Linguistic Knowledge and Complexity in an EBMT System Based on Translation Patterns

Kevin McTait

Department of Language Engineering
UMIST
Manchester, PO BOX 88, M60 1QD, UK
k.mctait@stud.umist.ac.uk

Abstract

An approach to Example-Based Machine Translation is presented which operates by extracting translation patterns from a bilingual corpus aligned at the level of the sentence. This is carried out using a language-neutral recursive machine-learning algorithm based on the principle of similar distributions of strings. The translation patterns extracted represent generalisations of sentences that are translations of each other and, to some extent, resemble transfer rules but with fewer constraints. The strings and variables, of which translations patterns are composed, are aligned in order to provide a more refined bilingual knowledge source, necessary for the recombination phase. A non-structural approach based on surface forms is error prone and liable to produce translation patterns that are false translations. Such errors are highlighted and solutions are proposed by the addition of external linguistic resources, namely morphological analysis and part-of-speech tagging. The amount of linguistic resources added has consequences for computational complexity and portability.

Introduction

A number of example-based machine translation (EBMT) systems operate by extracting and recombining translation patterns or templates from bilingual texts (Kaji et al, 1992; Güvenir & Cicekli, 1998; Brown 1999; Carl 1999; McTait & Trujillo, 1999). Translation patterns represent generalisations of sentences that are translations of each other in that various sequences of one or more words are replaced by variables, possibly with the alignments between word sequences and/or variables made explicit.

In this approach, which builds upon and improves that of McTait & Trujillo (1999), translation patterns are extracted from a bilingual corpus aligned at the level of the sentence. They are extracted by means of a language-neutral recursive machine-learning algorithm based on the principle of similar distributions of strings: source language (SL) and target language (TL) strings that co-occur in the same 2 (or more) sentence pairs of a bilingual corpus are likely to be translations of each other. The SL and TL strings that make up the translation patterns are aligned so that they provide not only sentential patterns of translation, but also a more refined bilingual knowledge source representing word / phrasal translations, necessary for the recombination phase, where TL translations are produced. Since the variables also represent strings, they too are aligned. Figure 1 is an example of a simple translation pattern indicating how a sentence in English containing *give…up*, may be translated by a sentence in French containing *abandonner*.

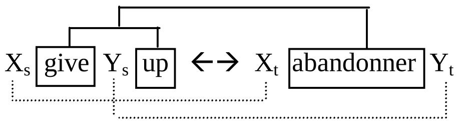


Figure 1: A Simple Translation Pattern

The translation pattern in figure 1 contains not only simple bijective or 1:1 alignments between text fragments or variables, but also non-bijective alignment types, such as the 2:1 alignment between *give…up* and *abandonner*. The ability to efficiently compute bijective and non-bijective alignments, as well as long distance dependencies, is conducive to more accurately describing translation phenomena.

Translation patterns are extracted, and the text fragments of which they are composed aligned, when the data is sparse, since strings only need to co-occur in a minimum of 2 sentence pairs. Furthermore, language-neutral techniques, such as cognates and bilingual lexical distribution, are used to align the strings or text fragments of which the patterns are composed.

The translation patterns extracted resemble, to some extent, transfer rules within a Rule-Based Machine Translation (RBMT) system, but with fewer constraints. A linear approach based on the distributions of surface forms within a corpus is liable to the extraction of translation patterns that are false translations. Solutions to this phenomenon are proposed by the addition of external linguistic knowledge sources such as morphological analysis and part-of-speech (POS) tagging. Figure 2 depicts the system architecture. The dotted lines indicate that the use of the knowledge sources is optional.

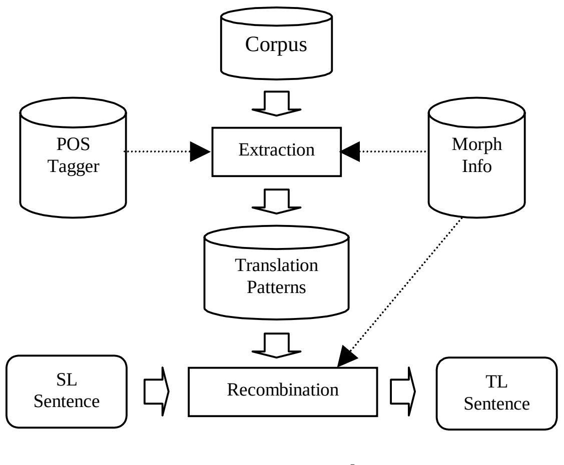


Figure 2: System Architecture

The addition of linguistic resources is intended to improve both accuracy and recall. However, their addition has consequences for portability and computational complexity. The more resources required, the less portable and the more complex the system.

This paper outlines the translation pattern extraction algorithm along with the corresponding recombination step where, given SL input sentences, TL translations are produced. The problems associated with a non-structural language-neutral approach are highlighted with solutions proposed involving the addition of external linguistic knowledge sources. Three variants of this approach are then ready for comparison, each with varying amounts of linguistic knowledge incorporated: i) the language-neutral approach based on surface forms, ii) the approach augmented to include morphological analysis and iii) the approach augmented to include both morphological analysis and POS tagging. Their performance is evaluated and compared, as is their complexity.

1 Existing Approaches

The concept of EBMT based on the extraction and recombination of translation patterns can be placed somewhere between 'traditional' linear EBMT - where the TL equivalents of overlapping partial exact matches of the SL input are computed dynamically and recombined (Nirenburg et al, 1993; Somers et al, 1994) - and systems that extract patterns that bear more resemblance to structural transfer rules (Kaji et al, 1992; Maruyama & Watanabe, 1992; Watanabe, 1995).

The extraction of translation patterns is typically reliant on the ability to generalise pairs of sentences in a corpus that are translations of each other. One method of classifying such systems is the method by which such generalisations are achieved and what constraints, if any, apply as a result. The broadest categorisation to make is those that use external linguistic resources and those that do not. However, distinctions may be made as to how and what external knowledge sources are used to generalise translation examples.

An approach that makes use of significant resources is Kaji et al. (1992). They use an English-Japanese bilingual dictionary and parsers to find correspondences at the phrase-structure level between two sentences that are translations of each other. These structures are then replaced by variables to produce translation patterns, similar to that in figure 1, except that the variables contain syntactic and possibly semantic constraints, due to the use of a thesaurus. The translation patterns described in Watanabe (1993) make use of a complex data structure involving a combination of lexical mappings and mappings between dependency structures, as is the case for the pattern-based context-free grammar rules found in Takeda (1996). Carl (1999) makes use of rich morphological analysis, enabling shallow parsing of the corpus to allow for the percolation of morpho-syntactic constraints in derivation trees. As Matsumoto & Kitamura (1995) show, it is also possible to generalise sentence pairs by replacing semantically similar words or dependency structures by means of a thesaurus.

Brown (1999) replaces certain strings denoting numbers, weekdays, country names etc. by an equivalence-class name, as well as including linguistic information such as number and gender. As Brown successfully shows, the level of abstraction or generalisation has consequences for coverage and accuracy.

Furuse & Iida (1992) describe three types of translation examples. The first type (2a) consists of literal examples, the second (2b) consists of a sentence pair with words replaced by variables and the third type (2c) are grammatical examples or context-sensitive rewrite rules, in effect, transfer rules of the kind found in traditional RBMT systems. The second type, despite its simplicity, most closely represents the translation patterns produced in this approach.

