Towards Memory and Template Based Translation Synthesis

Christos Malavazos[1,2] **Stelios Piperidis**[1,2] **George Carayannis**[1,2]
[1]Institute for Language and Speech Processing, [2]National Technical University of Athens
6 Artemidos & Epidavrou, 151 25 Marousi, Athens, Greece
{christos, spip, gcara}@ilsp.gr

Abstract

This paper describes a novel approach in enhancing the effectiveness of a typical translation memory system towards providing a more flexible translation framework. The system core consists of two basic modules: a) a self-modelling, incremental learning module for extracting translation rules from existing parallel corpora, and b) a translation module that efficiently utilizes various levels of available information for dealing with new input. Even though both modules do not rely on heavy linguistic processing, they are designed in such a way as to allow for additional information to be easily incorporated and improve system performance.

Keywords: Machine Translation/**MT**, Memory Based translation/**MBT**, Example Based Machine Translation/**EBMT**, Computer Aided Translation/**CAT**, Analogical Modelling, Translation by Analogy.

1. Introduction

So far, Translation Memory systems have presented limited success with respect to the type and the size of the text units involved in the translation process. Since their performance relies heavily on the existence of "good" matches they are characterised by considerable inflexibility and a rather ungraceful degradation curve when these matches are not found. Moreover, the real added value of any translation aid tool lies on its ability to encompass different levels of information and processing under a single framework towards providing optimal results.

Ideally, an EBMT system must determine correspondences at a sub-sentence level if optimal adaptation of matching fragments is to be achieved (Collins, B., & Cunningham, P. (1995)). Assuming that these text fragments have been extracted through an appropriate sub-sentential alignment process and have already been stored in the translation memory database, then a procedure is required to determine the best "cover" of the input text (Nirenburg, S. et al (1993), Cranias, L. et al (1994), Frederking, R., Nirenburg, S., (1994), Sato, S. (1995)). Although such approaches can be fully automated, the non-linearity of the translation problem makes them extremely vulnerable to low quality, especially when the produced segments are rather small.

Several approaches try to go a step further, by attempting to build a transfer-rule base in the form of abstract representations. This is achieved through different types of generalization processes, applied on available corpora and relying on different levels of linguistic information and processing (Kaji et al. (1992), Juola, P. (1994), Furuse, O., Iida, H. (1996), Veale, T. and Way, A. (1997), McTait, K., et al. (1999)), thus providing the translation phase with complete "context" information. The deeper the linguistic analysis, the more flexible the final translation structures will be and the better the quality of the results. However, this kind of analysis leads to more computationally expensive and difficult to obtain systems. Our approach consists in a fully modular analogical framework, which can cope with lack of resources, and will perform even better when these are available.

2. General

By "analogy" we mean the process of dealing with input patterns based on their similarities and differences from an existing database of stored examples (exemplars), than by referring to some pre-defined set of explicit translation rules. These examples are used to classify new items, without intermediate abstraction in the form of rules. In order to achieve this, an exhaustive database search is required and during this search, less relevant examples need to be discarded.

In contrast to most of the analogy-based systems, that perform run-time classification of input patterns without involving any intermediate processing on their knowledge base, our approach applies the same principles during a learning phase in an attempt to extract appropriate generalizations (translation rules) based on similarities and differences between existing exemplars. In this way, analogy is treated as more than simple pair-wise similarity between input and database exemplars; rather it is considered as the main relation underlying a more complex network of relations between database exemplars.

The main idea is based on the observation that given any source and target language sentence pair, any alteration of the source sentence will most likely result in one or more changes in the respective target sentence, while it is also highly likely that constant and variable units of the source sentence correspond to constant and variable target units respectively. Apart from cases of so-called "translational divergences" (Dorr, B. (1994)) in most cases the above assumption holds true. However, these do not affect the learning process since they do not fulfil the necessary criteria and are finally rejected.

The matching process described by Daelemans W., et al, (1997), based on Skousen's analogical modelling algorithm (Skousen, R. (1989)), consists of two subsequent stages. The first stage of the matching process is the construction of "**subcontexts**", i.e. sets of examples that are obtained by matching the input pattern, feature by feature, against each database item on an equal /not-equal base. These are later classified in the examples database accordingly. Taking the input pattern ABC as an example, eight ($=2^3$) different and mutually disjoint subcontexts would be constructed:

$$ABC \quad \bar{A}BC,\ A\bar{B}C,\ AB\bar{C},\ \bar{A}\bar{B}C,\ \bar{A}B\bar{C},\ A\bar{B}\bar{C},\ \bar{A}\bar{B}\bar{C}$$

where the macron denotes complementation. Thus exemplars in the second class share only the second and third feature with the input pattern.

