$,I-operator) and $h''(s$,D-operator) if s.trans is S-operator; and 3) the minimum value between $h''(s$,I-operator) and $h''(s$,D-operator) if s.trans is NIL.

4.2.2 Algorithm

The search algorithm is described below, where OPEN is a list of unexpanded states and CLOSED is a list of expanded states. The sameness of states in (5) means that two states are the same if they have the same value for each attribute except the paths.

1. set the value of cost upper limit and let OPEN be a list including the initial state alone.
2. terminate unless OPEN has a state of cost within the cost upper limit.
3. remove a state s of the least value of f^* from OPEN and put s into CLOSED.
4. if s is a goal state, keep s as a solution, change the value of cost upper limit with the cost of s and return to (2).
5. expand s into all of its successor states and for each successor state s', if $f^*(s')$ is within the cost upper limit, branch by the conditions:
   (a) if there is no same state as s' in either OPEN or CLOSED, put s' into OPEN;
   (b) if there is the same state as s' whose cost is larger than that of s' in OPEN or CLOSED, remove the same state and put s' into OPEN;

(c) if there is the same state as s' whose cost equals that of s' in CLOSED, remove the same state and put s' into OPEN;

(d) if there is the same state as s' whose cost equals that of s' in OPEN, add s'.paths to the paths of the same state.

6. return to (2).

4.2.3 Optimization

Word graphs tend to have the a larger number of edges originating from the start node than edges originating from another node. Therefore, when D-operator is applied to a state whose node attribute is the start node, many successor states are generated consuming processing time, which is the case when the head of a matching-sequence is a D record. We prepare a series of pseudo edges and nodes originating from the start node to avoid the generation of a large number of successor states. This time, when D-operator is applied to a state whose node is the start node, the state is expanded to a successor state whose node is the first pseudo node. The first pseudo node is the source of edges whose labels are the second words of candidate sentences and the edges flow into the ordinary network. A state of the first pseudo node is expanded into a state of an ordinary node by E-operator or S-operator and into a state of the second pseudo node by D-operator. It can be deduced from the possible maximum distance threshold how many steps of pseudo nodes we should prepare. On the condition that the length of a candidate sentence is L and the length of the series of D records on the head of a matching-sequence is d, the input sentence with the least possible distance is the sentence made from the candidate by deleting d words in the head. Then the distance is $d/((L-d)+L)$. If this distance is greater than the maximum distance threshold Θ, we can give up the search. Therefore, $d/((L-d)+L) \leq \Theta$ is the constraint on d and it deduces $d \leq 2\Theta L/(1+\Theta)$. The maximum integer of d on this condition is the number of steps of pseudo nodes we should prepare.

5 Evaluation

We evaluated the performance of D^3 implemented with the proposed retrieval method through experiments on Japanese-to-English translation in a travel conversation domain using a large-scale corpus.

5.1 Experimental Conditions

We employed a Japanese-and-English parallel corpus, the Basic Travel Expression Corpus (BTEC) (Takezawa and Kikui, 2003). BTEC is a collection of Japanese sentences and their English translations usually found in phrase-books for foreign tourists. The statistics of the corpus are shown in Table 1. For the experiments, a training set of 304,340 sentence pairs and a test set of 510 Japanese sentences were extracted from the corpus. We also used training sets of a half, a quarter, an eighth or a sixteenth the size of the original training set, where a larger set included a smaller set. We call the size of the original set 300K, and others 150K, 75K, 38K and 19K.

Table 1: Statistics of the Corpus

	Japanese	English
# of sentences	404,022	
# of words	2,870,280	2,473,711
avg. sentence length	7.10	6.12
vocabulary size	33,288	22,378

To evaluate translation quality, we employed objective measures and a subjective measure. The objective measures used were the BLEU score (Papineni et al., 2002) and Multi-reference Word Error Rate (mWER) (Ueffing et al., 2002). Sixteen references were used for these measures. Achieving a higher score by BLEU and a lower score by mWER means that the translation results can be regarded as more adequate translations.

For the subjective measure (SM), each translation result was graded into one of four ranks by a bilingual human translator who is a native speaker of the target language, American English. The four ranks were (A) Perfect: no problem in either information or grammar; (B) Fair: easy-to-understand with some unimportant information missing or flawed grammar; (C) Acceptable: broken but understandable with effort; and (D) Nonsense: important information has been translated incorrectly (Sumita et al., 1999). In the experimental results, we present the SM as the cumulative relative frequencies of the evaluation ranks: A, AB and ABC. ABCD is shown as an output rate.

We used thesauri whose hierarchies are based on the Kadokawa Ruigo-shin-jiten (Ohno and Hamanishi, 1984), to calculate the semantic distances. We used a personal computer with Pentium4/2GHz and Allegro Common Lisp 6.2.

Table 2: Training Corpus Size and Performance (θ is the distance threshold)

θ	size	mWER	BLEU	SM (%)			output	time (msec)	
				A	AB	ABC	rate (%)	avg.	max.
1/3	19K	0.4596	0.5448	53.7	65.9	71.0	82.5	62	550
	38K	0.4108	0.5967	58.2	70.6	75.1	86.3	85	930
	75K	0.3833	0.6295	64.7	74.1	79.0	88.8	121	1,690
	150K	0.3401	0.6554	71.2	80.4	83.3	91.8	218	3,310
	300K	0.3198	0.6508	71.2	81.8	84.5	93.3	320	6,650
1/4	19K	0.5060	0.4852	51.0	61.6	64.1	72.2	31	390
	38K	0.4583	0.5579	55.1	65.9	68.8	77.6	41	680
	75K	0.4150	0.6413	63.1	71.8	75.1	82.2	53	530
	150K	0.3602	0.6775	70.2	78.4	80.2	86.5	96	1,600
	300K	0.3349	0.6678	70.4	80.2	82.2	89.4	133	2,040

5.2 Performance

Table 2 shows the evaluation of translation results using the distance thresholds (θ) of 1/3 and 1/4. This table displays the relationship between training corpus size and performance, i.e., translation quality, output rate and processing time. Translation quality increases as the corpus size increases, where the subjective measure and objective measures roughly correspond to each other. When using the lower distance threshold, the processing time is clearly shorter although the output rate naturally decreases. Under three conditions, i.e., the conditions of using the distance threshold of 1/3 and the 150K or 300K corpus, or the distance threshold of 1/4 and the 300K corpus, the rate of rank A exceeds 70% and that of AB reaches above 80%. Under all the conditions, the average processing time is less than 0.4 second. Under the condition of using the threshold of 1/4 and the 300K corpus, where the translation quality is high, the average processing time is about 0.1 second and the maximum time is 2 seconds, indicating that the proposed retrieval method achieves efficient processing.

Figure 2 illustrates the relationship between corpus size and average processing time with axes of logarithmic scale. Although the processing time increases as the corpus size increases, the increasing scale is not linear but about a half power of the corpus size.

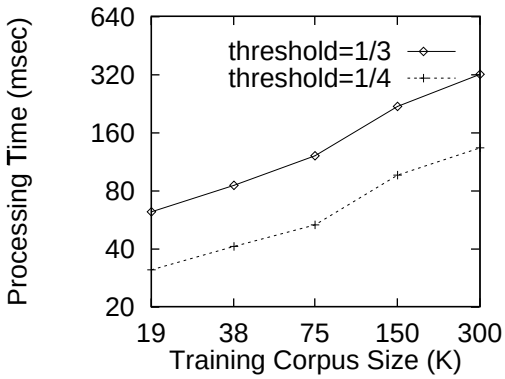


Figure 2: Training Corpus Size and Processing Time

5.3 Comparison with DP-matching

We compared the proposed graph-based retrieval method with naive methods repeating DP-matching. We prepared three methods called Simple-DP, Class-DP and Pruning-DP. Simple-DP uses a hash table to retrieve exactly matched sentences, and if there are no such sentences, repeats DP-matching for an input sentence against all different source sentences in the corpus to find the sentences with the minimum distance. Class-DP improves upon Simple-DP by using the candidate set classification described in Sect. 3.1. Pruning-DP improves upon Class-DP by aborting a DP-matching procedure as soon as the distance between the two concerned sentences is proved to be greater than the minimum distance so far or the threshold. These three methods and the proposed method retrieve the same set of similar sentences for a given input sentence. In the experiment, we executed translations using the retrieval methods and compared their processing time, where the distance threshold used was 1/3.

Table 3 shows the average processing time for each method. Pruning-DP actually improves upon Simple-DP and Class-DP. However, the proposed method far exceeds Pruning-DP. The proposed method is 8.7 times as efficient as Pruning-DP on the 19K corpus and 12.4 times

Table 3: Comparison with DP-matching-based Methods on Average Processing Time (processing time for each method is represented with the unit of milli-second)

corpus size	19K	38K	75K	150K	300K
different sentence#	15,923	29,785	54,657	97,116	199,664
Simple-DP	2,752	4,815	8,101	12,731	26,189
Class-DP	1,286	2,233	3,813	6,045	10,925
Pruning-DP	**539**	**880**	**1,449**	**2,310**	**3,961**
Proposed method	**62**	**85**	**121**	**218**	**320**

on the 300K corpus.

6 Conclusion

We reported on a retrieval method for a sentence-wise EBMT system using edit-distance, and the evaluation of its performance using a large-scale corpus. In experiments for performance evaluation, we used a bilingual corpus comprising hundreds of thousands of sentences from a travel conversation domain. Experimental results show that the EBMT system achieved a high-quality translation ability by using a large corpus, and also achieved efficient processing by using the proposed retrieval method.

Acknowledgements

The authors' heartfelt thanks go to Kadokawa-Shoten for providing the Ruigo-Shin-Jiten. This research was supported in part by the National Institute of Information and Communications Technology.

References

T. Baldwin and H. Tanaka. 2001. Balancing up efficiency and accuracy in translation retrieval. *Journal of Natural Language Processing*, 8(2):19–37.

J. A. Brzozowski. 1962. Canonical regular expressions and minimal state graphs for definite events. *Proc. of Symposium of Mathematical Theory of Automata, MRI Symposia Series*, 12:529–561.

H. T. Cormen, C. E. Leiserson, and L. R. Rivest. 1989. *Introduction to Algorithms*. The MIT Press, London.

L. Cranias, H. Papageorgiou, and S. Piperidis. 1997. Example retrieval from a translation memory. *Natural Language Engineering*, 3(4):255–277.

N. Nilsson. 1971. *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill, New York.

S. Ohno and M. Hamanishi. 1984. *Ruigo-Shin-Jiten (in Japanese)*. Kadokawa, Tokyo, Japan.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. *Proc. of 40th Annual Meeting of ACL*, pages 311–318.

E. Planas and O. Furuse. 1999. Formalizing translation memories. *Proc. of 7th MT Summit*, pages 331–339.

S. Sato. 1992. CTM: An example-based translation aid system. *Proc. of COLING '92*, pages 1259–1263.

H. Somers. 2003. An overview of ebmt. In M. Carl and A. Way, editors, *Recent Advances in Example-Based Machine Translation*, pages 3–57. Kluwer Academic Publishers, Boston/Dordrecht/London.

E. Sumita and H. Iida. 1991. Experiments and prospects of example-based machine translation. *Proc. of 29th Annual Meeting of ACL*, pages 185–192.

E. Sumita, S. Yamada, K. Yamamoto, M. Paul, H. Kashioka, K. Ishikawa, and S. Shirai. 1999. Solutions to problems inherent in spoken-language translation: The ATR-MATRIX approach. *Proc. of 7th MT Summit*, pages 229–235.

E. Sumita. 2003. An example-based machine translation system using DP-matching between word sequences. In M. Carl and A. Way, editors, *Recent Advances in Example-Based Machine Translation*, pages 189–209. Kluwer Academic Publishers, Boston/Dordrecht/London.

T. Takezawa and G. Kikui. 2003. Collecting machine-translation-aided bilingual dialogues for corpus-based speech translation. *Proc. of EUROSPEECH*, pages 2757–2760.

N. Ueffing, F.J. Och, and H. Ney. 2002. Generation of word graphs in statistical machine translation. *Proc. of Conf. on Empirical Methods for Natural Language Processing*, pages 156–163.

Assembling a parallel corpus from RSS news feeds

John Fry
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025-3493 USA
fry@ai.sri.com

Linguistics Department
San José State University
One Washington Square
San José, CA 95192-0093 USA
jfry@email.sjsu.edu

Abstract

We describe our use of RSS news feeds to quickly assemble a parallel English-Japanese corpus. Our method is simpler than other web mining approaches, and it produces a parallel corpus whose quality, quantity, and rate of growth are stable and predictable.

1 Motivation

A parallel corpus is an indispensable resource for work in machine translation and other multilingual NLP tasks. For some language pairs (e.g., English-French) data are plentiful. For most language pairs, however, parallel corpora are either nonexistent or not publicly available.

The need for parallel corpora led to the development of software for automatically discovering parallel text on the World Wide Web. Examples of such web mining systems include BITS (Ma and Liberman, 1999), PTMiner (Chen and Nie, 2000), and STRAND (Resnik and Smith, 2003).

These web mining systems, while extremely useful, do have a few drawbacks:

- They rely on a random walk through the WWW (through search engines or web spiders), which means that the quantity and, more importantly, the quality of the final results are unpredictable.
- Their 'generate-and-test' approaches are slow and inefficient. For example, STRAND and PTMiner work by applying sets of hand-crafted substitution rules (e.g., `english` → `big5`) to all candidate URLs and then checking those new URLs for content, while the BITS system considers the full cross product of web pages on each site as possible translation pairs.
- They sometimes misidentify web page pairs as translations when in fact they are not.
- Good translation pairs are often missed. STRAND, for example, reports recall scores of 60% for some language pairs (Resnik and Smith, 2003).
- Although their source code has not been published, some web mining systems appear to be quite complex to implement, requiring hand-crafted URL manipulation rules and expertise in HTML/XML, similarity scoring, web spiders, and machine learning.

This paper describes a much simpler approach to web mining that avoids these disadvantages. We used our method to quickly assemble an English-Japanese parallel corpus whose quality, quantity, and rate of growth are stable and predictable, obviating the need for quality control by bilingual human experts.

2 Approach

Our approach exploits two recent trends in the delivery of news over the WWW.

The first trend is the growing practice of multinational news organizations to publish the same content in multiple languages across a network of online news sites. Among the most prolific are online news sites in the domain of information technology (IT). For example, CNET Networks (`http://www.cnetnetworks.com`), a large IT media company, publishes stories in Chinese, English, French, German, Italian, Japanese, Korean, and Russian on its worldwide network of IT news sites. Another conglomerate, JupiterMedia (`http://www.jupiterweb.com`), publishes in English, German, Japanese, Korean, and Turkish.

The second trend is the use of RSS, an XML-based syndication format. RSS is increasingly used by both mainstream news web sites (e.g., `wired.com`, `news.yahoo.com`, and the IT news sites mentioned above) as well as sites that provide news-like content (e.g., `slashdot.org` and

Listing 1: Procmail code for creating our corpus

```
1  # .procmailrc file: extracts
2  # parallel URLs from RSS feeds
3  :0 HB
4  * ^User-Agent: rss2email
5  |url=`grep -o http:.*`\
6  ;wget -O - $url\
7  |egrep \
8  '(English)|CNET Networks|
9  target=original|<I>.*N</I></A>'\
10 |grep -o http:.*\
11 |sed -e 's/[ ?"].*//'\
12 |xargs -r echo -e "$url\t"\
13 >>parallel_url_list.txt
```

weblogs). RSS-aware client programs, called news aggregators, help readers keep up with such sites by displaying the latest headlines as soon as they are published. In other words, readers subscribe to the sites' RSS feeds, rather than checking the sites manually for new content.

In many cases, a story published in a target language (say Japanese) will include a link to the original story in the source language (usually English). When the target articles are published over RSS, as they increasingly are, then virtually all the ingredients of a parallel corpus are in place, with no random crawling required.

3 Assembling the parallel corpus

Using RSS feeds in the domain of technology news, we were able to automatically assemble an English-Japanese parallel corpus quickly with little programming effort.

The first step in assembling our corpus was to find web sites that publish Japanese-language news stories along with links to the original source articles in English. Table 1 lists the four RSS feeds we subscribed to. Instead of using a news aggregator, we subscribed to the sites in Table 1 using the open-source `rss2email` program (written by Aaron Swartz), which delivers news feed updates over email.

We then relied on standard UNIX tools like `procmail`, `grep`, `sed`, and `wget` to process the incoming RSS feeds as they arrived by email.

Listing 1 shows our `.procmailrc` configuration file that instructs procmail how to process incoming RSS feeds. First, the URL of the new Japanese story is extracted from the email (line 5), and the article is downloaded (line 6). Next, the link (if any) to the English source article is extracted from the Japanese article (lines 7-11). Finally, both the Japanese and English URLs are saved to a file (lines 12-13).

The regular expression in lines 8-9 of Listing 1 matches text that accompanies a link to the English source article. This is the only part of Listing 1 that is specific to the sites we used (Table 1) and that would need to be modified in order to adapt our method to different languages or web sites.

It should be noted that we do not record the *content* of the parallel news articles. Because material on the Web is subject to copyright restrictions, we cannot publish the content directly. Rather, we record the *URL* of each pair of Japanese and English articles, separated by a tab character. This same format, tab-separated URLs, is also used by the STRAND project for distributing their web-mined parallel corpora (Resnik and Smith, 2003). The STRAND web page (`http://umiacs.umd.edu/~resnik/strand`) offers a short Perl program for extracting the actual content from the URL pairs; this program works for our English-Japanese data as well.

4 Results

4.1 A five-week RSS corpus

We processed the RSS feeds from the Japanese sources listed in Table 1 over a period of five weeks. At the end of the fifth week, we had collected 333 parallel article pairs in Japanese and English. As Figure 1 shows, the bulk of the 333 article pairs were collected from HotWired (133) and CNET (125), followed by pairs from internet.com (65) and IT Media (10).

We then manually inspected all 333 translation pairs to check for problems. One of the URLs we collected, ostensibly a link to an English-language CNET article, turned out to be a stale link. Another link, from a Japanese internet.com article, did not in fact point to an English translation. Finally, two of the HotWired translation pairs were found to be repeats (HotWired occasionally republishes popular articles from the past). We discarded the two bad pairs and the two repeats, leaving us with a final corpus of 329 unique translation pairs.

4.2 Supplementing the corpus by crawling the archives

A corpus of 329 parallel news articles is of course insufficient for most tasks. We therefore recursively crawled the past archives of all four

URL of main news site	RSS feed
http://hotwired.goo.ne.jp	http://www.hotwired.co.jp/news/index.rdf
http://japan.cnet.com	http://japan.cnet.com/rss
http://japan.internet.com	http://bulknews.net/rss/rdf.cgi?InternetCom
http://www.itmedia.co.jp	http://bulknews.net/rss/rdf.cgi?ITmedia

Table 1: RSS feeds used to construct our parallel English-Japanese corpus

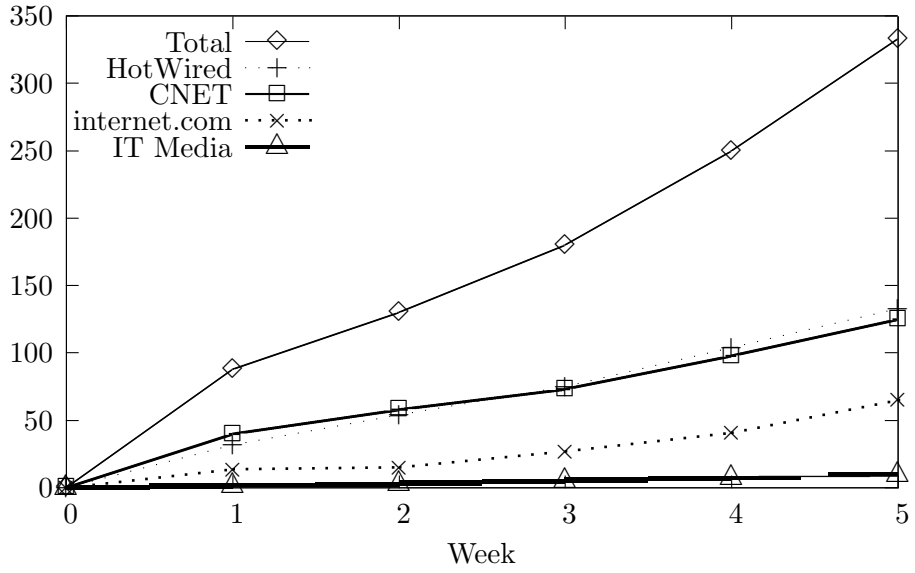


Figure 1: Sources of the 333 parallel English-Japanese article pairs collected over five weeks

HotWired	6,701
CNET	328
internet.com	8,227
ITMedia	2,021
Total	17,277

Table 2: Total article pairs after crawling

web sites in Table 1 using `wget -r` to find article pairs that were posted earlier than those in our five-week experiment with RSS feeds. This crawling netted more than 15,000 additional article pairs. The total count of collected article pairs as of the time of writing (duplicates removed) is shown in Table 2.

As Table 2 shows, the bulk of the article obtained through crawling came from the HotWired and internet.com sites, both of which maintain archives stretching back several years. ITMedia, on the other hand, observes a policy of removing content after one year, and so is not a substantial source for archived material.

4.3 Availability of our data

A regularly updated list of all English-Japanese article pairs we have collected so far can be downloaded from `http://johnfry.org/je_corpus`. At the time of writing, the list holds 17,277 English-Japanese article pairs (see Table 2), and is growing at a rate of approximately 70 pairs per week.

Where possible, our collected URLs point to 'printer-friendly' (as opposed to 'screen-friendly') versions of the content. Printer-friendly versions of news articles are structurally simpler, with fewer banners and advertisements cluttering the story content. In addition, printer-friendly versions typically contain the entire news article, whereas screen-friendly versions are sometimes published over several successive pages, making them more difficult to process. All the HotWired and IT Media articles in our corpus have printer-friendly versions in both English and Japanese. In the case of CNET and internet.com, the English-language articles offer printer-friendly versions, but the Japanese articles do not.

5 Other parallel English-Japanese corpora on the web

Our corpus of 17,277 article pairs is the largest, but not the only, parallel English-Japanese cor-

pus that is freely available on the web. The following are other free sources of English-Japanese data:

- The NTT Machine Translation Research Group offers a set of 3,718 Japanese-English sentence pairs at `http://www.kecl.ntt.co.jp/icl/mtg/resources`
- The OPUS project at `http://logos.uio.no/opus` offers 33,143 aligned Japanese-English sentence pairs, taken from the documentation for the OpenOffice software suite.
- A set of aligned translations of 114 works of literature (taken from Project Gutenberg and similar sources) is available from the homepage of NICT researcher Masao Utiyama at `http://www2.nict.go.jp/jt/a132/members/mutiyama`

A substantial, but not free, source of Japanese-English data is the set of 150,000 aligned sentence pairs collected from newspaper articles and aligned by Utiyama and Isahara (1993). This collection can be licensed from NICT (see the above link to Utiyama's home page).

6 Conclusion

We have demonstrated how RSS news feeds can be used to quickly assemble a parallel corpus. In the case of our Japanese-English corpus, we supplemented the RSS feeds with web crawling of the news archives in order to assemble a corpus of substantial size (17,277 article pairs and growing).

One drawback of our method is that it is feasible only for language pairs with a substantial online news media representation. On the other hand, our approach has two major advantages over web mining systems. First, it is considerably simpler to implement, requiring essentially one extended line of pipelined Unix shell commands (Listing 1). Second, our approach produces a parallel corpus whose quantity, rate of growth, and (most importantly) quality are stable and predictable. The burden of quality control (including article quality, translation quality, and identification of translation pairs) is shifted onto the news organization that publishes the RSS feed, rather than resting on the web crawling system or bilingual human experts.

References

Jiang Chen and Jian-Yun Nie. 2000. Automatic construction of parallel English-Chinese corpus for cross-language information retrieval. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 21–28, Seattle.

Xiaoyi Ma and Mark Liberman. 1999. BITS: a method for bilingual text search over the web. In *Proceedings of MT Summit VII*, September.

Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.

Masao Utiyama and Hitoshi Isahara. 1993. Reliable measures for aligning Japanese-English news articles and sentences. In *Proceedings of ACL-93*, pages 72–79, Columbus, Ohio.

Towards a definition of example-based machine translation

John HUTCHINS
89 Christchurch Road
Norwich NR2 3NG, UK
WJHutchins@compuserve.com

Abstract

The example-based approach to MT is becoming increasingly popular. However, such is the variety of techniques and methods used that it is difficult to discern the overall conception of what example-based machine translation (EBMT) is and/or what its practitioners conceive it to be. Although definitions of MT systems are notoriously complex, an attempt is made to define EBMT in contrast to other MT architectures (RBMT and SMT).

1 Introduction: why a definition is needed

The dominant framework until the late 1980s was what is now known as 'rule-based' machine translation (RBMT). Since then, research has been dominated by corpus-based approaches, among which the primary distinction is made between, on the one hand, statistical machine translation (SMT), based primarily on word frequency and word combinations, and on the other hand, example-based machine translation (EBMT), based on the extraction and combination of phrases (or other short parts of texts).

The overall conception of SMT is now fairly familiar – in essence, all described models derive from the design first formulated in 1988 by the IBM group (Brown et al. 1988).[1] Sentences of the bilingual corpus are first aligned, and then individual words of SL and TL texts are aligned, i.e. brought into correspondence. On the basis of these alignments are derived a 'translation model' of SL-TL frequencies and a 'language model' of TL word sequences. Translation involves the selection of most probable TL words for each input word and the determination of the most probable sequence of those selected words in the TL. The basic units for SMT systems are words; but recently longer segments are being taken into account (see section 8 below.)

The EBMT model is less clearly defined than the SMT model. Basically (if somewhat superficially), a system is an EBMT system if it uses segments (word sequences (strings) and not individual words) of source language (SL) texts extracted from a text corpus (its example database) to build texts in a target language (TL) with the same meaning. The basic units for EBMT are thus sequences of words (phrases).

Within EBMT there is however a plethora of different methods, a multiplicity of techniques, many of which derive from other approaches: methods used in RBMT systems, methods found in SMT, some techniques used with translation memories (TM), etc. In particular, there seems to be no clear consensus on what EBMT is or what it is not. In the introduction to their collection of EBMT papers (Carl & Way 2003), the editors – probably wisely – refrain from attempting a definition, arguing that scientific fields can prosper without clear watertight frameworks, indeed may thrive precisely because they are not so defined.

2 Original conceptions of EBMT

As a preliminary definition, we may identify the basic processes of EBMT as: the alignment of texts, the matching of input sentences against phrases (examples) in the corpus, the selection and extraction of equivalent TL phrases, and the adaptation and combining of TL phrases as acceptable output sentences.

In its original conception (e.g. Nagao 1984), EBMT seems to have been regarded primarily as a means of overcoming the deficiencies of RBMT systems, namely their weaknesses when translating between languages of greatly differing structures, such as English and Japanese, and therefore in generating good quality output – particularly in the treatment of collocations, e.g. translations of *yabureru* in: The bag *was broken* and The president *was defeated* in the election, and the different translations of Japanese *kakeru* according to 'context': *hang* something on a tree, *put* something on/over one's shoulder, *cover* someone or something.[2] Examples were thus to be treated like other SL and TL data, i.e. as tree representations. Hence, input sentences were analysed as far as possible, and transfer using examples was initiated when rules and trees failed.

[1] This SMT 'model' is not the only possibility, but others have rarely, if ever, been investigated.

[2] The first examples come from Nagao 1984, the second ones from Sato and Nagao 1190.

Two tendencies emerged: some researchers (e.g. Sumita et al. 1990) used examples to supplement (improve) RBMT systems and were unsure whether EBMT could or should deal with the whole process of translation, while others (e.g. Sato and Nagao 1990) were encouraged to investigate 'pure' EBMT systems, where the basic process was founded on finding examples of TL sentences "analogous to" input SL sentences, and rules were applied only when examples could not be found in the database.

These two tendencies persist. On one hand, example-based methods are used in what are basically RBMT systems and are essentially seen as developments of the MT tradition, and on the other hand, there is the conviction that EBMT represents in itself a new 'paradigm' – much as SMT researchers argue that their MT architecture represents a new paradigm. (Personally, I am reluctant to use the term 'paradigm' since it suggests the near complete overturn and virtual rejection of all preceding research in the field – as Kuhn (1962) originally conceived the term in connection with theories in the pure sciences, specifically physics. While some SMT researchers may see their approach as completely new, others in recent years have begun to incorporate methods from older periods. In the case of EBMT, most researchers appear to see their efforts as continuations of traditional approaches and readily acknowledge their predecessors. For such reasons, I prefer to refer to new 'architectures' or 'frameworks'.[3])

One argument for exploring EBMT approaches is that since it is based on actual texts, output translations should be more readable and more sensitive to contexts than RBMT systems, i.e. of higher quality in appropriateness and idiomaticity. A second argument is that EBMT systems can be more easily improved, by the addition of more examples from bilingual corpora; whereas the improvement of RBMT systems involves the modification and addition of complex rules and lexical entries. A third is that EBMT does not involve the complexities of lexical and structural transfer found in (most) recent RBMT systems, i.e. that the basic architecture of EBMT is simpler and less prone to failure than RBMT. As a fourth point, it is argued that EBMT can deal with cases of translation involving complex structural differences and subtle lexical choices that RBMT often fails at. In general, the argument in favour of EBMT is its potential to improve the generation of TL sentences.

3 Definitions of EBMT by Somers, and by Turcato and Popowich

As a starting point for approaching a definition of EBMT, we shall consider the article by Harold Somers (1999), reprinted in revised form in the Carl-Way collection. In this excellent overview of EBMT, he provides outline characterisations of the chief processes and methods encountered in EBMT research. These include the content, size and organisation of databases of parallel bilingual text corpora – how they are selected (e.g. for a domain, as controlled texts) and edited (e.g. to reduce redundancy and potentially disruptive 'unusual' examples), how they are aligned, whether texts are tagged, analysed as tree representations, etc. Likewise, there are options in the processes of matching (character based, word based, structure based), measures of similarity (e.g. statistical and/or by reference to thesauri), the adaptation of extracted examples and their 'recombination' to produce TL sentences. He points out that 'recombination', despite its crucial role for EBMT (whose major objective is to generate better quality output than RBMT), is the most neglected area of EBMT research – and the Carl-Way collection (2003) confirms this relative neglect. Finally, Somers outlines the actual and potential applications of EBMT (or EBMT-like) techniques and approaches in other MT architectures, specifically the derivation of dictionaries and grammar rules for RBMT systems, and the role of EBMT in multi-engine and 'hybrid' systems.

Somers rightly points out that the use of what are claimed to be 'EBMT methods' does not mean that systems are EBMT systems. The variety of methods and techniques, of the ways in which they interact, are all indicators of a thriving and productive research framework, but they do not make its definition any easier. What does Somers see as the essence? Firstly, "the use of a bilingual corpus is part of the definition, but this is not sufficient", since almost all current MT research (including RBMT systems) make use of text corpora to define and limit or constrain the range of data they are aiming to cover – at least in the initial stages of development. As a closer definition, Somers offers: "EBMT means that the

[3] It could be argued that corpus-based approaches as a whole represent a new departure in contrast to the preceding rule-based approaches. In so far as previous work is reconceptualised and reformulated in new frameworks the 'sudden' introduction of corpus-based MT in the late 1980s could be termed a 'paradigm shift' in the Kuhnian sense. This could be true, even though corpus-based approaches were quite common in the earliest days of MT research (e.g. the Rand project), before the rise of grammatical formalisms (Bar-Hillel, Harris, Chomsky, etc.) led to the domination of rule-based architectures in MT research.

main knowledge base stems from examples". But, example sentences can be used in RBMT systems as source data from which generalized rules and patterns can be derived,[4] and the databases of SMT systems are also derived from corpora of 'example' texts. A more restrictive and defining characteristic for EBMT is that "the examples are used at run-time". As Somers comments, this definition excludes SMT from the EBMT framework, since the data used in SMT is derived in advance of the translation process. In addition, the 'run-time' condition appears to exclude many of the EBMT systems described in the Carl-Way collection.

In an article following Somers' overview, Davide Turcato and Fred Popowich take issue with Somers' definition. Their aim is to set out a framework for defining the core processes of EBMT, i.e. to identify or isolate what makes a system example-based as opposed to rule-based. First they agree that use of a database of examples in a MT system is in itself no justification for labelling the system EBMT, since (they argue) the ways in which system knowledge is acquired or expressed is irrelevant; what matters is how knowledge is used in operation. On this basis, they compare 'linguistically-principled' EBMT systems and one type of transfer-based RBMT system (lexicalist 'shake-and-bake') – since this type (unlike other RBMT systems) also avoids structural transfer. The aim is to clarify the status of example databases. If EBMT can be shown to be equivalent in operation with a system (such as lexicalist RBMT) which makes no use of an example database, then either EBMT has to be defined in terms which make no reference to an example database or the characterization of EBMT rests upon knowledge acquisition rather than knowledge use – which, with Somers, they have already rejected as a valid defining characteristic. A crucial question is how sentences are decomposed during the EBMT matching process in comparison with decomposition (i.e. analysis) in lexicalist RBMT. Any MT system has to deal with constructions which cannot be translated compositionally; it needs to have access to a repository of 'non-monotonic contexts' (examples). In RBMT, the repository is extracted (created) from dictionary or text sources; in EBMT the repository used in operation *may* also be extracted from the resource (the example database) as 'explicit knowledge'. In this case, the "linguistic information used by EBMT is indistinguishable from the information used by lexicalist MT." However, in other EBMT architectures, there may be direct reference to the example database during the processing of sentences (i.e. during translation). In this case the repository is used as an 'implicit knowledge' database. Turcato and Popowich argue that it is only when EBMT has access to and makes use of the original full database of examples *during* the translation process that EBMT is clearly distinguished from RBMT systems. In other words, the original conception of 'translation by analogy' (as initially proposed by Nagao) represents "the most characteristic technique of EBMT" and it is "the one where the use of entire examples is most motivated." Such complete access can only be available if the EBMT system has *not* already processed examples (as 'explicit knowledge'). In other words, they suggest that the only true EBMT systems are those where the information is not pre-processed, is available intact and unanalysed throughout the matching and extraction processes, i.e. as the systems in the Carl-Way collection using example databases as 'implicit' knowledge during 'run time'. Even such use does not finally define EBMT since 'translation by analogy' could also "in principle… be an extension to a traditional transfer MT system, to solve cases of lexical ambiguity for which no direct evidence is found in a translation database".[5] In effect, Turcato and Popowich imply that a close definition of EBMT is unimportant; the main thing is to make good MT systems.

However, there are two major problems with such conclusions. Firstly, it does not help observers and indeed other MT researchers if it is said by EBMT practitioners themselves that there is no definition of EBMT; they need to know how EBMT differs from other MT architectures. Secondly, restriction of EBMT to the use of 'implicit knowledge' at run time only would seem to be too narrow, since it would exclude much of the research reported in the Carl-Way collection and at recent conferences. On the other hand, to say simply that, in effect, a system is an EBMT system if its authors say it is, is not the answer.

4 EBMT in the context of MT in general

The attempt here to define EBMT starts from a broader perspective, starting from identifying the core processes and components of *any* MT system and how these differ in RBMT, EBMT and SMT.

In any MT system the core must be the process by which elements (entities, structures, words, etc.)

[4] Carbonell et al. 2002 and Lavoie et al. 2001 describe current RBMT systems which induce rules from corpora.

[5] This was the motivation for the EBMT work of Sumita et al. 1990.

of the input (SL) text are converted into equivalent elements for the output (TL) text, where the output text means the same (or is functionally equivalent to) the input text.[6] In all cases there are processes of 'analysis' preceding this core conversion (or 'transfer') and processes of 'synthesis' (or 'generation') succeeding conversion.

1. In RBMT, the core process is mediated by bilingual dictionaries and rules for converting SL structures into TL structures, and/or by dictionaries and rules for deriving 'intermediary representations' from which output can be generated. The preceding stage of analysis interprets (surface) input SL strings into appropriate 'translation units' (e.g. canonical noun and verb forms) and relations (e.g. dependencies and syntactic units). The succeeding stage of synthesis (or generation) derives TL texts from the TL structures or representations produced by the core 'transfer' (or 'interlingual') process.

2. In SMT, the core process involves a 'translation model' which takes as input SL words or word sequences ('phrases') and produces as output TL words or word sequences. The following stage involves a 'language model' which synthesises the sets of TL words in 'meaningful' strings which are intended to be equivalent to the input sentences. In SMT the preceding 'analysis' stage is represented by the (trivial) process of matching individual words or word sequences of input SL text against entries in the translation model. More important is the essential preparatory stage of aligning SL and TL texts from a corpus and deriving the statistical frequency data for the 'translation model' (or adding statistical data from a corpus to a pre-existing 'translation model'.) The monolingual 'language model' may or may not be derived from the same corpus as the 'translation model'.