(2a) Sochira ni okeru ←→ We will send it to you
(2b) *X* o onegai shimasu ←→ may I speak to the *X′*
(2c) N_1 N_2 N_3 ←→ N_2 N_3 for N_1
N_1 = sanka / participation, N_2 = mōshikomi / application, N_3 = yōshi / form

Language-neutral techniques of extracting translation patterns are based on analogical reasoning (Güvenir & Cicekli, 1998; Malavazos & Piperidis, 2000) or inductive learning with genetic algorithms (Echizen-ya et al, 2000). The general principle applied is that given two sentence pairs in a corpus, the orthographically similar parts of the two SL sentences correspond to the orthographically similar parts of the two TL sentences. Similarly, the differing parts of the two SL sentences correspond to the differing parts of the TL sentences. The differences are replaced by variables to generalise the sentence pair. Highly inflective, or worse agglutinative, languages require an amount of linguistic pre-processing. In the case of Turkish, Güvenir & Cicekli (1998) use morphological analysis to alleviate orthographical differences.

Once generalisations of translation examples have been made, the SL and TL text fragments of which they are composed are generally aligned. In the case of the translation patterns of type (2b) in Furuse & Iida (1992) and also those of Carl (1999), there is no alignment problem to be solved since there are only *single* 1:1 or bijective mappings between strings or variables. In the case of Güvenir & Cicekli (1998), multiple 1:1 alignments in a translation template are solved by finding unambiguous or previously solved instances of the alignments in question from other translation templates. Few, if any, of the existing approaches cater for the fact that translation phenomena are not always bijective and that translation relations of a nature other than 1:1 exist i.e. the 2:1 relationship in figure 1.

The statistical models of Brown et al. (1993) cater for such relationships. However, they are computationally expensive, require large amounts of training data, rule out effective treatment of low-frequency words and are limited to unidirectional word-to-word translation models, thus ignoring the natural structuring of sentences into phrases. Later approaches (Dagan et al 1993; Wang & Waibel, 1998) address some, but not all, of these issues. The problem of aligning text fragments in translation examples is related to the bilingual vocabulary alignment problem. This includes words, terms and collocations (see Fung & McKeown (1997) and also Somers (1998) for an overview and bibliography). Generally, language-neutral or statistical vocabulary alignment techniques, based on distributions of word forms, are limited to computing 1:1 alignment patterns, again with large amounts of training data required.

2 Extraction & Recombination

2.1 Translation Patterns

A translation pattern can be defined formally as a 4-tuple $\{S, T, A_f, A_v\}$. S (T) represents a sequence of SL (TL) subsentential text fragments, separated by SL (TL) variables which represent subsentential text fragments (a subsentential text fragment is a series of one or more lexical items or tokens). In S, there can be any number p ($p>0$) of SL text fragments (F_p) with p, $p+1$, $p-1$ SL variables (V_p). In T, there can also be any number q ($q>0$) of TL text fragments (F_q) with q, $q+1$, $q-1$ TL variables (V_q). One possible configuration is depicted in figure 3.

$$F_1, V_1, F_2, V_2 \ldots F_p, V_p \leftrightarrow F_1, V_1, F_2, V_2 \ldots F_q, V_q$$

Figure 3: Possible Configuration of S and T

A_f represents the global alignment of text fragments between S and T, while A_v represents the global alignment of variables between S and T. The global alignment of the text fragments is represented as a set of local alignments $\{<A,B>_1,<A,B>_2 \ldots <A,B>_k\}$, where each local alignment is represented as a pair $<A,B>$. A (B) represents pointers to zero or more SL (TL) text fragments according to the local alignment patterns stipulated by the sequence comparison algorithm (section 2.2.3). The global alignment of the variables A_v is represented analogously.

2.2 Extracting Translation Patterns

The input to the translation-pattern extraction phase is a bilingual corpus aligned at the level of the sentence. The output is a set of translation patterns. The algorithm is language-neutral in nature and operates on the simple principles of string co-occurrence and frequency thresholds: possibly discontinuous pairs of SL and TL strings that co-occur in a minimum of 2 translation examples are likely to be translations of each other. Since strings are only required to co-occur a minimum of twice (frequency threshold), the algorithm is useful in instances of sparse data. However, the frequency threshold can be increased to improve the accuracy of the patterns (McTait & Trujillo, 1999). This section provides a highly simplified example, using the corpus in (3).

(3) The commission **gave** the plan **up** ←→
La commission **abandonna** le plan

Our government **gave** all laws **up** ←→
Notre gouvernement **abandonna** toutes les lois

2.2.1 Monolingual Phase

This stage is applied independently to the SL and TL sentences of the corpus. Lexical items (tokens) that occur in a minimum of 2 sentences are retrieved, together with a record of the sentences in which they were found: (4a) for the SL and (4b) for the TL.

(4a) *(gave)[1,2], (up)[1,2]*
(4b) *(abandonna) [1,2]*

The lexical items are allowed to combine to form longer word combinations (or *collocations*) constrained only by the sentences from which they were retrieved. The lexical items combine recursively to form a tree-like data structure of collocations. Each lexical item is tested to see if it can combine with the daughters of the root node and if so, recursively with each subsequent daughter, as long as there is an intersection of at least 2 sentence IDs (this enforces string co-occurrence in 2 or more sentences). The result is a tree of collocations of increasing length but decreasing frequency. The leaves become the most informative parts of the tree and are collected at the end of this phase. The longest provide more context and hence there is less chance of ambiguity.

As an example (figure 4), the SL lexical item *gave* is added to the root node (the integers denote the sentence IDs). The lexical item *up* is tested to see if it can combine with it since it is now a daughter of the root node. Since *gave* and *up* have an intersection of two sentence IDs, they are allowed to combine and form a new collocation node (*gave/up*). The TL lexical item is added to a separate tree and remains as in (4b) since there are no further TL lexical items with which it can combine.

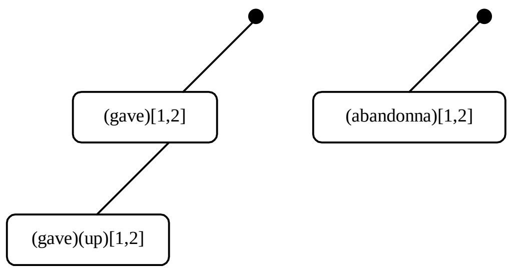


Figure 4: Collocation Trees

2.2.2 Bilingual Phase

SL and TL collocations are equated by simple co-occurrence criteria to form translation patterns: SL collocations that have exactly the same sentence IDs as TL collocations are considered to be translations of each other. This ensures that the patterns contain lexical items retrieved from the same sentences. The leaf-node collocations in figure 4 are equated to form the translation pattern (5). A translation pattern is formed from lexical items in 2 (or more) sentences, therefore its word order is determined from either of those sentences in the corpus. The discontinuities between the strings in (5) are represented as ellipses and denote variables.

(5) (…) gave (…) up ←→ (…) abandonna (…)

Translation patterns are not formed from inner leaves of the collocation trees since they would form patterns that are subsets of patterns from the leaf nodes. This would make them spurious. They also introduce ambiguity in that the effectiveness of EBMT lies in the retrieval of the longest possible matching segments. Furthermore, they would frequently be inaccurate. For example, if the node *gave[1,2]* were equated with *abandonna[1,2]* an incorrect pattern would be produced.

It is intuitive that if the text fragments - *F* in figure 3 - that make up the translation patterns are translations of each other, then the discontinuities or variables - *V* in figure 3 - that occur between them must also be translations. Since translation patterns are made up of lexical items from at least two translation examples, at least two *complement* translation patterns (6a) and (6b) are formed. They are created simply as the inverse of regular translation patterns and may contain lexical items that occur only once in the corpus.

The text fragments form the fundamental units of translation patterns such as (5) and (6a-b) and are not broken down further. The distinction into text fragments and variables, despite the formation of complement patterns, is a convenient representation for fragment alignment (2.2.3) and template matching (2.3).

(6a) The commission (…) the plan (…) ←→
La commission (…) le plan

(6b) Our government (…) all laws (…) ←→
Notre gouvernement (…) toutes les lois

2.2.3 Alignment of Text Fragments and Variables

Aligning the text fragments and variables of which translation patterns are composed produces translation patterns that are flexible enough for the recombination phase. In so doing, a more refined bilingual knowledge source (bilingual lexicon or "phrasicon") is produced. Aligning text fragments and variables is analogous to aligning words, phrases, terms or even sentences as in conventional bilingual alignment and involves similar problems.