In the following stage, "**supracontexts**" are constructed by generalising over specific feature values. This is done by systematically discarding features from the input pattern and taking the union of the subcontexts that are subsumed by this new pattern. Supracontexts can be ordered with respect to generality, so that the most specific supracontext contains items that share all features with the input pattern, while the less specific ones contain those items that share at least one feature. The most general supracontext contains all database examples whether or not they share any features with the input pattern. For example, the union of the first and fourth subcontext generates the following new supracontext: **[A B –]**.

In addition, we introduce an additional dimension to the above described process, that of language, by simultaneously performing the matching process to the target language equivalents and aligning individual results, based on the principles described earlier. Therefore, what we are ultimately searching for, is source and target sentence pairs for which evidence of correspondence between any or all the respective subcontexts within the training corpora is available. This will subsequently lead to links between respective supracontexts. For example:

$$\left.\begin{array}{c} [A_s\ B_s\ C_s] \Leftrightarrow [A_t\ B_t\ C_t] \\ \text{AND} \\ [A_s\ B_s\ \bar{C}_s] \Leftrightarrow [A_t\ B_t\ \bar{C}_t] \end{array}\right\} \Rightarrow [A_s\ B_s\ -] \Leftrightarrow [A_t\ B_t\ -]$$

(Where s = Source Language, **t** = Target Language**)**

Subcontexts   *Supracontexts*

3. The learning mechanism

To this respect, supracontexts constitute our translation templates, that is abstract expressions of bilingual pairs of "pseudo-sentences", consisting of sequences of constant and variable elements.

Discarded features (represented by the "–" symbol) of corresponding supracontexts, rising from different parts between matching sentences, correspond to single or multi-word **variable elements** (represented by the X_{ij} symbols) and comprise the bilingual lexicon of translation units for the respective translation patterns, while **similar/constant** parts act as the **context** within which variable units are instantiated. Matching between exemplars is performed in two dimensions simultaneously, that is between source and target sentences of matching pairs respectively. The results of the process, given that certain conditions are met, are stored in an **"analogical network"** (Federici, S. & Pirrelli V., (1994)) of inter-sentence and intrasentence relations between these exemplars and their generalizations. A rather simple example of this is presented in Figure 1.