3. In EBMT, the core process is the selection and extraction of TL fragments corresponding to SL fragments. It is preceded by an 'analysis' stage for the decomposition of input sentences into appropriate fragments (or templates with variables) and their matching against SL fragments (in a database). Whether the 'matching' involves pre-compiled fragments (templates derived from the corpus), whether the fragments are derived at 'run-time', and whether the fragments (chunks) contain variables or not, are all secondary factors. The succeeding stage of synthesis (or 'recombination' as most EBMT authors refer to it) adapts the extracted TL fragments and combines them into TL (output) sentences. As in SMT, there are essential preparatory stages which align SL and TL sentences in the bilingual database and which derive any templates or patterns used in the processes of matching and extracting.

We may note that in practice clear distinctions between stages may not be present, or some stages may even appear to be absent. In many RBMT systems there is a conflation of transfer and generation; some indeed conflate analysis and generation in a single 'transfer' process (in the transformer or 'direct translation' model). In various EBMT systems (or proposals) we see a conflation of matching and extraction – indeed, it could be argued that 'matching' is not a part of 'analysis' since it does not involve decomposition (or rather it follows decomposition) but is an integral part of the core (conversion or 'transfer') stage. In many EBMT systems, analysis may be as trivial as in SMT, consisting simply of the dividing of sentences into phrases or word strings on the basis of 'markers' (e.g. prepositions, conjunctions, punctuation; see e.g. Gough and Way 2004). In most cases, however, parts of the derived segments are further converted into templates or tree structures (i.e. 'normalised') before the matching process.

5 The database

However, the definition is not yet complete. Essential for any translation – a consequence of the aim to maintain 'meaning equivalence' – is access to information about correspondences of vocabulary in the SL and the TL. The information contained in a database may be derived from a variety of resources (bilingual and monolingual texts, bilingual and monolingual dictionaries, grammars, thesauri, etc.)

Before the arrival of corpus-based approaches (SMT and EBMT) it would be assumed that an MT system has to have a bilingual dictionary of some kind and a set of rules to deal (at very least) with differences of word order between SL and TL. In SMT, the dictionary is *largely* replaced by a bilingual text corpus (aligned in order to correlate SL sentences and words and TL sentences and words) and the rules are replaced by information about frequencies of correlations between SL words and TL words ('translation model') and collocations of TL words in texts ('language model'). In EBMT the dictionary is *largely* replaced by an aligned bilingual text corpus (the set of 'examples') and the rules are replaced by examples of TL strings in the text corpus. In both SMT and EBMT there may also be supplementary

[6] Meaning equivalence is the aim, but in practice MT output can be useful when falling short of this ideal, e.g. in contexts where readers need only to understand and grasp the 'essence' of messages and/or where output can be edited (post-edited) to produce appropriate and acceptable texts.

use of traditional bilingual dictionaries, and perhaps also of monolingual thesauri. If it is acknowledged that dictionaries represent generalisations of analyses by linguists and language users, culled from previous readings of texts, then bilingual RBMT dictionaries are also derived from text corpora.[7] In this light, the distinctions between RBMT on the one hand and SMT and EBMT on the other regarding the use of dictionaries and bilingual corpora also become secondary.

Can we go further and argue that it is essential also to have access to information necessary for decomposing (analysing) and combining (generating) sentences? Before EBMT and SMT it was assumed that systems require knowledge about the morphology and syntax (and probably also semantics) of both SL and TL. The rules used in RBMT were derived (explicitly or implicitly and indirectly) from observations of pattern frequencies between and within languages. In EBMT and SMT, information about well-formedness of sentences and strings is implicitly incorporated in the bilingual databases. The information is implicitly 'extracted' for matching and conversion in so far as input strings have to conform to the practices of the SL, otherwise matches will not be found. Likewise information is implicitly utilised in the synthesis stages by reference to a monolingual 'language model' (in SMT) and by the extraction of well-formed TL fragments (in EBMT). In sum, knowledge about sentence formation, explicit in RBMT, is still present implicitly in EBMT and SMT.

6 The essence of EBMT: a definition

If it is agreed that the essence of any MT system is to be located in the method(s) used to convert a SL string into a TL string, then this would locate the defining essences of MT architectures where they are most distinctive. RBMT systems are commonly distinguished by whether SL-TL transformation operates via an intermediary language-neutral representation (interlingua-based MT), via structure transduction from SL representation to TL representation (transfer-based MT), or via piece-by-piece conversion of SL fragments into TL fragments using dictionaries and rules ('direct translation' or transformer-based MT). Likewise, the comparable operation in SMT is the 'translation model' based on statistics derived from bilingual corpora which substitutes TL words or phrases for SL words or phrases. In TM systems, the comparable operation is performed by human translators who select equivalent TL phrases from the possibilities presented to them in a database (the translation memory).

In EBMT, therefore, the essence is the matching of SL fragments (from an input text) against SL fragments (in a database) and the extraction of the equivalent TL fragments (as potential partial translations). In this light, whether the 'matching' involves pre-compiled fragments (templates derived from the corpus), whether the fragments are derived at 'run-time', and whether the fragments (chunks) contain variables or not, are all secondary factors – however useful in distinguishing EBMT subtypes (as Carl and Way (2003) in their collection). Input sentences may be treated as wholes, divided into fragments or even analysed as tree structures; what matters is that in transfer (matching/extraction) there is reference to the example database and not, as in RBMT, the application of rules and features for the transduction of SL structures into TL structures. Consequently, the 'analysis' of SL input is secondary, its form dependent on the way examples are treated in the core 'transfer' process (and therefore stored in the database). Likewise, it can be argued that the operations of synthesis ('recombination'), perhaps the most difficult and complex in EBMT systems, are a consequence of the nature of the output from the matching/extraction process, i.e. because the input has been decomposed, because what are extracted from the database are not full sentences. Likewise, the alignment of bilingual corpora is a secondary process since it is a consequence of the requirement that the matching process has available sets of corresponding SL-TL fragments. Finally, in this framework, the use of variables, the use of 'fuzzy matching', of templates and patterns, etc., are all ancillary techniques in relation to the core EBMT process.

In the light of the definition being put forward here, the distinctions made by Turcato and Popowich (2003) between 'run time' EBMT and other systems are also secondary. In 'run time' systems, the full database of examples is made accessible and subject to any manipulation as required during matching and extracting processes (e.g. Sumita 2003). Such use of the database is ancillary (however essential) to the basic operation of converting SL input into TL output. In other EBMT systems, the analysis of the database is made in preparatory operations, before actual SL texts (input sentences) are processed for translation – i.e. as explicitly ancillary operations (e.g. McTait

[7] It follows that, as Somers and Turcato-Popowich point out, RBMT systems could also use bilingual corpora instead of (manually or automatically derived) bilingual dictionaries.

2003). The argument that systems which do not access the whole corpus during translation are not 'true' EBMT systems is no longer valid. What matters is the way SL fragments are converted into TL fragments in the core (transfer) process. The 'knowledge base', how it is derived and how it is structured, is secondary, albeit crucially important. Therefore, EBMT knowledge used during the core process can be either fully prepared in advance as 'explicit knowledge' or it can be adapted (adjusted) to the specific input as 'implicit knowledge' during translation operations. This may have important consequences computationally and for recall and precision in the retrieval and selection of examples, but choice between 'explicit' and 'implicit' knowledge remains secondary (as far as a definition of EBMT is concerned). What we have, therefore, are two sub-types of EBMT, both subsumed in the general framework outlined above. Indeed, if we consider the types of systems described in the Carl-Way collection we have probably more than two sub-types since it seems that clear differences are discernible between systems which use templates or patterns and systems which use derived (tree) structures.

To summarise the definition: MT systems are EBMT systems if the core 'transfer' (or SL-TL conversion) process involves the matching of SL fragments (sentences, phrases, strings) from an input text, the matching of such fragments against a database of bilingual example texts (in the form of strings, templates, tree representations), and the extraction of equivalent TL fragments (as partial potential translations). The databases of EBMT systems are derived primarily from bilingual corpora of (mainly) human translations, and are pre-processed in forms appropriate for the matching and extraction processes performed during translation (i.e. 'run-time' processes). The processes of analysis (decomposition) and synthesis (recombination) are designed, respectively, to prepare input text for matching against the database and to produce text from database output.

7 EBMT and RBMT

The proposed definition does not specify the structure of the 'knowledge base' (the database of examples) or the kinds of representations involved in the core 'transfer' process. However, whatever form they do have – simple 'surface' strings, strings with variables, templates, or structured (tree) representations – the crucial point is that they are derived from actual examples of SL and TL sentences.

However, when these representations are in forms similar to (or even identical with) those found in RBMT systems, the question arises whether their inclusion is stretching the framework of EBMT too far. The more input is analysed and the more structured the examples in the database, the less EBMT appears to differ from traditional RBMT.

There are clearly gradations in what can be accepted as EBMT representations, from unstructured strings with no variables at one end of the spectrum to dependency trees of input and example sentences at the other end of the spectrum. The 'simple' matching of input strings (after segmentation) against unstructured example SL sentences (strings of 'surface' forms) would be obviously accepted as true EBMT (e.g. Somers et al. 1994). Generalizations of strings in the form of sequences of words with variables (e.g. templates such as "I do not care for the X", "X gave the Y his particulars", "Do you want a room costing X dollars?") are seen as reasonable and natural developments designed to improve the recall of suitable examples from the database.

What is 'problematic' in EBMT (as far as defining the framework is concerned) is the analysis of sentences (clauses) as dependency and phrase structure tree representations, whether applied just to input sentences or also to example sentences in the database, or to both (e.g. Watanabe et al. 2003, Menezes and Richardson 2001). It would seem to be acceptable that systems are included within the EBMT framework if parsing is restricted to only one side of SL-TL correspondences, e.g. only to SL sentences in the database or only to their corresponding TL sentences, and if otherwise the system deals with 'surface' strings (with variables).

However, if *all* the processes of a system (pre-processing, input decomposition, matching, extraction, recombination) are based on parses as dependency trees and on comparisons of sub-trees, then what is the difference from tree transduction processes in RBMT systems (e.g. in the Eurotra architecture)? Although these systems stand at the edge of the EBMT spectrum – i.e. by taking generalisation of examples to the extreme – they are still not categorizable as being in effect RBMT systems. The reason is that the processes of tree transduction in these types of EBMT systems are based on comparisons and selections of tree (and subtree) representations which are comprised of lexical items and which are derived from bilingual corpora of SL and TL example sentences. That is to say that the 'transfer' processes are example based because they are performed *with reference to* databases of paired SL-TL sentences and phrases.

By contrast RBMT trees comprise both lexical items and grammatical categories (N, NP, PP, etc.) and trees are converted by rules operating on both lexical and grammatical nodes of trees (and subtrees). In RBMT systems tree transduction is based on rules applied to abstract representations consisting of categories as well as lexical items. RBMT systems may derive (all or some of) their rules from bilingual databases – whether manually or (semi)automatically – but the use of such resources does not make them EBMT systems.

Consequently, even though the processes of decomposition and recombination in such types of EBMT systems are identical to the processes of analysis and synthesis in RBMT systems, there remains a clear dividing line in principle with respect to the core process of 'transfer' – rule-based versus example-based. However, it can be argued that "there is no essential difference between translation examples and translation rules... they can be handled in a uniform way; that is, a translation example is a special case of translation rules, whose nodes are lexical entries rather than categories" (Maruyama and Watanabe 1992: 183). In this view, those EBMT systems with RBMT-like representations and RBMT-like tree processing appear to be variants of traditional RBMT. The uncertainty remains, and perhaps it would be better to refer to such systems as 'hybrids' of EBMT and RBMT.

8 EBMT and SMT

Initially, differences between SMT and EBMT were distinct: SMT input was decomposed into individual SL words and TL words were extracted by frequency data (in the 'translation model'), while in EBMT input was decomposed into SL fragments and TL examples (in the form of corresponding fragments) were extracted from the database. More recent developments of 'phrase-based' and 'syntax-based' SMT models have blurred these distinctions.

In phrase-based and syntax-based SMT systems parsing (i.e. statistical parsing) is performed for a variety of reasons: to improve alignments (e.g. Watanabe et al. 2002), or to facilitate the matching of input strings (rather than just individual words, e.g. Koen and Knight 2003), or to allow for the analysis of input sentences as phrase structures (e.g. Charniak et al. 2003) and matching against parsed sentences in the database.[8] There is thus a similar divergence as in EBMT between systems where parsing is part of the pre-processing stage and where it is (also) part of the analysis (decomposition) and matching stages. However, the SMT systems retain the distinctive use of 'translation models' and 'language models', and most processes remain word- and string-based.

This use in SMT of models based (partially or wholly) on dependency trees rather than surface strings represents a 'convergence' towards those EBMT systems which also operate with parsed representations. As far as phrase-based SMT and EBMT are concerned, it seems that both may be regarded as variants of a single framework. The only residual differences are that while SMT works mainly on the basis of statistical methods, EBMT works mainly on the basis of linguistic (symbolic) fragments and text examples.

9 Conclusion

The need for a definition of EBMT is motivated by the confusing variety of techniques which have been discussed as 'example-based' and the difficulty of locating the essential 'architecture' of EBMT from the great variety of descriptions of EBMT systems. As the last two sections demonstrate also there are some cases where EBMT approaches appear to differ little from those of RBMT and SMT approaches. The attempt to define EBMT is to provide researchers and observers with an 'archetype' (comparable to definitions of RBMT systems which distinguished transfer-based and interlingua-based systems, while in practice few operational systems conformed to the archetype in all details.)

Underlying the definition of EBMT attempted here is that the characteristic feature of EBMT remains the assumption (or hypothesis) that translation involves the finding of 'analogues' (similar in meaning and form) of SL sentences in existing TL texts. By contrast, neither SMT nor RBMT work with analogues: SMT uses statistically established word and phrase correspondences, and RBMT works with representations (of sentences, clauses, words, etc.) of 'equivalent' meanings. Since EBMT occupies an intermediary position between RBMT and SMT and it makes use of both statistical (SMT-like) and symbolic or linguistic (RBMT-like) methods, it is open to a wider variety of methodologies, and it is consequently less easy to characterise and define.

10 Acknowledgements

My thanks to the anonymous reviewers who contributed to improvements in this paper and to Michael Carl and Andy Way who encouraged me to write it.

[8] For a general model for parsing aligned bilingual texts see the work of Dekai Wu (e.g. Wu 2000, and references therein).

11 References

P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, R. M. Mercer and P. Roossin. 1988: A statistical approach to French/English translation. *Second International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*. Pittsburgh, Pennsylvania: Carnegie-Mellon University.

J. Carbonell, K. Probst, E. Peterson, C. Monson, A. Lavie, R. Brown, and L. Levin. 2002. Automatic rule learning for resource-limited MT. In: S. D. Richardson (ed.) *Machine translation: from research to real users, 5th Conference of the Association for Machine Translation in the Americas, AMTA 2002*, Tiburon, CA, USA, October 2002 (Berlin: Springer), 1-10.

M. Carl, and A. Way, eds. 2003. *Recent advances in example-based machine translation.* Dordrecht: Kluwer Academic Publishers.

E. Charniak, K. Knight, and K. Yamada. 2003. Syntax-based language models for statistical machine translation. *MT Summit IX: proceedings of the Ninth Machine Translation Summit,* New Orleans, USA, September 23-27, 2003; 40-46.

N. Gough and A. Way. 2004. Robust large-scale EBMT with marker-based segmentation. *TMI-2004: proceedings of the Tenth International Conference on Theoretical and Methodological Issues in Machine Translation,* October 4-6, 2004, Baltimore, Maryland, USA, 95-104.

P. Koen, and K. Knight. 2003. Feature-rich statistical translation of noun phrases. *ACL-2003*: 41st Annual Meeting of the Association for Computational Linguistics, July 7-12, 2003, Sapporo, Japan

T. S. Kuhn. 1962. *The structure of scientific revolutions*. Chicago: University of Chicago. (International Encyclopedia of Unified Science, vol.2, no.2)

B. Lavoie, M. White, and T. Korelsky. 2001. Inducing lexico-structural transfer rules from parsed bi-texts. *ACL-EACL 2001 workshop "Data-driven Machine Translation",* July 7, 2001, Toulouse, France, 17-24.

K. McTait. 2003. Translation patterns, linguistic knowledge and complexity in an approach to EBMT. In: Carl and Way (2003), 307-338.

H. Maruyama and H. Watanabe. 1992. Tree cover search algorithm for example-based translation. *Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages (TMI-90). Proceedings,* 11-13 June 1990, Austin, Texas, 173-184.

A. Menezes and S. D. Richardson. 2001. A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. *ACL-EACL 2001 workshop "Data-driven Machine Translation",* July 7, 2001, Toulouse, France, 39-46. Repr. in: Carl and Way (2003), 421-442.

M. Nagao. 1984. A framework of a mechanical translation between Japanese and English by analogy principle. In: A.Elithorn and R.Banerji (eds.) *Artificial and human intelligence* (Amsterdam: North-Holland), 173-180.

S. Sato and M. Nagao. 1990. Towards memory-based translation. In: H. Karlgren (ed.) *Coling-90: papers presented to the 13th International Conference on Computational Linguistics*, Helsinki, August 1990; vol.3, 247-252.

H. Somers. 1999. Review article: example-based machine translation. *Machine Translation* 14(2), 113-157. Revised as: An overview of EBMT. In Carl and Way (2003), 3-57.

H. Somers, I. McLean and D. Jones. 1994. Experiments in multilingual example-based generation. *CSNLP 1994: 3rd Conference on the Cognitive Science of Natural Language Processing*, Dublin City University, 6-8 July 1994.

E. Sumita, H. Iida, and H. Kohyama. 1990. Translating with examples: a new approach to machine translation. *Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages (TMI-90). Proceedings,* 11-13 June 1990, Austin, Texas, 203-212.

E. Sumita. 2003. An example-based machine translation system using DP-matching between word sequences. In Carl and Way (2003), 189-209.

D. Turcato and F. Popowich. 2003. What is example-based machine translation? In: Carl and Way (2003), 59-81.

H. Watanabe, S. Kurohashi, and E. Aramaki. 2003. Finding translation patterns from paired source and target dependency structures. In Carl and Way (2003), 397-420.

T. Watanabe and E. Sumita. 2002. Statistical machine translation based on hierarchical phrase alignment. *Ninth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-2002),* March 13-17, 2002, Keihanna, Japan; 188-198.

D. Wu. 2000. Bracketing and aligning words and constituents in parallel text using stochastic inversion transduction grammars. In: J. Véronis (ed.) *Parallel text processing: alignment and use of translation corpora* (Dordrecht: Kluwer Academic Publishers), 139-167.

EBMT by Tree-Phrasing: a Pilot Study

Philippe Langlais†, Fabrizio Gotti†, Didier Bourigault‡ and Claude Coulombe

†Univ. de Montréal
Succursale Centre-Ville
H3C 3J7 Montréal
Canada
rali.iro.umontreal.ca

‡Univ. de Toulouse-Le Mirail
5, allées Antonio Machado
F-31058 Toulouse Cedex 9
France
bourigault@univ-tlse2.fr

Lingua Technologies Inc.
555, Côte-des-Neiges
H3T 2A9 Montréal
Canada
www.linguatechnologies.com

Abstract

We present a study we conducted to build a repository storing associations between simple dependency treelets in a source language and their corresponding phrases in a target language. To assess the impact of this resource in EBMT, we used the repository to compute coverage statistics on a test bitext and on a n-best list of translation candidates produced by a standard phrase-based decoder.

1 Introduction

Phrase-based machine translation is nowadays a popular paradigm. It has the advantage of naturally capturing local reordering and is shown to outperform word-based machine translation (Koehn et al., 2003). The underlying unit (a pair of phrases), however, does not handle well languages with very different word orders and fails to generalise well upon the training corpus.

Several alternatives have been proposed to tackle some of these weaknesses. Matusov et al. (2005) propose to reorder the source text in order to mimic the target word order, and then let a phrase-based model do what it is good at. Hildebrand et al. (2005) show that it is possible to adapt the transfer table of a phrase-based model to the specificity of the text being translated. Simard et al. (2005) detail an approach where the standard phrases are extended to account for "gaps" either on the target or source side. They show that this representation has the potential to better generalise the training corpus and to nicely handle differences such as negations in French and English that are poorly handled by standard phrase-based models.

In this work, we consider a new kind of unit: a Tree-Phrase (TP), a combination of a treelet (TL) and a elastic phrase (EP), the tokens of which may be in non-contiguous positions. Several authors have used treelets as a prime unit to do translation (Gildea, 2003; Ding and Palmer, 2004; Quirk et al., 2005), but mostly with the idea of projecting a source treelet into its target counterpart.

In this study, we do not address the issue of projecting a treelet into a target one, but take the bet that collecting (without structure) the target words associated with the words encoded in the nodes of a treelet will suffice to handle translation. This set of target words is what we call an elastic phrase (EP). An elastic phrase is not only possibly a non-contiguous sequence of words, but also has the characteristic of having "gaps" of arbitrary size, which is not the case for the phrases considered by Simard et al. (2005).

The objective of this study is to show whether a memory populated with TPs can be of help in a translation task. We are in the early stages of this study and, at this time, do not have a full-fledged decoder using these units. For this reason, in this pilot study, we resorted first to compute coverage statistics of a Tree-Phrase memory on a test bitext and then made post-processing experiments on a n-best list produced by a classic phrase-based decoder. Arguably, if we can show (1) that our Tree-Phrases can cover much of the material to be translated as well as a reference translation, and (2) that these coverage statistics can be correlated with indicators of translation quality, then a memory populated with these units may have some interesting potential.

In order to answer these questions, we conducted the following experiment on the French-English Canadian Hansards. We first parsed the French material with a dependency parser called Syntex (Bourigault and Fabre, 2000) which will be briefly presented in Section 2. We collected from this parsed material a set of depth-one treelets that we associated with their target EPs, using a word alignment we computed offline. The main characteristics of this memory are reported in Section 3. Then, we computed several coverage statistics of this memory on a test bitext, employing different

pattern-matching methods. This is reported in Section 4. Finally, we use these coverage statistics in a translation context in Section 5.

2 Syntex

SYNTEX (Bourigault and Fabre, 2000) is a robust and efficient syntactic parser allowing the identification of syntactic dependency relations between words, as well as the extraction of nominal, adjectival and verbal phrases from a corpus. SYNTEX further builds a directed acyclic graph from these phrases, linked to each other by head or expansion relations. Two versions of this software have been created: one for English and one for French.

SYNTEX takes as input a text processed by TREETAGGER[1], a part-of-speech tagger developed at the University of Stuttgart. Some pre- and post-processing of the results from TREETAGGER are made, and through a pipeline of modules of syntactic relation recognition, SYNTEX outputs a number of dependency relations for each sentence.

Currently, the main relation types identified by this tool are subject, direct object, prepositional complement, adjectival modifier, and subordination. Each dependency relation identifies two words: one that acts as a governor, and another one that is its dependent. Each recognition module is "handcrafted" by linguists using the Perl language, and relies on grammatical knowledge and many heuristics to scan a sentence from a candidate governor to find its dependent (or vice-versa), using information from the previous modules.

For example, given the French source sentence "on a demandé des crédits fédéraux" (request for federal funding), SYNTEX outputs several dependency links that we can represent by the structure in Figure 1, where a root node contains the word governing the words of all its child nodes, which are called its dependents. The syntactic dependency relation is presented to the right of the dependent word. Note, however, that we do not consider this information in this work. In this study, SYNTEX was also used to segment sentences into individual tokens, as can be seen in the example in Figure 1.

An example of the output of SYNTEX for the English counterpart of our running example ("request for federal funding") is shown in Figure 2.

[1] http://www.ims.uni-stuttgart.de/projekte/corplex/.

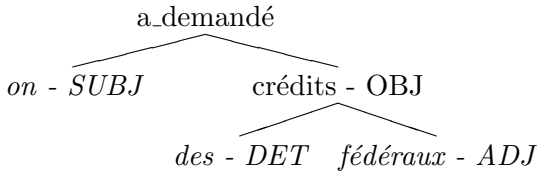


Figure 1: Parse of the sentence "on a demandé des crédits fédéraux" (request for federal funding). Note that the 2 words "a" and "demandé" (literally "have" and "asked") from the original sentence have been merged together by SYNTEX to form a single token. These tokens are the ones we use in this study.

SRC	Request for federal funding
SYNTEX	NOUN?S|request|Request|1|0|PREP;2
	PREP|for|for|2|PREP;1|NOUNPREP;4
	ADJ|federal|federal|3|ADJ;4|0
	NOUN?S|funding|funding|4|NOUNPREP;2|ADJ;3

Figure 2: An example of output from SYNTEX. Each line corresponds to a single SYNTEX token. Some tags have been translated in English to facilitate reading.

3 The Memory

We parsed with SYNTEX the source (French) part of our training bitext, that is, about 1.7 million sentences. From this material, we extracted all dependency subtrees of depth 1 from the complete dependency trees found by SYNTEX. For instance, the two treelets in Figure 3 will be collected out of the parse tree in Figure 1.

Prior to that, the full training corpus was aligned at the word level by the method described in (Simard and Langlais, 2003) which recursively splits in two parts both the source and target sentences and allows either a left-to-right alignment (the first part of the source sentence is aligned to the first part of the target sentence, the second parts are aligned together), or an inverted one (the first source part is aligned to the second target one and vice-versa). The best split found at each step is kept and we further split the two parts until we cannot split anymore (that is, when there is at most one token in one side). The computation of the quality of a split is done using a linear combination of two word models (one for each direction) that have been trained on the same training material. We used an IBM model 2 (Brown et al., 1993) for that purpose, whose parameters were

trained with the GIZA package (Och and Ney, 2000).

An illustration of the output of this alignment procedure is provided for the running example in Figure 3. Once both the word alignment and the treelets are computed, populating the memory with tree-phrases is just a matter of collecting them, and keeping their count over the total training corpus. The format we use to represent the treelets (see Figure 3) is similar to the one proposed in (Quirk et al., 2005): the left and right dependents of a given governor word are listed in order in two separate lists along with their respective offset (the governor/root token always has the offset 0). An elastic phrase is simply the list of tokens aligned to the words of the corresponding treelet as well as the respective offsets at which they were found in the target sentence, relative to the first token position. Note that TLs as well as the EPs might not be contiguous as is for instance the case with the first pair of structures listed in Figure 3.

alignment: a_demandé ≡ request for, fédéraux ≡ federal, crédits ≡ funding

treelets:

tree-phrases:

TL⋆ `{{on@-1} a_demandé {crédits@2}}`
EP⋆ `|request@0||for@1||funding@3|`

TL `{{des@-1} crédits {fédéraux@1}}`
EP `|federal@0||funding@1|`

Figure 3: The Tree-Phrases collected out of the SYNTEX parse for the sentence pair of Figure 1. Non-contiguous structures are marked by a star.

The tree-phrases (TPs) are stored in a database, whose main characteristics are reported in Table 1. Out of 1.7 million pairs of sentences, we collected more than 3 million different kinds of TLs from which we projected 6.5 million different kinds of EPs. Slightly less than half of the treelets are contiguous ones (that is involving a sequence of adjacent words); 40% of the EPs are contiguous. When the respective frequency of each TL or EP is factored in, we have roughly 11 million TLs and 10 million EPs.

We also observe that, as the treelet and the phrase sizes increase, the number of those that are contiguous drops, something that is to be expected.

s	$\|treelet\|$	%-c	$\|EP\|$	%-c
2	639 922	56.8	1 993 896	46.0
3	1 534 468	42.2	3 140 364	38.5
4	737 637	50.5	1 278 254	34.6
5	127 410	53.1	166 465	30.4
6	9 396	36.2	10 108	22.1
7	394	25.9	403	15.4
8	13	0.00	13	7.7
all	3 049 240	47.7	6 589 503	39.8

Table 1: Main statistics measured on the memory as a function of the structure size s. %-c stands for the percentage of structures (TL or EP) that are contiguous. The size of a structure corresponds to the number of tokens it contains. The figures presented here correspond to the number of the different kinds of structures populating the memory and does not account for their respective frequency.

The 5 most frequent tree-phrases as well as examples of very large ones are reported in Table 2. We note that the most frequent tree-phrases are contiguous ones that would have been captured as well by a "standard" phrase-based model.

4 Coverage Analysis

Rationale One way to get an idea of the exhaustiveness of the memory is to compute coverage statistics on a parallel test corpus disjoint from the training one. This will at least give us an idea of how many translation units a hypothetical TP-based decoder would be able to find for a sentence to be translated. A weak source coverage would be disappointing in our case. Moreover, by computing the coverage of the target (reference) sentence with the target material associated with the source treelets found in the previous step, we get a sense of how meaningful the associations stored in the memory are.

We can also evaluate the respective contributions of contiguous and non-contiguous units to that coverage. To do so, we randomly selected 1 000 pairs of parallel sentences from a subset of the Canadian Hansards not included in our training corpus and tried to match them against the units in our database using various matching methods.

Frequent Tree-phrases

freq	treelet	corresponding EP
75 051	{{{monsieur@-2} {Le@-1} président}}	|Mr@0| |.@1| |Speaker@2|
32 601	{{{Le@-1} gouvernement}}	|the@0| |Government@1|
26 347	{{{de@-2} {les@-1} voix}}	|Some@0| |Honourable@1||Members@2|
14 515	{{{Le@-1} ministre}}	|the@0| |Minister@1|
13 043	{{{Madame@-2}{la@-1}Présidente}}	|Madam@0| |Speaker@1|

Long Tree-phrases

|TL|	treelet	corresponding EP
8	{{par@-3} {un@-2} {excellent@-1} Chili {con@1} {carne@2} {servi@3} {Léger@6}}	|culmination@0| |a@2| |Chili@3| |con@4| |carne@5| |feast@6| |provided@7| |Leger@11|
8	{{sur@-2} {la@-1} question {fondamentale@1} {de@2} {à@8} {nationale@14} {de@15}}	|and@0| |on@2| |fundamental@4| |point@5| |to@11| |national @ 15|

Table 2: The 5 most frequent tree-phrases acquired and 2 examples of especially long ones.

Notation We describe here the notation we will use for the coverage analysis. Let S be a source (French) sentence, with n tokens $s_1 \dots s_n$. Let E be a target (English) sentence, with m tokens $e_1 \dots e_m$.

We also define $t_1 \dots t_k$ to be the tokens of the treelet T. $o_1, \dots, o_k$ are their associated offsets (recall that the root of the treelet has an offset of 0). We call r the token index in S at which T is rooted. It follows that s_r = root token of T.

4.1 Match Policies

We experimented with various matching methods between treelets and source sentences and between elastic phrases and target sentences. All of these methods share a criterion: to have a match, the words in the treelet or elastic phrase must be in the same order as those found in the source/target sentence. No token reordering is allowed.

Source match policies (`s-match`) For source treelets, we devised an *exact* (E) and a *relaxed* (R) match policy. We say that the treelet T *exactly* matches S if:

$$\forall i \in [1,k], e_i \equiv s_{r+o_i}$$

For the relaxed policy, all the tokens of T must be found in S, but the offsets constraint is relaxed. That is, to match a treelet T to a sentence S, we must find a strictly monotonous function $f : [1,k] \longrightarrow [1,n]$, such that:

$$\begin{array}{ll} h \rightarrow r & \text{where } o_h \equiv r \\ \forall i \in [1,k], & e_i \equiv s_{f(i)} \end{array}$$

Target match policies (`t-match`) When a treelet matches, its corresponding phrases are retrieved from the memory and matched against the target sentence E. We experimented with three different match policies for phrases. For all these match methods, the search starts at the beginning of E, i.e. at e_1.

With the *exact* (E) match method, we consider that we have a match when we find the phrase verbatim in the target sentence, with the same gaps between each token.

With the *relaxed* (R) method, we have a match if the tokens of the phrase are encountered in the same order in the target sentence, regardless of their offsets. This latter method allows the tokens of a phrase that are only separated by, say, 2 tokens to match a sentence where they become separated by 18 tokens. This goes against our intuition that the word gaps in non-contiguous phrases must not be stretched beyond a certain limit.

We therefore added a third method *relaxed with stretch limit* (R+S), similar to the second one, where we limit the "elasticity" of those gaps to a maximum of 3 times their original size.

4.2 Upper-Bound Coverage

We used the algorithm shown in Figure 4 to compute various source and target coverage fig-

ures. The idea is simple. We proceed in two steps. First, we find the set tl of all treelets `s-matching` the source sentence S. Then, for each treelet T in tl, we find all corresponding elastic phrases which `t-match` the target sentence E. The positions in S and E at which these pairs of corresponding treelet-phrases match are finally marked as "covered" by the algorithm. Any position s_i or e_j may therefore be covered by many units. This is why this algorithm gives us an upper-bound coverage. We will refine the idea of coverage in Subsection 4.3.

for all source tokens s_i of the sentence S **do**
  let tl be the set of TLs with root token s_i
  for all $T \in tl$ **do**
    if T `s-matches` S **then**
      let ep be the set of EPs associated with T in the repository
      for all $p \in ep$ **do**
        if p `t-matches` the target sentence E **then**
          mark the match positions of T and p in S and E as covered

Figure 4: Algorithm to compute the source and target coverage. The two matching functions `s-match` and `t-match` are discussed in the text.

Results Table 3 shows the results we gathered using the six possible combinations of these policies on 1 000 parallel pairs of sentences, corresponding to 17 798 source tokens and 16 219 target tokens. Expectedly, better coverage statistics are achieved when using less constraining methods. We also present there another figure of interest we gathered: the respective contribution of non-contiguous and contiguous units to this coverage. As can be seen, contiguous units account for most of the coverage, which means that a standard phrase-based model would probably have captured the same information. The extra coverage brought by non-contiguous units varies between roughly 10% and 20% (absolute), although it is difficult at this stage to assess how this could have translated into a better MT system.

In all cases, the coverage is very good, with, on average, roughly 75% coverage, both for the source and the target sentences.

4.3 Corrected Coverage

Raw coverage figures as we computed them only give a rough idea of the potential of TPs. The main drawback of our methodology is that *many* different overlapping units (TLs or EPs) are allowed to cover a given source or target sentence token, which might not reflect their true usefulness in a translation task, where, typically, a single translation unit is chosen to help in the translation of a given source token or groups of source tokens.

method		source		target	
src	tgt	%-cov	%-c	%-cov	%-c
E	E	68.70	62.55	71.90	68.63
E	R	69.58	63.21	75.31	70.33
E	R+S	69.10	62.76	73.80	67.66
R	E	79.29	72.35	77.86	74.78
R	R	80.39	73.23	80.85	76.36
R	R+S	79.80	72.72	79.57	73.69

Table 3: Source and target coverage statistics. `%-cov` stands the percentage of tokens that are covered, and `%-c` indicates the percentage of tokens covered by contiguous units.

Source coverage In order to better estimate the situation, we computed a corrected coverage by applying the algorithm in Figure 5.

The idea behind this algorithm is to select the minimum number of TLs covering as much as possible of the source sentence. All 6 combinations of our match policies have been tried for this experiment. We implemented a search algorithm in a way similar to the one embedded in a translation decoder, the main difference being that we do not build a translation, but just find the decomposition of the source sentence into TLs. Therefore the score we optimise is based on the source material only.

Conceptually, the algorithm builds the set of all the *valid* hypotheses that match the source sentence S. A valid hypothesis is a set of treelets that (at least) partially covers S and satisfies a certain number of properties, the main one being that none of the dependencies captured in the set of TLs is allowed to cross another one. Once all such hypotheses are built, the algorithm picks the one with the best score. In our case it is the one which covers S the most with the minimum number of treelets.

In practice, because of the combinatorial nature of the algorithm, these hypotheses are maintained into priority queues $Stack(i)$ that sometimes have to be pruned to achieve an acceptable computation time. The ith stack contains, at most, the b best ranked valid hypothe-

// init
for all $i \in [1, n]$ **do**
  $Stack(i) \leftarrow \phi$
let $tl[i] \leftarrow \{T \in \mathcal{M} \wedge \texttt{s-match}(T, s_i) = \texttt{true}\}$

// the search
for all $i \in [1, n]$ **do**
  for all $T \in tl[i]$ **do**
    $\texttt{add}(\epsilon, T, 1)$
    for all $j \in [1, n]$ **do**
      for all $h \in Stack(j)$ **do**
        if $\texttt{s-extend}(T, h)$ **then**
          $\texttt{add}(h, T, j + 1)$

// the best hypothesis
let *best* $\leftarrow$ the first hypothesis
for all $i \in [1, n]$ **do**
  for all $h \in Stack(i)$ **do**
    if $\texttt{score}(h) > \texttt{score}(best)$ **then**
      let *best* $\leftarrow h$

Figure 5: Algorithm to compute the corrected source coverage. $\mathcal{M}$ is the set of all treelets matching the source sentence S. `add(h,T,n)` is a function which adds in $Stack(n)$ the hypothesis h extended by the treelet T. `s-extend(T,h)` is a predicate which is true if the treelet T can extend the hypothesis h, and `score(h)` returns the score of a hypothesis h. Please read the text for more details.

ses built of i TLs. We used $b = 500$ for our experiments. The first stack (the one with only one TL per hypothesis) is seeded with all the different treelets `s-matching` the source sentence, with one treelet per hypothesis. The algorithm then goes along the source positions and systematically tries to extend previously built hypotheses with all of the treelets rooted at this very source position. A treelet may extend a hypothesis only if it does not introduce dependencies that cross other ones, and if at least one dependency is added to the hypothesis.