Given that a translation pattern contains a SL and a TL sequence of one or more subsentential text fragments (separated by variables which also represent a separate series of subsentential text fragments), the problem may be viewed as bilingual sequence alignment or computing the optimal or most probable alignment between two sequences of text fragments in two languages. The two sequences of variables are aligned analogously by considering the text fragments that the variables represent, as defined by the corpus, as a separate series of text fragments. The solution to the alignment problem involves a sequence-comparison algorithm and a bilingual similarity (distance) metric.

The bilingual similarity metric is language-neutral in nature and is a combined score based on bilingual lexical distribution of the text fragments (BLD) and the number of cognates the text fragments share (7). The bilingual lexical distribution score (a real value between zero and 1) is computed by Dice's co-efficient (Dice, 1945) and cognates are determined by computing the Levenshtein Distance (Levenshtein, 1966) between SL and TL strings. The Levenshtein Distance is normalised over the maximum distance between the two strings, returning a similarity score or probability that the two strings are cognates. If the similarity score is above an experimentally determined threshold, the two strings are considered to be cognates. Bilingual lexical distribution suffers from well-known problems such as data sparseness, the identification of translingual collocates as opposed to true translations and the varying distributions of morphological variants of words. The inclusion of cognates into the distance metric addresses, to some extent, the problem of low frequency words and provides an alternative score.

$$(7)\qquad \frac{BLD+|Cognates|}{1+|Cognates|}$$

The alignment of closed class words is particularly problematic. They are of such high frequency that they are unable to be aligned by lexical distribution. They are also not subject to cognate matching. Solutions to this problem include the non-alignment of text fragments composed uniquely of closed class words. An additional method is to 'fill in' the variable positions between text fragments where the variable is uniquely composed of closed class words. For instance *independent states (…) former Soviet Union* becomes *independent states of the former Soviet Union.* However, neither provides a complete solution.

The sequence-comparison algorithm needs to be able to compute alignments denoting translation relations of a more complex variety than simple 1:1 relationships, i.e 2:1, 1:0, etc. It must also compute alignments between adjacent and non-adjacent text fragments (long-distance dependencies) to allow for structural divergences between languages (figure 5). Finally, it needs to execute in a practicable asymptotic running time, therefore excluding an exhaustive search algorithm.

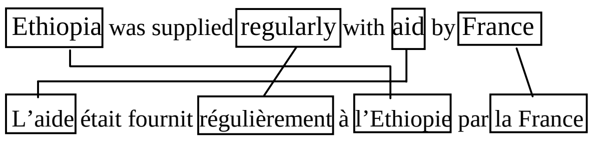

Ethiopia was supplied regularly with aid by France
L'aide était fournit régulièrement à l'Ethiopie par la France

Figure 5. Pattern with Non-Adjacent Alignments[1]

One solution to the problem of sequence alignment is the Dynamic Programming (DP) framework (8). While the problem of sentence alignment is suited to the DP algorithm in that the order of sentences between languages is often similar (Gale & Church, 1993), subsentential fragment alignment must include a search for non-adjacent alignments or long-distance dependencies – a crucial task which DP is inherently unable to perform.

$$(8)\quad D(i,j)=\min\begin{cases} D(i,j-1)+d(0,y_j) \\ D(i-1,j)+d(x_i,0) \\ D(i-1,j-1)+d(x_i,y_j) \\ D(i-1,j-2)+d(x_i,\{y_{j-1},y_j\}) \\ D(i-2,j-1)+d(\{x_{i-1},x_i\},y_j) \\ D(i-2,j-2)+d(x_i,y_{j-1})+d(x_{i-1},y_j) \\ D(i-1,j-3)+d(x_i,\{y_{j-2},y_{j-1},y_j\}) \\ D(i-3,j-1)+d(\{x_{i-2},x_{i-1},x_i\},y_j) \end{cases}$$

An algorithm is proposed that performs the alignment in two passes: the first pass, for which DP is ideally suited, takes into account substitutions (1:1), insertions (0:1), deletions (1:0), compression (2:1, 3:1), expansion (1:2, 1:3) and *swaps* of adjacent fragments (Lowrance & Wagner, 1975), assuming all local alignments are adjacent. The second-pass algorithm computes the (possibly empty) set of non-adjacent alignments of a 1:1 nature. If any are found that improve the initial global alignment computed by DP, they are recorded and removed from the two sequences, while the remaining text fragments are re-fed into the DP algorithm. The final global alignment is then a concatenation of the non-adjacent alignments and the results of the second application of the DP algorithm. The same bilingual similarity metric is used for both passes.

The second-pass algorithm is summarised as follows: given two sequences x and y of lengths m and n respectively, each element x_i for $1 \le i \le m$ is compared

[1] The text between the boxed text would appear in the translation patterns as variables. It is included here to make sense of the example.

with each element y_j for $1 \leq j \leq n$. For each potential alignment $<x_i,y_j>$, the similarity score $Score(x_i,y_j)$ is computed. If $Score(x_i,y_j)$ satisfies two conditions, the text fragments x_i and y_j along with $Score(x_i,y_j)$ are recorded as a triple and added to the list of candidate non-adjacent alignments (figure 6). The first condition states that the alignment must be high-scoring (above a threshold) and the second states that the alignment must be non-adjacent i.e. the absolute difference between i and j must be greater than 1.

```
for $i = 1$ to $m$
    for $j = 1$ to $n$
        if $Score(x_i,y_j) > Threshold \,\&\, abs(i-j) > 1$
            add $\{x_i, y_j, Score(x_i,y_j)\}$ to list of candidate
            non-adjacent alignments
```

Figure 6. Finding Candidate Non-Adjacent Alignments

Once a list of triples representing candidate non-adjacent alignments has been collected $\{(\alpha,\beta, Score(\alpha,\beta)),...\}$, for each triple, three conditions are applied before the alignment is declared a valid non-adjacent alignment. First, if the SL (TL) text fragment α (β) of the proposed alignment $<\alpha,\beta>$ has been used elsewhere in a *non*-adjacent alignment, it cannot be used to form a subsequent one. Second, if the alignment $<\alpha,\beta>$ is a subset of an alignment $<A,B>$ that has previously been computed by DP (where A and B represent a series of SL and TL text fragments respectively and where $\alpha \in A$ and $\beta \in B$), they cannot be considered further. The reason for this is simply that the DP algorithm has found that an alignment other than a 1:1 type is more probable, thus the proposed alignment is not likely to improve the likelihood of the final global alignment.

$$
\text{(9)} \quad \begin{array}{l} Score(\alpha,\beta) > Score\left(\langle A,B\rangle_p\right) \& \\ Score(\alpha,\beta) > Score\left(\langle A,B\rangle_q\right), \\ \alpha \in A_p,\ \beta \in B_q, 1 \leq p \leq k, 1 \leq q \leq k,\ p \neq q \end{array}
$$

Finally, the score of the proposed non-adjacent alignment $<\alpha,\beta>$ is tested to see whether it is greater than both the two individual scores of the two alignments computed by DP of which α and β are a member. In more detail, each adjacent alignment computed by DP is represented as a pair $<A,B>$, where A and B represent a series of SL and TL fragments according to the alignment patterns stipulated by the DP equation. The initial global alignment computed solely by DP is a set containing one or more of these alignments, $\{<A,B>_1,\ldots,<A,B>_k\}$. The proposed non-adjacent alignment can also be represented by a pair $<\alpha,\beta>$ where α and β represent one SL and TL text fragment respectively. This condition is summarised in (9)[2], which stipulates that the score of the non-adjacent alignment must be better than both of the two adjacent alignments which subsume it, which means that the addition of the proposed non-adjacent alignment would *improve* the overall global alignment.