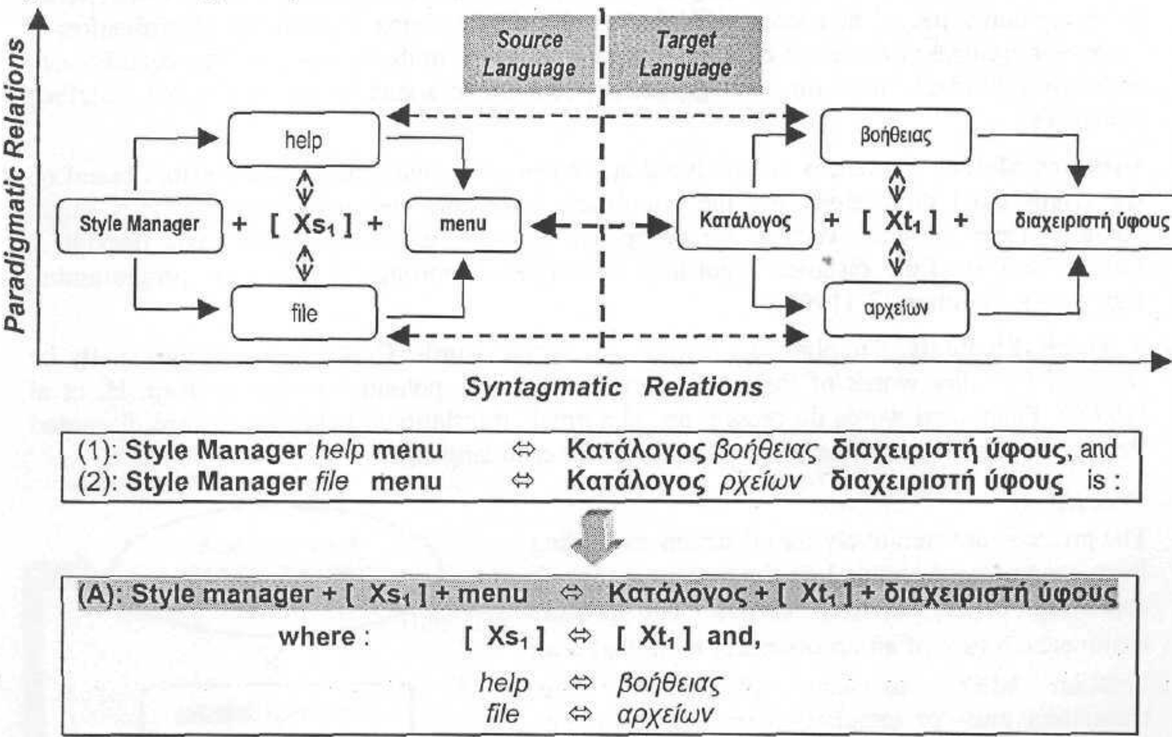

Figure 1

Syntagmatic links (horizontal axis) constitute the intrasentence relations/links between sentence constituents, that is, the way they actually appear and are ordered in the respective sentence, while paradigmatic ones (vertical axis) correspond to the intersentential relations, that is, the information concerning substrings that are in complementary distribution with respect to the same syntagmatic context. Furthermore, a third dimension is added to the whole framework, that of the **"language"**, since all principles are applied simultaneously to both source sentences and their target equivalents. In case linguistic annotations are available, they are appropriately incorporated in the respective nodes.

At this point no conflicts are resolved. All possible patterns are stored in the network, while links both paradigmatic and syntagmatic are weighted by frequency information. This will eventually provide the necessary information to disable and even discard certain false or useless variables or templates.

3.1 The Algorithm

Translation templates as well as translation units are treated as paradigmatic flexible structures that depend on the available evidence. As new data enter the system, rules can be extended or even replaced by other more general ones. It is usually assumed that there is only one fixed way to assign a structural representation to a symbolic object, either a translation unit or a translation template. However, in our approach there is no initial fixed definition of this particular structure, it is rather left up to the training corpus and the learning mechanism. As expected, within this analogy-based framework, linguistic objects are determined on the basis of the paradigmatic context they appear in, resulting in a more flexible and also corpus dependent definition of translation units.

Search Space Reduction: The SSR methodology depends on the specific needs of the particular task. Run-time pruning of possible matches can speed up the learning process, however it also reduces system recall & coverage. On the other hand, constraints on paradigmatic relations are more reliable in terms of providing better results but cannot contribute to the speed of the learning process. In our approach SSR was based on an efficient indexing and retrieval mechanism (Willman, N. (1994)) which allows fast identification of "relevant" sentences based on common single/multi-word units. In this way, the search space for each individual candidate was significantly reduced to a smaller set of possible matching sentences.

Distance Metric: Sentences are analysed and encoded in two-dimensional vectors based on the words (first dimension) and the linguistic annotations (second dimension) they might contain. Then sentence vectors are compared on an equal - not equal basis through a Levensthein or Edit distance algorithm implemented through a dynamic programming framework (Stephen, G. (1992)).

Variable Elements: Coupling is restricted to content words. Content words can usually be replaced by other words of the same category acting as potential variables (Kaji, H. et al (1992)). Functional words do present an "abnormal" translational behavior and are discarded through the use of predefined "exclusion lists" for each language. .

Workflow

The process runs iteratively for all sentences starting from sentences of length 1 to the maximum length appearing in the training corpus. The process terminates in case of an unsuccessful loop, that is an iteration where no new information either translation units or templates were extracted. The learning process is depicted in detail in **Fig. 2**:

<u>**Phase1**</u> **Search Space Reduction:** Extract an initial set of possibly relevant sentences for the current input sentence.

<u>**Phase2**</u> **Sentence Matching:** Match Input sentence against the previous set. Matching candidates are sorted based on distance score. Matches with fewer differences are examined first.

<u>**Phase3**</u> **Identification of Subcontexts: For** each matching candidate, identify the respective subcontext of the input sentence that it adheres to. Examine target language equivalents. Resolve differences between source and target language matching candidates based on already existing information contained in the bilingual lexicon. The translation unit lexicon is enriched with any successfully resolved difference.

<u>**Phase4**</u> **Identification of Supracontexts:** Given the identified subcontexts produce the respective supracontexts through unification of respective variable feature values.

<u>**Phase5**</u> **Extraction of Translation Patterns:** Construct corresponding translation patterns from existing supracontexts. Update analogical network.