An example of the output of this algorithm is given in Figure 6. Notice that, in this example, there were 8 candidate treelets found by `s-matching`, but only 4 were selected by the algorithm. Out of the 15 tokens, 8 tokens are covered (53%). Furthermore, we observe that 2 tokens are covered by the non-contiguous treelet {{`cette@-2`} `législature`}, which "conveniently" skips the token `33e` (thirty-third in English). In this example, it also happens that "droit à la propriété" (property rights) is captured here by 2 TLs, whereas it would have been captured as a single parameter in a standard phrase-based model.

This illustrates two strengths of the TP approach, at least regarding the source material and the treelets. First, a completely unknown token (`33e`) can be skipped by a treelet, while the tokens of the latter are still available to produce a translation for the surrounding known tokens. Second, a source token can be captured by many treelets, suggesting a way to combine them into a more elaborate tree during the decoding phase, possibly with more meaningful results.

Source sentence
Au_cours_de cette 33e législature nous avons examiné le droit à la propriété à trois égards
Treelets in corrected coverage
{{à@-2} {la@-1} propriété}
{{à@-1} trois}
{{cette@-2} législature}
{droit {propriété@3}}

Figure 6: Illustration of the corrected source coverage computed by the algorithm in Figure 5. Words merged together with an underscore form a SYNTEX token.

Target coverage Once a corrected source coverage is computed, we apply another algorithm to select among all the EPs that are associated with the TLs selected, the ones that maximally cover the target sentence T, once again, with the minimum number of phrases. This algorithm is presented in Figure 7.

The candidate EPs are those associated with the TLs obtained from the corrected source coverage computation, although the algorithm would work equally well with the treelets of the raw source coverage, albeit more slowly. These candidate EPs must also `t-match` the target sentence. The criteria used to find the score of a coverage hypothesis are, in order of importance, the target coverage (maximisation) and the number of covering EPs (minimisation). No target token e_i is allowed to be covered by more than one EP (no overlapping EPs). However, we did allow EPs to cover the target tokens contained in the "gaps" left by another EP. For example, given the target sentence `the white rabbit`, if an EP covers the words `the` and `rabbit`, then we allow another EP to cover the

word `white` contained in the gap, if there is such an EP, naturally.

One additional constraint that this algorithm enforces is that no two EPs in the corrected target coverage can share the same source treelet in $\mathcal{M}$, the set of treelets matching the source sentence.

Again, to avoid a combinatorial explosion of hypotheses (stored in *HypoSet* in Figure 7), we only kept the best 10 000 hypotheses at all times.

// init
$HypoSet \leftarrow$ 0-coverage hypothesis

// the search
for all $T \in \mathcal{M}$ **do**
  $AddSet \leftarrow \phi$
  let ep be the set of EPs associated with T which `t-match` E
  for all $p \in ep$ **do**
    for all $h \in HypoSet$ **do**
      if `t-extend`(p, h) **then**
        `add`$(p, h, AddSet)$
  $HypoSet \leftarrow HypoSet \cup AddSet$

// the best hypothesis
find in *HypoSet* the hypothesis h for which `score(h)` is the highest and return it.

Figure 7: Algorithm to compute the corrected target coverage. $\mathcal{M}$ is the set of all treelets matching the source sentence S. `t-extend(p,h)` is a predicate which is true if the elastic phrase p can extend the hypothesis h, `add(p,h,set)` adds to `set` the hypothesis `h` extended with `p`, and `score(h)` returns the score of a hypothesis h. Please read the text for more details.

We complete the example introduced in Figure 6 with the corresponding target coverage, presented in Figure 8. Out of the 11 target tokens, 5 are covered by 3 EPs, a 45% coverage. An interesting match has occurred: while the EP `|property@0||rights@5|` was acquired with a gap of 5 between the words `property` and `rights`, a match was possible with contiguous target words.

The corrected coverage figures are presented in Table 4. Without surprise, these figures are inferior to those reported in section 4.2, although the target coverage is the one which suffers the most from this optimisation. This may be due to the fact that the source coverage optimisation does not take into account the restrictions in the number of candidate EPs that it will eventually impose on the target coverage optimisation. Indeed, when we reach the target coverage optimisation, our options have been limited by the previous step.

Target sentence
This thirty-third Parliament is_dealing with property rights on three different fronts
Elastic phrases in corrected coverage
`|this@0||Parliament@1|`
`|property@0||rights@5|`
`|three@0|`

Figure 8: Illustration of the corrected target coverage. Words merged together with an underscore form a SYNTEX token.

Nonetheless, it is apparent from these results that non-contiguous units can contribute significantly to source and target coverage statistics.

method		source		target	
src	tgt	%-cov	%-c	%-cov	%-c
E	E	59.74	53.20	56.72	45.86
E	R	58.15	51.54	57.55	38.66
E	R+S	58.55	51.99	57.14	37.88
R	E	65.34	48.70	56.56	47.60
R	R	61.21	44.44	56.41	39.35
R	R+S	62.67	45.95	56.43	38.63

Table 4: Source and target corrected coverage statistics. `%-cov` stands for the percentage of tokens that are covered, and `%-c` indicates the percentage of tokens covered by contiguous units.

5 Towards EBMT

Without writing a specific decoder, it is difficult to determine whether TPs can be of help in MT. The RALI, the research group in applied computational linguistics at the Université de Montréal, is currently developing a decoder that will, hopefully, be able to handle tree-phrases. However, we could not wait for the final implementation of this decoder to measure the potential of tree-phrases in a translation context.

We therefore used pharaoh[2], a beam search decoder for phrase-based statistical machine translation models developed by (Koehn, 2004). However, since we do not have access to the code of this program, we cannot modify it to favour

[2] `www.isi.edu/licensed-sw/pharaoh/`

the treelets or phrases contained in our collection, or to propose and implement a new decoding strategy addressing our specific needs. We therefore resorted to a post-processing experiment, using a n-best list produced by pharaoh.

5.1 Experimental Set-Up

Using once again the training and test corpora described in Section 4, we had pharaoh produce a translation for the same 1000 randomly selected source sentences, as well as a n-best list of roughly 1000 different best candidates per translated sentence. We will call each source sentence S_i ($i = 1 \ldots 1000$), its corresponding reference sentence R_i and its candidates $C_i[j]$ ($j > 0$). The first candidate for a sentence S_i is $C_i[1]$ and is the best one, the candidate eventually output by pharaoh as the translation of S_i.

For each of these candidates, we calculated their word error rate (wer) when compared to their respective reference translation. For each set of candidates translated from the same source sentence S_i, we called *oracle* (O_i) the candidate with the lowest wer. When multiple candidates had the same wer, we randomly selected one among the candidates tied for lowest wer.

We then proceeded to compute a variety of coverage-related features for each candidate, like we did in the previous section. We did the same for the reference target sentence R_i and for the oracle O_i. To do so, we used the exact (E) matching policies both for the source and target sentences.

Our goal was to discover, if possible, a coverage feature f for which, on average, $f(R_i) > f(C_i[1])$ or $f(O_i) > f(C_i[1])$. This would mean that our tree-phrase approach could lend itself to a translation task. Indeed, if such a feature f exists, then it means that our memory better "recognizes", on average, R_i or O_i, than the best candidates $C_i[1]$, and the two former have the lowest word error rates: R_i has a wer of 0 by definition, and O_i is the candidate with the smallest wer. It could then be argued that our system is more likely to produce translation with lower wer's than a typical system.

Admittedly, this is a unorthodox way of assessing the usefulness of TPs in machine translation, but this is a pilot study and the resources at hand are still limited.

We computed the features in Table 5 for all the candidates and the reference. We attempted as well to integrate entropy-related features, but did not observe any interesting results. The results are presented in the following section.

f1	src cov. (%)
f2	trg cov. (%)
f3	src cov. w/ contiguous TLs (%)
f4	trg cov. w/ contiguous EPs (%)

Table 5: Various features computed for each candidate and reference in the n-best list for 1000 translations produced by pharaoh.

5.2 Results and Discussion

Table 6 presents the averages and standard deviations for the values of the different features introduced in Table 5, computed for the 1000 translations and their corresponding candidates. The "random" column is the average/standard deviation for each feature computed on a set composed of a randomly selected candidate for each S_i. It acts as a control group, making sure the differences we observe between the sets ref, best and oracle are not purely fortuitous.

feature	stat	ref	best	oracle	rnd
src cov	avg.	67.1	68.1	70.1	67.7
	stdev.	20.9	20.9	19.9	21.5
trg cov	avg.	70.6	71.6	75.4	70.4
	stdev.	22.0	22.1	20.4	22.2
src cov c	avg	61.0	62.4	63.7	61.8
	stdev.	21.8	21.7	21.3	22.3
trg cov c	avg.	67.9	69.4	73.4	68.4
	stdev.	22.0	22.1	20.5	22.2

Table 6: Averages and standard deviations for the values of features computed on a n-best list for 1000 translations produced by pharaoh. src cov is the source coverage, src cov c is the source coverage from contiguous units. ref is the reference R_i for each S_i, best is the first candidate $C_i[1]$, oracle is O_i, the candidate with the lowest wer, and rnd (random) is the set composed of a randomly selected candidate for each S_i. All values are expressed in percentage.

No set among ref, best, oracle clearly stands out, on average, for any of the features we chose. Nonetheless, the oracle set, the one composed of the candidates O_i with the lowest wer's, systematically exhibits the highest scores for each feature. For the target coverage, a difference of

3.9% (absolute) is observed between the oracle and the next best contender.

This assessment strategy is farfetched, we are the first to admit it, but it may argue in favour of the treelet/elastic phrase approach at this early stage of research. If, indeed, $f(O_i) > f(C_i[1])$ like the figures in Table 6 seem to suggest, then our memory could have—at least—the potential to generate translations with lower wer's than a classic phrase-based one, a promising perspective.

6 Discussion

We presented a pilot study aimed at appreciating the potential of Tree-Phrases as a base unit for example-based machine translation. Since we are in the early stages of this study and do not yet benefit from a decoder adapted to these units, we resorted to indirect measures of the potential of a repository populated with TPs.

Coverage statistics clearly show that, whether we allow restrictive match policies or more relaxed ones, our treelets and their corresponding elastic phrases cover most of the source and target material. We observe a slight coverage loss when we apply more rigorous match policies, but that was expected. This generally bodes well for a translation system based on TPs. We can at least rest assured that a given source sentence for which we need a translation will be recognized by the repository. Moreover, since the target coverage of the associated target sentence (reference) is also good, there is a distinct possibility that our system could generate a translation in many ways similar to the reference.

Coverage examples have also highlighted one of the most interesting features of treelets and elastic phrases: their capacity to conveniently skip unknown tokens in a given sequence of words in order to recognize the surrounding tokens. This is of major interest, since unknown or rare tokens usually confuse a standard phrase-based decoder, which does not benefit from the freedom of elastic gaps.

Our post-processing experiments using a n-best list generated by pharaoh, a phrase-based decoder, to attempt to highlight the interest of Tree-Phrases in the context of a translation met a limited success. Our somewhat unconventional approach suggests nonetheless that a TP repository could possibly generate translations with lower word error rates (compared to the reference) than those generated by a more traditional approach.

All this evidence leads us to believe that a TP-based MT system could be a viable alternative to a standard phrase-based one, that such a new repository might better generalise upon a training corpus.

Naturally, this is a preliminary study, and the metrics and features computed here as well as the conclusions drawn from them need to be validated in a more conventional approach, one that would benefit from a decoder capable of handling treelets and elastic phrases. We would then be able to directly measure the contributions of such translation units to a MT system. More efforts could also be invested in considering other translation unit pairs, namely elastic phrase-elastic phrase, or treelet-treelet.

7 Acknowledgements

This work has been financially supported by a grant from PRECARN.

References

Didier Bourigault and Cécile Fabre. 2000. Approche linguistique pour l'analyse syntaxique de corpus. *Cahiers de Grammaire*, (25):131–151. Toulouse le Mirail.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

Yuang Ding and Martha Palmer. 2004. Automatic learning of parallel dependency treelet pairs. In *First International Joint Conference on Natural Language Processing*.

Daniel Gildea. 2003. Loosely tree-based alignment for machine translation. In *ACL*.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In *10th EAMT*, pages 133–142, Budapest, Hongary, May 30-31.

P. Koehn, F.J. Och, and D. Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of HLT*, pages 127–133.

Philipp Koehn. 2004. Pharaoh: a Beam Search Decoder for Phrase-Based SMT. In *Proceedings of AMTA*, pages 115–124.

Evgeny Matusov, Stephan Kanthak, and Hermann Ney. 2005. Efficient statistical machine translation with constraint reordering. In *10th EAMT*, pages 181–188, Budapest, Hongary, May 30-31.

F.J. Och and H. Ney. 2000. Improved Statistical Alignment Models. In *Proceedings of ACL*, pages 440–447, Hongkong, China.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Michel Simard and Philippe Langlais. 2003. Statistical translation alignment with compositionnality constraints. In *HLT-NAACL workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 19–22, Edmonton, Canada, May.

Michel Simard, Nicola Cancedda, Bruno Cavestro, Marc Dymetmann, Éric Gaussier, Cyril Goutte, Arne Mauser, Philippe Langlais, and Kenji Yamada. 2005. Une approche à la traduction automatique statistique par segments discontinus. In *Proceedings of the 12th conference of Traitement Automatique des Langues Naturelles (TALN)*, Dourdan, France, June 6-10.

The 'purest' EBMT system ever built: no variables, no templates, no training, examples, just examples, only examples

Yves Lepage
yves.lepage@atr.jp

Etienne Denoual
etienne.denoual@atr.jp

ATR – Spoken language communication labs
Keihanna gakken tosi, 619-0288 Kyoto, Japan

Abstract

We designed, implemented and assessed an EBMT system that can be dubbed the "purest ever built": it strictly does not make any use of variables, templates or training, does not have any explicit transfer component, and does not require any preprocessing of the aligned examples. It uses a specific operation, namely proportional analogy, that implicitly neutralises divergences between languages and captures lexical and syntactical variations along the paradigmatic and syntagmatic axes without explicitly decomposing sentences into fragments. In an experiment with a test set of 510 input sentences and an unprocessed corpus of almost 160,000 aligned sentences in Japanese and English, we obtained BLEU, NIST and mWER scores of 0.53, 8.53 and 0.39 respectively, well above a baseline simulating a translation memory.

1 Introduction

In contrast to some "least effort" approaches to machine translation, which do not view linguistic data as specific data, we claim that natural language tasks are specific because their data are specific. The goal of this paper is to show that the use of a specific operation, namely proportional analogy in our present proposal, is profitable in terms of trading off preprocessing time of the data and quality of the results. Our proposed technique does not require any preprocessing of the data whatsoever, a definite advantage over techniques that require intensive preprocessing.

1.1 Dealing with the specificity of linguistic data

Trivially, any linguistic datum belongs to one specific natural language that constitutes a "system" in the Saussurian sense of the term. A consistent consequence is to process linguistic data using operations that specifically capture this systematicity. This systematicity appears at best in commutations exhibited by proportional analogies like in the following example.

$$\textit{I'd like to open these windows.} : \textit{Could you open a window?} :: \textit{I'd like to cash these traveler's checks.} : \textit{Could you cash a traveler's check?}$$

Such commutations make paradigmatic and syntagmatic variations explicit and allow for lexical and syntactical variations that ought to be exploited by machine translation system to express different meanings. Indeed, each sentence in any language can be cast into a wide number of such proportional analogies that form a kind of meshwork around it. In (Lepage and Peralta, 2004) we have shown how to automatically extract tables (or matrices) from a linguistic resource so as to visualize these meshworks: each cell in a table contains a sentence, and rectangles formed with four cells in the tables are proportional analogies.

1.2 Dealing with divergences across languages

Machine translation has specific problems to address: one of them, at the core of translation, is to tackle divergences across languages.

A classical and simple example of divergence is the exchange of the arguments of a predicate in Vauquois's famous example between English and French:

$$Elle_1\ lui_2\ plaît. \quad \leftrightarrow \quad He_2\ likes\ her_1.$$

To confirm the importance of the phenomenon, (Habash, 2002) quotes a study on a sample of 19,000 sentences between English and Spanish that shows that one sentence in three presents divergences that can be classified into five different types. An example of type 4 is the classical translation of a Spanish verb into an English preposition.

$$\begin{array}{l} 1:\ Atravesó_{\mathrm{V}} \\ 2:\ el\ río_{\mathrm{N}} \\ 3:\ flotando_{\mathrm{particip.}} \end{array} \quad \leftrightarrow \quad \begin{array}{l} 0:\ It \\ 3:\ floated_{\mathrm{V}} \\ 1:\ across_{\mathrm{prep.}} \\ 2:\ the\ river_{\mathrm{N}} \end{array}$$

Approaches that rely on the word as the unit of processing forget the fact that corresponding pieces of information in different languages are indeed distributed over the entire strings and do not necessarily correspond to complete words. For this reason, the correspondence between words given in the example above is in fact not detailed enough. Actually, the ending *-ó* of the first Spanish word accounts for 3rd person singular past tense. So, not only does *atravesó* correspond to the English preposition *across* for its meaning, but, in addition, it also corresponds to another complete word in English (the pronoun *it*), plus a portion of yet a third English word (the final ending *-ed* of *floated*).

1.3 Dealing with structures (meshworks of proportional analogies)

Following the previous idea that a sentence belongs to a meshwork of proportional analogies, any particular translation correspondence between two sentences belonging to two different languages should be viewed as a part of the global correspondence between the two languages at hand. The technique that we thus propose for automatic translation exploits the translation links that incidentally exist between sentences as part of the meshwork of proportional analogies found around them.

$$\begin{array}{cccccccc} \textit{I'd like to open these windows.} & : & \textit{Could you open a window?} & :: & \textit{I'd like to cash these traveler's checks.} & : & \textit{Could you cash a traveler's check?} \\ \updownarrow & & \updownarrow & & \updownarrow & & \updownarrow \\ \textit{Est-ce que ces fenêtres, là, je peux les ouvrir?} & : & \textit{Est-ce que vous pouvez m'ouvrir une fenêtre?} & :: & \textit{Ces chèques de voyage, là, je peux les échanger?} & : & \textit{Vous pouvez m'échanger un chèque de voyage?} \end{array}$$

Figure 1: Two proportional analogies in two different languages that correspond.

Figure 1 gives the example of the two following sentences taken as part of particular proportional analogies that correspond.

$$\textit{Could you cash a traveler's check?} \quad \leftrightarrow \quad \textit{Vous pouvez m'échanger un chèque de voyage?}$$

The correspondence can only be established because each sentence in the lower part of the figure is a possible translation of the sentence above it in the upper part of the figure.

A consequence of this view is that the difficulty which is usually seen in translating between some particular pairs of languages simply vanishes. The claim that it is costly to translate between some specific language pairs like, *e.g.*, Japanese and English, relies indeed on the idea that translating would basically consist of rearranging, transforming, or decoding. However, to make a comparison with clothes, to localise what corresponds to the left shoulder of a shirt on, say, a jacket, one does not take material from the left shoulder of the shirt, unweave it, weave it back again in a different way, and then patch it somewhere on the jacket. Although this sounds strange, this is precisely what second generation MT systems do when they use lexical and structural transfer rules; and SMT systems (Brown et al., 1993) when they use lexicon models with distortion models.

Rather, it is reasonable to point at the left shoulder of the jacket by looking at the gen-

eral constitution of the jacket, and by following the different wooves and threads *on* the jacket to localise some point more precisely if needed, as the jacket is made of a different material from the shirt. Transposing to machine translation, the translation of a source sentence should be looked for by relying on the paradigmatic and syntagmatic meshworks, *i.e.*, by using the proportional analogies in the target language which correspond to the proportional analogies of the source language that involve the source sentence, until a corresponding sentence is obtained.

2 Example-based machine translation (EBMT) by proportional analogy

2.1 The algorithm

Suppose we have a corpus of aligned sentences in two languages (a bicorpus) at our disposal. The following gives the basic outline of our method to perform the translation of an input sentence:

Form all analogical equations with the input **sentence** D and with all relevant pairs of **sentences** (A_i, B_i) from the source part of the bicorpus[1];

$$A_i : B_i :: x : D$$

For those sentences that are solutions of the previous analogical equations which do not belong to the bicorpus, translate them using the present method recursively. Add them with their newly generated translations to the bicorpus;

For those sentences $x = C_{i,j}$ that are solutions of the previous analogical equations[2] which belong to the bicorpus, do the following;

Form all analogical equations with all possible target language sentences corresponding to the source language sentences[3];

$$\widehat{A_i}^k : \widehat{B_i}^k :: \widehat{C_{i,j}}^k : y$$

Output the solutions $y = \widehat{D_{i,j}}^k$ of the analogical equations as a translation of D, sorted by frequencies[4].

2.2 An example

Suppose that we wanted to translate the following Japanese input sentence:

濃いコーヒーが飲みたい。[5]

Among all possible pairs of sentences from the bicorpus, we may find the following two Japanese sentences:

紅茶をください。	↔	*May I have some tea, please?*
コーヒーをください。	↔	*May I have a cup of coffee?*

that will allow us to form the following analogical equation:

紅茶をください。	:	コーヒーをください。	::	x	:	濃いコーヒーが飲みたい。

This equation yields $x =$ 濃い紅茶が飲みたい。[6] as a solution. If this sentence already belongs to the bicorpus, *i.e.*, if the following translation pair is found in the data

濃い紅茶が飲みたい。	↔	*I'd like some strong tea, please.*

the following analogical equation is formed with the corresponding English translations:

May I have some tea, please?	:	*May I have a cup of coffee?*	::	*I'd like some strong tea, please.*	:	x

By construction, the solution: $x =$ *I'd like a cup of strong coffee.* is a candidate translation of the input sentence: 濃いコーヒーが飲みたい。

[1] Relevant pairs of sentences are selected on-the-fly according to a similarity criterion. A_i,B_i and D are **sentences**; they are **not fragments** of sentences. Sentences are **not cut into pieces**. Also, **pairs** of sentences are retrieved to form an analogical equation with D; consequently, there is no such thing as **analogous examples**, as such an expression does not make any sense in this framework; indeed, A_i's and B_i's may be quite "far away" from D.

[2] One analogical equation may yield several solutions.

[3] Several target sentences may correspond to the same source sentence.

[4] Different analogical equations may yield identical solutions.

[5] Gloss: strong coffee NOMINATIVE-PARTICLE drink-VOLITIVE. Literally: *I want to drink strong coffee.*

[6] Lit.: *I want to drink strong tea.*

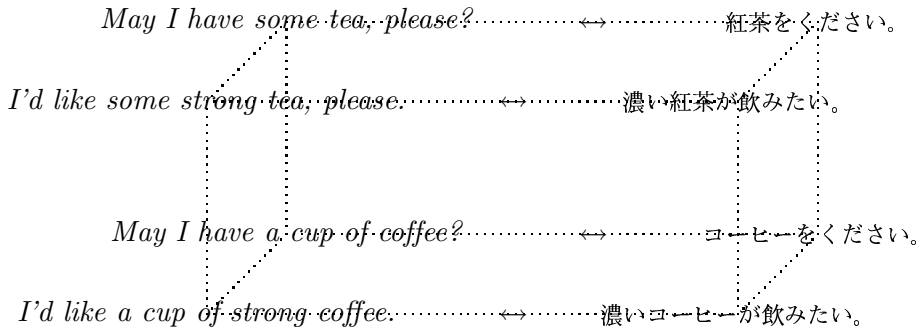


Figure 2: The parallelopiped: in each language, four sentences form a proportional analogy. There exist four translation relations between the sentences.

2.3 A geometric view of the principle

The processing of the previous example, which is reminiscent of distributionalism (Harris, 1954), can be viewed in the shape of a parallelopiped as shown in Figure 2. The left plane of this parallelopiped is the plane of the English analogy. The right plane is the Japanese one. Because each of these planes resides in one and only one language, the terms of the proportional analogy involve monolingual data only so that they can be processed by algorithms like the one proposed in (Lepage, 1998).

3 Features of the method

3.1 No transfer

To stress that the choice of a correct translation is really left to an implicit use of the structure of the target language, and does not imply any explicit transfer processing, consider the Spanish example of Section 1.2 again. The correspondences between the source and the target language in a proportional analogy will be entirely responsible not only for the selection of the correct lemmas wit their lexical POS, but also for the correct word order[7].

This could be compared to some extent to the translation of the adnominal particle N_1 *no* N_2 from Japanese into English in (Sumita and Iida, 1991) where the choice of the correct preposition (or word order) is left to the list of examples.

They swam in the sea.	:	*They swam across the river.*	::	*It floated in the sea.*	:	*It floated across the river.*
↕		↕		↕		↕
Nadaron en el mar.	:	*Atravesaron el rio nadando*	::	*Flotó en el mar.*	:	x

However, it should be stressed that in proportional analogies like the two above, nowhere is it said which word corresponds to which word, or which syntactic structure corresponds to which syntactic structure. The sole action of proportional analogy with (necessarily) **the character as the only unit of processing**, is sufficient to produce the exact translation of *It floated across the river*, that is, the correct Spanish sentence: $x =$ *Atravesó el rio flotando*, provided that the three sentence pairs on the left are valid translation pairs.

3.1.1 No extraction of symbolic knowledge

In a second generation MT system, one makes the knowledge relevant to such divergences explicit in the form of lexical and structural transfer rules. In the EBMT approach too, one makes this knowledge explicit by automatically acquiring templates that capture these divergences. In both cases, the knowledge about these divergences has to be made explicit. In

[7] As for reordering of words, with its translation knowledge reduced to the sole two translation pairs: $abc \leftrightarrow abc$, $abcabc \leftrightarrow aabbcc$, the system needs only to solve $2 \times (n-2)$ proportional analogies recursively to translate members of the regular language $\{\,(abc)^n \mid n \in \mathbb{N}^*\,\}$ into the corresponding members of the context-sensitive language $\{\,a^n b^n c^n \mid n \in \mathbb{N}^*\,\}$, and reciprocally: $(abc)^n \leftrightarrow a^n b^n c^n$.

our view the choice of the correct expression ought to be left implicit as it pertains to the structure of the target language. Indeed, paradigmatic and syntagmatic commutations neutralise these divergences as they are the implicit constitutive material of proportional analogies.

Our system definitely positions itself in the EBMT stream, however it departs from it in one important aspect: it does not make any use of explicit symbolic knowledge such as templates with variables. Direct use of bicorpus data in their raw form is made, without any preprocessing.

The reason for doing so is that we consider that templates may well be insufficient in representing all of the implicit knowledge contained in examples. Indeed, variables in templates allow for paradigmatic variations at some predefined positions only[8]. For instance, extracting the template *X salts Y* from the example sentence *the butcher salts the slice* where *X* may be replaced by *the butcher*, etc. and *Y* by *the slice*, etc. [9] does not make the most of the potential of the example. Firstly, it prevents *the butcher* from being changed into a plural: *the butchers*. Moreover, it misses the fact that *salts* may also commute with its past and future forms, etc.: *salted*, *will salt*, etc., or with *cuts*, *smokes*, etc.; and so forth. To summarise, there is a risk of loss of information when replacing examples with templates.

The situation is in no way better with translation patterns. They make explicit which variables in the source have to be replaced by which variables in the target[10]. But it is well known that a single variable at one single position in a source template often needs to be linked to several positions distributed over a target template, and may even imply different levels of description (morphological, syntactical, etc.) For instance, negation is expressed at one single position in Japanese, whereas it may also imply a change in the form of the main verb in English: *he eats → he does not eat.*

Our view is that *every* position in a language datum is subject to paradigmatic variation[11]. The consequence being that a lot more exploitable information is to be found in unprocessed examples than in templates. And it may well be the case that the number of templates necessary to encode the same amount of information contained in a set of examples is much larger in size than the actual size of these unprocessed examples themselves. Thus, extracting templates from examples may well entail a loss in generative power as well as in space. It must however be stressed that the generative power of the unprocessed examples does not actually reside in their bare listing but in their capacity for getting involved in proportional analogies.

3.2 No training, no preprocessing

As a consequence of the abovementioned features, there is no such thing as a training phase or a preprocessing phase in our system: the bicorpus is just loaded into memory at program startup. No language model is computed; no other alignment than the one given by the bicorpus is extracted; no segmentation or tagging whatsoever is performed. Needless to say, the possibility of adding new information to the bicorpus is left open. For instance, adding dictionaries or paraphrases to the corpus is a possibility that may improve results but leaves the structure of the system absolutely unchanged (see Sections 4.4.2 and 4.4.3).

4 Evaluation and comparison with other systems

4.1 Resources used in the evaluation

To assess the performance of the proposed method, we used the C-STAR Basic Traveler's Expressions Corpus[12]. It is a multilingual resource of expressions from the travel and tourism domain that contains almost 160,000 aligned translations in English and Japanese. In this resource, the sentences are quite short as the figures in the following table show. As the same sentence may appear several times with different translations, the number of different

[8] In (Sato, 1991), so as to acquire a grammar, sentences are fed into a system, which differ by one word only.

[9] Examples from (Carl, 1998).

[10] (Sasayama et al., 2003) for the use of arrays describing these kinds of associations.

[11] Putting it to the extreme, even phonetic variations have to be considered: *wolf* : *wolves* :: *leaf* : *leaves*. So that one definitely has to go below words. For this reason, our system processes **strings of characters,** not strings of words.

[12] `http://www.c-star.org/`.

コーヒーのおかわりをいただけますか。

2318 *I'd like another cup of coffee.*
2296 *May I have another cup of coffee?*
1993 *Another coffee, please.*
1982 *May I trouble you for another cup of coffee?*
1982 *Can I get some more coffee?*
530 *Another cup of coffee, please.*
516 *Another cup of coffee.*
466 *Can I have another cup of coffee?*
337 *May I get some more coffee?*
205 *May I trouble you for another cup of coffee, please?*

小銭をまぜてください。

924 *Can you include some small change?*
922 *Can you include some small change, please?*
899 *Would you include some small change?*
896 *Include some small change, please.*
895 *I'd like to have smaller bills mixed in.*
895 *Please change this into small money.*
895 *Will you include some small change?*
885 *Could you include some small change, please?*
880 *May I have some small change, too?*

Figure 3: Two examples of translations. The figures on the left are the frequencies with which each translation candidate has been output.

sentences in each language is indicated in the following table.

	Number of $\neq$ sentences	Size in characters avg. $\pm$ std. dev.
English	97,395	35.17 $\pm$ 18.83
Japanese	103,051	16.22 $\pm$ 7.84

The method relies on the assumption that analogies of form are almost always analogies of meaning. Thus, prior to its application, we (Lepage, 2004) estimated the relative number of analogies of form which are not analogies of meaning in the resource used: less than 4% (p-value = 0.1% on a sample of 666 analogies). This proportion is too small to seriously endanger the quality of the results obtained during translation.

4.2 Gold Stantard and baseline

In order to evaluate the performance of our system, we use a test set of 510 input sentences. These sentences are from the same domain as the bicorpus. For each of them, we also have a set of 16 translation references in the target language at our disposal.

This allows us to perform an evaluation using several standard objective measures, like BLEU, NIST or mWER.

Firstly, we determined a Gold Standard in the following way. For each sentence of the test set, we evaluated the first reference translation as if it were given by an MT system. In this way, we obtained the "best" values for each of the measures considered (see Table 1).

Then, we determined a baseline by simulating a translation memory. For each sentence of the test set, we took the closest sentence in the corpus according to edit distance and output its translation that we evaluated with each of the objective measures. This gives baseline scores for each of the measures considered.

4.3 Results with the resource only

Our system was then evaluated on the translations it output for the sentences of the test set, with the sole source of examples being the resource data (see Table 1, line: resource only). Some examples of translations are shown in Figure 3, with the frequencies for each candidate[13]. As we assumed that the most frequent candidate should be the most reliable one, the evaluation was performed on the first candidates only.

4.4 Choice and influence of linguistic resources

4.4.1 Influence of the amount of examples

In an EBMT system, one would trivially expect the amount and nature of examples to strongly influence translation quality. The figures in Table 1 on the lines marked 1/2 resource and 1/4 resource, which were obtained by sampling the original resource confirm this fact. In this case, the more data, the better the results.

[13]Different analogical equations may yield the same solutions (see Section 2.1).

Table 1: Scores for the Gold Standard, the baseline, and the system with various data. We also compare with two other EBMT systems that require heavy preprocessing of the bicorpus to extract patterns either automatically (system A) or by hand (system B).

System:	Number of translation pairs	BLEU	NIST	mWER	PER	GTM
Gold Standard	n.r.	1.00	14.95	0.00	0.00	0.91
System A	unknown	0.66	10.36			
+ Src + tgt paraphrases	438,817	0.50	**8.98**	0.46	0.42	0.67
+ Tgt paraphrases	158,409	0.49	8.91	0.47	0.43	0.67
+ Src paraphrases	158,409	0.53	8.53	**0.38**	**0.35**	**0.68**
+ Dictionary	206,382	**0.54**	8.54	0.39	0.36	**0.68**
Resource only	158,409	0.53	8.53	0.39	0.36	**0.68**
1/2 resource	81,058	0.45	7.78	0.50	0.45	0.63
1/4 resource	40,580	0.42	7.18	0.53	0.49	0.60
System B	unknown	0.41	9.00			
Baseline: transl. memory	n.r.	0.38	7.54	0.58	0.53	0.61

4.4.2 Dictionaries as lists of particular examples

Whole sentences contained in the resource (as opposed to isolated words or idioms) may not allow the translation of particular expressions if commutations cannot be found between them. This case is particularly plausible when translating sentences that contain multi-word expressions or numbers, for instance.

A possible remedy is to add dictionary entries to the original resource to be used as additional examples. As a matter of fact, this system does not make any difference between a bicorpus or a dictionary as long as both are aligned strings of data, be they sentences or words. The following examples illustrate that the data format for a bicorpus or a dictionary does not differ in any way.

フィルムを買いたいのですが。 ↔ *I'd like a film, please.*

三十六枚撮りを二本ください。 ↔ *Two rolls of thirty-six exposure film, please.*

このカメラの電池がほしいのです。 ↔ *I'd like a battery for this camera, please.*

フィルム ↔ *film*
映画 ↔ *film*
電池 ↔ *battery*
砲台 ↔ *battery*

The scores obtained by adding a dictionary to our resource are not different from those with the resource only, except for a slight improvement in BLEU.

4.4.3 Paraphrases generated from the resource as additional examples

Previous research has shown that the introduction of paraphrases may improve the quality of machine translation output. Paraphrases may be added in the source language (Yamamoto, 2004) or in the target language (Habash, 2002).

In order to increase the chances of a sentence entering into proportional analogies, we grouped sentences in the source language data by paraphrases. To do so, we grouped sentences that share at least one common translation because, in this case, they share the same meaning, (*i.e.*, they are paraphrases). In our bicor-

pus, an average of 3.03 paraphrases per source sentence was obtained[14]. This new information allows the translation process to test a larger number of proportional analogies. When a pair of sentences (A, B) is proposed for an input sentence D, not only the equation $A : B :: x : D$ will be tried, but also all possible equations of the form $A' : B' :: x : D$, where A' and B' are paraphrases of A and B.

The evaluation of translation quality when adding paraphrases in the source language are shown in Table 1 on the line marked: + Src paraph. They show a slight improvement in word error rate.

The same thing can be done on the target language side with a similar effect of increasing the number of proportional analogies tried, this time in the target language. As for scores, they decrease in BLEU but show a real improvement in NIST.

The scores obtained when adding paraphrases in the source and in the target language are shown on the line marked: + Src + tgt paraph. They are not better than those with the resource only, except for NIST, as paraphrases are expected to have introduced lexical and syntactical variation in expressing identical meanings. An explanation for the loss in quality according to all other measures may be that the increase in computation to perform may have overloaded the system (all experiments are done with the same time-out).

5 Discussion and future work

5.1 Translation time

It could have been feared that the complexity of the algorithm, which is basically square in the amount of data, would have enormously impaired the method. However, using a simple heuristics to select only relevant pairs entering in analogical equations allowed us to keep translation times reasonable. Within a time-out of 1 CPU second, the average translation time per sentence was 0.73 second on a 2.8 GHz processor machine with 4 Gb memory.

[14]However, the distribution is not uniform: 71,192 sentences (out of 103,274) don't get any new paraphrase, while 54 sentences get more than 100 paraphrases, with a maximum of 410 paraphrases for one sentence.