To illustrate the 2nd pass algorithm, the translation pattern in figure 7 is first aligned using the DP algorithm. The alignment of the text fragments (boxed text) is incorrect. After the application of the 2nd pass algorithm, a non-adjacent alignment (*project-projet*) is computed. Subsequent application of the DP algorithm produces the correspondences *maternal-maternelle*, and *neonatal-néonatals*.

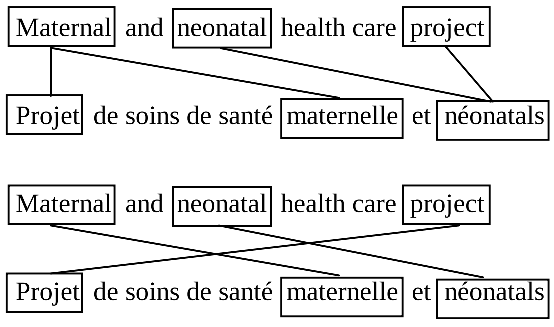


Figure 7: Before and After 2nd Pass Algorithm

Only non-adjacent alignments of a 1:1 nature are considered to ensure that the alignment process executes in a practical asymptotic running time. The second-pass algorithm therefore runs with a worst-case complexity of $\mathcal{O}(mn)$, as does DP.

2.3 Recombination

In the case of EBMT systems where examples are stored as tree structures with the correspondences between the fragments explicitly labelled, the recombination problem is a matter of tree traversal (Sato, 1995; Watanabe, 1995). In this approach, the input to the recombination phase is an SL input sentence and a set of translation patterns with their constituent text fragments and variables aligned. The output is one or more TL translations which can be ranked best first according to confidence.

Given an SL input sentence, translation patterns that share lexical items with the SL input and partially (or fully) cover it are retrieved in a pattern-matching process. From these, the patterns whose SL side cover the SL input to the greatest extent (longest cover) are selected. They are termed *base patterns*, as they provide sentential context in the translation process. It is intuitive that the greater the extent of the cover of the base pattern, the more context and hence the less ambiguity and complexity in the translation process. If the SL side of the base pattern does not fully cover the SL input, any unmatched parts of the SL input are bound to the variables on the SL side of the base pattern. Translation patterns from the remaining set, containing SL text fragments that match the string value of the instantiated variables on the SL side of the base pattern, are retrieved. Since the text fragments and variables in translation patterns are aligned, their TL equivalents are easily retrieved and bound to the relevant TL variables in the TL side of the base pattern.

The following is a simple example: given the SL input in (10), suppose the longest covering base pattern is (11). To complete the match between (10) and the SL side of (11), a translation pattern containing the text fragment *Ethiopia* is required (12). The TL translation (13) is generated by recombining the text fragments: *Ethiopia* and *Etiopía* are aligned in (12) as are the variables in the base pattern (11). Since *Ethiopia* and *Etiopía* are aligned on a 1:1 basis and so are the variables in the base pattern (11), the TL text fragment *Etiopía* is bound to the variable on the TL side of (11) to produce (13). It is clearer to think of the algorithm proceeding in terms of matching

[2] The notation A_p is shorthand for "the A in $<A,B>_p$".

successive alignments of the variables in the base pattern, rather than successive variables on the SL side of the base pattern. This is because it is not only text fragments that are matched but also the alignment type. As pointed out in 2.2.3, alignments may be non-bijective.

(10) AIDS control programme for Ethiopia

(11) AIDS control programme for (…) ←→ programa contra el SIDA para (…)

(12) (…) Ethiopia ←→ (…) Etiopía

(13) programa contra el SIDA para Etiopía

2.3.1 Translation Confidence

On account of translation ambiguity, there may be more than one translation per SL input. The list of competing translations is ranked best-first according to a score of translation confidence (14). The score is based on three components. The first is a score based on a bigram model of the TL formed from the TL side of the corpus (*BG*). The model is used as a test of the “grammaticality” of the TL sentence. The second is a score that checks the extent of the overlap of the inserted TL text fragments (*OV*) in an attempt to reduce boundary friction. The third is a score reflecting the bilingual distance between translated text fragments (*BD*). Finally a penalty *p* is applied if any TL variables in the base pattern remain unfilled.

$$(14)\quad \left(\omega_1(BG)+\omega_2(OV)+\omega_3(BD)\right)(1-p)$$

The TL bigram model is computed from the TL side of the corpus. It is easily improved by adding more TL text. The strength of association between the words in the bigrams is computed by Dice’s co-efficient. For each TL sentence produced of length *n*, *BG* in (14) is computed by taking the average score of each bigram within it, or each word pair $<w_i, w_{i+1}>$. This is summarised in (15): For each bigram in the TL sentence, its frequency in the TL corpus is divided by the sum of the frequencies of either word in the corpus and multiplied by 2. This score is added to a running total. The total is then divided by the number of bigrams in the sentence or *n-1*.

$$(15)\quad \frac{\sum_{i=1}^{n-1} 2\left(\frac{|w_i, w_{i+1}|}{|w_i|+|w_{i+1}|}\right)}{n-1}$$

The overlap term (*OV*) in (14) makes use of the fact that the text fragments have been extracted from real texts and so there is information about the contexts in which the fragments are known to have occurred. The idea is borrowed from Somers et al. (1994) who use the image of hooks being attached before and after the text fragment, indicating the words (and POS tags) that can occur before and after it. A window of five words before and after the text fragment bound to each TL variable of the base pattern is considered. If the same sequence of 5 words or less preceding or subsequent to the text fragment is found in the TL side of the corpus, a score reflecting the extent of the overlap between the two sequences is returned. Where multiple match sequences are found, the maximum overlap value is returned. As an example, when the TL text fragment *Etiopía* is inserted into the TL base variable in (11), the word sequence [*programa, contra, el, SIDA, para*] is looked for in the corpus preceding *Etiopía*. Starting from *Etiopía*, the extent of the overlap is determined. If only *para* appears before *Etiopía* in the corpus, the overlap score returned would be 1/5 = 0.2. If there were a match sequence subsequent to *Etiopía*, the average of the previous and the subsequent overlap would be returned. The formula is given in (16): For each base pattern with *n (n>0)* TL variable positions, the overlap score for each variable V_i, is the average of the sum of the previous and subsequent overlaps. The final overlap score is the average overlap score for each V_i. If a V_i occurs at a sentence boundary, only the previous or subsequent overlap score is considered.

$$(16)\quad \frac{\sum_{i=1}^{n}\left(\frac{PreviousOverlap(V_i)}{5}+\frac{SubsequentOverlap(V_i)}{5}\right)\Big/2}{n}$$

For each variable alignment, the bilingual distance score is calculated between the SL text fragments bound to the SL variables of the base pattern and their TL equivalents using equation (7) in the alignment process (section 2.2.3). The average score for each alignment is returned to become *BD* in equation (14).

Finally a penalty *p* is applied if there are any variables on the TL side of the base pattern that have not been instantiated on account of *partial* matches between the SL input and the SL side of the base pattern. Partial matches occur when not all variables on the SL side of the base pattern can be matched by SL text fragments in the remaining set of translation patterns. For example, if there were no translation pattern containing the text fragment *Ethiopia*, the TL variable would not be instantiated, resulting in the partial translation: *programa contra el SIDA para (…)*. The penalty *p* is simply the number of unfilled variables on the TL side of the base pattern over the total number of variables on the TL side of the base pattern. In the case where *p = 1* i.e. when all TL variables remain unfilled, *p* is scaled by 0.9 to avoid *(1-p)* becoming zero.

The three terms *BG*, *OV* and *BD* return a real value between 0 and 1.0. To ensure that the sum of the three terms remains between 0 and 1.0, $\omega_1 + \omega_2 + \omega_3 = 1.0$. In the experiments undertaken, $\omega_1 = \omega_2 = \omega_2 = 1/3$, thus providing equal weighting to all terms. The term containing the penalty *(1-p)*, returns a value between 0.1 and 1.0, ensuring that the entire score remains a real value no greater than 1.0.