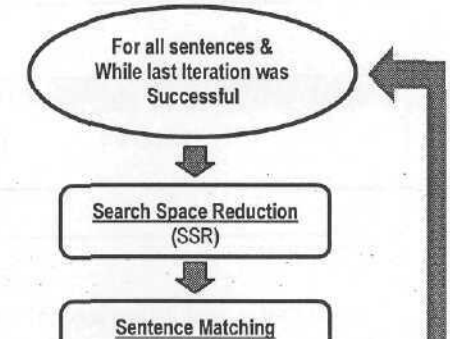

Figure. 2

3.2 Network Refinement

Conflict resolution and network refinement is performed at the end where all information is available. Special attention was paid to cases of: a) **Translation alternatives** b) **Conflicting templates**, c) **Overlapping templates** and d) **Complementary Templates** (Malavazos, C., Piperidis, S. (2000)).

4. Translation

Under the previous framework three different types of resources can be identified: a) **Sentences**, b) **Translation Units** (below sentence fragments), and c) **Translation Templates**. The aim of the overall translation process should be to optimally combine these different types of resources in order to improve the quality of the proposed translation, thus providing a more flexible architecture, which requires less post-editing from the user. Our first attempt consists in a rather top-down approach, where the system first searches for full matching sentences and then for templates that could fully "cover" the input sentence. If both steps fail, the system proceeds on a sentence fuzzy match process, which however is also enhanced by a "local matching" process between the DB of translation units and the parts of the input sentence that have not been covered by the best matching DB sentence. The Overall Sentence Translation Process is presented in Figure 3. It consists of 4 subsequent phases:

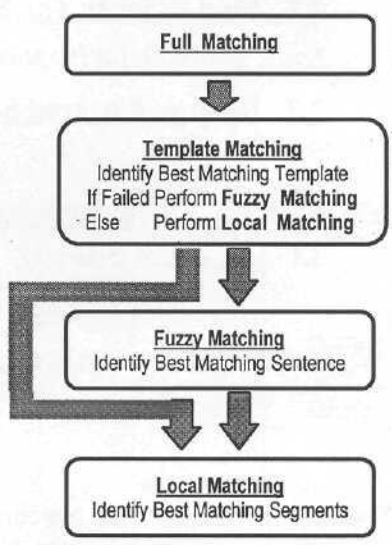

Figure 3. Translation Process

Phase 1 Full Matching: Performs a very fast search for potential full matching sentences. If one is found then the translation process ends.

Phase 2 Template Matching: Identifies the best matching template that provides the "optimal cover" for the given input sentence. This is based on a Dynamic Programming framework that assigns higher scores to matches with fewer and longer contiguous segments and fewer variables. This is described in detail in the next section. In case a matching template is found, then local matching is performed in order to find those translation units that cover possible unmatched input segments (resulting from variable elements of the matching template).

Phase 3 Fuzzy Matching: Identifies the best matching DB sentence for the given input based on similar DP framework. Again, Local matching will be performed on unmatched segments.

Phase 4 Local Matching: Identifies full matching translation units (if any) for all unmatched input segments.

Unmatched are those parts of the input pattern that correspond to template variables (template matching) or those that do not have any corresponding parts on the best matching sentence (fuzzy matching). Matching is performed on the context of both input segment and translation unit in case more than one translation alternatives exist. For example, if sentence **[A B C]** is matched with the template **[AB X_{var}]** (template matching) or to sentence **[A B D]** then word **[C]** will have to be searched in the translation unit database. In case more than one translation alternatives exist, then the context (±3 words) of word **[C]** in the input pattern **[AB -]** will be matched against the context of all translation unit alternatives. In the case of template matching, the target equivalents of matching translation units of word **[C]** will automatically substitute the target equivalent of variable **[X_{var}]**. This is actually the reason why the template matching process precedes the fuzzy matching one.

4.1 Template Matching

As already mentioned, template matching is based on DP framework. A table is constructed between the input sentences and each matching template including their words (and variables) on the horizontal and vertical axis respectively. The algorithm is explained in more detail hereafter.