5.2 Proportion of successful analogies

As the fundamental operation in the system is analogy, we measured the proportion of analogical equations sucessfully solved over the total number of analogies formed in the source language. Between half a million and one million analogical equations $(687,641)$ are formed on average to translate one sentence from the test set. The proportion of analogical equations sucessfully solved is 28%. In other words, the heuristics used to select sentence pairs from the corpus in order to form analogical equations is successful only a quarter of the time. Future work should include finding a heuristics that would increase this proportion so as to reduce the number of unnecessary trials.

5.3 Recursion level needed

As was explained in Section 2.1, recursive applications are expected to be made in order to reach translations of a single input sentence. Over all input sentences of the test set, one recursive call is needed on average, and a maximum of two is necessary on some sentences. This shows that the sentences in the test set were in fact quite "close" to the resource used: the number of recursive calls is a measure of how "far" a sentence is to a corpus.

5.4 Relevance / suitability of the examples

The translation of an input sentence depends crucially on the two following points. Firstly, whether the input sentence belongs to the domain (and the style) of the corpus of examples. Secondly, whether the corpus covers the linguistic phenomena present in the input sentence. A positive point of our system is that the absence of any training phase reduces the development cycle to the problem of choosing / coining suitable examples that cover a given domain and the linguistic phenomena of the language. To address these two issues, we see two possible directions of research.

Firstly, as was mentioned in Sections 4.4.3 and 4.4.2, we are studying various ways to add paraphrases or dictionaries and how to improve their efficiency in terms of lexical and syntactical variation, so as to further densify the bicorpus in terms of coverage

Secondly, we are investigating the possibility of designing a core grammar by examples, *i.e.*,

a collection of examples that would cover the basic linguistic phenomena in a given language. In the same way as school grammars illustrate rules by examples, our methodology will be to choose a formal grammar known to have a large coverage, and to illustrate its rules with examples. Distributionalist grammars (Harris, 1982) seem to be better candidates for this purpose as they rely on the notion of the expansion and embedding of strings, a notion that is precisely captured by proportional analogy. In particular, *string grammars* (Sager, 1981) or (Salkoff, 1973) are well known for having a large coverage.

6 Conclusion

In this paper, we have shown that the use of a specific operation, namely proportional analogy, leads to reasonable results in machine translation without any preprocessing of the data whatsoever, an advantage over techniques requiring intensive preprocessing. In an experiment with a test set of 510 input sentences and an unprocessed corpus of almost 160,000 aligned sentences in Japanese and English, we obtained BLEU, NIST and mWER scores of 0.53, 8.53 and 0.39, respectively, well above a baseline simulating a translation memory. Slight improvements could be obtained by adding paraphrases.

The use of an operation that suits by essence the specific nature of linguistic data, *i.e.*, their capacity of commutation on the paradigmatic and syntagmatic axes, allowed us to dispense with any preprocessing of the data whatsoever. In addition, this operation has the advantage of tackling the issue of divergences between languages in an elegant way: it neutralises them implicitly. As a consequence, the system implemented does not include any transfer component (either lexical or structural).

To summarise, we designed, implemented and assessed an EBMT system that, we think, can be dubbed the "purest ever built" as it strictly does not make any use of variables, templates or training, does not have any explicit transfer component, and does not require any preprocessing of the aligned examples, a knowledge that is, of course, indispensable.

As an extra feature, the system is learning as it keeps translating. Recursive calls add translation knowledge to the bicorpus, so that, in standard use, the history of translations will influence the results of coming translations. In the reported experiment we had to disallow this feature to be placed in conditions comparable with, say, SMT systems. However, such a use denatures our system.

7 Acknowledgements

The research reported here was supported in part by a contract with the National Institute of Information and Communications Technology entitled "A study of speech dialogue translation technology based on a large corpus".

References

Peter E. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics, Special Issue on Using Large Corpora: II*, 19(2):263–311, March.

Michael Carl. 1998. A constructivist approach to machine translation. In *Proceedings of NeMLaP'98*, Sydney.

Nizar Habash. 2002. Generation-heavy hybrid machine translation. In *Proceedings of the International Natural Language Generation Conference (INLG'02)*, New York.

Zellig Harris. 1954. Distributional structure. *Word*, 10:146–162.

Zellig Harris. 1982. *A grammar of English on mathematical principles.* John Wiley & Sons, New York.

Yves Lepage and Guilhem Peralta. 2004. Using paradigm tables to generate new utterances similar to those existing in linguistic resources. In *Proceedings of LREC-2004*, volume 1, pages 243–246, Lisbonne, May.

Yves Lepage. 1998. Solving analogies on words: an algorithm. In *Proceedings of COLING-ACL'98*, volume I, pages 728–735, Montréal, August.

Yves Lepage. 2004. Lower and higher estimates of the number of "true analogies" between sentences contained in a large multilingual corpus. In *Proceedings of COLING-2004*, volume 1, pages 736–742, Genève, August.

Naomi Sager. 1981. *Natural language information processing: a computer grammar of English and its applications*. Adelson-Wesley, Reading, Massachusetts.

Morris Salkoff. 1973. *Une grammaire en chaîne du français*. Dunod, Paris.

Manabu Sasayama, Fuji Ren, and Shigo Kuroiwa. 2003. Super-function based Japanese-English machine translation system. In *Proceedings of Natural Language Processing and Knowledge Engineering*, volume 1, pages 555–560, Beijing, October.

Satoshi Sato. 1991. *Example-based Machine Translation*. Ph.d. thesis, Kyoto University, September.

Eiichiro Sumita and Hitoshi Iida. 1991. Experiments and prospects of example-based machine translation. In *Proceedings of the 29th Conference on Association for Computational Linguistics*, pages 185–192, Morristown, NJ, USA. Association for Computational Linguistics.

Kazuhide Yamamoto. 2004. Interaction between paraphraser and transfer for spoken language translation. *Journal of Natural Language Processing*, 11(5):63–86, October.

Monolingual Corpus-based MT using Chunks

Stella Markantonatou[1], **Sokratis Sofianopoulos**[2], **Vassiliki Spilioti**[3], **Yiorgos Tambouratzis**[4], **Marina Vassiliou**[5], **Olga Yannoutsou**[6], **Nikos Ioannou**[7]
Machine Translation Department, Institute for Language & Speech Processing
6 Artemidos & Epidavrou Str., Paradissos Amaroussiou, Athens, GREECE 151 25
[1]marks, [2]s_sofian, [3]vspiliot, [4]giorg_t, [5]mvas, [6]olga@{ilsp.gr}, [7]nick.ioannou@gmail.com

Abstract

In the present article, a hybrid approach is proposed for implementing a machine translation system using a large monolingual corpus coupled with a bilingual lexicon and basic NLP tools. In the first phase of the METIS system, a source language (SL) sentence, after being tagged, lemmatised and translated by a flat lemma-to-lemma lexicon, was matched against a tagged and lemmatised target language (TL) corpus using a pattern matching algorithm. In the second phase, translations are generated by combining sub-sentential structures. In this paper, the main features of the second phase are discussed while the system architecture and the corresponding translation approach are presented. The proposed methodology is illustrated with examples of the translation process.

Keywords: MT, monolingual corpus, chunks, METIS-II

1 Introduction

In this article we present on-going work on a hybrid approach for implementing a machine translation system which uses a large monolingual corpus coupled with a bilingual lexicon, a tagger, a lemmatiser and a chunker. Translating without bilingual parallel corpora has been the focus of the METIS[1] projects. In the first phase of the METIS system (Dologlou et al., 2003 and Ioannou, 2003), a source language (SL) clause was tagged, lemmatised and translated by a flat lemma-to-lemma lexicon. The string resulting from these procedures was matched against a tagged and lemmatised target language (TL) corpus using a pattern matching algorithm. Results of adequate quality were received, only when a similar clause did exist in the TL corpus. However, even for very large corpora this proved to be unlikely. The next step was to attempt to generate a translation by combining translations of the chunks of the SL clause.

In the present paper, we first present the main features of our approach and then the architecture of the system. Finally, we use concrete examples to illustrate the translation process.

2 The main features of METIS

Resources have been one of the major problems in MT regardless of the approach, whether RBMT, EBMT, SMT or other: lexica, grammars/parsers, parallel corpora are some of the required resources. EBMT (Nagao, 1984) and statistics-based approaches (Brown et al., 1990) originally aimed at avoiding the problem of great expenditure resources in human expertise. The argument, however, was proven to be weak in two respects. First, from the days of early SMT (Brown et al., 1990), it was admitted that some amount of linguistic knowledge was necessary. This wisdom does not seem to have been altered much by today, at least as regards the need for bilingual lexica (Brown et al., 1990 and Popovic et al., 2005). Second, all corpus-based approaches rely on large bitexts (McTait, 2003) in order to produce reasonable results, and such bitexts are rare, may be of questionable linguistic quality (Al-Onaizan, 2000), and are usually confined to a sublanguage, while their register identity is a parameter rather difficult to control. The approach selected for METIS is innovative, exactly because it relies on a monolingual corpus, still a relatively low-cost and easy-to-construct resource, whose quality and register type are more controllable issues than in the case of bitexts.

Working at sub-sentential level has been proposed as a promising way of achieving better exploitation of the linguistic knowledge in a corpus (Cranias, 1997). A variety of ways of fragmenting

[1] METIS was funded by EU under the FET Open Scheme (METIS-I, IST-2001-32775), while METIS-II, the continuation of METIS, is being funded under the FET-STREP scheme of FP6 (METIS-II, IST-FP6-003768). The assessment project METIS ended in February 2003, while the second phase started in October 2004 and has a 36 month duration.

sentences for MT purposes have been proposed ranging from the exploitation of highly structured representations of linguistic knowledge (Way, 2003) to the establishment of string correspondences with little/trivial linguistic knowledge representation adhered to them (Brown et al., 1990 and McTait, 2003). However, any method relying on the combination of sub-sentential strings faces the problem of boundary friction, while 'more linguistic' methods are reported to be less affected by it than 'less linguistic' ones (Way, 2003).

The hybrid approach described here presupposes work at sub-sentential level and freely draws on the EBMT, RBMT and SMT paradigms. It aims to be modular, language-independent and with a small number of language-pair specific tools and resources being added to the core engine. In order to illustrate its principles, the Greek (SL) to English (TL) language pair was selected by ILSP within the METIS projects.

3 A Methodology for Implementing the Machine Translation Task

In order to translate with a monolingual corpus, we have defined a sequence of steps shown in Figure 1 where different colours signal the two main parts of the system architecture. The first part (white-coloured entities) consists of processes that are performed initially so as to obtain a translation. The second part (grey-coloured entities) consists of processes performed only when the first part results are of a non-satisfactory quality. The source sentence and the target corpus are annotated before the sentence matching algorithm applies. The overall translation process comprises the following steps:

1. Annotation of the TL corpus (off-line)
2. Annotation of the SL sentence (on-the-fly)
3. Exploitation of the TL corpus to create the best translation (on-the-fly)
4. Synthesising the translation output (on-the-fly)

3.1 Annotation of the TL Corpus

In order to be searched efficiently for candidate translations of SL sentences, the TL corpus is annotated. For the purposes of METIS-II, the British National Corpus (BNC)[2] has been selected as the TL corpus, because it has been established as the largest, general-purpose balanced corpus for this language. Annotation is performed off-line and only once: BNC is tagged with the CLAWS5[3] tagset (it actually comes with a large part of it golden-tagged as standard) and is lemmatised with a purpose-built lemmatiser[4]. It is then exhaustively annotated with a purpose-built tool for clauses containing a finite verb (non-finite clauses such as gerunds or infinitival clauses are not considered: [*Walking the dog I met Iris*] [*who wanted to pick flowers*]). Clauses are then annotated for VGs, NPs, PPs (at the moment) with the ShaRPa 2.0 chunker (Vandeghinste, 2005).

To ensure a fast and efficient search for a best match, clauses are indexed according to their finite verb and chunks are classified into sets according to their label (sets of NPs, PPs etc.) and their head.

3.2 Annotation of the SL Sentence

The SL sentence is annotated with the linguistic information necessary to guide the matching algorithm before being fed to the matching algorithm. First, it is tagged and lemmatised with a PAROLE compatible ILSP tool (Labropoulou et al., 1996). It is then annotated for finite clauses and their constituent chunks with the ILSP chunker (Boutsis et al., 2000). The output of the chunker consists of a sequence of labelled chunks and the words contained in each chunk. A purpose-made script marks the respective heads. Next, two flat bilingual lexica are sequentially applied on the tagged-lemmatised string; first the Expression Lexicon, which contains the translations for multi-word units and second, the Word Lexicon with single-word units. The output of the lookup is a list of sets of TL lemmata (each list containing all possible translations for a given term in the source language) with PoS information for the Word lexicon, while word forms are maintained in the Expression one, (ILSP: Internal Document, Specifications for METIS lexicon, 2004).

Up to this point only basic resources have been used for both the SL and the TL. Apart from the bilingual lexica, they are all monolingual general purpose NLP tools not dedicated solely to MT. In our case, bilingual lexica have been constructed by drawing on existing resources, which after being checked for consistency and accuracy, were homogenised to fit to the system's requirements.

3.3 Employing Mapping Rules

The system, as presented in Figure 1, allows for the possibility of employing a limited set of mapping rules aimed to map the string obtained by the

[2] http://www.natcorp.ox.ac.uk/index.html

[3] http://www.comp.lancs.ac.uk/ucrel/claws5tags.html

[4] http://iai.iai.uni-sb.de/~carl/metis/lemmatiser

lemma-to lemma-translation onto a string which is closer to what we expect to find in the target language. Analogies respected, this process has been shown to greatly enhance the translation quality in rule-based systems (Dyvik, 1995). Mapping rules will not be used to deal with local problems but rather to accommodate significant linguistic differences across a given language pair. Subsets of these rules may be (re)used for any pair of languages presenting the same typological differences. As an indicative example we use NP order, which the pattern matching algorithm treats in a way that makes sure that Modern Greek NP nominatives correspond to preverbal English NPs (typical features of subject NP in Modern Greek and English respectively). This case obviously reflects the typological difference between languages which use case and languages which employ strict word order to mark functional relations.

3.4 The Sentence Matching and the Synthesising Algorithm

All steps up to this point belong to the annotation stage. The material collected during the SL sentence annotation phase is input to the Sentence Matching Algorithm, which compares this information with the corresponding information retrieved from BNC.

As a first step, the algorithm, which examines both the sentence structure (in terms of number and types of chunks) and sentence contents (in terms of lemmata and tags within each chunk), searches the BNC for a very similar sentence. If one exists, it is retrieved and sent to the synthesising algorithm. If, however, no candidate sentence has a very high similarity to the input, the phrase matching algorithm searches within the BNC to retrieve chunks originating in different sentences in order to replace the mismatching chunks of the best-matching sentence.

In the unlikely case that no overall structure is found, the system attempts to modify the structure and provide translations for as many phrasal parts of the SL sentence as possible by searching again within the BNC for appropriate chunks, extracted from different sentences.

The synthesising algorithm combines the essential parts of the best-matching sentence (the '*framework*', see Section 4) with the material from other BNC sentences to generate a sentence of satisfactory quality.

In the most general case the pattern matching based search algorithm yields a set of fragments (chunks and sets of chunks), which are fed to the synthesising algorithm. The latter roughly comprises two tasks: (a) the modification and rearrangement of the retrieved chunks, so that they can be meaningfully combined into a sentence and (b) the handling of morphological phenomena. Task (a) draws mainly on a number of synthesis rules, while for task (b) a morphological generator is employed [see footnote 4].

Below, we present in more detail the rationale and the practical steps taken at the matching phase.

4 The Matching Procedure: rationale

The mechanism employed for making the SL and the TL languages "meet" relies on the already mentioned notion of a clause '*framework*' (Section 3.4), which represents the main clause structure with the verb head-lexicalised. We thus seek to retrieve from the monolingual corpus clauses that contain the TL verb[5], which is the exact translation of the SL verb (the lexicon may provide more than one such solutions), in a context consisting of the same amount of referential expressions.

The idea behind this requirement is that sentences express events with a certain number of participants. The event is basically denoted by the verb while the participants mainly by referential expressions, embedded within some grammatical information functor, call it Case (from a purely morphological point of view) or Preposition or both. For instance, the Modern Greek sentence

O	Petros	mpike	sto	-	dhomatio
The-Nom	Peter-Nom	enter-3rd-SG-Past	in-the-Pr		room-Acc

'Peter entered the room'

denotes an event with two participants, one embedded under the Nominative Case and the other one under a preposition and the Accusative Case. Its English correspondent differs from it as regards the grammatical functor of the second referential expression.

For our approach, it is important that, although we avail ourselves to no information about the subcategorisation preferences of the verbs involved, we end up with the proper verb and the proper referential expressions embedded under the proper functor.

[5] One could look for families of verbs occurring in the same syntactic environment. We would first like to exhaust the present approach and then move to a more abstract description of phrase structure.

To this end, our pattern-matching algorithm generalises over these two types of grammatical functor, Case and Prepositions. Thus, while the matching algorithm takes care of the essential cross-language information (the verb predicate and the amount of referential expressions), the grammatical particularities of either language are supplied by their well-formed strings (this viewed as mapping from the SL->TL implies that the corpus plays the role of the supplier of grammatical information about the TL). In the example above, our algorithm will select a TL sentence with the verb 'enter' in the appropriate grammatical context, which is not a one-to-one copy of the SL grammatical context.

Of course, the assumption underlying this approach is that verbal expressions are translated to verbal expressions and referential expressions to referential ones. This might be a strong hypothesis; however, it is considerably less strong than requiring grammatical equivalence across language pairs.

On a similar par, that of generalising linguistic patterns at the matching phase, we have chosen to work with lemma-to-lemma bilingual lexica rather than looking for tokens in the TL corpus. Morphological information is, in general, relatively simple to incorporate at the end of the overall translation procedure.

Having said the above, it must be noted that all SL information is kept as default information, overwritten only by corpus information. For instance, when no framework is found containing all the appropriate chunks, an appropriate one is introduced by directly mapping information from the SL onto the TL.

We now proceed to present the matching procedure step by step.

4.1 Matching Step by Step

Step 1: As explained before, clauses from the BNC are retrieved, based on the main verb and the number of chunks. For each different translation of the SL verb a different set of clauses is created.

The multiple translations provided by the lexicon are reduced by calculating the relative frequencies of co-occurrence of chunk heads (i.e. verbs with nouns, verbs with prepositions, prepositions with their noun complements) within the BNC. Consequently the number of combinations the system has to check against the BNC material is reduced. The alternative candidate combinations are checked and ranked in the following way:

Initially, the relative frequency $R_{((i,j),(a,b))}$, where *(i,j)* denotes the *j*-th translation of the *i*-th chunk-head in relation to a *a*-th translation of a *b*-th chunk-head *((a,b))*, is calculated:

$$R_{((i,j),(a,b))} = \frac{c_{\{j,b\}}}{\sum_{b=1}^{v} c_{\{j,b\}}} \quad \text{(eq. 1)}$$

where *C{j,b}* is the number of co-occurrences of the *(i,j)* lemma with the *(a,b)* lemma and *v* is the number of translations provided by the lexicon for the *a*-th chunk-head.

Then, every possible combination is determined by (eq.2):

$$\prod_{\substack{i=1 \\ 1 \le j \le trj \\ 1 \le b \le trb}}^{\mu-1} R_{((i,j),(i+1,b))} \quad \text{(eq. 2)}$$

where μ is the number of the chunks in the sentence, and *trj*, *trb* are the numbers of translations for the *i*-th and *(i+1)*-th chunk-head, respectively. The combination with the higher score is chosen.

Step 2: For each translation of the SL clause, which has scored high, a comparison is run between the SL clauses and the BNC clauses. The search originates within the class of clauses containing the given verb. If no matches ('good frameworks') are found, searching has failed (at this level of development of the system). The result of each comparison is a score for the SL clause and TL clause pair, based on general chunk information, such as the number of chunks in the clause, chunk labels and chunk heads, using a pattern recognition-based method. The formula for calculating the score is

$$ClauseScore = \sum_{n-1}^{m} \left\{ ocf_n \times \frac{ChunkScore_n}{\sum_{n=1}^{m} ocf_n} \right\} \text{(eq.3)}$$

where *m* is the number of chunks in the SL clause and *ocf* is the overall cost factor of each chunk (based on the chunk type).

Chunk scores are calculated by combining the partial scores obtained after comparing the chunk label as well as the tag and the lemma of the chunk head. Given that not all chunk types are of the same significance, we need to introduce a series of weights. The formula for calculating the score for each chunk is the following:

$$ChunkScore_n = (1 - tcf_n - lcf_n) \times LabelComp_n + tcf_n \times TagComp_n + lcf_n \times LemmaComp_n$$

where *tcf* is the tag cost factor, *lcf* the lemma cost factor and *(1-tcf-lcf)* the chunk label cost factor.

Step 3: In the third step of the algorithm, the comparison is more detailed and involves comparing the tokens contained in each chunk. The SL chunks are checked against the respective chunks in the BNC clause, again using a pattern recognition-based method. At the end of this step a second score is given to each clause pair (and to each chunk of the clause) in a similar way to the second step.

The final score for each pair is the product of the clause scores obtained at steps 2 and 3. Final scores are calculated for each chunk as well. The BNC clause of the comparison pair with the highest clause score will serve as the best-matching and form the archetype of the translation. The chunk comparison pairs of the clause are then classified on the basis of their final score. Chunks scoring higher than A% will be used in the final translation without any changes. Chunks scoring between A% and B% (A > B) will be used in the final translation after modifications are made. Finally, chunks with a score lower than B% are not considered eligible candidate translations. To translate these SL chunks, we need to search the BNC again for chunks based on chunk label and head token information. Values A and B are entered as parameters to the system, so that the translator can tune the precision of the final translation.

4.2 Example of the Translation Process

The process proposed for translating a sentence with the approach presented so far is summarised in Table 1 where rows are numbered.
In (1), the SL string is a Modern Greek declarative sentence with a VSO word order.
In (2), (3) & (4), the results of tagging, lemmatising and chunking the SL sentence are shown.
In (5), the result of the dictionary look-up is shown. All possible translations are managed through the relative frequency of co-occurrence algorithm.
In (6) the chunks from the SL string are copied on the lemma-to lemma string.
In (7), the core engine searches and finds a similar string in terms of chunks and lexical heads. Furthermore, by applying the NP order mapping rule (Section 3.3), the algorithm has established an implicit link between the NPs in the SL and the TL so that TL 'cuban officers' is linked to the SL 'american officer' and TL 'continuous animosity' is linked to SL '{constant, continuous, unabated}, {tension, intensity}'.
In (8) the found BNC chunks are shown. As, in this example, the sentences are isomorphic, they coincide in terms of the number and type of chunks.
In (9) the retrieved string after synthesising appears.

4.3 Experimental Results

In Table 2, the translation results obtained from the prototype for a sample experiment are briefly presented. For this experiment, a simple sentence was used (Row 1). The results of the analysis of the sentence are shown in Rows 2 to 5, while the reference translation is shown in lemmatised form in Row 6. The experiment was carried out using a prototype of the system running under Java. The monolingual corpus consisted of 1,703,551 sentences, and the translation process was completed in 31.44 seconds on a Dell 670 Precision workstation. The top 12 sentences retrieved from the corpus as candidate translations are shown in the bottom part of Table 2, ranked according to their overall score, together with their associated scores. As can be seen, the score for step 2 is generally higher than that for step 3. In certain cases, the score of step 3 is higher for a lower-ranked sentence, though the overall score agrees to a large extent with that of step2. The system is successful in retrieving the sentences with the highest similarity to the SL sentence (sentences 1 to 6). Lower-ranked sentences seem to indicate a decreasing similarity to the reference translation. The exact ranking depends on the exact values of the weights, which are currently being fine-tuned.

5 Future Work

In the present article we have described a methodology for a machine translation system employing a limited set of resources. The approach exploits sub-sentential structure information and is based on searching and retrieving the most appropriate translation from a large monolingual corpus. It is self-evident that the accuracy and quality of the retrieved translations is heavily dependent upon the size and coverage of the given corpus.

Currently, we are experimenting on the optimisation of the proposed algorithm along the following lines:

* Extending the corpus indexing scheme, in order to accelerate the search process and improve its effectiveness
* Narrowing down the search space
* Exploring further the issue of synthesising the final translation from multiple segments (chunks/clauses)

* Studying the issue of automatic evaluation (METEOR, NIST, Papineni et al., 2002) of the output of the algorithm.

6 Acknowledgements

This work is partially supported by European Community under the Information Society Technology (IST) RTD programme. The authors are solely responsible for the content of this communication. It does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of data appearing therein.

References

Y. Al-Onaizan, U. Germann, U. Hermjakob, K. Knight, P. Koehn, D. Marcu, & K. Yamada. 2000. *Translating with Scarce Resources.* American Association for Artificial Intelligence Conference (AAAI'00), 30 July – 3 August, Austin, Texas, pages 672-678 (http://www.isi.edu/natural-language/projects/rewrite).

S. Boutsis, P. Prokopidis, V. Giouli & S. Piperidis. 2000. *A Robust Parser for Unrestricted Greek Text.* In "Proceedings of the Second International Conference on Language Resources and Evaluation", 31 May-2 June, Athens, Greece, Vol. 1, pp. 467-482.

P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer & P. S. Roosin. 1990. A Statistical Approach to Machine Translation. *Computational Linguistics*, 16(2):79-85.

L. Cranias, H. Papageorgiou & S. Piperidis. 1997. Example Retrieval from a Translation Memory. *Natural Language Engineering*, 3:255-277.

I. Dologlou, S. Markantonatou, G. Tambouratzis, O. Yannoutsou, A. Fourla & N. Ioannou. 2003. *Using Monolingual Corpora for Statistical Machine Translation.* In "Proceedings of EAMT/CLAW 2003", Dublin, Ireland, 15-17 May.

H. Dyvik. 1995. Exploiting Structural Similarities in Machine Translation. *Computers and the Humanities*, 28:225-234.

ILSP Internal Document 2004. *Specifications for METIS lexicon.*

N. Ioannou. 2003. METIS: *Statistical Machine Translation Using Monolingual Corpora.* In "Proceedings of the Workshop on Text Processing for Modern Greek: From Symbolic to Statistical Approaches" (held in conjunction with the 6th International Conference of Greek Linguistics), Rethymno, Greece, 20 September. ISBN:960-88268-0-2.

P. Labropoulou, E. Mantzari & M. Gavrilidou. 1996. *Lexicon-Morphosyntactic Specifications: Language Specific Instantiation* (Greek), PP-PAROLE, MLAP 63-386 report.

METEOR: http://www-2.cs.cmu.edu/~banerjee/MT/METEOR/

K. McTait. 2003. *Translation Patterns, Linguistic Knowledge and Complexity in EBMT.* In "Recent Advances in Example-Based Machine Translation", M. Carl and A. Way (eds.) Kluwer Academic Publishers, pp. 307-338.

M. Nagao. 1984. *A Framework of a Mechanical Translation between Japanese and English by Analogy Principle.* In "Artificial and Human Intelligence", A. Elithorn and R. Banerji (eds). North-Holland.

NIST: http://www.nist.gov/speech/tests/mt/

K. A. Papineni, S. Roukos, T. Ward & W. J. Zhu. 2002. *Bleu, a method for automatic evaluation of Machine Translation*. In "Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics", Philadelphia (USA), pages 311-318.

M. Popovic and H. Ney. 2005. *Exploiting Phrasal Lexica and Additional Morpho-syntactic Language Resources for Statistical Machine Translation with Scarce Training Data.* EAMT 10th Annual Conference, 30-31 May, Budapest, Hungary.

V. Vandeghinste 2005. *Manual for ShaRPa 2.0.* Internal Report, Centre for Computational Linguistics, K.U.Leuven.

A. Way 2003. *Translating with Examples: The LFG-DOT Models of Translation,* In "Recent Advances in Example-Based Machine Translation", M. Carl and A. Way (eds.). Kluwer Academic Publishers, pages 443-472.

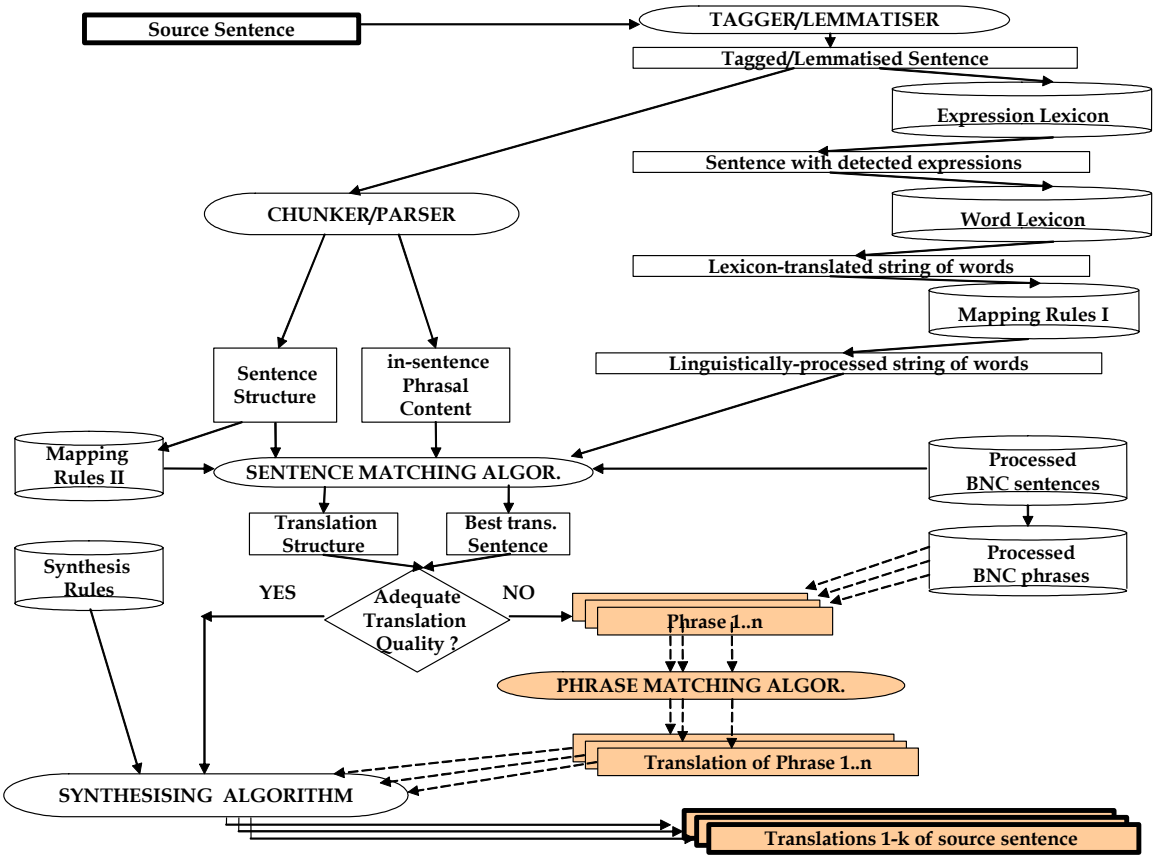


Figure 1: General architecture of the proposed methodology

1	περιγράφουν	Αμερικανοί	αξιωματούχοι	τη	διαρκή	ένταση	μεταξύ	Ελλάδας	και	Τουρκίας
2	**Vb**	**Aj**	**No**	**At**	**Aj**	**No**	**AsPp**	**No**	**Cj**	**No**
3	περιγράφω	Αμερικανός	αξιωματούχος	ο	διαρκής	ένταση	μεταξύ	Ελλάδα	και	Τουρκία
4	**VG**	**NP**		**NP**			**PP**			
5	describe	American	officer official	the	constant continuous unabated	tension intensity	between mean-while	Greece	and	Turkey
6	**VG**	**NP**		**NP**			**PP**			

Searching for match in pre-processed BNC										
7	cuban	officers	describe	the	continuous	animosity	between	Greece	and	Turkey
8	**NP**		**VG**	**NP**			**PP**			
9	american	officers	describe	the	continuous	tension intensity	between	Greece	and	Turkey

Table 1: An example of the translation approach

level	Sentence							Score (step 2)	Score (step3)	Overall Score
SL string (1)	Η	γυναίκα	έχασε	έναν	αδελφό	στον	πόλεμο			
Tags	**At**	**No**	**Vb**	**Card**	**No**		**AsPp**			
Lemmata (3)	Ο	γυναίκα	χάνω	ένας	αδελφός	στου	πόλεμο			
SL string chunked (4)	NP		VG	NP		PP				
Lemma-to-lemma (5)	The	woman wife lady	lose miss misplace	a one	brother	in during	war battle			
Reference translation (6)	The	woman	lose	a	brother	in	war			
Retrieved sentences from pre-processed corpus										
Retrieved sentence 1	The poor woman		lost	her older brother		in war		100	95.9	95.9
Retrieved sentence 2	The woman		lost	her brother		during the great war		94.9	88.2	83.8
Retrieved sentence 3	The woman		lost	a dog		in war		93.3	88.6	79.9
Retrieved sentence 4	Both women		lost	their husbands		in the war		93.3	82.4	76.9
Retrieved sentence 5	The man		lost	a brother		in war		88.2	78.5	69.3
Retrieved sentence 6	The brother		lost	his wife		in war		86.6	74.9	64.9
Retrieved sentence 7	The woman		lost	an apple		in the kitchen		86.6	73.2	63.4
Retrieved sentence 8	Britain		lost	a lot		in that war too		86.6	71.8	62.2
Retrieved sentence 9	The brother		lost	her		in the war		83.8	72.2	60.5
Retrieved sentence 10	He		lost	two sons		in the Great war		83.8	66.1	55.3
Retrieved sentence 11	They both		lost	their husbands		in the war		81.0	68.0	55.1
Retrieved sentence 12	Pitch Barratt Developments		lost	9p		to 173p		80.0	61.2	48.9

Table 2: Translation results generated by the prototype for a sample sentence

Dependency Treelet Translation: The convergence of statistical and example-based machine-translation?

Arul Menezes and Chris Quirk
Microsoft Research
One Microsoft Way, Redmond, WA 98052
{arulm,chrisq}@microsoft.com

Abstract

We describe a novel approach to machine translation that combines the strengths of the two leading corpus-based approaches: Phrasal SMT and EBMT. We use a syntactically informed decoder and reordering model based on the source dependency tree, in combination with conventional SMT models to incorporate the power of phrasal SMT with the linguistic generality available in a parser. We show that this approach significantly outperforms a leading string-based Phrasal SMT decoder and an EBMT system. We present results from two radically different language pairs, and investigate the sensitivity of this approach to parse quality by using two distinct parsers and oracle experiments. We also validate our automated BLEU scores with a small human evaluation.

1. Introduction

Current example-based (EBMT) and statistical (SMT) machine translation systems both use phrases learned from parallel corpora, yet while the two approaches are closer than ever, some critical differences remain. (Way & Gough, 2005) On the one hand, while statistical systems excel at producing correct, even idiomatic translations at the local level, they are still challenged by many linguistic phenomena, such as global constituent ordering. While SMT excels at translating domain-specific terminology and fixed phrases, grammatical generalizations are poorly captured and often mangled in translation (Thurmair, 04).

On the other hand, many EBMT systems do not fully exploit the power that results from a combination of multiple powerful statistical models. In particular, we believe that the recent dominance of SMT systems in competitive evaluations indicates that an end-to-end search over a weighted linear combination of statistical models is essential for high-quality translation. However, there is no indication that these models must necessarily be linguistically uninformed.

1.1. Limitations of string-based phrasal SMT

State-of-the-art phrasal SMT systems such as (Koehn et al., 03) and (Vogel et al., 03) model translations of *phrases* (here, strings of adjacent words, not syntactic constituents) rather than individual words. Arbitrary reordering of words is allowed within memorized phrases, but typically only a small amount of phrase reordering is allowed, modeled in terms of offset positions at the string level. This reordering model is very limited in terms of linguistic generalizations. For instance, when translating English to Japanese, an ideal system would automatically learn large-scale typological differences: English SVO clauses generally become Japanese SOV clauses, English post-modifying prepositional phrases become Japanese pre-modifying postpositional phrases, etc. A phrasal SMT system may learn the internal reordering of specific common phrases, but it cannot generalize to unseen phrases that share the same linguistic structure.

In addition, these systems are limited to phrases contiguous in both source and target, and thus cannot learn the generalization that English *not* may translate as French *ne…pas* except in the context of specific intervening words.

1.2. Previous work on syntactic SMT and statistical EBMT

The hope in the SMT community has been that the incorporation of syntax would address these issues, but that promise has yet to be realized[1].

[1] Note that as the focus of this paper is decoding, we do not discuss the large body of work incorporating syntax into the word alignment process.