3 Adding Linguistic Knowledge

An approach based on the distributions of surface forms is error prone and liable to the production of translation patterns that are false translations. Linguistic knowledge, in the form of morphological analysis and POS tagging, is added in an attempt to increase the accuracy of the translation patterns and coverage. This results in three variants of this approach to EBMT: The first variant, outlined in section 2, is based on surface forms and requires no significant external linguistic knowledge, apart from the sentence-aligned parallel corpus. However, in order to remove obvious errors a very small amount of (optional) minor linguistic information is added in the form of stop lists of closed-class words and information for tokenisation, which compromises, to a certain extent, language-neutrality and portability. We believe this information to be obtainable in a few person-hours. The second variant is augmented with morphological information in the form of a set of two-level morphological rules (Koskenniemi, 1983), a list of lemmas and a finite state transducer (PC-Kimmo). The corpus is lemmatised so that translation patterns composed of lemmas and translation patterns on the morphological level are extracted. Furthermore, clitics such as the French *des*, *du* etc are expanded in an effort to increase coverage and reduce boundary friction. TL surface forms are generated by reversing the direction of the transducer, combing TL lemmas with TL tags. Since no POS information nor word grammar is used, the surface forms are validated by means of a list of TL surface forms (a spelling dictionary in this case). The third variant is extended even further to include POS annotation, using TreeTagger (Schmid, 1994).

As more linguistic knowledge sources are added to a system, portability generally decreases. In this case, the linguistic knowledge added is deliberately kept to a minimum and to the sort of knowledge sources that are reasonably widely available, at least for European languages. Even if they are not, it is not too difficult nor time-consuming to produce a list of stems and a set of two-level spelling rules. POS taggers such as TreeTagger and Brill's tagger (Brill, 1992) are also easily trained for new languages if they do not already exist.

3.1 Morphological Analysis

The principle source of errors is the phenomenon where SL words that are orthographically identical (homographs, polysemes, orthographically identical variants of the same lemma) have different TL equivalents. This can be corrected, to a certain extent, by the addition of morphological analysis and POS tagging to distinguish them. Consider the example corpus in (17) where the morphological variants of *give* are orthographically identical, but their TL equivalents are not. This results in a translation pattern with an empty TL side and is consequently a false translation (18).

(17) He **gave the** plans **up** ←→
Il abandonna les projets
They **gave the** suggestion **up** ←→
Ils abandonnèrent la suggestion

(18) (…) gave the (…) up ←→

Lemmatising the corpus (19) produces a more accurate, if more general, translation pattern (20) composed of lemmas. Furthermore, abstracting all morphological variants to their root form increases the chance of string co-occurrence and collocation formation, resulting in a greater number of translation patterns with the potential to increase coverage.

(19) he **give the** plan **up** ←→
il **abandonner le** projet
they **give the** suggestion **up** ←→
ils **abandonner le** suggestion

(20) (…) give the (…) up ←→
(…) abandonner le (…)

3.2 Part-of-Speech Tagging

A further case is SL homographs that are of different parts-of-speech and are consequently different 'words' relating to different concepts. It is very likely that in such cases each individual homograph will have a different TL translation. Consider *wave* as a noun and a verb in the corpus (21) and the resulting erroneous translation pattern (22).

(21) She **waves** goodbye ←→
Elle agite la main et dit au revoir
The **waves** are enormous ←→
Les vagues sont énormes

(22) (…) waves (…) ←→

Lemmatisation is not a solution since both forms of *wave* abstract to the (orthographically) same root. The solution lies in POS tagging the corpus and modifying the string co-occurrence algorithm for extracting patterns to include the POS tag as a feature. In such a case, the two forms of *wave* would not 'co-occur' to form a collocation, since their respective POS tags would distinguish one form of *wave* from the other and consider them as different words.

3.2 Semantic Tagging

The solution to the false translation problem in this approach appears to lie in being able to distinguish different 'words' from each other, on criteria than just their orthography. Distinguishing different 'words' i.e. strings of characters that relate to different concepts is not entirely possible by morphological analysis and POS tagging alone. Consider the corpus in (23) where the two senses of the polysemous verb *mark* occur with identical orthographies and POS tag, but different translations, to produce (24). While the two meanings of *mark* may or not have been historically related, they are clearly separate and consequently have different translations. The solution appears to lie in effective semantic tagging, if it were feasible, and including that tag as part of a feature structure in the co-occurrence algorithm. Again this would enable the different semantic forms of a verb such as *mark* to be distinguished and prevented from forming collocations.

(23) She **marked the** papers ←→
Elle corrigea les examens
He **marked the** table ←→
Il tâcha la table

(24) (…) marked the (…) ←→

3.3 Other Sources of Error

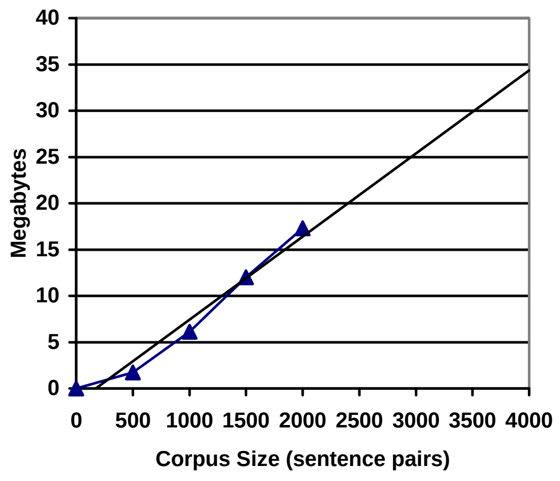
The fact that translation patterns are formed using an algorithm based on the distribution of strings in a corpus means that there are fewer constraints than in a traditional rule-based system or system that generalizes translation examples by morphological, syntactic or even semantic means. The patterns do not contain any restrictions on the morphology, number, gender or syntactic category of the text fragment that binds to the variables in the base pattern. Furthermore, adding external knowledge sources is no guarantee of increasing the accuracy (or coverage) of a system. Taggers, parsers and other analysers are prone to a certain degree of error themselves.

4 Complexity Issues

4.1 Translation Pattern Extraction

The translation pattern extraction algorithm, in particular the construction of the collocation trees (section 2.2.1) is a notable source of complexity. As each lexical item occurring in two or more sentences is added, the tree increases in size. Since each lexical item may be added to the daughter(s) of the root node and any subsequent node, the trees grow to become very much larger than the corpus itself. On account of their high frequency, closed class words are particularly problematic in that they dramatically increase the size of the trees.

In this instance, space becomes an issue. As the size of the corpus increases, the collocation trees become very large, in effect, placing a restriction on extensibility. Graph 1 shows how the size of the collocation trees for a corpus of French (measured in megabytes) increases in relation to the size of the corpus (measured in sentence pairs). The dark line indicates the extrapolation of the data. Although the relationship appears to be linear, the trees take a substantial portion of memory and the limit on corpus size is dictated by the memory of a particular machine (unless hard-drive space is efficiently utilised). The size of the collocation trees is the greatest obstacle to scaling up this approach. The solution lies in reducing the space required by the trees, in particular, reducing the number of levels. This may be achieved by adding the longest substrings that appear in 2 or more sentence pairs to the trees instead of individual word tokens.



Graph 1: Corpus Size vs. Size of Collocation Trees

The polynomial two-pass alignment algorithm outlined in 2.2.3 is also a notable source of complexity. In fact it is the part of the extraction algorithm that incurs the largest time processing overhead The well-known DP algorithm consists of two embedded loops and is clearly of a time and space complexity of $\mathcal{O}(mn)$, where m and n are the lengths of the two sequences to be aligned. DP is practical in terms of time and space, but not fast. In this application, the values of m and n are rarely large.