1. - INITIALIZE TABLE

*Each Point Local Score (i,j) in the table is assigned a value of 1 if the **i-th** input word matches the **j-th** template word (or is contained in set of words corresponding to template variable at position **j**).*

2. - WHILE THERE ARE STILL TABLE ELEMENTS

2.1. Estimate All Local Scores Left to Right

for *(i = 0 .. maxValue, j = 0..maxValue)*
*Local_Score(**i, j**) = Local_Score(i, j) + Local_Score(**i-1, j-1**)*

2.2. Find & Store The Maximum Local Score

*Local_Score(**k**) (at iteration –**k**)*

2.3. Remove Covered Segment from Table

3. - ESTIMATE TOTAL SCORE

3.1 $[\Sigma_k [Local\_Score(k)]^2$ **for all –k]**

Figure 4: Template Matching Algorithm

The first two steps of the matching process between an input sentence **[A B C D E F G]** and three alternative templates **[A B $\boxed{X_1}$ D E F $\boxed{X_2}$] [A $\boxed{X_1}$ C D E F $\boxed{X_2}$] [A B $\boxed{X_1}$ D E F $\boxed{X_2}$ G]**, are shown in the following figures. In these examples, none of the words **[A-G]** is included in the set of words instantiating the variables of the first two templates however, word **[C]** is included in the set of words corresponding to variable $\boxed{X_1}$ of the last template. Therefore, the columns corresponding to all variables (marked in grey) are filled with zero values, and only in the last case a value of **1** is added into the column of the first variable for the word **[C]**.

The second template is assigned a higher score than the first one since it covers the input sentence through a longer contiguous text segment. The last template would be expected to produce an even lower score ($2^2 + 2^2 + 1^2 = 9$). However, the first variable matches word [C], thus covering words [A] to [E] with a single segment and therefore it is finally assigned a higher score (26).

G	0	0	0	0	0	0	0
F	0	0	0	0	0	1	0
E	0	0	0	0	1	0	0
D	0	0	0	1	0	0	0
C	0	0	0	0	0	0	0
B	0	1	0	0	0	0	0
A	1	0	0	0	0	0	0
	A	B	X	D	E	F	X

Ex. 1: Initialization

G	0	0	0	0	0	0	0
F	0	0	0	0	0	3	0
E	0	0	0	0	2	0	0
D	0	0	0	1	0	0	0
C	0	0	0	0	0	0	0
B	0	2	0	0	0	0	0
A	1	0	0	0	0	0	0
	A	B	X	D	E	F	X

Ex. 1: Estimation of Local Scores

Input
[AB C DEF G]
Template
[AB $\boxed{X_1}$ DEF $\boxed{X_2}$]
- 2 Variables
- 2 Segments
No hit between variables and words Total Score
Total Score
($3^2 + 2^2 = \underline{13}$)

G	0	0	0	0	0	0	0
F	0	0	0	0	0	1	0
E	0	0	0	0	1	0	0
D	0	0	0	1	0	0	0
C	0	0	1	0	0	0	0
B	0	0	0	0	0	0	0
A	1	0	0	0	0	0	0
	A	X	C	D	E	F	X

Ex. 2: Initialization

G	0	0	0	0	0	0	0
F	0	0	0	0	0	4	0
E	0	0	0	0	3	0	0
D	0	0	0	2	0	0	0
C	0	0	1	0	0	0	0
B	0	0	0	0	0	0	0
A	1	0	0	0	0	0	0
	A	X	C	D	E	F	X

Ex. 2: Estimation of Local Scores

Input
[A B CDEF G]
Template
[A $\boxed{X_1}$ CDEF $\boxed{X_2}$]
- 2 Variables
- 2 Segments
No hit between variables and words
Total Score
($4^2 + 1^2 = \underline{17}$)

G	0	0	0	0	0	0	1
F	0	0	0	0	0	0	0
E	0	0	0	0	1	0	0
D	0	0	0	1	0	0	0
C	0	0	1	0	0	0	0
B	0	1	0	0	0	0	0
A	1	0	0	0	0	0	0
	A	B	X	D	E	X	G

Ex. 3: Initialization

G	0	0	0	0	0	0	1
F	0	0	0	0	0	0	0
E	0	0	0	0	5	0	0
D	0	0	0	4	0	0	0
C	0	0	3	0	0	0	0
B	0	2	0	0	0	0	0
A	1	0	0	0	0	0	0
	A	B	X	D	E	X	G

Ex. 3: Estimation of Local Scores

Input
[AB C DE F G]
Template
[AB $\boxed{X_1}$ DE$\boxed{X_2}$G]
- 2 Variables
- 2 Segments
One hit between variable [X_1] and word [C]
Total Score
($5^2 + 1^2 = 26$)

Figure 5: Template Matching Examples

5. Evaluation

The training set consisted of a bilingual (EN-GR) technical corpus (automotive industry) of 5K sentences, ~20K wordforms for each language. The process resulted in ~550 translation rules, and 350 translation units (~50 multi-word ones). The precision estimated through manual evaluation was ~75%. Coverage of the final translation rule set against the corpus was measured 38%. More specifically, the set of 500 rules proved sufficient to generate 38% of the corpus sentences through an inverse process.