One simple means of incorporating syntax into SMT decoding is by re-ranking the *n*-best list of a baseline SMT system using various syntactic models, but Och et al. (04) found very little positive impact with this approach. However, an *n*-best list of even 16,000 translations captures only a tiny fraction of the ordering possibilities of a 20 word sentence; re-ranking provides the syntactic model no opportunity to boost or prune large sections of that search space.

Inversion Transduction Grammars (Wu, 97), or ITGs, treat translation as a process of parallel parsing of the source and target language via a synchronized grammar. To make this process computationally efficient, however, some severe simplifying assumptions are made, such as using a single non-terminal label. This results in the model simply learning a very high level preference regarding how often nodes should switch order without any contextual information. Also these translation models are intrinsically word-based; phrasal combinations are not modeled directly, and results have not been competitive with the top phrasal SMT systems.

Along similar lines, Alshawi et al. (2000) treat translation as a process of simultaneous induction of source and target dependency trees using head-transduction; again, no separate parser is used.

Yamada and Knight (01) employ a parser in the target language to train probabilities on a set of operations that convert a target language tree to a source language string. This improves fluency slightly (Charniak et al., 03), but fails to significantly impact overall translation quality. This may be because the parser is applied to MT output, which is notoriously unlike native language, and no additional insight is gained via source language analysis.

Lin (04) translates dependency trees using paths. This is the first attempt to incorporate large phrasal SMT-style memorized patterns together with a separate source dependency parser and SMT models. However the phrases are limited to linear paths in the tree, the only SMT model used is a maximum likelihood channel model and there is no ordering model. Reported BLEU scores are far below the leading phrasal SMT systems.

Aue et al. (04) recently reported incorporating a logical form (LF) or dependency tree-based statistical language model into an existing EBMT system. MSR-MT (Menezes & Richardson, 03) parses both source and target languages to obtain a logical form (LF), and translates source LFs using memorized aligned LF examples to produce a target LF. It utilizes a separate sentence realization component (Ringger et al., 04) to turn this into a target sentence. As a result, Aue could not use an end-to-end search over a linear combination of models, and the simple addition of a single target language model did not provide much improvement.

2. Dependency Treelet Translation

In this paper we propose a novel dependency tree-based approach to phrasal SMT which uses tree-based 'phrases' and a tree-based ordering model in combination with conventional SMT models to produce translations significantly better than a leading string-based system.

Our system employs a source-language dependency parser, a target language word segmentation component, and an unsupervised word alignment component to learn treelet translations from a parallel sentence-aligned corpus. We begin by parsing the source text to obtain dependency trees and word-segmenting the target side, then applying an off-the-shelf word alignment component to the bitext.

The word alignments are used to project the source dependency parses onto the target sentences. From this aligned parallel dependency corpus we extract a treelet translation model incorporating source and target treelet pairs, where a *treelet* is defined to be an arbitrary connected subgraph of the dependency tree. A unique feature is that we allow treelets with a wildcard root, effectively allowing mappings for siblings in the dependency tree. This allows us to model important phenomena, such as *not* … → *ne…pas*. We also train a variety of statistical models on this aligned dependency tree corpus, including a channel model and an order model.

To translate an input sentence, we parse the sentence, producing a dependency tree for that sentence. We then employ a decoder to find a combination and ordering of treelet translation pairs that cover the source tree and are optimal according to a set of models that are combined in a log-linear framework as in (Och, 03).

This approach offers the following advantages over string-based SMT systems: Instead of

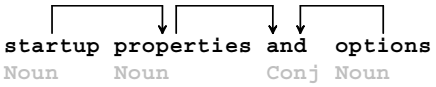


Figure 1. An example dependency tree.

limiting learned phrases to contiguous word sequences, we allow translation by all possible phrases that form connected subgraphs (treelets) in the source and target dependency trees. This is a powerful extension: the vast majority of surface-contiguous phrases are also treelets of the tree; in addition, we gain discontiguous phrases, including combinations such as verb-object, article-noun, adjective-noun etc. regardless of the number of intervening words.

Another major advantage is the ability to employ more powerful models for reordering source language constituents. These models can incorporate information from the source analysis. For example, we may model directly the probability that the translation of an object of a preposition in English should precede the corresponding postposition in Japanese, or the probability that a pre-modifying adjective in English translates into a post-modifier in French.

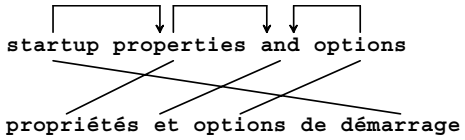
2.1. Parsing and alignment

We require a source language dependency parser that produces unlabeled, ordered dependency trees and annotates each source word with a part-of-speech (POS). An example dependency tree is shown in Figure 1. The arrows indicate the head annotation, and the POS for each candidate is listed underneath. For the target language we only require word segmentation.

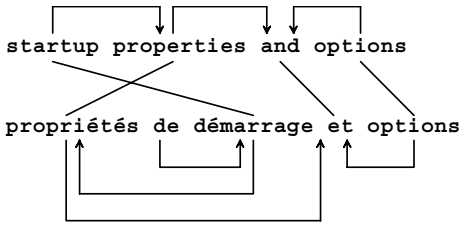
To obtain word alignments we currently use GIZA++ (Och & Ney, 03). We follow the common practice of deriving many-to-many alignments by running the IBM models in both directions and combining the results heuristically. Our heuristics differ in that they constrain many-to-one alignments to be contiguous in the source dependency tree. A detailed description of these heuristics can be found in (Quirk et al, 2004).

2.2. Projecting dependency trees

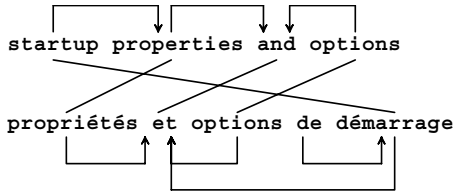
Given a word aligned sentence pair and a source dependency tree, we use the alignment to project the source structure onto the target sentence. One-to-one alignments project directly to create a



(a) Word alignment.



(b) Dependencies after initial projection.



(c) Dependencies after reattachment step.

Figure 2. Projection of dependencies.

target tree isomorphic to the source. Many-to-one alignments project similarly; since the 'many' source nodes are connected in the tree, they act as if condensed into a single node. In the case of one-to-many alignments we project the source node to the rightmost[2] of the 'many' target words, and make the rest of the target words dependent on it.

Unaligned target words[3] are attached into the dependency structure as follows: assume there is an unaligned word t_j in position j. Let $i < j$ and $k > j$ be the target positions closest to j such that t_i depends on t_k or vice versa: attach t_j to the lower of t_i or t_k. If all the nodes to the left (or right) of position j are unaligned, attach t_j to the left-most (or right-most) word that is aligned.

The target dependency tree created in this process may not read off in the same order as the target string, since our alignments do not enforce phrasal cohesion. For instance, consider the projection of the parse in Figure 1 using the word alignment in Figure 2a. Our algorithm produces the dependency tree in Figure 2b. If we read off the leaves in a left-to-right in-order traversal, we

[2] If the target language is Japanese, leftmost may be more appropriate.
[3] Source unaligned nodes do not present a problem, with the exception that if the root is unaligned, the projection process produces a forest of target trees anchored by a dummy root.

do not get the original input string: *de démarrage* appears in the wrong place.

A second reattachment pass corrects this situation. For each node in the wrong order, we reattach it to the lowest of its ancestors such that it is in the correct place relative to its siblings and parent. In Figure 2c, reattaching *démarrage* to *et* suffices to produce the correct order.

2.3. Extracting treelet translation pairs

From the aligned pairs of dependency trees we extract all pairs of aligned source and target treelets along with word-level alignment linkages, up to a configurable maximum size. We also keep treelet counts for maximum likelihood estimation.

2.4. Order model

Phrasal SMT systems often use a model to score the ordering of a set of phrases. One approach is to penalize any deviation from monotone decoding; another is to estimate the probability that a source phrase in position *i* translates to a target phrase in position *j* (Koehn et al., 03).

We attempt to improve on these approaches by incorporating syntactic information. Our model assigns a probability to the order of a target tree given a source tree. Under the assumption that constituents generally move as a whole, we predict the probability of each given ordering of modifiers independently. That is, we make the following simplifying assumption (where *c* is a function returning the set of nodes modifying *t*):

$$\mathrm{P}(order(T) \mid S,T) = \prod_{t \in T} \mathrm{P}(order(c(t)) \mid S,T)$$

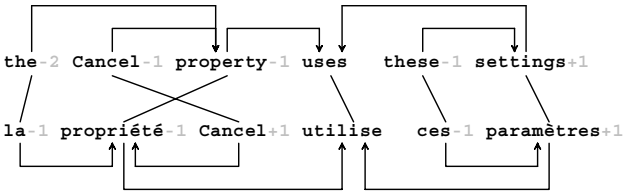
Furthermore, we assume that the position of each child can be modeled independently in terms of a head-relative position:

$$\mathrm{P}(order(c(t)) \mid S,T) = \prod_{m \in c(t)} \mathrm{P}(pos(m,t) \mid S,T)$$

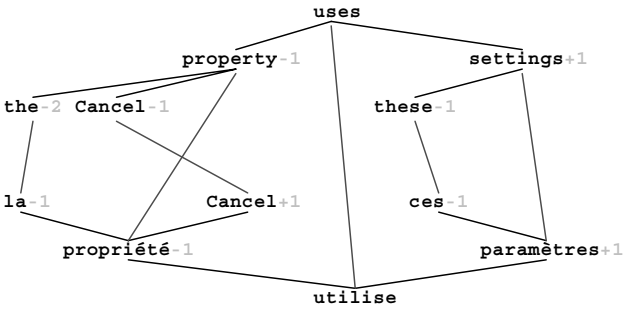
Figure 3a demonstrates an aligned dependency tree pair annotated with head-relative positions; Figure 3b presents the same information in an alternate tree-like representation.

We currently use a small set of features reflecting very local information in the dependency tree to model P(pos(*m*,*t*) | *S*, *T*):

- The lexical items of the head and modifier.
- The lexical items of the source nodes aligned to the head and modifier.



(a) Head annotation representation



(b) Branching structure representation.

Figure 3. Aligned dependency tree pair, annotated with head-relative positions

- The part-of-speech ("cat") of the source nodes aligned to the head and modifier.
- The head-relative position of the source node aligned to the source modifier. (One can also include features of siblings to produce a Markov ordering model. However, we found that this had little impact in practice).

As an example, consider the children of *propriété* in Figure 3. The head-relative positions of its modifiers *la* and *Cancel* are -1 and +1, respectively. Thus we try to predict as follows:

P(pos(m_1) = -1 |
lex(m_1)="*la*", lex(*h*)="*propriété*",
lex(src(m_1))="*the*", lex(src(*h*)="*property*",
cat(src(m_1))=Determiner, cat(src(*h*))=Noun,
position(src(m_1))=-2) ·
P(pos(m_2) = +1 |
lex(m_2)="*Cancel*", lex(*h*)="*propriété*",
lex(src(m_2))="*Cancel*", lex(src(h))="*property*",
cat(src(m_2))=Noun, cat(src(h))=Noun,
position(src(m_2))=-1)

The training corpus acts as a supervised training set: we extract a training feature vector from each of the target language nodes in the aligned dependency tree pairs. Together these feature vectors are used to train a decision tree (Chickering, 02). The distribution at each leaf of the DT can be used to assign a probability to each possible target language position. A more detailed description is available in (Quirk at al, 2004).

2.5. Other models

Channel Models: We incorporate two distinct channel models, a maximum likelihood estimate (MLE) model and a model computed using Model-1 word-to-word alignment probabilities as in (Vogel et al., 03). The MLE model effectively captures non-literal phrasal translations such as idioms, but suffers from data sparsity. The word-to-word model does not typically suffer from data sparsity, but prefers more literal translations.

Given a set of treelet translation pairs that cover a given input dependency tree and produce a target dependency tree, we model the probability of source given target as the product of the individual treelet translation probabilities: we assume a uniform probability distribution over the decompositions of a tree into treelets.

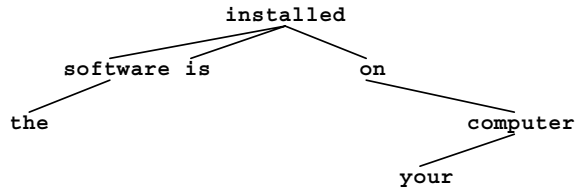
Target Model: Given an ordered target language dependency tree, it is trivial to read off the surface string. We evaluate this string using a trigram model with modified Kneser-Ney smoothing.

Miscellaneous Feature Functions: The log-linear framework allows us to incorporate other feature functions as 'models' in the translation process. For instance, using fewer, larger treelet translation pairs often provides better translations, since they capture more context and allow fewer possibilities for search and model error. Therefore we add a feature function that counts the number of phrases used. We also add a feature that counts the number of target words; this acts as an insertion/deletion bonus/penalty.

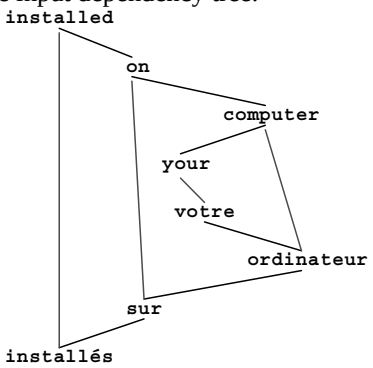
3. Decoding

The challenge of tree-based decoding is that the traditional left-to-right decoding approach of string-based systems is inapplicable. Additional challenges are posed by the need to handle treelets—perhaps discontiguous or overlapping—and a combinatorially explosive ordering space.

Our decoding approach is influenced by ITG (Wu, 97) with several important extensions. First, we employ treelet translation pairs instead of single word translations. Second, instead of modeling rearrangements as either preserving source order or swapping source order, we allow the dependents of a node to be ordered in any arbitrary manner and use the order model described in section 2.4 to estimate probabilities.



(a) Example input dependency tree.



(b) Example treelet translation pair.

Figure 4. Example decoder structures.

Finally, we use a log-linear framework for model combination that allows any amount of other information to be modeled.

We will initially approach the decoding problem as a bottom up, exhaustive search. We define the set of all possible treelet translation pairs of the subtree rooted at each input node in the following manner: A treelet translation pair *x* is said to *match* the input dependency tree *S* iff there is some connected subgraph *S'* that is identical to the source side of *x*. We say that *x* *covers* all the nodes in *S'* and is *rooted* at source node *s*, where *s* is the root of matched subgraph *S'*.

We first find all treelet translation pairs that match the input dependency tree. Each matched pair is placed on a list associated with the input node where the match is rooted. Moving bottom-up through the input dependency tree, we compute a list of *candidate translations* for the input subtree rooted at each node *s*, as follows:

Consider in turn each treelet translation pair *x* rooted at *s*. The treelet pair *x* may cover only a portion of the input subtree rooted at *s*. Find all descendents *s'* of *s* that are *not* covered by *x*, but whose parent *s''* is covered by *x*. At each such node *s''* look at all interleavings of the children of *s''* specified by *x*, if any, with each translation *t'* from the candidate translation list[4] of each child

[4] Computed by the previous application of this procedure to *s'* during the bottom-up traversal.

s'. Each such interleaving is scored using the models previously described and added to the candidate translation list for that input node. The resultant translation is the best scoring candidate for the root input node.

As an example, see the example dependency tree in Figure 4a and treelet translation pair in 4b. This treelet translation pair covers all the nodes in 4a except the subtrees rooted at *software* and *is*. We first compute (and cache) the candidate translation lists for the subtrees rooted at *software* and *is*, then construct full translation candidates by attaching those subtree translations to *installés* in all possible ways. The order of *sur* relative to *installés* is fixed; it remains to place the translated subtrees for *the software* and *is*. Note that if c is the count of children specified in the mapping and r is the count of subtrees translated via recursive calls, then there are $(c+r+1)!/(c+1)!$ orderings. Thus $(1+2+1)!/(1+1)! = 12$ candidate translations are produced for each combination of translations of *the software* and *is*.

3.1. Optimality-preserving optimizations

Dynamic Programming

Converting this exhaustive search to dynamic programming relies on the observation that scoring a translation candidate at a node depends on the following information from its descendents: the order model requires features from the root of a translated subtree, and the target language model is affected by the first and last two words in each subtree. Therefore, we need to keep the best scoring translation candidate for a given subtree for each combination of (head, leading bigram, trailing bigram), which is, in the worst case, $O(V^5)$, where V is the vocabulary size. The dynamic programming approach therefore does not allow for great savings in practice because a trigram target language model forces consideration of context external to each subtree.

3.2. Lossy optimizations

The following optimizations do not preserve optimality, but work well in practice.

N-best lists

Instead of keeping the full list of translation candidates for a given input node, we keep a top-scoring subset of the candidates. While the decoder is no longer guaranteed to find the optimal translation, in practice the quality impact is minimal with a list size ≥ 10 (see Table 5.6).

Variable-sized n-best lists: A further speedup can be obtained by noting that the number of translations using a given treelet pair is exponential in the number of subtrees of the input not covered by that pair. To limit this explosion we vary the size of the *n*-best list on any recursive call in inverse proportion to the number of subtrees uncovered by the current treelet. This has the intuitive appeal of allowing a more thorough exploration of large treelet translation pairs (that are likely to result in better translations) than of smaller, less promising pairs.

Pruning treelet translation pairs

Channel model scores and treelet size are powerful predictors of translation quality. Heuristically pruning low scoring treelet translation pairs before the search starts allows the decoder to focus on combinations and orderings of high quality treelet pairs.

- Only keep those treelet translation pairs with an MLE probability above a threshold t.
- Given a set of treelet translation pairs with identical sources, keep those with an MLE probability within a ratio r of the best pair.
- At each input node, keep only the top k treelet translation pairs rooted at that node, as ranked first by size, then by MLE channel model score, then by Model 1 score. The impact of this optimization is explored in Table 5.6.

Greedy ordering

The complexity of the ordering step at each node grows with the factorial of the number of children to be ordered. This can be tamed by noting that given a fixed pre- and post-modifier count, our order model is capable of evaluating a single ordering decision independently from other ordering decisions.

One version of the decoder takes advantage of this to severely limit the number of ordering possibilities considered. Instead of considering all interleavings, it considers each potential modifier position in turn, greedily picking the most probable child for that slot, moving on to the next slot, picking the most probable among the remaining children for that slot and so on.

		English	French	English	Japanese
Training	Sentences	500,000		500,000	
	Words	6,598,914	7,234,153	7,909,198	9,379,240
	Vocabulary	72,440	80,758	66,731	68,048
	Singletons	38,037	39,496	50,381	52,911
Test	Sentences	10,000		10,000	
	Words	133,402	153,701	175,655	211,139

Table 4.1 Data characteristics

The complexity of greedy ordering is linear, but at the cost of a noticeable drop in BLEU score (see Table 5.4). Under default settings our system tries to decode a sentence with exhaustive ordering until a specified timeout, at which point it falls back to greedy ordering.

4. Experiments

We evaluated the translation quality of the system using the BLEU metric (Papineni et al., 02) under a variety of configurations. We compared against two radically different types of systems to demonstrate the competitiveness of this approach:

- Pharaoh: A leading phrasal SMT decoder (Koehn et al., 03).
- The MSR-MT system described in Section 1, an EBMT/hybrid MT system.

4.1. Language pairs

We ran experiments in English→French and English→Japanese. The latter was chosen deliberately to highlight the challenges facing string-based MT approaches in language pairs with significant word-order differences.

Word order in Japanese is fundamentally very different from English. English is generally SVO (subject first, then verb, then object), where Japanese is SOV with a strong bias for head-final structures. Several other differences include:

- Word order is more flexible, since verbal arguments are generally indicated by postpositions, e.g. a direct object is indicated by the postposition を (o), a subject by が (ga).
- Most post-modifying English phrases (such as relative clauses and prepositional phrases) are translated as Japanese pre-modifiers; demonstratives and adjectives remain pre-modifiers.
- Verbal and adjectival morphology in Japanese is relatively complex: information contained in English pre-modifying modals and auxiliaries is often represented as verbal morphology.
- Japanese nouns and noun phrases are not marked for definiteness or number.

The word-aligned sentence pair in Figure 1 demonstrates many of these phenomena.

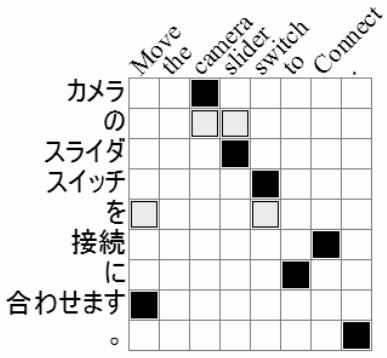


Figure 1. English-Japanese word alignment

4.2. Data

We used a corpus of Microsoft technical data (e.g., support articles, product documentation) containing over 1 million sentence pairs for each language-pair. We excluded sentences containing XML or HTML tags and for each language pair randomly selected training data sets ranging from 1,000 to 500,000 sentence pairs as well as 10,000 sentences for development testing and parameter tuning, 250 sentences for lambda training and 10,000 sentences for testing. Table 4.1 presents some characteristics of this corpus.

4.3. Training

We parsed the source (English) side of the corpus using two different parsers: NLPWIN, a broad-coverage rule-based parser developed at Microsoft Research able to produce syntactic analyses at varying levels of depth (Heidorn, 02)

	English→French, 100K		English→Japanese, 500K	
	BLEU	**Sents/min**	**BLEU**	**Sents/min**
Pharaoh monotone	37.06	4286	25.06	1600
Pharaoh	38.83	162	30.58	82
MSR-MT	35.26	453	-	-
Treelet	40.66	10.1	33.18	21

Table 5.1 System comparisons

		1k	3k	10k	30k	100k	300k	500K
English → French	Pharaoh	17.20	22.51	27.70	33.73	38.83	42.75	-
	Treelet	18.70	25.39	30.96	35.81	40.66	44.32	-
English → Japanese	Pharaoh	14.85	15.99	18.18	21.89	23.01	26.67	30.58
	Treelet	13.90	15.39	18.94	23.99	25.68	29.97	33.18

Table 5.2 BLEU scores at different training set sizes, phrase/treelet size 4

and a Treebank parser (Bikel, 04). For the purposes of these experiments we used a dependency tree output with part-of-speech tags and unstemmed, case-normalized surface words.

For word alignment, we used GIZA++, following a standard training regimen of five iterations of Model 1, five iterations of the HMM Model, and five iterations of Model 4, in both directions.

The target language models were trained using only the French and Japanese sides, respectively, of the parallel corpus; additional monolingual data may improve its performance. Finally we trained lambdas via Maximum BLEU (Och, 03) on 250 held-out sentences with a single reference translation, and tuned the decoder optimization parameters (*n*-best list size, timeouts etc) on the development test set.

Pharaoh

The same GIZA++ alignments as above were used in the Pharaoh decoder. We used the heuristic combination described in (Och & Ney, 03) and extracted phrasal translation pairs from this combined alignment as described in (Koehn et al., 03). Except for the order model (Pharaoh uses a penalty on the deviance from monotone), the same models were used: MLE channel model, Model 1 channel model, target language model, phrase count, and word count. Lambdas were trained in the same manner (Och, 03).

MSR-MT

MSR-MT used its own word alignment approach as described in (Menezes & Richardson, 03) on the same training data. MSR-MT does not use lambdas or a target language model.

5. Results

We present BLEU scores on an unseen 10,000 sentence test set using a single reference translation for each sentence. Speed numbers are the end-to-end translation speed in sentences per minute. Unless otherwise specified all results are based on a phrase size of 4 and a training set size of 100,000 sentences for English→French and 500,000 sentences for English→ Japanese. Unless otherwise noted all the differences between systems are statistically significant at $P < 0.01$

Comparative results are presented in Table 5.1. Pharaoh monotone refers to Pharaoh with phrase reordering disabled.

Table 5.2 compares the systems at different training corpus sizes. All the differences are statistically significant at $P < 0.01$ except for English→Japanese at training set sizes less than 30K. Note that in English→French, where word order differences are mainly local, the gap between the systems narrows slightly with larger corpus sizes, however in English→Japanese, with global ordering differences, the treelet system's margin over Pharaoh (initially negative) actually increases with increasing corpus size.

Table 5.3 compares Pharaoh and the Treelet system at different phrase sizes. The wide gap at smaller phrase sizes is particularly striking. It appears that while Pharaoh depends heavily on long phrases to encapsulate reordering, our dependency tree-based ordering model enables credible performance even with short phrases/treelets. Our treelet system with two-word treelets outperforms Pharaoh with six-word phrases.

Max size	English→French, 100K		English→Japanese, 100K		English→Japanese, 500K	
	Treelet BLEU	**Pharaoh BLEU**	**Treelet BLEU**	**Pharaoh BLEU**	**Treelet BLEU**	**Pharaoh BLEU**
1	37.50	23.18	22.36	12.75	26.95	17.72
2	39.84	32.07	24.53	18.63	31.33	24.30
3	40.36	37.09	25.44	21.37	32.58	28.15
4 (default)	40.66	38.83	25.68	23.01	33.18	30.58
5	40.71	39.41	25.87	23.82	-	-
6	40.74	39.72	25.92	24.43	-	-

Table 5.3 Effect of maximum treelet/phrase size

		English→French, 100K		English→Japanese, 500K	
		BLEU	**Sents/min**	**BLEU**	**Sents/min**
Monotone	Pharaoh	37.06	4286	25.06	1600
	Treelet with no order model	35.35	39.7	26.43	67
Non-monotone	Pharaoh (default)	38.83	162	30.58	82
	Treelet: greedy ordering	38.85	13.1	31.99	43
	Treelet: exhaustive (default)	40.66	10.1	33.18	21

Table 5.4 Effect of ordering strategy

Table 5.4 compares different ordering strategies. In contrast to results reported for English-Chinese (Vogel et al., 03), monotone decoding severely degrades the performance of both systems in English→Japanese, presumably due to the large ordering variation between the two languages. In English-French the degradation is less marked.

	BLEU
Pharaoh	23.01
NLPWIN parser: top parse only	25.68
Bikel parser: top parse only	24.15

Table 5.5 Using different parsers
(English→Japanese, data size 100k, phrase size 4)

Table 5.5 shows the translation results are not dependent on one particular parser, though a a parser trained on a different domain (here, the Treebank) is at a disadvantage.

	BLEU
Pharaoh	30.58
Single NLPWIN parse	33.18
Top 100 NLPWIN parses	34.13
Oracle selection (top 100 NLPWIN parses)	36.91

Table 5.6 Using multiple parses, parse oracle
(English→Japanese, data size 500k, phrase size 4)

Table 5.6 shows the impact of using the top 100 NLPWIN parses even without any parse scoring. The last line in the table is a parse oracle experiment to explore the potential quality impact of better parse selection – the oracle picks and translates the one best parse from the top 100 parses.

Table 5.7 is a translation oracle experiment that demonstrates the impact of model error. The oracle picks the translation with the highest BLEU score from among the top N translations produced by the treelet system. Better models may improve performance, though Och et al. (04) suggests achieving this gain this may be difficult.

Number of translations available to oracle	**BLEU**
1	33.18
4	35.30
16	37.38
64	38.56
256	38.70

Table 5.7 Translation oracle
(English→Japanese, data size 500k, phrase size 4)

5.1. Human Evaluation

Two human raters were presented (in random order) both Pharaoh and Treelet translations of 100 sentences between 10 and 25 words and corresponding source and reference translations. They were asked to pick the more accurate translation. Table 5.8 shows that for most of the sentences, humans prefer the Treelet translations, which is consistent with the BLEU scores above.

		Rater 1			
		Treelet	*Neither*	*Pharaoh*	
Rater 2	*Treelet*	26	21	3	**50**
	Neither	4	27	3	**34**
	Pharaoh	0	11	5	**16**
		30	**59**	**11**	

Table 5. 8 Human evaluation of 100 sentences (English→Japanese, data size 500k, phrase size 4)

6. Conclusions and Future Work

We presented a novel approach to syntactically-informed statistical machine translation that leverages a parsed dependency tree representation of the source language via a tree-based ordering model and a syntactically informed decoder. We showed that it outperforms a leading phrasal SMT decoder in BLEU and human quality judgments. We also showed that it out-performed our own logical form-based EBMT/hybrid MT system.

Even in the absence of a parse quality metric, we found that employing multiple parses could improve translation quality. Adding a parse probability may help further the gains from these additional possible analyses.

The syntactic information used in these models is still rather shallow. Order modeling may benefit from additional information such as semantic roles or morphological features. Furthermore, different model structures, machine learning techniques, and target feature representations all have the potential for significant improvements.

References

Alshawi, Hiyan, Srinivas Bangalore, and Shona Douglas. Learning dependency translation models as collections of finite-state head transducers. Computational Linguistics, 26(1):45–60, 2000.

Aue, Anthony, Arul Menezes, Robert C. Moore, Chris Quirk, and Eric Ringger. Statistical machine translation using labeled semantic dependency graphs. TMI 2004.

Charniak, Eugene, Kevin Knight, and Kenji Yamada. Syntax-based language models for statistical machine translation. MT Summit 2003.

Cherry, Colin and Dekang Lin. A probability model to improve word alignment. ACL 2003.

Chickering, David Maxwell. The WinMine Toolkit. Microsoft Research Technical Report: MSR-TR-2002-103.

Ding, Yuan and Martha Palmer. Automatic learning of parallel dependency treelet pairs. IJCNLP 2004.

Heidorn, George. (2000). "Intelligent writing assistance". In Dale et al. Handbook of Natural Language Processing, Marcel Dekker.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. Statistical phrase based translation. NAACL 2003.

Lin, Dekang. A path-based transfer model for machine translation. COLING 2004.

Menezes, Arul and Stephen D. Richardson. A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In Recent Advances in Example-Based Machine Translation, M. Carl & A. Way, Eds, Kluwer Academic Publishers, 2003.

Och, Franz Josef and Hermann Ney. A systematic comparison of various statistical alignment models, Computational Linguistics, 29(1):19-51, 2003.

Och, Franz Josef. Minimum error rate training in statistical machine translation. ACL 2003.

Och, Franz Josef, et al. A smorgasbord of features for statistical machine translation. HLT/NAACL 2004.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. ACL 2002.

Quirk, Chris, Arul Menezes, and Colin Cherry. Dependency Tree Translation. Microsoft Research Technical Report: MSR-TR-2004-113.

Ringger, Eric, et al. Linguistically informed statistical models of constituent structure for ordering in sentence realization. COLING 2004.

Thurmair, Gregor. Comparing rule-based and statistical MT output. Workshop on the amazing utility of parallel and comparable corpora, LREC, 2004.

Vogel, Stephan, Ying Zhang, Fei Huang, Alicia Tribble, Ashish Venugopal, Bing Zhao, and Alex Waibel. The CMU statistical machine translation system. MT Summit 2003.

Way, A. and N. Gough. Comparing Example-Based and Statistical Machine Translation. Journal of Natural Language Engineering, June 2005

Wu, Dekai. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. Computational Linguistics, 23(3):377–403, 1997.

Yamada, Kenji and Kevin Knight. A syntax-based statistical translation model. ACL, 2001.

An Example-Based Approach to Translating Sign Language

Sara Morrissey
School of Computing
Dublin City University
Dublin 9, Ireland
smorri@computing.dcu.ie

Andy Way
School of Computing
Dublin City University
Dublin 9, Ireland
away@computing.dcu.ie

Abstract

Users of sign languages are often forced to use a language in which they have reduced competence simply because documentation in their preferred format is not available. While some research exists on translating between natural and sign languages, we present here what we believe to be the first attempt to tackle this problem using an example-based (EBMT) approach.

Having obtained a set of English–Dutch Sign Language examples, we employ an approach to EBMT using the 'Marker Hypothesis' (Green, 1979), analogous to the successful system of (Way & Gough, 2003), (Gough & Way, 2004a) and (Gough & Way, 2004b). In a set of experiments, we show that encouragingly good translation quality may be obtained using such an approach.

Key-words: Example-based machine translation, sign languages, Marker Hypothesis, ECHO corpus.

1 Introduction

Just like speakers of a less widely spoken language are often not catered for properly with respect to the provision of documentation in their preferred language, users of sign languages (SLs) observe similar restrictions. Having to read documents in the *lingua franca* often causes them some hindrance. This is because a system of 'oralism' (the practice of teaching Deaf students through spoken language using amplification devices and lip-reading, to the exclusion of all sign language communication) is used in most Deaf schools. As the students lack the ability to hear the language, on average their literacy competencies remain at approximately that of a ten year old (Holt, 1991).

A small body of work has attempted to alleviate the situation for SL users by developing machine translation (MT) systems capable of translating texts written in natural languages into various SLs. This field of SLMT is still in its infancy with research into the area dating back approximately ten years. Many of the systems proposed to date are rule-based systems, based on transfer approaches (Grieve-Smith, 1999), interlingual systems (e.g. the Zardoz system, (Veale et al., 1998)), or hybrid models where these approaches are combined (Huenerfauth, 2004, 2005). On a rather smaller scale, corpus-based approaches have also been proposed (Bauer et al., 1999).

Example-Based MT (EBMT) has been around for over 20 years now, from the seminal paper of (Nagao, 1984) to the more recent collection of (Carl & Way, 2003) and beyond. However, as far as we are aware, no previous approaches to SL translation have used such a method. In the medium to long term, our main goal is to develop an EBMT system for the language pair English–Irish Sign Language (ISL), in both directions. However, at this early stage of the project no ISL corpus is available, though one is in the process of being constructed by the Centre for Deaf Studies[1] in Dublin.

In order to demonstrate proof-of-concept of our approach, therefore, we present a system which instead translates between English and Nederlandse Gebarentaal/Sign Language of the Netherlands (NGT). We obtained a corpus of NGT examples from the ECHO project website.[2] As consultants on the ISL corpus-building process, we are aware that the ISL corpus is being constructed using the same annotation process and toolkit as that of the ECHO corpus, so developing an English–NGT EBMT system is a reasonable approximation of the task with which we will eventually be confronted. In initial experiments, we devised a set of sentences for testing the system and used manual analysis to evaluate the results. At this preliminary stage, the results obtained are encouraging.

The remainder of the paper is constructed as follows. In section 2, we describe previous related research in this area. In section 3, we present some of the issues involved in projects of this type, in particular the ECHO project, by showing the internal representation of an NGT object and describing how an EBMT approach may avail of this data. In section 4, we briefly summarize the main ideas behind typical models of EBMT, as well as the particular system used here. Section 5 presents the results obtained by our prototype EBMT system, and discussion of the major findings. Finally, we conclude and present avenues for further research.

[1] http://www.tcd.ie/Deaf_Studies/
[2] http://www.let.kun.nl/sign-lang/echo/

2 Related Work

It is only in the last ten years or so that an interest has been taken in using MT techniques to automate the translation of sign languages. Most of the research that has been carried out has involved the development of a system for the language pair English–American Sign Language (ASL), although there have been a few other language pair models. The most common approaches to date have been rule-based with more SL corpora being created we can reasonably expect corpus-based approaches to become more prevalent in this field mirroring the situation in 'regular' MT. The majority of systems work at translating spoken languages in text format into sign language that is then reproduced as either an avatar of a signing mannequin or a literal orthography (written annotation of the sign language).

Transfer systems have been developed by:

- (Grieve-Smith, 1999), who modelled a system for English–ASL using the limited domain of Albuquerque weather reports;
- (Marshall & Sáfár, 2002), (Sáfár & Marshall, 2002), whose English–ASL system is semantically driven and uses HPSG semantic feature structures and Discourse Representation Structures to represent the internal structure of linguistic objects;
- (Van Zijl & Barker, 2003), who proposed a system for English–South African Sign Language.

In terms of Interlingual approaches:

- (Zhao et al., 2000) developed an English–ASL system that uses synchronised tree adjoining grammars;
- (Veale et al., 1998) developed the Zardoz system for translating English into ISL, ASL and Japanese Sign Language.

In addition, (Huenerfauth, 2004, 2005) has proposed a hybrid multi-path system where English is translated into ASL using a combination of an interlingua, transfer methods and direct methods. This work focuses in particular on models for classifier predicates.

Systems translating from sign language into written oral-language text have also been developed, one such system being that of (Bauer et al., 1999). This is a statistical MT (SMT) system that uses Hidden Markov Models in the recognition of signs before using a translation model and a language model for translation in the usual SMT manner.

3 Sign Language

Despite common misconceptions, sign languages are indigenous, fully accessible languages for Deaf people, with their own unique syntax and grammar. Each country has its own sign language and these languages can vary slightly from region to region just like the dialects of a spoken language. In recent years, more and more national sign languages have begun to be officially recognised in the countries where they are used, as they are the primary means of communication for Deaf people. Regrettably, in many others "provision is not made for deaf people to access public information, or receive vital services such as education and health in their first language" (Ó'Baoill & Matthews, 2000), namely sign language. This is also true for the accessibility of public or private information in the form of written documents. This is an area in which an automated translation of written text could prove invaluable to members of the Deaf community, particularly in areas of low interpreter availability.