The grain-size of the local alignment types is also pertinent. Further text fragment alignment patterns such as 2:3, 3:2 and even 2:4 could easily be included as terms in the DP equation. More terms entails a larger processing overhead, but as grain-size increases, the problem of long distance dependencies becomes less of an issue as their likelihood diminishes. Moreover, the granularity of the alignment types involves the classic trade-off between accuracy and flexibility. While matches with longer fragments reduce ambiguity, flexibility suffers since there is a lower probability of a match. Matches with shorter segments involve a greater amount of translation ambiguity, passage and boundary friction (Nirenburg et al. 1993: 48). Fine as opposed to coarse-grained local alignment types have been favoured in this approach in order to maintain flexibility and recall.

4.2 Recombination

The core recombination algorithm is an area where the amount of entropy incurred by translation ambiguity and the lack of constraints in the translation patterns produce the most notable source of computational complexity. Since any one SL fragment can have more than one TL equivalent and there is no information present to favour one fragment over another, all possible TL fragments are considered. This results in (possibly) numerous translations per base pattern which are ranked by the translation confidence score (14).

This complexity is increased if recursive matches are incorporated. In the examples (10)-(13), the string values bound to the variables on the SL side of the base pattern are matched against *single* SL text fragments in the remaining translation patterns (direct matches). If no direct matches are found, a recursive matching algorithm is invoked. The algorithm matches a text fragment by attempting to recursively match successively shorter portions of it against the SL text fragments in the translation patterns. The TL equivalents are concatenated naively according to the order of the matches with the SL fragments. Increased complexity comes about since each SL fragment that forms a subpart of the entire match can have more than one TL equivalent, each of which has to be taken into account.

The addition of knowledge sources has consequence for the complexity of the recombination phase. The second variant of this approach, which is augmented to included morphological analysis includes the extraction of not only translation patterns composed of lemmas, but translation patterns composed of morphological tags or *morphological patterns*. They are analogous to regular translation patterns (see 2.1) except that they are composed of morphological tags instead of lexical items.

Morphological patterns are formed trivially by the replacement of lemmas in regular translation patterns extracted with the corresponding morphological tags computed during lemmatisation of the corpus. The tags represent (inflectional) morphological change from the root lexical form. The alignment between the sequences of fragments (composed of tags) and variables in a morphological pattern is carried out trivially by transposing the alignments from the regular translation pattern from which it was formed.

(25) The telephones worked ←→
Les téléphones fonctionnaient

The telephone failed ←→
Le téléphone échouait

(26) **The telephone**+*s* work+*ed* ←→
Le+*s* **téléphone**+*s* fonctionner+*aient*

The telephone fail+*ed* ↔
Le téléphone échouer+*ait*

The corpus (25) is lemmatised to form (26), from which translation pattern (27a) is extracted (among others). By replacing the lemmas with the corresponding morphological tags from (26), the morphological patterns (27b) and (27c) are formed. The empty parentheses denote cases of no morphological change from the root lexical form. The fragments and variables in (27b) and (27c) are aligned according to alignment of the fragments in (27a).

(27a) The telephone (…) ↔ Le téléphone (…)
(27b) [] +s (…) ↔ +s +s (…)
(27c) [] [] (…) ↔ [] [] (…)

The recombination algorithm is updated to operate with translation patterns composed of lemmas and morphological patterns. First, the SL input is lemmatised and recombination proceeds in exactly the same manner as described in 2.3, except that matching takes place on the lexical level and the result is a set of one or more sequences of TL lemmas. Morphological patterns are retrieved and recombined analogously to cover the sequence of SL morphological tags computed when the SL input is lemmatised. This results in a set of one or more sequences of TL morphological tags. TL sentences are produced by laying the sequences of TL lemmas and tags, generating sequences of surface forms. As an example, the sequence of lemmas (28a) is layered with the sequence of tags (28b) to produce (28c). It is clear that recombination that takes place on more than one level (here the lexical and morphological) increases the processing overhead.

(28a) <Programme, contre, le, maladie, diarrhéique>
(28b) <[], [], +s, +s, +s>
(28c) Programme contre les maladies diarrhéiques

One further complexity issue is ambiguity of analysis. When a surface form is lemmatised using just a list of lemmas and a set of two-level spelling rules, there may be ambiguity about the lemma to which it belongs and consequently ambiguity of the tag. The layering algorithm takes into account all alternatives thus adding a further parameter, increasing the complexity of the algorithm. The third variant of this approach takes this factor into account. The TreeTagger, which incorporates a lemmatiser, is used to analyse the corpus and resolve ambiguous analyses. This has the effect of removing this extra parameter and reducing the complexity of the layering algorithm.

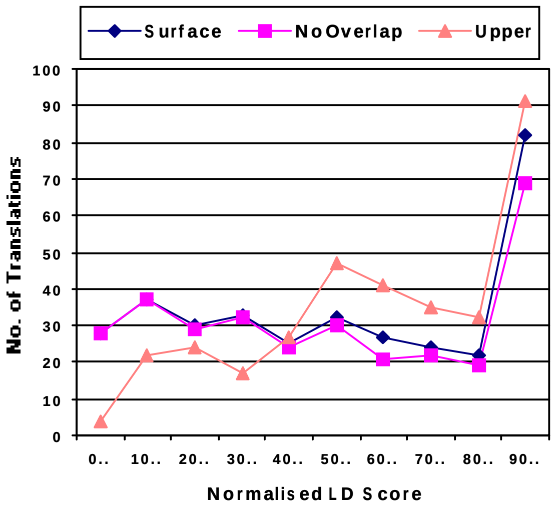
5 Experiments

Using the first of the three variants - the one based on surface forms - translation patterns were extracted from a 3,000-sentence-pair sample of the World Health Organisation AFI corpus of 4,858 French and English titles.[3] From the remaining set of 1,858 sentence pairs, 1,000 were randomly selected and used as a test set of unseen SL input. 36 SL sentences in the test set occurred in the training set. Furthermore, 728 SL sentences in the test set were entirely composed of lexical items found in the training set, leaving 272 that contained one or more unseen words. 10,767 patterns were extracted and the most frequent values of p and q (figure 3) were 2 or 3.

$$(29)\quad 1-\frac{LD(TR,RT)}{Longest(TR,RT)}$$

For each SL input, one ore more translation solutions were produced. The best translation among this set (*TR*) was selected according to the translation confidence score (14). The correctness of the translation was determined automatically by comparing it with the reference translation given in the corpus (*RT*). The Levenshtein Distance (*LD*) between the translation (*TR*) and the reference translation (*RT*) was normalised to account for the maximum distance between the two strings (maximum string length), returning a "percentage of similarity" score (29).

Graph 2 shows the distributions of the results. Only full translations were considered. The line depicted by 'Surface' indicates the results for the test set of 1,000 SL sentences. 340 SL sentences were successfully translated. Recall - the number of SL sentences translated divided by the number of sentences in the test set - stands at 340/1000 = 34%. Maximal recall is obtained by dividing 340 by 728 (47%) i.e. the number of SL sentences in the test set uniquely composed of lexical items found in the training set. 82 of the 340 translations (24%) were 90-100% accurate. Moreover, 73 translations (21.5%) were 100% correct. Accurate translations were largely produced by base patterns whose SL side substantially covered the SL input. Moreover, such patterns, of which (11) is typical, represent extremely frequent phenomena in the corpus.