The test set for the translation module consisted of an additional 1K english sentences, randomly selected from the same corpus. Translation coverage on this new set was estimated around 31% by applying only the full matching component (a rather high repetition rate was expected after all for this type of domain). By also utilizing the template matching component (along with the local matching one) the translation coverage reached ~43%, which increased even more (~49%) by applying the fuzzy matching component as well (with a sentence similarity threshold equal to 50%).

6. Conclusion & Future Work

We have presented a self-modelling, incremental analogical algorithm for extracting translation patterns from existing bilingual corpora, a method for efficient storage and representation of extracted relations between various units of text as well as a first attempt to efficiently combine these during the translation phase. Not surprisingly, the quality of the results depends on the available information in terms of quantity as well as quality and depth. Lack of any kind of linguistic information will consequently result in translation rules based only on "shallow" evidence that would obviously be less efficient translation-wise. Similarly, information of low quality will generate erroneous rules. However, this is a basic presupposition of any EBMT system: "what you give is what you get".

The proposed framework was initially evaluated on the basis of string form information. However, the model can easily take into account "deeper" linguistic knowledge during the learning as well as the matching phase, thus improving the quality of the final results. This is the focus of on going research. Finally, future work will also focus on a more flexible bottom-up template matching technique that could provide "draft" translations of new input by composing translations of constituent translation units.

7. References

Collins, B., & Cunningham, P. (1995) A Methodology for EBMT. *4th International Conference on the Cognitive Science of Natural Language Processing, Dublin 1995.*

Cranias, L., Papageorgiou, H. and Piperidis, S. (1994). A matching technique in Example-Based Machine Translation, *Proc. of COLING-94, pp 100-105.*

Daelemans, W., Gillis, S. & Durieux, G., (1997) Skousen's analogical modelling algorithm: a comparison with lazy learning. *New Methods in Language Processing: Edited by Daniel Jones & Harold Somers, UCL Press, p.3-15.*

Dorr, B. (1994) Machine Translation Divergences: A Formal Description and Proposed Solution. *Association for Computational Linguistics, Vol. 20, 1994.*

Federici, S. & Pirrelli, V. (1994). The compilation of large pronunciation lexica: the elicitation of letter to sound patterns through analogy based networks. *Papers in Computational Lexicography, Complex '94, Budapest, 59-67.*

Frederking, R., Nirenburg, S., (1994) Three Heads are Better then One. *Proceedings of the fourth Conference on Applied Natural Language Processing, ANLP-94, Stuttgart, Germany*

Furuse, O., Iida, H. (1996) Incremental Translation Utilizing Constituent Boundary Patterns. *Proc. Coling-96, pp 412-417.*

Juola, P. (1994) Self-Organizing Machine Translation: Example-Driven Induction of Transfer Functions. *University of Colorado at Boulder, Technical Report CU-CS-722-94.*

Kaji, H., Kida, Y., and Morimoto, Y., (1992) Learning Translation Templates from Bilingual *Text. Proc. Coling., p. 672-678, 1992.*

Malavazos, C., Piperidis, S. (2000) Application of Analogical Modelling to Example Based Machine Translation. *COLING 2000. Saarbrucken.*

McTait, K., Olohan, M., Trujillo, A. (1999) A Building Blocks Approach to Translation Memory. *Proc. From the 21st ASLIB Conference, London, 1999.*

Nirenburg, S. Domashnev, C., Grannes, D. (1993) Two Approaches to Matching in Example-Based Machine Translation. *Proc. of TMI-93, Kyoto, Japan, 1993.*

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

Skousen, R. (1989) Analogical Modelling of language. *Dordrecht: Kluwer.*

Stephen, G. (1992) String Search. *University College of North Wales, Technical Report TR-92-gas-01.*

Veale, T. and Way, A. (1997) Gaijin: A Bootstrapping Approach to Example-Based Machine Translation. *International Conf., Recent Advances in Natural Language Processing, Tzigov Chark, Bulgaria, 239-244.*

Willman, N. (1994) A Prototype Information Retrieval System to Perform a Best-Match Search for Names. *Conference Proceeding of RIAO '94.*