3.1 SLMT Issues

The development of an SLMT system requires a number of issues to be taken into consideration. An SLMT system has to deal with some of the problems that models of translation for non-SLs have to handle, such as varying and free syntax, morphological issues (e.g. repetition and pluralisation), and lexical gaps. In addition, models of SLMT should also have the ability to deal with sign language-specific phenomena: non-manual features (NMFs), classifiers, the spatial nature of sign language and its discourse mapping onto the signing space, topic–comment structures, and co-articulation of signs. It should also have an adequate notation system/literal orthography to describe the sign language, as they have no officially recognised written forms.

3.2 Sign Language Corpora

Corpora of sign languages are not widely available and the few that are often contain little or no annotation. Annotation is necessary as the corpora usually take the form of sign language videos owing to the lack of a standardised written form for SLs. This is one way in which SLMT differs from spoken language text based MT. SignWriting[3] may fill this gap as there are SL corpora available in this form. In terms of its suitability as a candidate for use in an EBMT system, SignWriting lacks the explicit linguistic detail necessary for the generation of signs using an avatar. Annotated corpora on the other hand have the potential to carry varying degrees of granularity of linguistic detail, therefore bypassing the need to translate using SingWriting and then deriving such details from the resulting SignWriting symbol. Another issue with SignWriting is that the majority of signers are unfamiliar with it which lowers its appeal for use as final output translation.

By contrast to poorly or unannotated data, the ECHO project is a pilot venture to make fully annotated digitised corpora available on the Internet. The project is based in the Netherlands and contains annotated corpora in NGT, British Sign Language (BSL) and Swedish Sign Language (SSL). These corpora have been annotated using ELAN annotation

[3] http://www.signwriting.org

software.[4] ELAN provides a graphical user interface (Fig. 1) from which corpora can be viewed in video format with their corresponding aligned annotations. These can be seen in the lower half of Fig. 1 where the tiers are named on the left-hand column and the annotations appear horizontally in line with their corresponding tier.

The ECHO corpora have been annotated to include a time-aligned translation in the native spoken language and in English. Other annotation tiers include glosses of the signs articulated by the right and left hand in both spoken languages and various NMF descriptions. An example of some annotations used in one of the NGT corpora can be found in (1). The initial numbers indicate the time span of annotation, the text in brackets shows the name of the annotation tiers and the final text is the annotation itself:

(1) a. `1459490 1461360`
`(Gloss RH English)CONSCIOUS`

b. `1459490 1461360`
`(Gloss RH) BEWUST`

c. `1459490 1461310`
`(Mouth) 'bewussssss'`

3.3 Suitability of ECHO Corpora

Suitably annotated corpora, such as those provided by the ECHO project, are ideal for use in an EBMT approach to SL translation. The provision of an English translation in the form of an annotation tier for each signed sentence along with the other time-aligned annotations allows for easy alignment of corpora on a sentential level as annotations within the time limits of the English translation annotation can then be aligned with that annotation. The presence of time spans for each annotation also aids in the aligning of annotations from each annotation tier to form chunks that can then be aligned with chunks derived from the English tier. Time-aligned annotations are also useful for tackling the issue of co-articulation of signs. The phenomenon of co-articulation in sign languages is analogous to co-articulation in spoken languages where the articulation of a phoneme may be altered relative to its neighbouring phonemes (Jerde et al., 2003). In sign languages phonemes are articulated using the hands. Examples of sign language phonemes include handshape and palm orientation. Co-articulation can occur in fluent signing when the the shape of the hand for one sign is altered relative to the handshape for the subsequent sign. Even if signs are co-articulated in the videos, the annotations for the signs will be separate and either contiguous or overlapping in different tiers, either way they are easy to separate using the time span figures. As it is these annotations that are used in the translation output, the issue of separating co-articulated words is removed automatically.

[4] `http://www.mpi.nl/tools/elan.html`

These annotated corpora also provide a solution to the SL translation issue of NMFs. In sign language meaning is conveyed using several parts of the body in parallel (Huenerfauth, 2005), not solely the hands which is a common misconception. NMFs are sign language units that use parts of the body other than the hands to express semantic information. Some examples of NMFs are eyebrow, cheek or eye movements, mouth patterns, head tilting or upper body and shoulder movements. They are used to express emotion or intensity, but also can be used morphologically and syntactically as markers (Ó'Baoill & Matthews, 2000). The annotations of the ECHO corpora contain explicit NMF detail in varying tiers such as eye aperture and mouth that combine with other tiers to form complete signs and therefore more linguistically complete translations. The example in (2) shows the effect NMFs have on a sign. The *Gloss RH/LH English* is the manual hand sign articulated by the right and left hand, in this case showing that of a hare running. The annotation *n* on the *head* tier indicates a nod of the head. This combined with the furrowing marked in the *brows* tier (signified by *f*), the squinting marked in the *eye aperture* tier (signified by *s*) and the puffing of the cheeks marked in the *cheeks* tier (signified by *p*) shows the intensity of the running that the hare is doing. Without these NMFs the hare would be understood to be running at a normal running pace.

(2)
`(Gloss RH English) running hare`
`(Gloss LH English) running hare`
`(Head) n`
`(Brows) f`
`(Eye Aperture) s`
`(Cheeks) p`

In many cases NMFs are essential for providing the full sense of the sign. The more detail that is contained in the annotation tiers, the better the translation and the more suitable the translations will be for use with a signing avatar. Currently, research is focused on the translation modules of the system and it is for this reason that annotations are produced as final input as opposed to a signing avatar.

4 Example-Based Machine Translation

A prerequisite for EBMT is a set of sentences in one language aligned with their translations in another. Given a new input string, EBMT models use three separate processes in order to derive translations:

1. Searching the source side of the bitext for 'close' matches and their translations;
2. Determining the sub-sentential translation links in those retrieved examples;
3. Recombining relevant parts of the target translation links to derive the translation.

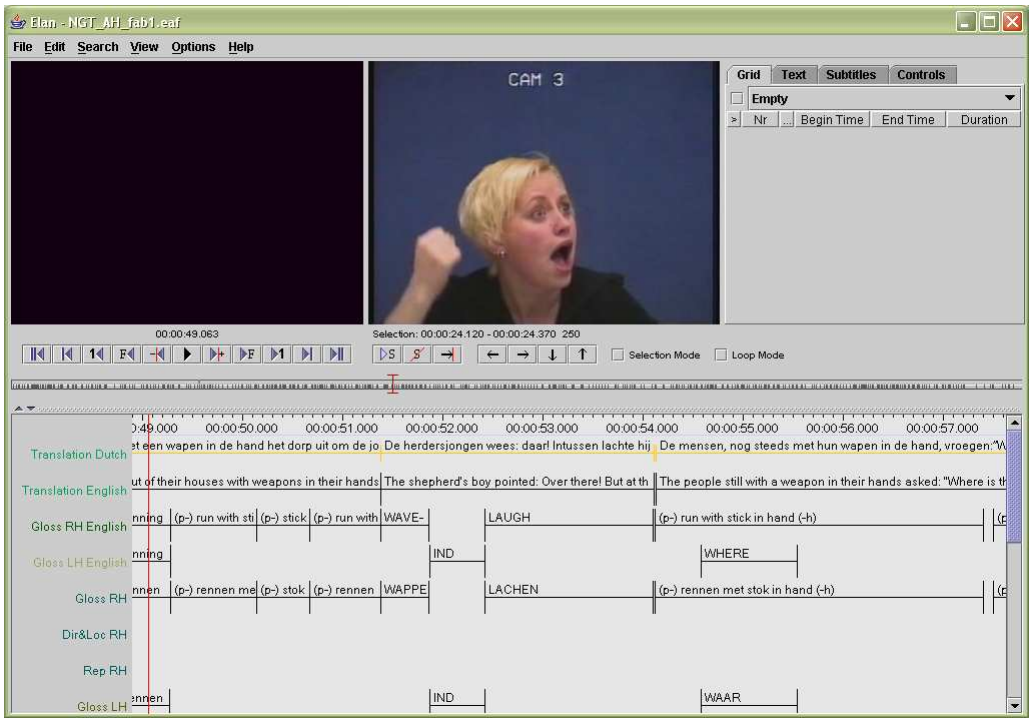


Figure 1: ELAN user interface

Searching for the best matches involves determining a similarity metric based on word occurrences and part-of-speech labels, generalised templates and bilingual dictionaries. The recombination process depends on the nature of the examples used in the first place: from aligning phrase-structure (sub-)trees (Hearne & Way, 2003) or dependency trees (Watanabe et al., 2003), to the use of placeables (Brown, 1999) as indicators of chunk boundaries.

Another method—and the one used in the EBMT system in our experiments—is to use a set of closed-class words to segment aligned source and target sentences and to derive an additional set of lexical and phrasal resources. (Way & Gough, 2003), (Gough & Way, 2004a) and (Gough & Way, 2004b) base their work on the 'Marker Hypothesis' (Green, 1979), a universal psycholinguistic constraint which posits that languages are 'marked' for syntactic structure at surface level by a closed set of specific lexemes and morphemes. In a pre-processing stage, (Gough & Way, 2004b) use 7 sets of marker words for English and French (e.g. determiners, quantifiers, conjunctions etc.), which together with cognate matches and mutual information scores are used to derive three new data sources: sets of marker chunks, generalised templates and a lexicon.

In order to describe this in more detail, we revisit an example from (Gough & Way, 2004a), namely:

(3) each layer has a layer number $\Longrightarrow$chaque couche a un nombre de la couche

From the sentence pair in (3), the strings in (4) are generated, where marker words are automatically tagged with their marker categories:

(4) <QUANT> each layer has <DET> a layer number $\Longrightarrow$<QUANT> chaque couche a <DET> un nombre <PREP> de la couche

Taking into account marker tag information (label, and relative sentence position), and lexical similarity, the marker chunks in (5) are automatically generated from the marker-tagged strings in (4):

(5) a. <QUANT> each layer has: <QUANT> chaque couche a

b. <DET> a layer number: <DET> un nombre de la couche

(5b) shows that *n:m* alignments are possible (the two French marker chunks *un nombre* and *de la couche* are absorbed into one following the lexical similarities between *layer* and *couche* and *number* and *nombre*, respectively) given the sub-sentential alignment algorithm of (Gough & Way, 2004b).

By generalising over the marker lexicon, a set of marker templates is produced by replacing the marker word by its relevant tag. From the examples in (5), the generalised templates in (6) are derived:

(6) a. <QUANT> layer has: <QUANT> couche a

b. <DET> layer number: <DET> nombre de la couche

These templates increase the robustness of the system and make the matching process more flexible. Now any marker word can be inserted after the relevant tag if it appears with its translation in the lexicon, so that (say) *the layer number* can now be handled by the generalised template in (6b) and inserting a (or all) translation(s) for *the* in the system's lexicon.

However, since SLs display a considerably reduced number of marker words, an alternative method is used for segmenting the SL texts. This is discussed in section 5.1.

5 Experiments and Results

Our corpus consists of 561 sentences with an average sentence length of 7.89 words, (min. 1 word, max. 53 words). The sign language side of the corpus consists of annotations that describe the signs used in the video. As the English translation annotation tier and the other annotation tiers are time-aligned, sentence alignments were easy to extract automatically.

5.1 Segmentation and Alignment

The Marker Hypothesis described in section 4 was used to segment the English sentences according to the same set of closed-class words used in (Way & Gough, 2003; Gough & Way, 2004a/b). This results in segments that start with a closed class word and usually encapsulate a concept or an attribute of a concept being described, for example the concept of darkness as shown in (7) where the angle-bracketed text refers to the marker tag representing the pronoun *it*.

(7) <PRON> it was almost dark

On the sign language side it was necessary to adopt a different approach as a result of the sparseness of the English closed class item markers in the SL text. This is normal in SLs, where often closed class items are not signed, as is the case with many determiners, or are subsumed into the sign for the neighbouring noun as is sometimes the case with prepositions. Initially experiments were performed on different divisions of the SL annotations. The NGT gloss tier was segmented based on the time spans of its annotations. The remaining annotations on other tiers were then grouped with the NGT gloss tier annotations within the appropriate matching time frame. It was found that these segmentations divided the SL corpus into concept chunks. Upon examination these concept chunks were found to be similar in form to the chunks that were formed using the the Marker Hypothesis on the English text and suitable for forming alignments, thereby providing a viable option for chunking the SL side of the corpus. The following example shows segments from both data sets and their usability for chunk alignment. (8) shows the results of the different chunking process on both sentences, (8a) being taken from the English chunking process and (8b) from the SL chunking process. (9) shows specific chunks that can be successfully aligned following the chunking process, (9a) being taken from the English chunked text and (9b) from the SL chunked text. Angled brackets contain the markers, round bracketed text names the tier, the remaining text is the annotation content of that tier and each tier is separated by a colon.

(8) a. <DET> the hare takes off <PREP> in a flash.

b. <CHUNK> (Gloss RH English) (p-) running hare :
(Mouth) closed-ao :
(Mouth SE) /AIRSTREAM/ :
(Cheeks) p :
(Gloss LH English) (p-) running hare :
(Gloss RH) (p-) rennen haas :
(Gloss LH) (p-) rennen haas :
<CHUNK> (Gloss RH English) FLASH-BY :
(Gloss RH) VOORBIJ-SCHIETEN :
(Mouth) closed, forward :
(Mouth SE) /PURSED/ :
(Eye gaze) rh

(9) a. <DET> the hare takes off

b. <CHUNK> (Gloss RH English) (p-) running hare :
(Mouth) closed-ao :
(Mouth SE) /AIRSTREAM/ :
(Cheeks) p :
(Gloss LH English) (p-) running hare :
(Gloss RH) (p-) rennen haas :
(Gloss LH) (p-) rennen haas

The main concept expressed in (9a) and (9b) is the running of the hare. The English chunk encapsulates this concept with the words *the hare takes off*. This same concept is expressed in the SL chunk in the combination of annotations. The 'Gloss RH English' and 'Gloss LH English' show the running of the hare and the additional semantic information of the effort involved in *takes off* as opposed to running at ease is expressed in the NMF tiers with the indication of puffing of the cheeks (*p* in the *cheeks* tier) and the closed mouth with breath being exhaled (closed-ao and */AIRSTREAM/* in the *mouth* and *mouth SE* tiers respectively). Despite the different methods used, they are successful in forming potentially alignable chunks.

5.2 Evaluation

As there is not formally recognised writing system for SLs and as annotation maybe be considered subjective to the author to a degree, it is uncertain that consistent gold standard sentences for evaluation purposes could be produced, (Huenerfauth, 2005).To better evaluate the performance of the system we decided to formulate our own test set. Test sets were manually constructed in four groups of ten sentences. The groups are as follows: (i) full sentences taken directly from the corpus, (ii) grammatical sentences formed by combining chunks taken from different parts of the corpus, (iii) sentences made of combined chunks from the corpus and chunks not in the corpus, (iv) sentences of words present in the corpus but not forming alignable chunks and of words not in the corpus. These test sets were constructed with a view to making the most of the limited data we had.

Each sentence was run through the translator and the resulting output manually evaluated based on the alignments of the corpus. The results are evaluated and divided into four categories depending on their quality: good, fair, poor and bad. Below is an explanation of the metric employed with examples using the sentence *it was almost dark*.

Good: contains the correct grammatical information (i.e. adverbs, prepositions that provide detail about the concept) and content (i.e. head noun or verb) information.

(10) `Gloss RH English`: DARK
`Gloss LH English`: DARK
`Mouth`: 'donker'
`Brows`: f
`Eye Aperture`: s.

Fair: contains the correct content information but is missing some of the grammatical detail.

(11) `Gloss RH English`: DARK
`Gloss LH English`: DARK
`Mouth`: 'donker'
(no brow or eye movement shown, alters meaning of phrase)

Poor: contains only some correct content information and either lacks grammatical detail or contains the incorrect grammatical detail.

(12) `Gloss RH English`: DARK
`Eye Aperture`: c.

Bad: contains an entirely incorrect translation.

(13) `Gloss RH English`: WHAT

5.3 Discussion

The manual evaluations performed on the test results show that the system is competent in translating sentences that occur fully intact in the corpus as would be expected from any EBMT system. These results also show that more than half the translations of sentences made up of chunks from the corpus provide reasonable-to-good translations. The system is able to segment the input and find adequate matches in the corpus to produce coherent translations for 60% of the sentences tested from (ii). This is also the case for almost a third of test sentences where data consists of combined corpus and external chunks (sentence type (iii)). The more data that is not present in the training set that is introduced in the test set the lower the rating, as can be seen from the results of type (iii) and (iv) where an increased amount of material not present in the corpus is tested. In these cases, translations are still produced but are of poor to bad quality. For sentence type (iii), only a third of the sentences were of fair quality. For sentence type (iv), more than two thirds of the translations were considered bad and the remainder poor. As with EBMT systems in general, were the corpus to be larger and to contain a richer word-level dictionary, the system would be able to produce closer, if not exact, matches for an increased number of chunks and words, thus improving the ratings. Currently the approach to aligning segments for the bilingual corpus is in its infancy. Further research and development in this area will also improve the quality of alignments and thus the translations.

6 Conclusions and Future Work

In light of the absence of documentation available to the Deaf in their first language, in this paper we aimed to test the applicability of EBMT techniques to SLMT with a view to developing a prototype MT system for SLs. Corpora of English-NGT data were obtained from the ECHO project website and their annotations were extracted. These annotations were then used as a written representation of NGT from which example alignments could be deduced following the segmentation phase. We found the Marker Hypothesis a sufficient approach for segmenting the English data but found it necessary to employ a time frame based technique to segment the SL annotations. We found that employing these segmentation approaches provided us with chunks of a similar format from which adequate alignments could be constructed for use in the translation process. Despite the small corpus and dictionary size, initial results are promising and indicate further development is plausible and worthwhile. Further research into the chunking and aligning processes, combined with an enhanced corpus and dictionary, will improve the quality of results and provide a clearer picture of the success of an EBMT system for sign languages. This prototype system has allowed us to identify some areas which require particular focus.

Subsequent to the work carried out to date, we intend to continue developing the system using the current language pair English–NGT. Initial plans include enhancing the annotation alignments by incor-

porating non-time-aligned annotations into the data set and using the information in the complete annotation set to determine closer matches with the English data and thus improve alignment at all levels. This should also allow for the automatic creation of generalised templates which would further aid the translation process. A large part of the work on this system will involve the improvement of the word-level dictionary. If possible, this task will be automated and the word alignments extracted from the corpora as opposed to an external source. We also intend to undertake increased manual evaluations of the translation results to determine specific problem areas that need work. Once a successful system has been produced for this language pair we intend to expand the system to translate from Dutch to NGT and to apply the system to other language pairs for which we have similar data, i.e. English-British Sign Language.

The ISL corpus under construction at the Centre for Deaf Studies (Dublin) will be much larger than the NGT corpus we are currently using and will contain richer annotations. The ISL corpus consists of roughly 20 hours of video data in comparison to the 40 minutes of the current NGT corpus we are using. This will allow for the creation of larger test–training sets, which should improve the results of the system on the basis that the more data a system has, the more possible matches can be found for input sentences. The richer annotations incorporated into the ISL corpus, including phonological information such as hand shape and palm orientation, will provide a more detailed translated output from which real sign language may be synthesised using an avatar. This is the ultimate goal for our work, to develop a fully automated text to sign language translation system where the signers can enter English written data and have it translated for viewing in their first language.

Acknowledgements

We would like to thank the anonymous reviewers whose valuable comments helped improve the quality of the paper. This research was partially funded by an IRCSET[5] PhD scholarship.

References

Britta Bauer, Sonja Nießen and Hermann Heinz. 1999. Towards an Automatic Sign Language Translation System. In *Proceedings of the International Workshop on Physicality and Tangibility in Interaction: Towards New Paradigms for Interaction Beyond the Desktop*, Siena, Italy.

Ralf Brown. 1999. Adding Linguistic Knowledge to a Lexical Example-based Translation System. In *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-99)*, Chester, England, pp.22–32.

Michael Carl and Andy Way (eds). 2003. *Recent Advances in Example-Based Machine Translation*. Kluwer, Dordrecht, The Netherlands.

Nano Gough and Andy Way. 2004a. Example-Based Controlled Translation. In *Proceedings of the Ninth EAMT Workshop*, Valetta, Malta, pp.73–81.

Nano Gough and Andy Way. 2004b. Robust Large-Scale EBMT with Marker-Based Segmentation. In *Proceedings of the Tenth Conference on Theoretical and Methodological Issues in Machine Translation (TMI-04)*, Baltimore, MD., pp.95–104.

Thomas Green. 1979. The Necessity of Syntax Markers. Two experiments with artificial languages. *Journal of Verbal Learning and Behavior* **18**:481–496.

Angus B. Grieve-Smith. 1999. English to American Sign Language Machine Translation of Weather Reports. In D. Nordquist (ed.) *Proceedings of the Second High Desert Student Conference in Linguistics (HDSL2)*, Albuquerque, NM., pp.23–30.

Mary Hearne and Andy Way. 2003. Seeing the Wood for the Trees: Data-Oriented Translation. In *MT Summit IX*, New Orleans, LA., pp.165–172.

Judith Holt. 1991. *Demographic, Stanford Achievement Test - 8th Edition for Deaf and Hard of Hearing Students: Reading Comprehension Subgroup Results.*

Matt Huenerfauth. 2004. Spatial and Planning Models of ASL Classifier Predicates for Machine Translation. In *The 10th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2004)* Baltimore, MD., pp.65–74

Matt Huenerfauth. 2005. American Sign Language Generation: Multimodal NLG with Multiple Linguistic Channels. In *Proceedings of the ACL Student Research Workshop (ACL 2005)* Ann Arbor, MI., pp.37–42

Thomas E. Jerde, John F. Soechting and Martha Flanders. 2003. Coarticulation in Fluent Fingerspelling. *Journal of Neuroscience* **23(6)**:2383–2393.

Ian Marshall and Éva Sáfár. 2002. Sign Language Generation using HPSG. In *Proceedings of the 9th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-02)*, Keihanna, Japan, pp.105–114.

Makoto Nagao. 1984. A Framework of a Mechanical Translation between Japanese and English by Analogy Principle. In A. Elithorn and R. Banerji (eds.) *Artificial and Human Intelligence*, North-Holland, Amsterdam, The Netherlands, pp.173–180.

Dónall Ó'Baoill and Patrick A. Matthews. 2000. *The Irish Deaf Community (Volume 2): The Structure*

[5] http://www.ircset.ie

of Irish Sign Language. The Linguistics Institute of Ireland, Dublin, Ireland.

Éva Sáfár and Ian Marshall. 2002. The Architecture of an English-Text-to-Sign-Languages Translation System. In *Proceedings of Recent Advances in Natural Language Processing (RANLP-01)*, Tzigov Chark, Bulgaria, pp.223–228.

Lynette Van Zijl and Dean Barker. 2003. South African Sign Language Machine Translation System. In *Proceedings of the Second International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa (ACM SIGGRAPH)*, Cape Town, South Africa, pp.49–52.

Tony Veale, Alan Conway, and Bróna Collins. 2000. The Challenges of Cross-Modal Translation: English to Sign Language Translation in the Zardoz System. *Machine Translation* **13**(1):81–106.

Hideo Watanabe, Sadao Kurohashi and Eiji Aramaki. 2003. Finding Translation Patterns from Paired Source and Target Dependency Structures. In M. Carl & A. Way (eds.) *Recent Advances in Example-Based Machine Translation*, Kluwer Academic Publishers, Dordrecht, The Netherlands, pp.397–420.

Andy Way and Nano Gough. 2003. Developing and Validating an EBMT System using the World Wide Web. *Computational Linguistics* **29**(3):421–457.

Liwei Zhao, Karin Kipper, William Schuler, Christian Vogler, Norman Badler, and Martha Palmer. 2000. A Machine Translation System from English to American Sign Language. In *Envisioning Machine Translation in the Information Future: Proceedings of the Fourth Conference of the Association for Machine Translation (AMTA-00)*, Cuernavaca, Mexico, pp.293–300.

A Machine Learning Approach to Hypotheses Selection of Greedy Decoding for SMT

Michael Paul and Eiichiro Sumita and Seiichi Yamamoto
ATR Spoken Language Communication Research Laboratories
2-2-2 Hikaridai, Kansai Science City, Kyoto, 619-0288 Japan
{michael.paul, eiichiro.sumita, seiichi.yamamoto}@atr.jp

Abstract

This paper proposes a method for integrating example-based and rule-based machine translation systems with statistical methods. It extends a greedy decoder for statistical machine translation (SMT), which searches for an optimal translation by using SMT models starting from a decoder seed, i.e., the source language input paired with an initial translation hypothesis. In order to reduce *local optima* problems inherent in the search, the outputs generated by *multiple translation engines*, such as rule-based (RBMT) and example-based (EBMT) systems, are utilized as the initial translation hypotheses. This method outperforms conventional greedy decoding approaches using initial translation hypotheses based on translation examples retrieved from a parallel text corpus. However, the decoding of multiple initial translation hypotheses is computationally expensive. This paper proposes a method to select a single initial translation hypothesis *before decoding* based on a machine learning approach that judges the appropriateness of multiple initial translation hypotheses and selects the most confident *one* for decoding. Our approach is evaluated for the translation of dialogues in the travel domain, and the results show that it drastically reduces computational costs without a loss in translation quality.

1 Introduction

This paper proposes a method for integrating example-based and rule-based machine translation systems with statistical methods. It extends a greedy decoder for statistical machine translation (cf. Section 2), which searches for an optimal translation by using SMT models starting from a decoder seed, i.e., the source language input paired with an initial translation hypothesis. Despite a high performance on average, the greedy decoding approach can often produce translations with severe errors.

A major problem of the greedy decoding approach is that the translation output depends on the initial translation hypothesis to start the search, which may lead to a local optimum translation but not to the global optimum translation. Therefore, the selection of the starting point is crucial to avoid local optima in the search.

Previous methods addressed this problem by creating an initial translation hypothesis based on translation examples obtained from a parallel text corpus (Marcu, 2001), (Watanabe and Sumita, 2003) or by using diverse starting points generated by multiple translation engines (Paul et al., 2004). Combining multiple MT systems has the advantage of exploiting the strengths of each MT engine. Quite different initial translation hypotheses are produced due to particular output characteristics of each MT engine. Therefore, larger parts of the search space can be explored while avoiding local optima problems of the search algorithm. This method outperforms conventional greedy decoding approaches using initial translation hypotheses based on translation examples retrieved from a parallel text corpus. However, the sequential decoding of multiple decoder seeds is *computationally expensive*.

In this paper, we propose a method to select a single initial translation hypothesis *before decoding* in order to reduce computational costs. A machine learning approach (*decision tree*), that judges the appropriateness of a given initial translation hypothesis, is combined with a ranking method based on statistical model scores in order to select the most confident initial translation hypothesis for decoding. Section 3 extends the greedy decoding approach as follows: (1) the initial translation hypotheses are produced by multiple MT engines, (2) a machine learning approach using a decision tree classifier is proposed to identify and eliminate hypotheses that might be wrongly modified by the greedy decoder thus leading to translations of lower quality, and (3) information about the classification

result and statistical model scores of the remaining initial translation hypotheses are combined in order to select the best suited hypothesis.

The effects of the proposed method are demonstrated in Section 4 for the Japanese-to-English translation of dialogues in the travel domain.

2 Greedy Decoding for SMT

In this section, we explain the outline of SMT and greedy decoding in short.

2.1 Statistical Machine Translation

Statistical machine translation formulates the problem of translating a sentence from a source language S into a target language T as the maximization problem:

$$\text{argmax}_T \; p(S|T) * p(T), \qquad (1)$$

where $p(S|T)$ is called a *translation model* (TM), representing the generation probability from T into S, and $p(T)$ is called a *language model* (LM), which represents the likelihood of the target language (Brown et al., 1993). During the translation process (*decoding*), a statistical score based on TM and LM is assigned to each translation. In this paper, we call this score **TM·LM**. The translation with the highest TM·LM score is selected as the output.

We used the *IBM-4* translation model (Brown et al., 1993) in the experiments in Section 4, which consists of probabilities for word translations (*lexicon model*), the number of source words produced by a target word (*fertility model*), word insertions (*generation model*), and word order changes (*distortion model*). LM is based on the frequency of consecutive word sequences (*n-gram*). The TM and LM probabilities are trained automatically from a parallel text corpus.

Figure 1 gives an example for the process of transferring a Japanese source sentence into an English target sentence and illustrates which translation knowledge is captured by the respective statistical models mentioned above.

2.2 Greedy Decoding

Various decoding algorithms have been proposed, including *stack-based* (Wang and Waibel, 1997), *beam search* (Tillmann and Ney, 2000), and *greedy decoding* (Germann et al., 2001). This paper concentrates on the greedy decoding approach described in details in Section 2.2.1. The local optima problem of this approach is illustrated in Section 2.2.2.

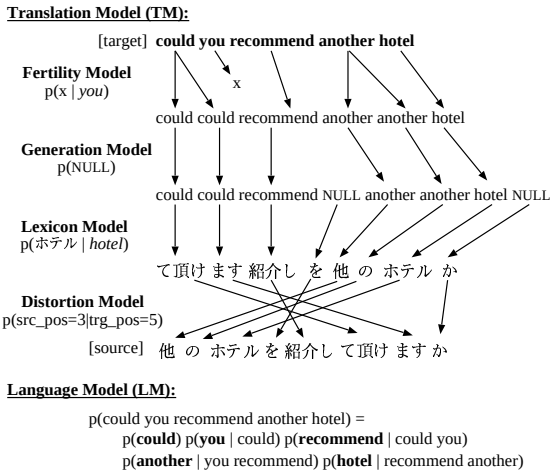


Figure 1: Statistical Models

2.2.1 Algorithm

Figure 2 illustrates the decoding algorithm, which is described in detail in (Germann et al., 2001), and summarizes the terminology used throughout this paper.

The input of the decoder (*decoder seed*) consists of the input, i.e., the source language sentence, paired with an initial translation hypothesis, whereby the initial translation hypothesis is formed by a word-by-word translation of the source language sentence. The following steps attempt to improve the quality of the translation hypothesis by greedily exploring alternative translations starting from the initial translation hypothesis. The algorithm modifies the hypothesis iteratively using a set of word operations such as *inserting*, *deleting*, *joining*, and *swapping*. After each modification, the statistical scores of the previous and modified input-hypothesis pairs are calculated. If the modified pair has a higher TM·LM score, it is used in the next iteration. Otherwise, the modified hypothesis is ignored and the search is continued using the previous input-hypothesis pair. The decoding algorithm stops if no further improvement can be achieved by any operation and outputs the hypothesis with the *highest statistical score*.

If multiple initial translation hypotheses are used for a given source language input, the decoder is applied to each of the initial translation hypotheses, resulting in multiple translation candidates, and the candidate with the highest statistical score is selected as the translation.

2.2.2 Local Optima Problem of Greedy Decoding

A major problem of the greedy decoding approach is that the translation output depends

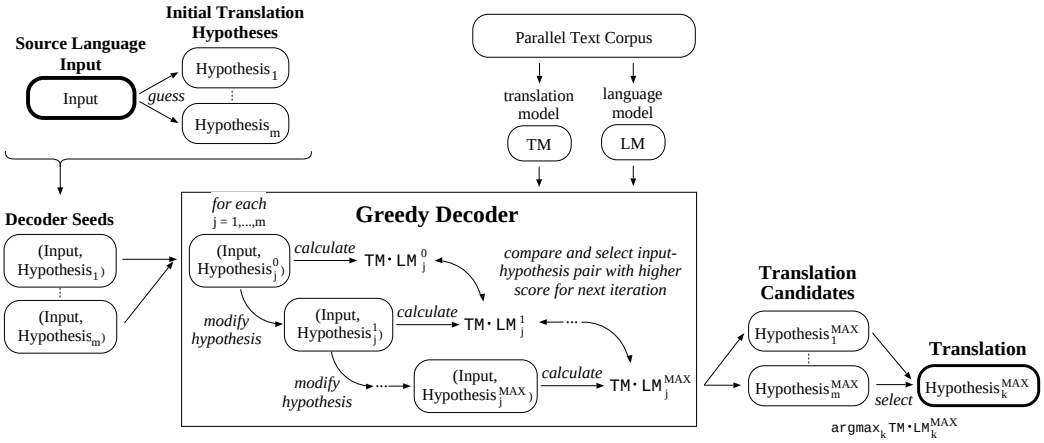


Figure 2: Greedy Decoding

on the initial translation hypothesis to start the search, which may lead to a local optimum translation but not to the global optimum translation.

This problem is illustrated in Figure 3. Given the decoder seed $seed_1$, the greedy decoder modifies the initial translation hypothesis based on its statistical models (along the dotted line) as long as the TM·LM score increases and finally outputs the translation candidate with maximal score ($cand_1$). Similarly, the local optimum translation candidate $cand_2$ is obtained when $seed_2$ is used as the decoder seed. However, using $seed_3$ as the starting point, the decoder finds the global optimum translation candidate $cand_3$ that cannot be found by using the other seeds.

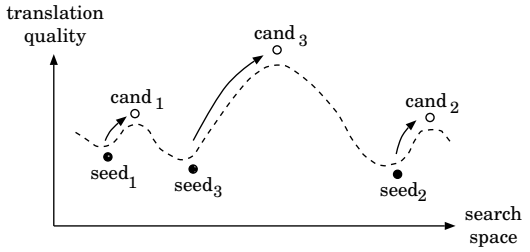


Figure 3: Local Optima Problem of the Greedy Search

2.3 Greedy Decoding Using Translation-Engine-Based Hypotheses

To solve the local optima problem, (Paul et al., 2004) proposed to use diverse starting points generated by multiple translation engines. Combining multiple MT systems has the advantage of exploiting the strengths of each MT engine. Quite different initial translation hypotheses are obtained, because they are produced by independently developed translation engines that use different dictionaries, grammars, and translation rules. Therefore, larger parts of the search space can be explored, increasing the chance to catch the global optimum.

The greedy decoder is applied sequentially to each of the initial translation hypotheses, where the best translation is selected according to an edit-distance-based rescoring method that compensates the statistical scores of each generated translation candidate by information on how much the initial translation hypothesis is modified during decoding.

This method outperforms conventional greedy decoding approaches solely based on statistical models. However, a shortcoming of this approach is that the decoder has to be applied to *all* initial translation hypotheses. Therefore, high computational costs are involved to identify the best translation.

3 Machine Learning Approach for Hypotheses Selection

The method proposed in this paper is based on the greedy decoding approach described in Section 2.3. In order to reduce computational costs, our approach selects a single hypothesis out of the set of initial translation hypotheses obtained from multiple MT engines *before* the greedy decoder is applied to generate the translation output.

The initial translation hypotheses are produced by multiple MT engines as described in Section 3.1.

In order to select the most appropriate initial translation hypothesis for decoding, we propose a machine learning approach using a decision tree classifier to identify and eliminate hypotheses that might be wrongly modified by the

greedy decoder thus leading to translations of lower quality (cf. Section 3.2).

Finally, information about the classification result and statistical model scores of the remaining initial translation hypotheses are combined in order to select the best suited hypothesis as described in Section 3.3.

3.1 Translation-Engine-based Hypotheses

For our experiments, we used the five MT engines listed in Table 1[1].

Table 1: Utilized MT Engines

EBMT	D3 (Sumita, 2001) HPAT (Imamura, 2002)
RBMT	ATLAS (Fujitsu, 2003) LOGOVISTA (LogoVista, 2001) THEHONYAKU (Toshiba, 2003)

Two of them (MT_{1-2}) are *example-based MT* (EBMT) systems that are trained on the same training set as the greedy decoder. The remaining three (MT_{3-5}) are *off-the-shelf rule-based MT* (RBMT) systems that are based on lexicons, grammars, and translation rules. Examples of MT-based hypotheses are given in Table 2.

Table 2: Translation-Engine-Based Hypotheses

(source language input)
ハイアットリージェンシーホテルをお願いします
シングルルームに泊まりたいのですが
(→ *i would prefer the hyatt regency please and if possible i want a single room*)

(initial translation hypothesis)
MT_1: i 'm asked do i want to stay to room single room
MT_2: i 'll send a hyatt i 'd like to stay in a single room
MT_3: i want to stay at the single room which asks you for the hyatt regency hotel
MT_4: i want to stay at a single room in which it asks for the hyatt regency hotel
MT_5: i want to stay at the single room which you may ask for hyatt regency hotel with

The outputs of each MT engine show large variations, because they are produced by independently developed translation engines that use different translation knowledge resources.

3.2 Decision Tree Classifier

We use a machine learning approach in order to learn an automatic decision tree classifier (Rulequest, 2004) that distinguishes between initial translation hypotheses being decoded into translations of low vs. high quality.

[1]The MT engines are listed alphabetically, where the order is unrelated to the indexing scheme (MT_i) used for the examples and the discussion of the evaluation results given in this paper.