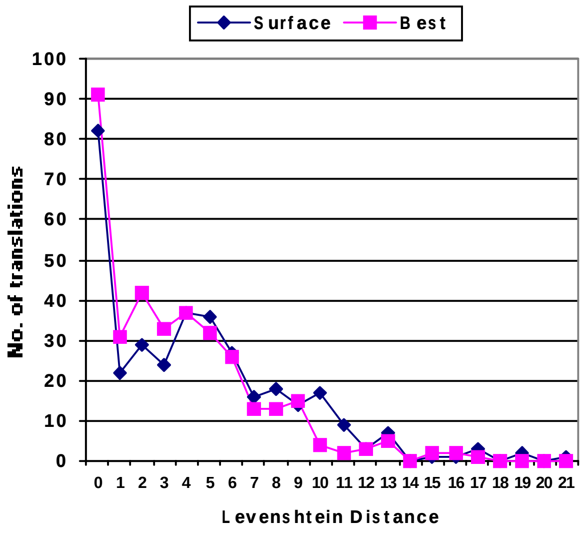


Graph 2: Distribution of Results

The line depicted by 'NoOverlap' shows the results for the test set with the 36 sentences which occur in the training set removed. Although no direct matches between the SL inputs and the SL sentences in the training set are sought (only the EBMT mechanism is evaluated), 10 of the 100% accurate translations are no longer present and there are less high-scoring translations generally. The line depicted by 'Upper' represents the theoretical upper bound of the system. For each SL input, the best possible translation solution – the one which returns the highest percentage of similarity score (29) among the set of all translation solutions – is selected, irrespective of its

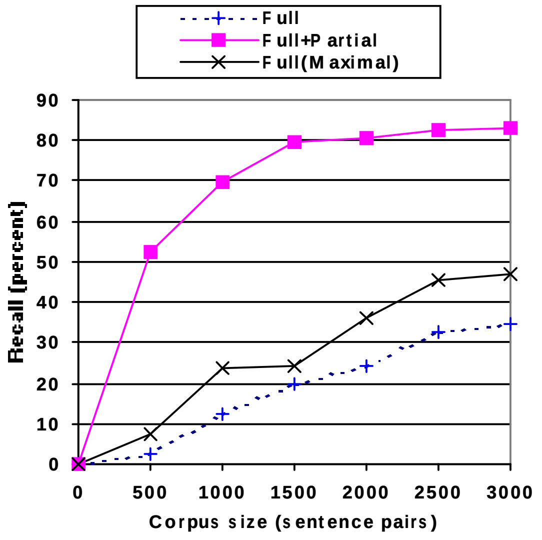
[3] http://www.who.int/pll/cat/cat_resources.html

translation confidence score (14). This line shows that less low-scoring translations and more high-scoring translations are produced. The upper bound is not reached due to the inefficacy of the translation confidence score i.e. it does not always assign the best translation with the highest score.



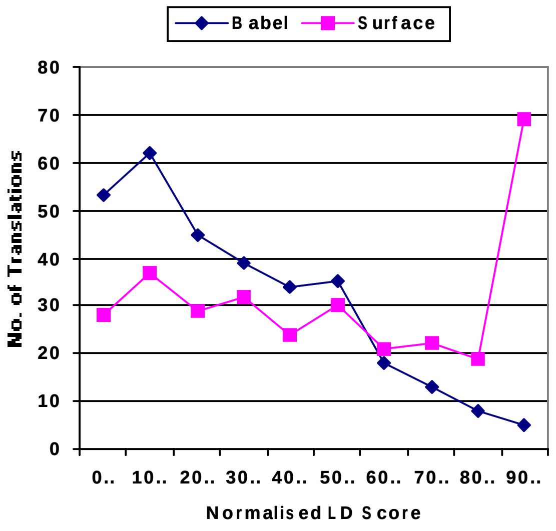
Graph 3: Word Error Rate

Graph 3 shows, for the same test set of 1,000 sentences, the distribution of word errors. For each translation solution, the error rate is calculated as the plain Levenshtein distance, as opposed to normalised Levenshtein Distance (29). In this case, the Levenshtein Distance represents the number of insertions, deletions or substitutions required to transform the translation string into the reference translation given in the corpus. The line 'Surface' represents the errors of the translation solutions selected according to (14) and the line 'Upper' represents the errors of the upper bound translations. In each case, the number of translations with zero errors is equal to the number of translations that were 100% accurate .



Graph 4: Recall vs. Corpus Size

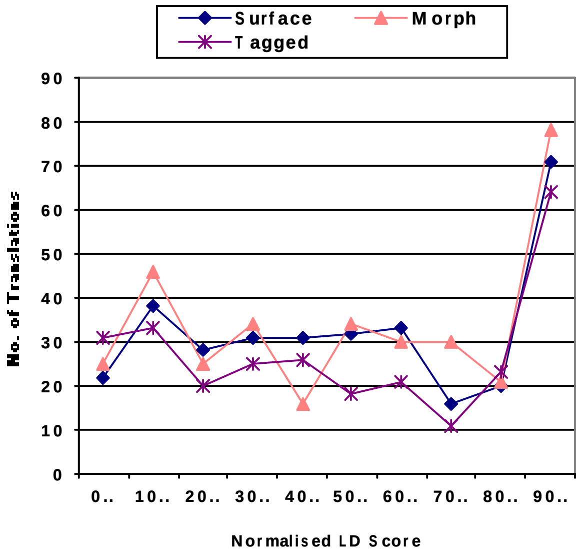
Graph 4 shows increasing rates of recall as corpus size, measured in sentence pairs, increases. A test set of 500 unseen SL input sentences was used in each case. The line 'Full' represents recall calculated by dividing the number of SL sentences fully translated by the number of SL sentences in the test set. The line 'Full+Partial' represents not only SL sentences that were fully translated, but includes those that were only partially matched against the translation patterns. The line 'Full(Maximal)' represents recall calculated by dividing the number of SL sentences fully translated by the number of SL sentences in the test set that are uniquely composed of lexical items found in the training set. The rates of recall rise fairly slowly if only full matches are considered, indicating the need for larger training sets. However, high rates of recall are rapidly achieved if incomplete or partial translations are considered. Including partial translations results in many more low-scoring than high-scoring translations.



Graph 5: Comparison with Babel Fish

In order to compare this approach with a commercial MT system, the same test set of 1,000 SL input sentences was translated using the *BabelFish/Systran* translation facility on `http://www.alta-vista.com`. *BabelFish* is robust enough to achieve 100% recall for the test set of SL input sentences, far outperforming this approach. However, in order to compare the rates of *precision*, graph 5 only depicts the distributions of results for the sentences that this approach was able to translate fully. To make the test fairer, the 36 SL sentences in the test set that occur in the training set were removed. The graph clearly shows that for that test set, this approach outperformed *BabelFish*. This is to be expected since this approach is tuned to the text type.

In order to test the hypothesis that adding linguistic knowledge improves accuracy and coverage (section 3), a comparison of the distributions of results of each of the three variants of this approach is shown in graph 6. For each variant, translation patterns were extracted from the same 2,500-sentence-pair sample in order to make a fair test. Complexity issues (as outlined in section 4) prevented the use of a larger training set. The variant based on surface forms extracted 9,327 patterns, the variant including morphological information extracted 9,610 while the variant augmented further with POS information extracted only 7,237 patterns. The same test set of 1,000 unseen input sentences from the previous experiments was used. Only full translations were considered. The results are summarised in table 1.



Graph 6: Comparison of Distributions of each Variant

The distributions of the three variants are largely similar. The salient differences between them are established in terms of recall and the number of high-scoring translations. The morphological variant translated more SL sentences than the variant based on surface forms. Furthermore, it produced a greater number of accurate (defined as 90-100% correct) or 100% correct translations. The variant including POS information performed less well than the variant based on surface forms. Fewer translation patterns were produced, resulting in a lower number of SL sentences translated and a lower number of high-scoring translations. The reasons for the poor performance of this variant include tagger errors and the fact that more constraints operate in the translation pattern extraction phase (section 3.2), resulting in fewer translation patterns.