The decision tree classifier is trained on monolingual as well as bilingual features obtained for pairs of source language input sentences and MT engine outputs. The features were selected in order to cover inter-hypotheses characteristics as well as general features for the identification of appropriate initial translation hypotheses. The *inter-hypotheses features* consist of the following:

- **Similarity features** between initial translation hypotheses produced by different MT engines.
  - the number of identical initial translation hypotheses
  - the *average edit-distance* between the given hypotheses and those of other MT engines, whereby the edit-distance is defined as the sum of the costs of *insertion*, *deletion*, and *substitution* operations required to map one word sequence into the other (Wagner, 1974).
  - differences in the length of a given initial translation hypothesis toward the shortest/longest initial translation hypothesis.

Moreover, we added also *statistical features* and *syntactic/semantic features* for the experiments described in this paper, some of which were used in previous research on the automatic evaluation of machine translation output (Corston-Oliver et al., 2001).

- **Perplexity** of the source language input and the initial translation hypothesis calculated on the basis of trigram language models.
- **Translation model and language model scores** of the input-hypothesis pairs.
- **Dictionary features** including the number of OOV (out-of-vocabulary) words and the number of target words in the initial translation hypothesis that are possible translations of source words.
- **Syntactic features** that are extracted from the syntactic structure of the source language input and the initial translation hypotheses, respectively. These can be subcategorized as follows.

- *sentence length*
- *sentence type*
- *sentence parse* (success of parsing, number of nodes in the parse-tree, number/length of pre/post-modifiers of noun phrases, number of coordinated constituents, coordination balance, i.e., the maximal length difference in coordinated constituents)
- *size of constituents*
- *density features*, i.e., ratio of function words to content words

- **Semantic features** of content words that are extracted from a thesaurus (Ohno and Hamanishi, 1984).

During the *learning phase*, all MT engines listed in Table 1 are used to translate parts of the training corpus and to extract the above mentioned features automatically. Next, the greedy decoder is applied to each initial translation hypothesis, and the obtained results are evaluate automatically using the WER metrics introduced in Section 4.1.2. Based on this evaluation, each input-hypothesis pair is assigned to one of the following two classes:

$$class = \begin{cases} OK & \text{, if } WER(\textit{decoder output}) \\ & \quad < WER(\textit{initial translation hypothesis}) \\ NG & \text{, otherwise} \end{cases}$$

During the *application phase*, the obtained decision tree classifier is applied to each input-hypothesis pair. All initial translation hypotheses classified as *NG* are removed from the hypothesis set. In addition to the classification result, a *confidence score*, i.e., the percentage of training samples classified correctly using the same decision tree path, is assigned to each input-hypothesis pair.

3.3 Selection Algorithm

Statistical model scores are in general good indicators of translation quality and can be used to compare translation hypotheses directly. *The higher the statistical model score, the higher the translation quality is supposed to be.* However, the greedy decoding approach can often produce translations with severe errors. This occurs partly because the decoder might modify hypotheses wrongly resulting in translations of lower quality with higher statistical scores.

On the other hand, the decision tree classifier provides us with information about how reliable the decision is, i.e., *the higher the confidence score* derived from the classification result, *the more likely it is that a good starting point is found.* However, it is not possible to compare directly two hypotheses on the basis which one is more reliable than the other one, because the decision tree classifier is applied independently.

In order to select the most appropriate initial translation hypothesis classified as *OK*, we propose to use both types of information by combining the confidence score derived from the decision tree with the statistical model scores of the input-hypothesis pair (I, H) as follows:

$$\textbf{CONF·TM·LM}(\text{I,H}) = \frac{2*\text{conf}(I,H)*\text{logP}(TM \cdot LM)}{\text{conf}(I,H)+\text{logP}(TM \cdot LM)},$$

where $conf(I, H)$ is the confidence score derived from the classification result and *logP(TM·LM)* denotes the positive log-probabilities of the statistical model score for the given input-hypothesis pair (I, H).

The input-hypothesis pair with the highest CONF·TM·LM score is selected for decoding.

4 Evaluation

Section 4.1 describes the experimental setting. In order to train the translation[2] and language[3] models used for decoding, we utilize two corpora from the *travel* domain. The proposed method is evaluated by using an automatic evaluation metrics and a human assessment of *translation accuracy.* The baseline performance of the greedy decoder using multiple translation-engine-based hypotheses is given in Section 4.2. The effects of the hypotheses selection method proposed in this paper are summarized in Section 4.3 and the obtained results are discussed in Section 4.4.

4.1 Experimental Setting

In this section, we describe the corpora and evaluation metrics.

4.1.1 Corpora

The evaluation of our approach is carried out using two Japanese(J)-English(E) parallel corpora of the *travel* domain.

[2]The translation models are trained using the GIZA++ toolkit, http://www.fjoch.com

[3]The language models are trained using the CMU-Cambridge Statistical Language Modeling Toolkit v2, http://mi.eng.cam.ac.uk/∼prc14/toolkit.html

- *Basic Travel Expression Corpus* (BTEC)
  The BTEC corpus is a large collection of sentences[4] that bilingual travel experts consider useful for people going to or coming from countries with different languages. The BTEC sentences are not transcriptions of actual interactions, but were written by experts (Takezawa et al., 2002).
- *Machine Aided Dialogue Corpus* (MAD)
  The MAD corpus is a collection of dialogues between a native speaker of Japanese and a native speaker of English that is mediated by a speech-to-speech translation system (Kikui et al., 2003).

The statistics of the corpora are given in Table 3, where *word token* refers to the number of words in the corpus and *word type* refers to the vocabulary size. Since the MAD corpus consists of dialogues, it contains more complex and compound sentences as well as filled pauses, resulting in longer sentences that are more difficult to translate.

Table 3: Corpus Statistics

corpus	sentence count	lang uage	word tokens	word types	words per sentence
BTEC	162,318	J	1,114,186	18,781	6.9
		E	952,300	12,404	5.9
MAD	4,894	J	62,529	2,607	10.0
		E	57,500	2,158	10.3

The BTEC corpus was used for the acquisition of translation knowledge (*training set*) and the MAD corpus was used for the training of the decision tree classifier. In addition, we used 502 sentences from the MAD corpus reserved for evaluation purposes as the test set.

4.1.2 Evaluation Metrics

For the evaluation, we used the following automatic scoring measure and human assessment.

- *Word Error Rate* (Su et al., 1992) (WER), which penalizes edit operations against reference translations..
- *Translation Accuracy* (Sumita et al., 1999) (ABC): subjective evaluation ranks ranging from A to D (A: perfect, B: fair, C: acceptable and D: nonsense), judged by a native speaker. Hereafter, we use the total count of translations ranked A, B, or C as the ABC score.

[4]Parts of the BTEC corpus were used in the International Workshop of Spoken Language Translation (http://www.slt.atr.jp/IWSLT2004/) and will be made publicly available through GSK (http://www.gsk.or.jp).

In contrast to WER, higher ABC scores indicate better translations. For the automatic scoring measure we utilized up to 16 human reference translations.

4.2 Translation-Engine-based Hypotheses

Table 4 summarizes the translation quality of the MT engines used to create the initial translation hypotheses.

Table 4: Utilized MT Engines

initial translation hypotheses		evaluation WER (%)	ABC (%)
EBMT	MT_1	49.6	60.3
	MT_2	52.0	66.3
RBMT	MT_3	69.6	54.5
	MT_4	69.4	59.3
	MT_5	71.4	54.1

Table 5 summarizes the translation quality of the greedy decoder using the combination of all MT engine outputs as the initial translation hypotheses.

Table 5: Greedy Decoder Output

initial translation hypotheses	evaluation WER (%)	ABC (%)
EBMT+RBMT (MT_{1-5})	45.8	67.7

The results demonstrate experimentally the effectiveness of using multiple translation-engine-based hypotheses for decoding. The greedy decoding approach (EBMT+RBMT) outperforms all MT engines used to create the initial hypotheses, gaining 3.8% in WER and 1.4% in ABC toward the best MT engine.

4.3 Hypotheses Selection Method

The translation of the MAD corpus by all MT engines listed in Table 1, resulted in 24,470 input-hypothesis pairs from which the feature sets described in Section 3.2 were extracted automatically. Based on this data set, a decision tree classifier was learned and its performance was evaluated as described in Section 4.3.1.

Next, the decision tree classifier was used to filter-out inappropriate initial translation hypotheses and the performance of the proposed selection method was evaluated as described in Section 4.3.2.

4.3.1 Performance of Decision Tree Classifier

Table 6 gives the percentage of sentences classified correctly ($actual = predicted$) and the total amount of classification errors for the training and test sentences, respectively.

Table 6: Decision Tree Classifier

(training corpus)

	actual		
predicted	OK	NG	total
OK	53.5	21.5	75.0
NG	6.9	18.1	25.0
total	60.4	39.6	

(test corpus)

	actual		
predicted	OK	NG	total
OK	66.5	14.5	81.0
NG	14.6	4.4	19.0
total	81.1	18.9	

In total, 72.6%/70.9% of the training/test set were classified correctly, where 21.5%/14.5% of the sentences were accepted falsely. However, 6.9%/14.6% of good initial translation hypotheses were ignored resulting in a total error of 18.4% for the training set and 29.1% for the test set.

4.3.2 Selection of Initial Translation Hypothesis

In order to investigate the effects of applying the decision tree classifier to the test sentences, we evaluated two different selection methods: the hypothesis with (1) the highest statistical score (*TM·LM*), and (2) the highest CONF·TM·LM score is selected as the initial translation hypothesis to be used for decoding.

Table 7: Hypothesis Selection

selection method	evaluation	
	WER (%)	ABC (%)
TM·LM	53.2	61.1
CONF·TM·LM	48.1	67.9

The results summarized in Table 7 show, that:

- a large gain in performance is achieved for the combination of confidence scores with statistical model scores.
- the proposed method outperforms all single MT engines (cf. Table 4)
- it achieves the same level of performance as the sequential decoding of *all* initial translation hypotheses (cf. Table 5)

4.4 Discussion

In order to investigate the effects of the proposed method on the computational costs, we compared the processing time of the EBMT+RBMT system that decodes all five initial hypothesis toward the proposed CONF·TM·LM method that selects a single hypothesis. The results show that the proposed method is 7 times faster than the EBMT+RBMT system, thus reducing the computational costs by 85.7%.

Moreover, an investigation into the feature dependency revealed, that *inter-hypotheses* features are most important. For example, if two or more MT engines produce the same initial translation hypothesis, it is an indicator of good quality. Therefore, similarity features like "*the number of identical initial translation hypotheses*" appear at the top of the decision tree classifier.

On the other hand, *general features* like language perplexity or information about the sentence structure seems to be less important. They are used in the decision tree classifier, but appear mainly on lower levels of the decision tree.

However, the set of features used in our experiments is not exclusive. Further investigations have to verify the usefulness of additional features not used in the above experiments like the *minimal tiling of substrings* (Quirk, 2004).

Moreover, the lower total error rate obtained for the classification of the training compared to the test data set indicates the problem of overfitting. Therefore, the application of pruning techniques and the careful selection of features might help to improve the classifier performance and thus the overall system performance of the proposed method.

5 Conclusion

This paper described a machine learning approach to seeding a greedy decoder effectively. The proposed method used a decision tree classifier to judge the appropriateness of multiple translation-engine-based hypotheses and selects a single initial translation hypothesis *before* decoding based on statistical model scores of the input-hypothesis pairs as well as confidence scores derived from the decision tree classification results.

The proposed method was integrated into the greedy decoding approach and the effectiveness of this approach was verified for Japanese-to-English translation of dialogues in the travel domain.

An analysis of the evaluation results showed that *the proposed hypotheses selection method avoids high computational costs* by limiting the decoding process to a single initial hypothesis *without a loss in translation quality.*

Acknowledgments

The authors' heartfelt thanks go to Kadokawa-Shoten for providing the Ruigo-Shin-Jiten. The research reported here was supported in part by a contract with the National Institute of Information and Communications Technology entitled "A study of speech dialogue translation technology based on a large corpus."

References

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

S. Corston-Oliver, M. Gamon, and C. Brockett. 2001. A machine learning approach to the the automatic evaluation of machine translation. In *Proc. of 39th ACL*, pages 148–155, Toulouse, France.

Fujitsu. 2003. ATLAS Honyaku Superpack V9. http://software.fujitsu.com/jp/atlas.

U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proc. of 39th ACL*, pages 228–235, Toulouse, France.

K. Imamura. 2002. Application of translation knowledge acquired by hierarchical phrase alignment for pattern-based MT. In *Proc. of 9th TMI*, pages 74–84, Kyoto, Japan.

G. Kikui, E. Sumita, T. Takezawa, and S. Yamamoto. 2003. Creating corpora for speech-to-speech translation. In *Proc. of the EUROSPEECH03*, pages 381–384, Geneve, Switzerland.

LogoVista. 2001. X PRO Multilingual Edition Ver.2.0. http://www.logovista.co.jp.

D. Marcu. 2001. Towards a unified approach to memory- and statistical-based machine translation. In *Proc. of the 39th ACL*, pages 378–385, Toulouse, France.

S. Ohno and M. Hamanishi. 1984. *Ruigo-Shin-Jiten*. Kadokawa.

M. Paul, E. Sumita, and S. Yamamoto. 2004. Example-based rescoring of statistical machine translation output. In *Proc of the HLT-NAACL, Companion Volume*, pages 9–12, Boston, USA.

C.B. Quirk. 2004. Training a sentence-level machine translation confidence measure. In *Proc. of 4th LREC*, pages 825–828, Lisbon, Portugal.

Rulequest. 2004. Data mining tool c5.0. http://rulequest.com/see5-info.html.

K. Su, M. Wu, and J. Chang. 1992. A new quantitative quality measure for machine translation systems. In *Proc. of the 14th COLING*, pages 433–439, Nantes, France.

E. Sumita, S. Yamada, K. Yamamoto, M. Paul, H. Kashioka, K. Ishikawa, and S. Shirai. 1999. Solutions to problems inherent in spoken-language translation: The ATR-MATRIX approach. In *Proc. of the Machine Translation Summit VII*, pages 229–235, Singapore.

E. Sumita. 2001. Example-based machine translation using DP-matching between word sequences. In *Proc. of the 39th ACL, Workshop: Data-Driven Methods in Machine Translation*, pages 1–8, Toulouse, France.

T. Takezawa, E. Sumita, F. Sugaya, H. Yamamoto, and S. Yamamoto. 2002. Toward a broad-coverage bilingual corpus for speech translation of travel conversations in the real world. In *Proc. of the 3rd LREC*, pages 147–152, Las Palmas, Spain.

C. Tillmann and H. Ney. 2000. Word reordering and dp-based search in statistical machine translation. In *Proc. of COLING 2000*, Saarbruecken, Germany.

Toshiba. 2003. TheHonyaku Ver.7.0. http://pf.toshiba-sol.co.jp/prod/hon_yaku/index_j.htm.

R.W. Wagner. 1974. The string-to-string correction problem. *Journal of the ACM*, 21(1):169–173.

Y. Wang and A. Waibel. 1997. Decoding algorithm in statistical machine translation. In *Proc. of 36th ACL*, Madrid, Spain.

T. Watanabe and E. Sumita. 2003. Example-based decoding for statistical machine translation. In *Proc. of the Machine Translation Summit IX*, pages 410–417, New Orleans, USA.

A Semantics-based English-Bengali EBMT System for translating News Headlines

Diganta Saha
Computer Science and Engineering Department
Jadavpur University
Kolkata, India, 700032
neruda0101@yahoo.com

Sivaji Bandyopadhyay
Computer Science and Engineering Department
Jadavpur University
Kolkata, India, 700032
sivaji_cse_ju@yahoo.com

Abstract

The paper reports an Example based Machine Translation System for translating News Headlines from English to Bengali. The input headline is initially searched in the Direct Example Base. If it cannot be found, the input headline is tagged and the tagged headline is searched in the Generalized Tagged Example Base. If a match is obtained, the tagged headline in Bengali is retrieved from the example base, the output Bengali headline is generated after retrieving the Bengali equivalents of the English words from appropriate dictionaries and then applying relevant synthesis rules for generating the Bengali surface level words. If some named entities and acronyms are not present in the dictionary, transliteration scheme is applied for obtaining the Bengali equivalent. If a match is not found, the tagged input headline is analysed to identify the constituent phrase(s). The target translation is generated using English-Bengali phrasal example base, appropriate dictionaries and a set of heuristics for Bengali phrase reordering. If the headline still cannot be translated using example base strategy, a heuristic translation strategy will be applied. Any new input tagged headline along with its translation by the user will be inserted in the tagged Example base after generalization.

1 Introduction

The present work aims to develop a methodology for a semantics-based Example Based Machine Translation (EBMT) system for translating news headlines from English to Indian languages. The methodology is being deployed to implement a machine translation system for translating news headlines from English to Bengali, a major Indian language and the fifth language in the world in terms of the number of native speakers. It is the official language of Bangladesh. The reason for choosing English as the source language is that most news are generated in English, even in India, and the vernacular dailies carry out a translation before publishing them. The semantic and syntactic classification schemes developed for English news headlines may be useful for building news headline machine translation systems from English to other languages.

Most of the International and National news wire service agencies send news items in English. Manual translation of these news items into any other language is slow and tedious. The inflow of news items is not evenly distributed, therefore there is burst of translation required just before the newspaper is to go out. The domain of news items has attracted the attention of Machine Translation (MT) researchers all over the world. The internet editions of newspapers in English and regional languages are now a reality.

Translation of news headlines plays a crucial role in the translation of a news item. The headline is an important component in a news item. The headline must be informative, i.e., it should indicate sufficiently about the content of the news item. At the same time it must attract the attention of the reader, i.e., it must have its own style. The informative property of the news headlines must be retained as far as possible while translating into the target language. Each language has its own style of writing headlines. The style of the source language news headline can be preserved by assigning semantic tags to the words in the news headline in addition to the syntactic tags. The style of the news headline in the target language can be maintained by developing a parallel example base of news headlines in source and target languages, assigning semantic as well as the syntactic tags to both sides for generalizing the paralle example base, aligning the entries and then following an example based machine translation strategy. A direct parallel example base of news headlines may be necessary for those headline pairs which are unique in nature

and thus cannot be generalized. The system described in the present work follows this strategy.

News headlines are generally not grammatical sentences in nature. They can be or can consist of root word(s), surface level word(s), named entity(ies) (person names, location names, organization names, and miscellaneous e.g., temporal expressions, monetary expressions, cinema names, book names, hotel names, train names), acronym(s), noun phrase(s), sentence without an auxiliary verb, quotation or a grammatical sentence. The syntactic structure of the news headlines suggests that while they cannot always be defined by the sentence level grammar formalisms, news headlines follow a sublanguage of its own. Thus, the Rule based machine translation strategy that uses sentence level grammar formalisms is not suitable for the translation of news headlines. If the input news headline cannot be translated using either the direct or the generalized example base, the tagged input headline may be analysed to identify the constituent phrase(s). The target translation is then generated using the parallel phrasal example base, appropriate dictionaries and a set of heuristics for target language phrase reordering. This rule based machine translation strategy has been followed in the present work.

In India, most English newspapers have their vernacular publication but the layout of news and their headlines are not parallel, i.e., not exact translation of each other. Thus corresponding news headlines in English and vernacular editions cannot be directly used to create a large parallel example base of English-Vernacular news headlines. The two machine translation systems for translating English news headlines to Hindi (Sinha, 2002; Rao et. al., 2000) do not have a large parallel example base of English-Hindi news headlines. Thus Statistical machine translation (SMT) system is also not suitable for machine translation of English news headlines to Indian languages. In this work, we are creating the tagged parallel example base of news headlines with the help of English and Bengali newspapers of the same date. The present system generalizes the tagged English news headlines. The corresponding set of tagged Bengali news headlines may not be identical. The system displays the possible generalized tagged Bengali news headlines and the developer chooses one of them. The chosen target language news headline may be edited to maintain the informative nature and the style. This collection of parallel example base is not large enough to attempt SMT. In view of these, it has been considered that EBMT strategy is most suitable for translation of news headlines. The EBMT strategy also allows the system to integrate different resources, namely, Direct example base, Generalized tagged example base and the Phrasal example base which are discussed later.

Related works on machine translation of news headlines in India as well as elsewhere in the world are discussed in section 2. Semantic and syntactic classification of news headlines have been outlined in section 3 and 4 respectively. Tag set definition and tagging of English and Bengali news headlines are discussed in section 5. Creation of generalized tagged example base of English and Bengali news headlines are described in sections 6 and 7 respectively. Section 8 describes the different example bases in the system, specifically the Phrasal example base. The dictionary design is outlined in section 9. MT system development methodology is described in section 10 and the conclusion is drawn in section 11.

2 Related Works

In India, a heuristic approach for translating news headings from English to Hindi is found in (Sinha, 2002). A human-aided MT system for translating English news texts to Hindi is being developed at the Centre for Development of Advanced Computing, Mumbai (Rao et al, 2000). The system is now being enhanced and adopted for web translation service to the news agencies. A hybrid system for translating news items from English to Bengali (Naskar & Bandyopadhyay, 2005; Bandyopadhyay & Saha, 2002; Bandyopadhyay, 2000a, 2000b) is being developed at the Jadavpur University, India.

The NHK System of Japan which translates English newspaper articles to Japanese is described in (Hutchins, 1999). The improvement of translation quality of English newspaper headlines by automatic pre-editing in the English to Japanese machine translation system being developed at the Sharp Corporation of Japan is discussed in (Yoshimi, 2001). The work focuses on the absence of the verb *be* and formulates a set of rewriting rules for putting the verb properly into headlines, based on information obtained by morpholexical and rough syntactic analysis. The improvement of translation style and the target words of English news headlines by identifying and resolving the coreference of acronyms, abbreviations and proper names in the English to Japanese machine translation system being developed at the Toshiba Corporation of Japan is discussed in (Ono, 2003).

3 Semantic Classification of News Headlines

News items in a news paper generally follow a classification on the basis of geographical

hierarchy (Metro -> State -> Country -> World) as well as a separate topic based one (Sports, Business etc.). Though there does not exist any standard classification of news items in the journalistic world, we conducted a study on six English news papers that are published from Kolkata. It has been observed that all of them follow the same geographic classification as well as a topic based one, though the names of the classes are not always identical. Named entities and acronyms occur in very large number in news items as well as in the associated news headlines and these named entities and acronyms tend to cluster around each class in the classification scheme. The classification of the news items as well as the associated headline, on the basis of the content of the news has been termed as the **Semantic Classification**. Having separate bilingual named entity and acronym dictionaries under each semantic class help in the disambiguation of these words also. In the present work, English news headlines have been semantically classified as follows: *Front Page, World, India, Bengal, Kolkata, Business and Sports*. The news in the *Front Page* include the important news events for the day that may belong to any category. There are further classification like *Editorial, Perspective, Cinema, Entertainment or Campus* whose contents are mainly feature based. Headlines for these items have not been considered in the present work. The Bengali news papers published from Kolkata carry more news from the state and hence they follow a more detailed classification on *Bengal*. In the present work, we have followed identical classification schemes for both English and Bengali news headlines.

News items can be further classified into the following two categories on the basis of the number of paragraphs in the news item: (i) Short Single Paragraph News Items and (ii) Long Multi-paragraph News Items as they follow distinct styles. Long multi-paragraph news items are more informative in nature. Headlines for both these types of news items also differ in their style and informative nature. Headlines for short single paragraph news items are generally one-, two- or three words long; may be a named entity, compound noun or noun phrase and occassionally may be sentences. Long multi-paragraph news items may include two separate headlines. Sometimes, within the bodies of these news items short news along with a separate headline are found, either originating from the same place as the main news or dealing with a related topic. Apart from these syntactic differences which are discussed in the next section, headlines from these two categories of news are also different on their information content. For example, the headlines *Tea Strike* and *Garden workers go on an indefinite strike for pay hike / Trouble brews in tea estates* correspond to the short and long versions of the same news event. It may be noted that there are two headlines for the long news. On the basis of these observations, the following semantic classes *Front Page, World, India, Bengal, Kolkata, Business and Sports* have been further divided into *short* and *long* classifications. The example news headlines for the various semantic classes are shown in Table 1:

Semantic Class	Example News Headline
Front Page	Snaps say error camps exist: Natwar
Front Page – short	FB threat
World	Rice no-show invites criticism
World-short	Van Gogh trial
India	PM assures left on eve of US trip
India-short	Ex-servicemen
Bengal	Bandh to protest against blasts
Bengal-short	SFI clash
Kolkata-	More courses at Presidency
Kolkata-short	Train services hit
Business	Assam Tea workers want basic pay revised Productivity-linked wages rejected by ACMS
Business-short	Microsoft
Sports	ICC says 2004-05 was corruption-free
Sports-short	Selections

Table 1: Semantic Classification of News Headlines

4 Syntactic Classification of News Headlines

News headlines for short single paragraph news and those for the long multi-paragraph news show different syntactic structures. Headlines for short single paragraph news items can be classified at the top level on the basis of the number of words they contain, viz., one-, two-, three- or more than

three words. Similarly, headlines for long multi-paragraph news items can be classified at the top level on the basis of the number of words they contain, viz., three- or more than three words. Such headlines are generally grammatical sentences in nature. A headline may consist of two sentences, also. Sometimes, within the body of these news items short news are found, either originating from the same place as the main news or dealing with a related topic. It has been observed that the structure of these news headlines follow the same for the short single paragraph news headlines.

The *single word headlines* can be a root word, named entity, acronyms or a surface level word. *Two word headlines* can be a compound noun, sentence without an auxiliary verb, grammatical sentence or a collocation. *Three word headlines* can be a noun phrase, compound noun, sentence without an auxiliary verb or a grammatical sentence. *Headlines with more than three words* can be a noun phrase, sentence without an auxiliary verb, grammatical sentence or quotation.

On the basis of above observations, the news headlines have been syntactically classified at the top level on the basis of the number of words, viz., one-, two-, three- and more than three words. Since, three words or more than three words headlines appear for both short and long news items, each of the 7 semantic classes have been synactically classified at the top level further into 4 classes as above. In the present work, a total of 28 parallel example bases have been designed. Further syntactic classification (i.e., root word, named entity, acronym, surface word, compound noun, collocation, noun phrase, sentence without an auxiliary verb, grammatical sentence, quotation) is included as an attribute of the example news headline. This organization of the example bases makes it easier to identify the appropriate example base for an input news headline, where it is most likely to be present, given its semantic class and the number of words present in it. The syntactic classification provides appropriate information for alignment of the tagged source and target language news headlines. The syntactic class of the input news headline helps in the application of the appropriate rule based translation strategy when it cannot be translated using the example based translation methods. The example news headlines for the various syntactic classes are shown in Table 2.

Some headlines are elliptical in nature. An example of ellipsis is in the headline *Train kills 1* where the number *1* is not explicitly qualified but the implicit qualification is *person.* The elliptical resolution in this case is not necessary for translating it to Bengali as the ellipsis is retained in Bengali. Another example of an elliptical news headline is *Bhajji claims a couple*. In this case, ellipsis resolution is necessary for translation. Since, the news headline is for a sports news in which *claiming a couple* means *claiming a couple of wickets*, the headline will be extended as *Bhajji claims a couple of wickets* and then translated.

Syntactic Class	Example News Headline
One word	• Accident (root word) • Kirloskar (named entity) • HDFC (acronym) • Selections (surface word)
Two words	• Flight problem (compound noun) • Buddha's gesture (compound noun) • RBI report (compound noun) • Kanika critical (sentence without an auxiliary verb) • India wins (grammatical sentence) • Pulse Polio (collocation)
Three words	• Woods on top (noun phrase) • Shastri Bhavan fire (compound noun) • Tour de France (compound noun) • Sania No. 70 (sentence without • an auxiliary verb) • Train services hit (sentence without an auxiliary verb) • Australia wins again (grammatical sentence)
More than three words	• Breakthrough in diagnosing HIV (noun phrase) • Rail contracts under cloud (sentence without an auxiliary verb) • Family health drive enters fifth round (grammatical sentence) • Intelligence couldn't have prevented attack: Blair (quotation)

Table 2: Syntactic Classification of News Headlines

5 Tag set definition and Tagging of News Headlines

The present system is being developed for translating English news headlines to Bengali. Since, parallel example base of English and Bengali headlines is not available, we started with the collection of English news headlines under different semantic and syntactic classes. The headlines in English and Bengali are tagged with a set of syntactic and semantic tags.

5.1 Tag Sets

Noun and verb words are tagged with the corresponding Wordnet Lexicographer file names.

The following are some example tags used for noun words:

noun.act: noun denoting acts or actions,

noun.animal: nouns denoting animals,

noun.artifact: noun denoting man-made objects,

noun.body: noun denoting body parts,

noun.event: noun denoting natural events,

noun.food: noun denoting foods and drinks,

noun.group: noun denoting grouping of people or objects,

noun.location: noun denoting spatial position,

noun.person: noun denoting people,

noun.time: noun denoting time and temporal relations.

It may be noted that when the noun words in the last four types identify a specific object they denote a named entity and are appropriately tagged.

The following are some example tags used for verb words:

verb.change: verbs of change of size, temperature, intensity, etc.,

verb.cognition: verbs of thinking, judging, analyzing, doubting, etc.,

verb.communication: verbs of telling, asking, ordering, singing, etc.,

verb.competition: verbs of fighting, athletic activities, etc.,

verb.consumption: verbs of eating and drinking,

verb.contact: verbs of touching, hitting, tying, digging, etc.,

verb.creation: verbs of sewing, baking, painting, performing, etc.,

verb.motion: verbs of walking, flying, swimming, etc.,

verb.possession: verbs of buying, selling, owning and transfer,

verb.social: verbs of politicial and social activities and events,

verb.weather: verbs of raining, snowing, thawing, thundering, etc..

The tagging of the verb words in the headlines helps to identify the source and the target language verb patterns (Kim et. al., 2002). Each verb can have several meanings and each meaning of a verb is represented by a verb pattern. A verb pattern consists of a source language pattern part for the analysis and the corresponding target language pattern part for the generation. The meaning of a verb can be identified using the associated noun and the adjective words. For example, the verb *kill* is tagged as *verb.contact*. The associated noun words for one meaning of the verb are *accident, attack* etc. and the adjective word *dead* is associated with the same meaning of the verb.

Named entities are further tagged as *Person Name*, *Location Name*, *Organization Name* and *Miscellaneous* e.g. *temporal expressions*, *monetary expressions*, *cinema names*, *book names*, *hotel names*, *train names* etc.. Strictly speaking, further tagging of named entities are not necessary for headline translation except in tagging of *person names* and *organization names* and that too, when the headline includes a verb word. The verb form in Bengali depends on the associated named entity. Words of other parts of speech (*adjective, adverb, preposition, article, conjunction*) are tagged by their part of speech category only. Further tag sets are Anaphora Classes (*personal pronoun, demonstrative pronoun, abbreviation, special symbol*) and *Numbers*. Personal and demonstrative pronouns generally occur when the headline is a quotation. Special symbols like *$* have been considered as a separate anaphora class as in many

target language headlines the transliteration of the the full form of the symbol, i.e., *dollar*, is used. The *abbreviation* class includes both abbreviated words and acronyms. *Abbreviations* have been considered as a special class of anaphora as they are incomplete in nature and have to be resolved by either looking into the dictionary or in the first paragraph of the associated news item. The first paragraph in a news items is likely to include the content words of the associated headline. The abbreviated words can be resolved while the news headlines are collected from the corpus of English news items. Numbers can appear either in the form of digits or words in the source and the target language headlines and hence these are tagged separately.

5.2 Tagger / Recognizer / Classifier

The words / terms in the input English headlines in English are identified with the help of a tokenizer, a morphological analyzer and a Named entity recognizer(NER) and classifier. The system uses a lexicon of English words developed from the Wordnet 2.0 which includes the lexicographer file level tags associated with each word and term. The lexicon is being developed at the bilingual level and the Bengali meaning of the words are being entered in phases. A separate bilingual list is maintained for words that are pronouns, prepositions, articles and conjunctions. The words / terms are initially tagged at the part of speech (POS) level and then further tagged by a semantic tagger. The semantic tagger uses separate bilingual tables for abbreviations, acronyms, special symbols and various types of named entities. Identification of acronyms in long multi-paragraph news items causes problem as all words in the headline are sometimes written in all capital. Acronyms in short news headlines can be identified by looking for words which are all capital or may include a vowel in small case (e.g., HoD) or a special symbol (e.g., J & K). The system uses a Named Entity Recognizer and Classifier System for English developed in-house as part of a separate research activity. The NER system uses a *frequent starter's list* containing words that appear at the beginning of headlines but are not named entities themselves. This list has been prepared by looking into the English headlines collected in the example base. The NE classifier system is basically table driven and uses a limited set of features. A shallow parser for English (Naskar & Bandyopadhyay, 2005) is being used for identifying the compound nouns, noun phrases and the verb phrases in the input headline. The shallow parser can also detect whether the input headline is a quotation or a grammatical sentence. Thus, the shallow parser is identifying the syntactic category of the English news headline. The system also maintains a bilingual collection of collocations from English to Bengali. The Bengali portion of the parallel news headline is tagged by searching each word of the English headline in the appropriate dictionary or list and finding the Bengali equivalent. A match for the Bengali word is searched in the headline using a Bengali morphological analyzer. If a named entity or an acronym cannot be found in the bilingual dictionary, it is transliterated into Bengali and then searched in the Bengali headline. The Bengali headline may include additional words (nouns or adjectives) which are associated with the verb in the English headline. The system maintains a list of noun and adjective words associated with each meaning of a verb. These additional words are tagged separately using the Bengali lexicon which associates each Bengali noun and verb word with tags similar to those for English.

Let us consider the following examples of parallel English-Bengali headlines. The English gloss of the Bengali words are shown in brackets.

(i) Train kills two ←→ ট্রেনে কাটা পড়ে মৃত দুই
[*traine kaataa parhe mrita dui*]

(ii) Bus kills 1 ←→ বাস দুর্ঘটনায় মৃত এক
[*bus durghatanaaya mrita ek*]

(iii) Train kills 3 ←→ ট্রেন দুর্ঘটনায় মৃত তিন
[*train durghatanaaya mrita tin*]

(iv) Elephant kills three ←→ হাতির আক্রমণে মৃত তিন
[*haatir aakramane mrita tin*]

(v) Two killed in train accident ←→ ট্রেন দুর্ঘটনায় মৃত দুই
[*train durghatanaaya mrita dui*]

The parallel example base of headlines after tagging will look like

(i) <*train*, noun.artifact> <*kill*, verb.contact> <*two,*number> ←→ < ট্রেন [*train*], noun.artifact> <-ে [*-e*]> < কাটা পড় [*kaataa parh*], noun.event> <-ে [*-e*]> <মৃত, [*mrita*], adjective>< দুই [*dui*], number>

(ii) <*bus*, noun.artifact> <*kill*, verb.contact> <*1,*number> ←→ <বাস [*bus*],

noun.artifact> <দুর্ঘটনা.[*durghatanaa*], noun.event> <-য় [*-ya*]> <মৃত [*mrita*], adjective> <এক [*ek*], number>

(iii) <*train*, noun.artifact> <*kill*, verb.contact> <*3,*number> ←→ <ট্রেন [*train*], noun.artifact> <দুর্ঘটনা. [*durghatanaa*], noun.event> <-য় [*-ya*]> <মৃত [*mrita*], adjective> <তিন [*tin*], number>

(iv) <*elephant*, noun.animal> <*kill*, verb.contact> <*three,*number> ←→ <হাতি [*haati*], noun.animal> <-র [*-r*]> <আক্রমণ [*aakraman*], noun.event> <-ে [*-e*]> <মৃত [*mrita*], adjective> <তিন [*tin*], number>

(v) <*two,*number> <*kill*, verb.contact> <*in*, preposition> <*train*, noun.artifact> <*accident*, noun.event> ←→ <ট্রেন [*train*], noun.artifact> <দুর্ঘটনা. [*durghatanaa*], noun.event> <-য় [*-ya*]> <মৃত [*mrita*], adjective> < দুই [*dui*], number>

The two tags <noun.artifact> and <number> can be directly aligned. The tags <-য়>, <-র> and <-ে> are Bengali inflections to be attached to the preceeding word. The two tags <noun.event> and <adjective> are associated with the tag <verb.contact>. The system maintains a list of *verb.contact* words and the associated *noun.event* word. The *adjective* word is basically used to qualify the object of the *verb.contact* and the system maintains a list of such Bengali adjectives for the verbs.