	Surface	Morph	Tagged
Patterns	9,327	9,610	7,237
Translations	322	339	272
Recall	32.2%	33.9%	27.2%
Max recall	44.2%	46.5%	37.3%
Accurate	71	78	64
100% correct	61	68	53

Table 1: Comparison of the Three Variants

Since the evaluation method utilised in these experiments is automatic, it is possible that some of the lower scoring translations may be “correct” or useful by some other criteria i.e. acceptability to a post-editor as raw MT output. Its usefulness could be evaluated by the amount of time or effort required to revise it to the required standard, as in (Minnis, 1994; Whyman & Somers, 1999). Alternatively, one could use any of the various methodologies in the MT evaluation literature based on human observations and scales of correctness.

Discussion

The addition of morphological information to the approach based on surface forms increased recall, if only slightly, and returned a slightly higher number of translations that are 100% accurate or very nearly so. There are numerous possibilities as to why there is only a slight improvement: first, only a relatively small amount of linguistic information is added. Second, the nature of the corpus (titles) is such that there are few morphological variants per root form. There tends to be a higher density of root forms in titles than in ‘ordinary’ texts. Third, the recursive matching algorithm in recombination (section 4.2) adds an amount of noise to an already noisy channel. This has the effect of blurring the improvements made by adding morphological information. Finally, and more importantly, more training data is required.

The results of the variant where POS information is included are disappointing and do not improve upon the variant based on surface forms. Given the results as they stand, the variant augmented uniquely with morphological information performs only slightly better than the variant based on surface forms. Therefore, the effort or cost (financial and computational) of adding linguistic knowledge sources may not be justified, since they affect portability to new language pairs and domains.

The rates of recall are currently low. This can only be attributed to the fact that there is simply not enough data in the corpus of the size reported. The approach must be scaled up to include corpora of millions of words. However, this requirement is in contrast to the observations in graph 1 where the size of the collocation trees severely limits corpus size (unless hard-drive space is efficiently utilised). Therefore, scalability is problematic. On the other hand, if one considers the satisfactory rates of recall when partial matches are included (graph 4), one has to consider whether this methodology is of more use as a flexible translation memory system (Langé et al. 1997; Planas & Furuse, 1999; McTait et al. 1999) where information from more than one translation example can be used to create new TL suggestions.

References

Brill, E. (1992) A Simple Rule-based Part-of-Speech Tagger. *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, 152-155.

Brown, P.F, S.A. Della Pietra, V.J. Della Pietra & R.L. Mercer (1993) The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics* **19**: 263–311.

Brown, R.D. (1999) Adding Linguistic Knowledge to a Lexical Example-based Translation System. *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 99)*, Chester, England, 22–32.

Carl, M. (1999) Inducing Translation Templates for Example-Based Machine Translation. *Machine Translation Summit VII*, Singapore, 617–624.

Dagan, I., K.W. Church & W.A. Gale (1993) Robust Bilingual Word Alignment for Machine Aided Translation. *Proceedings of the Workshop on Very Large Corpora: Academic and Industrial Perspectives*, Columbus, Ohio, 1–8.

Dice, L.R. (1945) Measures of the Amount of Ecological Association Between Species. *Geology* **26**: 297-302.

Echizen-ya, H., K. Araki, Y. Momouchi & K. Tochinai (2000) Effectiveness of Layering Translation Rules based on Transition Networks in Machine Translation using Inductive Learning with Genetic Algorithms. *MT 2000, Machine Translation and Multilingual Applications in the New Millennium*, Exeter, England, 5-1-8.

Fung, P. & K. McKeown (1997) A Technical Word- and Term-Translation Aid Using Noisy Parallel Corpora Across Language Groups. *Machine Translation* **12**: 53–87.

Furuse, O. & H. Iida (1992) An Example-Based Method for Transfer-Driven Machine Translation. *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation. Empiricist v. Rationalist Methods in MT. TMI-92*, Montréal, Québec, 139-150.

Gale, W.A. & K.W. Church (1993) A Program for Aligning Sentences in Bilingual Corpora. *Computational Linguistics* **19**: 75-102.

Güvenir, H.A. & I. Cicekli (1998) Learning Translation Templates from Examples. *Information Systems* **23**: 353–363.

Kaji, H., Y. Kida & Y. Morimoto (1992) Learning Translation Templates from Bilingual Text. *Proceedings of the fifteenth* [sic] *International Conference on Computational Linguistics: COLING-92*, Nantes, France, 672–678.

Koskenniemi, K. (1983) Two-Level Model for Morphological Analysis. *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany, 683-685.

Langé, J-M, E. Gaussier & B. Daille (1997) Bricks and Skeletons: Some Ideas for the Near Future of MAHT. *Machine Translation* **12**: 75-102.

Levenshtein, V.I. (1966) Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Cybernetics and Control Theory* **10**: 707-710.

Lowrance, R. & R.A. Wagner (1975) An Extension of the String-to-String Correction Problem. *Journal of the Association for Computing Machinery* **22**: 177-183.

McTait, K. & A. Trujillo (1999) A Language-Neutral Sparse-Data Algorithm for Extracting Translation Patterns. *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-99)*, Chester, England, 98–108.

McTait, K, M. Olohan & A. Trujillo (1999) A Building Blocks Approach to Translation Memory. *Proceedings of the 21st Aslib International Conference on Translating and the Computer*, London, England.

Malavazos, C. & S. Piperidis (2000) Application of Analogical Modelling to Example Based Machine Translation. *The 18th International Conference on Computational Linguistics: COLING 2000 in Europe*, Saarbrücken, Germany, 516–522.

Maruyama, H. & H. Watanabe (1992) Tree Cover Search Algorithm for Example-Based Translation. *Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-92)*, Montréal, Québec, 173–184.

Matsumoto, Y. & M. Kitamura (1995) Acquisition of Translation Rules from Parallel Corpora, in Mitkov & Nikolov (1997), pp. 405-416.

Minnis, S. (1994) A Simple and Practical Method for Evaluating Machine Translation Quality. *Machine Translation* **9**: 133-149.

Mitkov, R. & N. Nicolov (eds) (1997) *Recent Advances in Natural Language Processing: Selected Papers from RANLP '95*, Amsterdam: John Benjamins.

Planas, E. & O. Furuse (1999) Formalizing Translation Memories. *Machine Translation Summit VII*, Singapore, 331-339.

Nirenburg, S., C. Domashnev & D.J. Grannes (1993) Two Approaches to Matching in Example-Based Machine Translation. *Proceedings of the Fifth International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium, 47–57.

Sato, S. (1995) MBT2: A Method for Combining Fragments of Examples in Example Based Machine Translation. *Artificial Intelligence* **75**: 31–49.

Schmid, H. (1994) Probabilistic Part-of-Speech Tagging Using Decision Trees. *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK, 44-49.

Somers, H.L., I. McLean & D. Jones (1994) Experiments in Multilingual Example-Based Generation. *CSNLP 1994: Third Conference on the Cognitive Science of Natural Language Processing*, Dublin, Ireland.

Somers, H. L. (1998) Further Experiments in Bilingual Text Alignment. *International Journal of Corpus Linguistics* **3**: 115-150.

Takeda, K. (1996) Pattern-Based Context-Free Grammars for Machine Translation. *Proceedings of the 34th Meeting of the Association for Computational Linguistics*, Santa Cruz, CA, 144-151.

Wang, Y-Y. & A. Waibel (1998) Modeling with Structures in Statistical Machine Translation. *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Quebec, 1357–1363.

Watanabe, H. (1992) A Similarity-Driven Transfer System. *Proceedings of the fifteenth* [sic] *International Conference on Computational Linguistics: COLING-92*, Nantes, France, 770-776.

Watanabe, H. (1993) A Method For Extracting Translation Patterns from Translation Examples. *Proceedings of the 5th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-93), MT in the Next Generation*, Kyoto, Japan, 292-301.

Watanabe, H. (1995) A Model of a Bi-Directional Transfer Mechanism Using Rule Combinations. *Machine Translation* **10**: 269–291.

Whyman, E.K. & H.L. Somers (1999) Evaluation Metrics for a Translation Memory System. *Software-Practice and Experience* **29**: 1265-1284.