6 Creation of Generalized Tagged Example Base of English News Headlines

We are creating the tagged parallel example base of news headlines with the help of English and Bengali newspapers of the same date. The tagged English headlines are automatically generalized. The generalization process of tagged news headlines is basically identifying the identical tagged news headlines and then generalizing them. Two tagged headlines can be considered identical if they have identical tags at all the corresponding positions. In the above example, tagged headlines (i) <*train*, noun.artifact> <*kill*, verb.contact> <*two,*number> and (ii) <*bus*, noun.artifact> <*kill*, verb.contact> <*1,*number> and (iii) <*train*, noun.artifact> <*kill*, verb.contact> <*3,*number> are identical and they can be generalized to <noun.artifact> <verb.contact> <number>. Two tagged headlines can be considered similar if all the tags present in one headline are present in the other and the later headline includes noun and adjective tags which can be derived from the verb tag. In the above example, tagged headlines (i)<*train*, noun.artifact> <*kill*, verb.contact> <*two,*number>, (ii)<*bus*, noun.artifact> <*kill*, verb.contact> <*1,*number>, (iii) <*train*, noun.artifact> <*kill*, verb.contact> <*3,*number> and (v) <*two,*number> <*kill*, verb.contact> <*in*, preposition> <*train*, noun.artifact> <*accident*, noun.event> are considered similar since all the tags present in (i), (ii) and (iii) are present in (v) and (v) includes the noun.event tag with the word *accident* that can be derived from the verb tag <*kill*, verb.contact>. The system will maintain the list of such noun and adjective words that can be derived from the verb word. Headlines which are similar can be generalized at the next level.

The headlines that do not take part in any generalization are kept in the Direct Example base alongwith the tagging. During the development of the parallel example base further match with headlines in the Direct example base can occur and the generalized headline can then be included in the Generalized Tagged Example Base. It appears from the above that the headlines (i), (ii) and (iii) can be generalized on the English side and the generalized tagged headline will be stored as <noun.artifact> <verb.contact> <number> in the Generalized Tagged Example Base. Since the headlines (iv) and (v) cannot be generalized, the original headlines will be kept in the Direct Example base with its tags.

7 Creation of Generalized Tagged Example Base of Bengali News Headlines

Let us consider the five example parallel English-Bengali news headlines as mentioned in the section 5.2.. It can be seen that the set of Bengali news headlines corresponding to the English news headlines (i), (ii) and (iii), which have been generalized, are not identical. This is a general phenomenon and has been observed during the devlopment of the example base. It also shows the stylistic variations in news headlines across languages and within the same language also. This module identifies the tagged Bengali news headlines that are identical for a tagged English news headline and generalizes the Bengali news headlines under each subset. In this example, the two generalized tagged Bengali headlines corresponding to the English news headlines (i), (ii) and (iii) are identified as <noun.artifact>

<noun.event> <-য়> <adjective> <number> and <noun.artifact> <-ো> <noun.event> <-ো> <adjective><number>. These two generalized Bengali news headlines are shown to the developer for choosing one of them. The chosen Bengali tagged news headline can also be edited by the developer to maintain the informative nature and the style and the edited tagged Bengali news headline is associated with the generalized tagged English news headline and stored in the generalized tagged Example base. The parallel headlines are automatically aligned by their tags.

8 Creation of Example Bases (Direct Example Base, Generalized Tagged Example Base, Phrasal Example Base)

The system consists of three types of Example bases: (i) Direct Example Base, (ii) Generalized Tagged Example Base and (iii) Phrasal Example Base. The Direct Example Base is like the Translation Memory that stores the headlines in the source language and their translation in the target language. The Generalized Tagged Example Base stores the tagged examples with proper alignment. The Phrasal Example Base, stores the various phrase patterns in terms of the part of speech of the constituent words in the source language and their corresponding translation in the target language. The system uses the phrasal example base of English and Bengali phrase patterns used in a phrasal example based machine translation system (Naskar and Bandyopadhyay, 2005). The phrasal EBMT system is being developed for translating English news items to Bengali.

The Phrasal Example base consists of translation examples (phrasal templates) that store the part of speech of the constituent words of the phrases along with necessary syntactic information. Some examples of noun phrasal examples are:

(i) <art $ a / an> <noun & singular, human, nominative> ←→<একজন [*ekjan*]> <noun>

(ii) <art $ the> <noun & singular, human, objective> ←→ <noun> <-টিকে [*-tike*]>

(iii) <art $ a / an> <adj> <noun & singular, inanimate, objective> ←→ <একটি [*ekti*]> <adj> <noun>

An example of a prepositional phrase is

(iv) <prep $ to / at / in> <art $ the> <noun & singular, place> ←→ <noun> <- ে / েত [*-e/te*]>.

The headline “A hearty walk” may be translated by using the phrasal example base as the headline matches with the Noun phrasal example (iii).

9 Dictionary Design

The system uses a lexicon of English words developed from the Wordnet 2.0 which includes the lexicographer file level tags associated with each word and term. The lexicon is being developed at the bilingual level and the Bengali meaning of the words are being entered in phases. A separate bilingual list is maintained for words that are pronouns, prepositions, articles and conjunctions. There are separate bilingual tables for abbreviations and special symbols. Separate bilingual dictionaries for named entities and acronyms are maintained for each semantic and syntactic class in which the named entities and acronyms are most likely to occur. The named entity recognizer uses a *frequent starter’s list* containing words that appear at the beginning of headlines but are not named entities themselves. This list has been prepared by looking into the English headlines collected in the example base. The system also maintains a bilingual collection of collocations from English to Bengali. The Bengali headline may include additional words (nouns or adjectives) which are associated with the verb in the English headline. The system maintains a list of noun and adjective words associated with each meaning of a verb.

10 MT System Development Methodology

During translation, the input headline is initially searched in the Direct example base for an exact match. If a match is obtained, the Bengali headline from the example base is produced as output. If there is no match, the headline is tagged and the tagged headline is searched in the Generalized Tagged Example base. If a match is obtained, the output Bengali headline is to be generated after appropriate synthesis. If a match is not found, the Phrasal example base will be used to generate the target translation. If the headline still cannot be translated, the following heuristic translation strategy will be applied: translation of the individual words or terms in their order of appearance in the input headline will generate the translation of the input headline. Appropriate dictionaries will be consulted to attempt a translation of the news headline.

Let us consider the five example parallel English-Bengali news headlines as mentioned in section 5.2.. If the headline *Elephant kills three* is given for translation, a match will be obtained in the Direct Example Base. The corresponding Bengali translation will be retrieved and then shown as output. If the headline *Tiger kills 1* is to be translated there will be no exact match in the Direct Example Base but the tagged version of the input headline will match with the tagged version of *Elephant kills three.* The two headlines will now be generalized on the source side and the tagged Bengali headline corresponding to *Elephant kills three* will be considered as the generalized tagged Bengali headline. The generalized tagged headlines will be included in the appropriate example base. The headline *Elephant kills three* will be removed from the direct example base. If the input headline is *Tram kills 2*, it will obtain a match in the generalized tagged example base and will be translated accordingly. If the input headline is *A sweet dream,* it will not find any match with either the direct or the generalized tagged example base. The Phrasal example base will then be consulted and the input headline will match with the noun phrase structure. The Bengali translation can be obtained accordingly. If the input headline is *Shastri Bhavan fire,* it will not find any match even in the phrasal example base. The headline will be iedntified as a compound noun as *Shastri Bhavan* is a named entity and *fire* is a noun. The system will produce an output following the heuristic. The named entity will be transliterated and the Bengali equivalent of the word *fire* will be obtained from the dictionary. The sequence of the two Bengali words will be presented as the output. The output will not be accepted by the user in this case as an inflection is necessary after the transliterated named entity and the heuristics could not produce that.

A preliminary version of the machine translation has been developed. The different example bases and the dictionaries are under development. Work is also going on for the development of a Bengali lexicon that includes the tags which are similar to those used in the English Wordnet at the Lexicographer file level.

11 Conclusion

Our news headline corpus is a collection of 2000 news headlines from the Kolkata edition of the News paper 'The Statesman'. A preliminary version of the machine translation has been developed. The different example bases and the dictionaries are under development. Work is also going on for the development of a Bengali lexicon that includes the tags which are similar to those used in the English Wordnet at the Lexicographer file level. Initial testing of the MT System has started and no formal evaluation of the system has been carried out.

References

R.M.K. Sinha. 2002. *Translating News Headings from English to Hindi*. In "The 6th IASTED International Conference on Artificial Intelligence and Soft Computing (ASC2002) Proceedings", Banff, Canada.

D. Rao, and K. Mohanraj et al. 2000. *A Practical Framework For Syntactic Transfer Of Compound – Complex Sentences For English – Hindi Machine Translation*. In "International Conference KBCS-2000 Proceedings", Mumbai, India : 343-354.

S. Bandyopadhyay & D. Saha. 2002. *Anaphora / Coreference in Machine Translation of News Headlines*. In "Discourse Anaphora and Anaphora Resolution Colloquium (DAARC) 2002 Proceedings", Portugal.

S. Bandyopadhyay. 2000a. *An Example Based MT System in News Items Domain from English to Indian Languages*. In "Machine Translation and Multilingual Applications in the New Millennium Proceedings", Exeter, UK : 10-1 – 10-5.

S. Bandyopadhyay. 2000b. *ANUBAAD – Translating News Items from English to Bengali*. In "International Conference KBCS2000 Proceedings", Mumbai, India : 297-307.

J. Hutchins. 1999. *The Development & Use of Machine Translation Systems and Computer-based translation tools*. "The International Symposium on Machine Translation and Computer Language Information Processing Proceeding", China.

T. Yoshimi. 2001. Improvement of Translation Quality of English Newspaper Headlines by Automatic Pre-editing. *Journal of Machine Translation*, 16(4): 233-250.

C. Kim, M. Hong, Y. Huang, Y. Kim, S. Yang, Y. Seo and S. Choi. 2002. *Korean-Chinese Machine Translation Based on Verb Patterns.* Lecture Notes in Computer Science, Volume 2499: 94-103.

Kenji Ono. 2003. *Translation of News Headline*. In "MT SUMMIT IX Proceedings", New Orleans, Louisiana, USA.

Sudip Naskar and Sivaji Bandyopadhyay. 2005. *A Phrasal EBMT System for translating English to Bengali.* MT SUMMIT X, Phuket, Thailand.

Example-based Translation without Parallel Corpora: First experiments on a prototype

Vincent Vandeghinste, Peter Dirix and **Ineke Schuurman**
Centre for Computational Linguistics
Katholieke Universiteit Leuven
Maria Theresiastraat 21
B-3000 Leuven
Belgium
firstname.lastname@ccl.kuleuven.be

Abstract

For the METIS-II project (IST, start: 10-2004 – end: 09-2007) we are working on an example-based machine translation system, making use of minimal resources and tools for both source and target language, i.e. making use of a target language corpus, but not of any parallel corpora.

In the current paper, we present the results of the first experiments with our approach (CCL) within the METIS consortium : the translation of noun phrases from Dutch to English, using the British National Corpus as a target language corpus.

Future research is planned along similar lines for the sentence as is presented here for the noun phrase.

1 Introduction: Background of METIS-II

The METIS approach differs from other known statistical or example-based approaches to machine translation in that it does not make use of parallel corpora (or bitexts) (Dologlou et al., 2003).

It is conceived as a system to be used in those circumstances in which other MT-systems that are around cannot be used, for example, because there are no sufficiently large parallel corpora available, at least not in the given domain (be it a specific subdomain, such as the automotive domain, or the domain of free language) and/or for a given language pair. The latter will often be the case in the European context when smaller languages are involved.

Constructing a rule based system would take too much time (and therefore be too costly). An alternative solution would be to use a hybrid system, not relying on parallel corpora and with relatively few rules. METIS-II is meant to become such a system.

The rationale behind the METIS projects is that a monolingual corpus in the target language guiding the validation of translations (choice of translation alternatives, word order), together with a bilingual dictionary guiding the raw lemma-to-lemma translation, should in principle suffice to generate good translations using a combination of statistics and linguistic rules, i.e. a hybrid approach. This monolingual target language corpus is likely to contain (parts of) sentences with the target words in them, serving as target-language examples. Finding and recombining these is in fact what METIS-II is about. The target language corpus helps disambiguating between different translation possibilities and it is used to retrieve the target language word order.

The development of such a machine translation system which uses simple tools and cheap resources for a rather complex task could give natural language processing in circumstances in which little resources are available a real boost: tasks for which parallel corpora and other expensive resources were conceived to be indispensable, can become feasible without them.

Although languages for which parallel corpora are not available in a large quantity tend to lack other resources like lemmatizers or taggers, it is much cheaper to create such resources than to create a large enough parallel corpus that links the source language with the target language.

METIS-I aimed at constructing free text translations by relying on pattern matching techniques and by retrieving the basic stock for translations from large monolingual corpora. METIS-II aims at further enhancing the system's performance and adaptability by:

- Breaking sentence-internal barriers: the system will retrieve pieces of sentences (chunks) and will recombine them to produce a final translation. This approach was also used by (Veale and Way, 1997), (Nirenburg et al. 1994), and (Brown, 1996).
- Extending the resources and integrating new languages using post-editing facilities.
- Adopting semi-automated techniques for adapting the system to different translation needs.
- Taking into account real user needs, especially

as far as the post-editing facilities are concerned.

This paper describes the approach of the Centre for Computational Linguistics within the METIS consortium. Other approaches can be found in (Markantonatou et al., 2005, this volume) and (Badia et al., 2005, this volume).

The experiments in this article are part of the investigations in the breaking of the sentence-internal barriers. We use noun phrase (NP) translation as a test case.

Our NP translation system differs from the approach explained in (Sato, 1993), in that we do not use parallel corpora, but a bilingual dictionary, and that our system is not domain specific. We also use a different weighing mechanism (cf. section 2.3.2).

Dutch is used as a source language with the parts-of-speech tagset of (Van Eynde, 2004). English is used as a target language, the British National Corpus (BNC) as target-language corpus with the CLAWS5 tagset. A reason why not to use the world wide web as a resource like (Grefenstette, 1999) is that our corpus needs to be preprocessed (tagged, chunked, lemmatized) and our target language is English from native speakers.

For a more extensive description of the METIS system see (Dirix et al., 2005, this volume).

2 System Description

In this section we describe our prototype system, which is used in the experiments in section 3, and which is implemented in perl 5.8.5 (Wall, 2004).

In figure 1, we present the general system flow (at the sentence level). The prototype we use is part of this general system as it translates noun phrase chunks.

First we describe how the source language analysis is performed (section 2.1), then we describe how we map the source language to the target language (section 2.2), and finally we describe the target language generation (section 2.3).

2.1 Source Language Analysis

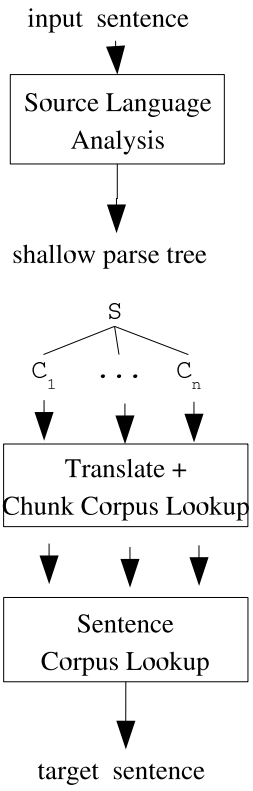
The source language (Dutch) text is analysed in a number of steps: tokenization (section 2.1.1), part-of-speech tagging (section 2.1.2), lemmatization (section 2.1.3) and chunking (section 2.1.4).

For the experiments in section 3, we used a test set of already analysed source language noun phrases.

Nevertheless, the prototype system is capable of doing its own source language analysis.

Let's take the following Dutch NP as an example: *een jonge champignon* [a young mushroom]

Figure 1: General System Flow



2.1.1 Tokenization

The first processing step in the source language analysis is the tokenization of the input sentence. The input sentence is converted into a series of tokens, representing separate words. All punctuation is considered as separate tokens.

Example		
"een jonge champignon"	tokenized into	*"een"* *"jonge"* *"champignon"*

2.1.2 Part-of-Speech Tagging

The part-of-speech (PoS) tagger we use is TnT (Brants, 2001), which was trained on the spoken Dutch corpus (CGN) internal release 6. It is reported to have an accuracy of 96.2% (Oostdijk et al., 2002). The tagset which was used is the CGN-tagset (Van Eynde, 2004).

Example		
een	gets the tag	*LID(onbep,stan, agr)* (indefinite article)
jonge		*ADJ(prenom,basis, met-e,stan)* (prenominal adjective)
champignon		*N(soort,ev,basis, zijd,stan)* (non-neutre singular common noun)

2.1.3 Lemmatization

Each token is lemmatized, by looking up the token and its PoS-tag in the CGN-lexicon (Piepenbrock, 2004), and retrieving the words lemma. For some tokens, the lemmatization process results in more than one lemma. By using the PoS-tag as additional input for the lemmatizer, the amount of ambiguity can be strongly reduced. For instance, the Dutch word *was* can be a noun meaning *wax* or *laundry* or the past tense singular of a verb meaning *to be*. It can thus be lemmatized as *was* (noun) or as *zijn* (verb). By using the PoS-tag as additional input, we can disambiguate between these two lemmas[1].

Example		
een	lemmatized into	*een*
jonge		*jong*
champignon		*champignon*

In future versions of our system we plan to implement a rule-based lemmatizer for Dutch, which would only use the lexicon for the exceptions to the rules and would have a larger coverage as it would also return lemmas for previously unseen words.

2.1.4 Chunking

The sentence is sent to the ShaRPa chunker, which was adapted for the METIS-II project and already used in (Vandeghinste and Pan, 2004) and (Vandeghinste and Tjong Kim Sang, 2004). The updated version of ShaRPa is using the same rules as before, but is now able to detect the heads of phrases, which was necessary for the approach described in this paper.

In the experiment described in this paper, it is used only to detect NPs and their heads. As described in Vandeghinste and Tjong Kim Sang, the chunking accuracy for noun phrases has an F-value of 94.7%.

Example	
een jonge champignon	
chunk type	NP
head	*champignon*

2.2 Source to Target Language Mapping

Source to target language mapping contains two stages: the translation of the source language lemmas into target language lemmas, using a bilingual dictionary (section 2.2.1) with a treatment for missing entries (section 2.2.2), and the conversion of the source language tags into the target language tags (section 2.2.3).

2.2.1 Bilingual Dictionary

For the mapping of the analysed source language NP to the target language, we use a bilingual dictionary, taking a lemma and a PoS-tag (without features) as input and returning a target language lemma and a partial target language tag.

The initial bilingual dictionary was compiled from various sources, like the Ergane Internet Dictionaries (Travlang Inc., http://www.travlang.com/Ergane) and the Dutch WordNet (Vossen et al., 1999) and manually edited and improved (Dirixa, 2002).

After some more editing and correcting the resulting dictionary contains about 37000 different source language lemmas. The average source language lemma has more or less three translations.

Note that one source language lemma can be translated into several consecutive target language lemmas.

Example		
een	is translated into	*a / an / one* *anybody* *some* *somebody* *someone*
jonge		*young*
champignon		*mushroom*

Together with the target-language lemmas, we retrieve target-language lemma tags from the dictionary. These tags contain only partial information,

[1] As far as *was* as a noun is concerned, this is a homonym meaning either *laundry* or *wax*. The tag associated with both meanings is not identical: they differ in gender. *Was* (laundry) is non-neuter, whereas *was* (wax) can be used both as neuter and non-neuter. Whenever the word is used in a neuter context (determiner, neuter form of adjective), we know for sure that it is to be translated as *wax*. In the other cases we are to derive the proper translation via the BNC (searching for adjectival and verbal contexts in which *laundry*, resp. *wax* are used).

Making use of this information still needs to be implemented.

compared with the CLAWS5 target-language tagset. Because the tag contains information about a lemma and not about a token it cannot contain certain feature values (e.g. number), but it can contain others (e.g. gender). In our current system it only contains the PoS, and no feature-information.

In some cases, one word in the source language is translated into several consecutive words in the target language. The dictionary should contain the PoS information for each of those words, which is not yet the case in the current version, where we use underspecification in those cases where that information is missing.

There is certainly room for other improvements to the dictionary, as it still contains mistakes and some high-frequency words are still missing (especially Belgian Dutch items). Future versions of our system will use updates of this dictionary.

As Dutch is a language with productive word formation processes (amongst others, Booij and van Santen, 1995), it is impossible to include all words in the dictionary.

As a weight for the different translation alternatives we use the frequency of that lemma and tag combination in the target-language corpus, divided by the total frequency of all the translation alternatives for that entry. If the translation alternative contains two words, we look up the frequency of that bi-gram in the target-language corpus instead of the frequencies of the separate words. When there are more than two words in the translation of the word, for now we use a back-off procedure of giving them the frequency of 1.

2.2.2 Out-of-Vocabulary Treatment

When translating NPs, there are always words missing from our lexicon. In these cases we apply the following approach:

- If tokens are tagged as *proper nouns* in the source language, keep them as they are. If there are no translation alternatives, set the weight for the translated entry to 1.
- Check if the tokens are *compounds*. If this is the case, then translate the compounds' modifier and head instead of the token as a whole. Here we use the same hybrid decompounding/compounding module as in (Vandeghinste, 2002), which is used in its decompounding mode. It takes a word (lemma or token) as its input and generates the word parts plus a confidence value. The modifier and the head are considered as separate tokens for the rest of the processing, and they are treated like dictionary entries which contain one word on the source language side and two on the target language side.

  It is clear from our experiments that this approach works only in a number of cases but fails in others. Nevertheless it improves translation accuracy.

  For instance, the word *maffiakenner* is not present in our lexicon. The word is split up into two parts: *maffia* and *kenner*, which are both in our lexicon. This results in the translation *Maffia expert*, which is a correct translation.

  The word *fractieleider* (leader of a parliamentary party) is also missing from our lexicon. We could also split it up into two parts: *fractie* and *leider*, which could both be in our lexicon. This would result in the translation *fraction leader* which is an inaccurate translation.
- If none of the above apply[2] , keep the word as it is, as we do not have a clue on how to translate it. In the experiment, we do not produce a translation in this case as it is definitely incorrect.

2.2.3 Tag Mapping Rules

Apart from what is described in the previous sections, tag mapping rules are used (Dirix, 2002b). For each source language PoS tag, the equivalent target language tags were identified and put in a database. Some of the morpho-syntactic features are *'translated'* from source to target language. The source language tagset is described in (Van Eynde, 2004) and the target language tagset CLAWS5 is described on the UCREL website (University Centre for Computer Corpus Research on Language), http://www.comp.lancs.ac.uk/ucrel/claws5tags.html.

Example		
LID()	into	*AT0*
ADJ(prenom,basis)		*AJ0*
N(soort,ev,stan)		*NN0* or *NN1*

By combining the partial tag from the dictionary and the tag mapping rules, we can reduce a number of ambiguities which would otherwise arise.

2.3 Target Language Generation

Generating the target language by using the BNC as a data-set of examples is a rather complex task.

The target language generation uses the head of the NP, plus the bag of the other lemmas in the NP,

[2]Some other regularities in the translation of compounds will be implemented at a latter stage (e.g. *parlementslid* into *member of parliament* instead of *parliament member*).

together with their target language tags. In order to find out the exact word order, and disambiguate the different translation possibilities coming from the bilingual dictionary we use the BNC, which is preprocessed as described in the following section.

First, we describe how the target language corpus was preprocessed (section 2.3.1), and then we describe how we match the bag with the corpus (sections 2.3.2, 2.3.3 and 2.3.4).

2.3.1 Preprocessing of the Target Language Corpus

We lemmatized the BNC, using the lemmatizer described in (Carl et al., 2005). Then, we chunked the BNC, using ShaRPa2.0 with a rule-set for English. This was done only up to the lowest NP level.

This results in a huge number of NPs, for which we have their head and the structure of the chunk (containing the tags of the leaf nodes and possible intermediate levels between the NP and the leaf nodes).

We put this in a database, indexed on the head, allowing fast retrieval of NPs based on their head.

If an NP is found for which the lemmas exactly match the lemmas in the bag of lemmas, we use this NP as a possible translation. The frequency with which this NP occurs in the BNC, divided by the total frequency of all the possible translations found this way is used as the weight for that translation.

If there is no exact match with the bag of lemmas, we try to find an NP with the same head, but for which the tags of the tokens in the NP match the tags in the bag of lemmas, and replace the words which are not occurring in the retrieved NP from the BNC, hence producing a translated NP.

2.3.2 NP Retrieval from BNC

When having the bag of lemmas and the head as input, we retrieve all noun phrases from BNC with this head. From these noun phrases, we extract the noun phrases in which each lemma of each word corresponds with the lemmas from the words in the bag.

When such a noun phrase is found, it is considered a translation alternative, with weight w_k which is calculated as follows:

$$w_k = \frac{freq(a_k)}{\sum_k freq(a_k)}$$

The frequency with which the alternative occurs in the BNC, divided by the total frequency of all matching NPs is used as the weight for that translation, ignoring the information about the frequency of the separate tokens in the BNC. When we cannot find such a noun phrase, we switch to NP Template Retrieval, which is described in the next section.

Example

In the BNC we find 273 different NPs with *mushroom* as the lemma of their head. Of these, there is only one which contains all the words from the bag, but it contains also a number of other tokens, which are not present in the bag, and therefore we switch from NP Retrieval to Head-based NP Template Retrieval.

2.3.3 Head-based NP Template Retrieval from BNC

When no noun phrase can be retrieved from the BNC in which all the lemmas in the bag correspond to the target language, we try to retrieve a noun phrase template, with the same head. In order to do so, we retrieve all the noun phrases from the BNC with the current head, and try matching the tag structure of these noun phrases with the tags of the translations coming from the dictionary.

When we find a matching template, we have to replace the original words in the retrieved noun phrase with the actual translations of the input words, where the tags of the original words match the tags of our dictionary translations. In this process, we replace as minimal as possible, maximizing the influence of the target language corpus.

This greatly enhances the coverage of using the noun phrases of the BNC.

Example

Of the 273 different NPs with *mushroom* as its head, there are 9 NPs which only differ one word with the bag of TL lemmas derived from the dictionary. They all contain three tokens, of which two are present in the lists of translation alternatives from the dictionary. Only the adjective differs. So we replace the adjective in these NPs by the translations of the adjective coming from our dictionary, which leads to the desired result, being *a young mushroom*.

Again, the relative frequency of occurrence of the NP Template is used as a weight for the different translation alternatives.

2.3.4 Other Cases

It still happens that no matching NP Template can be found in the BNC with the same head. When this is the case, we want to apply an even more general template approach, in which the head word does

not play any role anymore, but all the noun phrase structures we find in the BNC are taken into account (with their frequencies), so we match the words and target tags coming from the dictionary with the different tag-structures we find in the BNC, giving the most frequent tag-structure the highest translation priority.

As this is not yet implemented, when no solution is found following the procedure described in the previous sections, we generate a word-by-word translation, using the word frequency based weights to rank different translation alternatives.

3 Experiments

In these experiments we wanted to validate our approach by testing it on noun phrase translations. Different teams in the METIS2 consortium are investigating different approaches.

First we describe the methodology of our experiments (cf. section 3.1), and then an overview of the results is given (cf. section 3.2).

3.1 Methodology

For our experiments, we used a test set of 685 NPs, of which 467 come out of the Spoken Dutch Corpus[3], and 218 noun phrases out of recent newspaper texts.

All the input NPs are correctly tagged and chunked. When they were not correctly tagged or chunked, they were left out of the test set. This concerns a small number (about 1%) of mainly complex NPs[4].

We did only take NPs into account which contain at least one noun. NPs containing only a personal pronoun are not taken into account.

For the rest, the tool as it is currently implemented for these experiments follows what is described in section 2.

As the results described in this paper are only first prototype results, we did not apply any of the automated evaluation approaches for machine translation, like BLEU (Papineni et al., 2002), but evaluated our results manually by judging the translation quality.

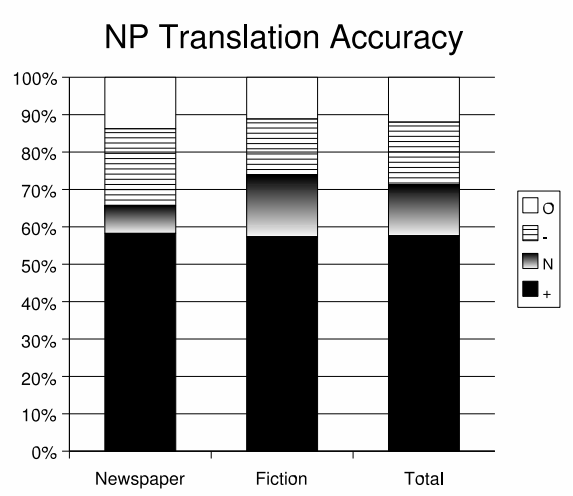
3.2 Results

Table 1 and figure 2 show the results of our evaluation. Each NP translation resulted in a number of translation alternatives, ranked by their weight. For each NP translation, we judged whether the first translation alternative was *correct* (+). When this was not correct we looked among the other translation alternatives. When a correct translation was present this response was classified as *N-best* correct (N). We did not limit N, because we wanted to see whether our system was capable of generating a correct translation. When only incorrect output was generated, the respons was classified as *incorrect* (-). In some cases the sytem did return *no output* (0).

	Newspaper	Fiction	All
Correct	58.26%	57.39%	57.66%
N-best correct	7.34%	16.49%	13.58%
Incorrect	20.64%	14.99%	16.79%
No output	13.76%	11.13%	11.97%

Table 1: NP translation accuracy

Figure 2: NP translation accuracy



Our system produces several translation alternatives, ranked according to their weight. In 57.66% of the cases, the system provides a correct translation. In another 13.58% of the cases, the correct translation is among the translation alternatives, but did not receive rank 1. This implies that, by only changing the weighing mechanism, we could get a maximum of 71.24% correct NP translations.

There are slight differences between newspaper texts and fiction texts. Fiction seems a little easier to translate (at least when we include the N-best solutions)

The fact that these results are not higher is due to the coverage of the lexicon, as illustrated in table 2. Although some of these uncovered cases are solved by the decompounding module, most of them re-

[3]They were extracted from the section of the Spoken Dutch Corpus (CGN) which contains read-aloud fiction.

[4]With complex NPs, we mean NPs which consist of a number of elements amongst which a lower level NP

main unsolved and hence result in an incomplete translation or no translation at all. One of the test texts contained a high number of exclusively Belgian Dutch words, which are missing from our lexicon and which explains the low translation accuracy of that text (50.59% correct + 2.35% N-best).

Also, a number of cases where no output was generated can be explained due to bugs in our prototype system, which we expect to solve in future versions.

	Coverage
Newspaper	80.28%
Fiction	80.51%
Total	80.44%

Table 2: Coverage of the dictionary by token

4 Conclusions

Looking at these results and some of the reasons why the results are not better than they are, we can conclude that the approach adopted in our system works reasonably well for the translation of noun phrases.

As this is work in progress (initial version of the code, the dictionary and the weighing system), we expect our system to perform better in future versions.

NP translation is a substantial part of full sentence translation, but it is not safe to assume that because our approach works for noun phrase translation, it will work for full sentence translation.

In NP translation from Dutch to English, there are not many word order issues to solve. Translating VPs is already much more difficult (Way and Gough, 2003), and we want to translate full sentences. There are also no agreement issues to solve, which certainly would be the case when translating full sentences (like the agreement between the subject and the verb).

But still, as the approach seems promising, we plan to use the same strategy when implementing our full sentence translation system, although many issues will have to be solved during the process.

5 The Near and Not Too Distant Future

In the near future, we plan to implement a full sentence translation system. In order to do so, there are a number of tasks which need to be executed.

Amongst others, we need to ameliorate the Dutch language analysis tools, because when mistakes are made in the SL-analysis, this will most certainly lead to incorrect translations.

We also need to improve the English language analysis tools, with which we preprocess the TL-corpus, because the better the TL-corpus is preprocessed, the higher the probability is to retrieve useful information from the corpus.

Work on the bilingual dictionary is also not finished. We need to extend and ameliorate it, because when dictionary information is incorrect or missing, it becomes almost impossible to generate a correct translation. We also need to add some words which are typical for Belgian Dutch, as they tend to be left out of the dictionary.

As mentioned in section 2.1.3 we are using the PoS-tag to assign the correct lemma to a word. We may also make use of the further features of the PoS-tag to distinguish between the various meanings (plus associated translations) of a lemma.

For the experiments described in this paper we used a lexicon with very underspecified PoS (only main PoS (N,ADJ etc.), cf. section 2.2.1, without further features), we are in the process of adding some features in those cases where it might help translation (like the noun *was*). Further experiments will have to prove of this.

The TL-corpus needs to be preprocessed at the sentence level, analoguous to the way it is preprocessed now at the NP level.

The Head-based Template Retrieval mechanism needs to be enhances to get more information out of the corpus, and we need to implement the general Template Retrieval mechanism, which does not make us of heads.

We need to implement some extra language analysis tools (e.g. a subject detector) to enable us to enhance translation quality.

A number of frequency tables need to be created, derived from the TL-corpus, which will allow for a more accurate weighing system

We need to come up with a solution concerning prepositional phrase attachment and the translation of light verbs.

In all, there are numerous tasks still to be performed to get to a "good" translation system, but the general system outline is emerging in the process.

6 References

T. Badia, G. Boleda, M. Melero, A. Oliver. 2005. An n-gram approach to exploiting a monolingual corpus for Machine Translation. In *Proceedings EBMT Workshop 2005 - this volume*.

G. Booij and A. van Santen. 1995. *Morfologie. De woordstructuur van het Nederlands*. Amsterdam University Press, Amsterdam.

T. Brants. 2001. *TnT - A Statistical Part-of-Speech Tagger*. Published online at http://www.coli.uni-sb.de/thorsten/tnt.

R.D. Brown. 1996. Example-Based Machine Translation in the Pangloss System. *COLING 1996*, Copenhagen, Danmark. pp. 169-174.

M. Carl, P. Schmidt and J. Schütz. 2005. Reversible Template-based Shake & Bake Generation. In *Proceedings EBMT Workshop 2005 - this volume*.

P. Dirix. 2002a. *The METIS Project: Lexical Resources.* Internship Report, KULeuven.

P. Dirix, 2002b. *The METIS Project: Tag-mapping rules.* Paper, KULeuven.

P. Dirix, I. Schuurman, V. Vandeghinste. 2005. METIS: Example-Based Machine Translation Using Monolingual Corpora - System Description. In *Proceedings EBMT Workshop 2005 - this volume*.

Y. Dologlou, S. Markantonatou, G. Tambouratzis, O. Yannoutsou, A. Fourla, and N. Ioannou. 2003. Using Monolingual Corpora for Statistical Machine Translation: The METIS System. In *Proceedings of EAMT - CLAW 2003*, Dublin, pp. 61-68.

G. Grefenstette. 1999. The World Wide Web as a Resource for Example-Based Machine Translation Tasks. *ASLIB, Translating and the Computer 21*. London.

S. Markantonatou, S. Sofianopoulos, V. Spilioti, Y. Tambouratzkis, M. Vassiliou, O. Yannoutsou, N. Ioannou. 2005. Monolingual Corpus-based MT using Chunks. In *Proceedings EBMT Workshop 2005 - this volume*.

S. Nirenburg, S. Beale, and C. Domashnev. 1994. A Full-Text Experiment in Example-Based Machine Translation. In *Proceedings of the Int. Conf. on New Methods in Language Processing*. Manchester, UK. pp. 78-87.

N. Oostdijk, W. Goedertier, F. Van Eynde, L. Boves, J.P. Martens, M. Moortgat, and H. Baayen. 2002. Experiences from the Spoken Dutch Corpus Project. In *Proceedings of LREC 2002*, vol. 1, pp. 340-347.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40st Annual Meeting of the Association for Computational Linguistics*.

R. Piepenbrock. 2004. *CGN Lexion v.9.3*. Spoken Dutch Corpus.

S. Sato. 1993. Example-Based Translation of Technical Terms. *Proceedings of TMI 1993*. Kyoto, Japan.

V. Vandeghinste. 2002. Maximizing Lexical Coverage in Speech Recognition through Automated Compounding. In *Proceedings of LREC2004*. ELRA. Paris.

V. Vandeghinste and Y. Pan. 2004. Sentence Compression for Automated Subtitling. A Hybrid Approach. In *Proceedings of ACL-workshop on Text Summarization*. Barcelona.

V. Vandeghinste and E. Tjong Kim Sang. 2004. Using a Parallel Transcript/Subtitle Corpus for Sentence Compression. In *Proceedings of LREC2004*. ELRA. Paris.

F. Van Eynde. 2004. *Tagging and Lemmatisation for the Spoken Dutch Corpus*. Internal report.

T. Veale and A. Way. 1997. Gaijin: A Bootstrapping, Template-Driven Approach to Example-Based MT. In *Proc. of the 2nd Int. Conf. on Recent Advances in NLP*. Tzigov Chark, Bulgaria. pp. 239-244.

P. Vossen, L. Bloksma, and P. Boersma. 1999. *The Dutch WordNet*. University of Amsterdam.

L. Wall. 2004. *Perl 5.8.5*. http://www.perl.com.

A. Way and N. Gough. 2003. wEBMT: Developing and Validating an Example-Based Machine Translation System using the World Wide Web. *Computational Linguistics*, 29 (3), pp. 421-457.

List of Authors