Sub-sentential Exploitation of Translation Memories

Michel Simard, Philippe Langlais

Laboratoire de recherche appliquée en linguistique informatique (RALI)
Département d'Informatique et recherche opérationnelle
Université de Montréal
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada, H3C 3J7
{SimardM,Felipe}@IRO.UMontreal.CA

Abstract

Translation memory systems (TMS) are a family of computer tools whose purpose is to facilitate and encourage the re-use of existing translations. By searching a database of past translations, these systems can retrieve the translation of whole segments of text and propose them to the translator for re-use. However, the usefulness of existing TMS's is limited by the nature of the text segments that that they are able to put in correspondence, generally whole sentences. This article examines the potential of a type of system that is able to recuperate the translation of sub-sentential sequences of words.

Keywords

Translation memory, translation alignment, translation support tools, example-based machine translation.

Introduction

Translation memory systems (TMS) are a family of computer tools whose purpose is to facilitate and encourage the re-use of existing translations. Ultimately, within an environment where such a system is used, the same piece of text should never be translated twice (for a given pair of source and target languages). The mechanism that is proposed to attain this goal is to systematically archive the translators' production, as pairs of mutually translated segments. When, within a new text to be translated, a segment of text is encountered that matches some couple in this "translation memory", its translation can be retrieved and re-used by the translator.

A number of vendors now market this sort of TMS, most notably Trados (*Translator's WorkBench*), IBM (*Translation Manager/2*), Atril (*Déja Vu*) and Star-AG (*Transit*). In all these systems, the couples in the translation memory are normally pairs of sentences. Given a new sentence to be translated, they will usually look for couples whose source-language sentence matches the new sentence in its entirety. This match need not be exact (all the above systems feature some sort of "fuzzy" matching), but obviously, better-matching couples stand a better chance of being re-usable.

It is instructive, as Macklovitch and Russell (2000) suggest, to view TM applications in an *information retrieval* (IR) perspective: when using a TMS, the translator is actually just searching for documents that might help in translating a given sentence. In this case, these "documents" happen to be pairs of mutually translated sentences, and the translator is likely to deem a document relevant if its target-language (TL) segment constitutes an acceptable translation for the source-language (SL) sentence. As for the query, it is constructed automatically by the TMS, from the SL sentence to be translated. The retrieval operation is carried out by matching this query as closely as possible.

In this perspective, the default strategy of existing TMS's is an extreme form of high-precision, low-recall search: return only the best matching document, and only if it displays a sufficient resemblance to the source sentence. In IR terms, this is like having an assistant hit the "I'm feeling lucky" button on the *Google* search engine interface[1] for you, and warn you whenever this brings back a Web page that matches your query exactly.

To a certain extent, this strategy is appropriate for the TMS application, because it is based on the assumption that the user is not willing to go through large quantities of information before translating every sentence. In this perspective, complete sentences are "sure hits" for the TMS, because their translation stands a good chance of being recuperable[2]. Yet, the net effect of this strategy is that TMS's are applicable only in very specific contexts, such as software localization, documentation updates, administrative forms, etc. In the general case, the repetition of complete sentences is an extremely rare event.

Nevertheless, this state of affairs is somewhat frustrating: intuitively, just because a sentence has never been translated before does not necessarily mean that the TM does not contain smaller segments that could be useful to the translator.

The *TransSearch* system (Macklovitch et al. 2000) is one radically different type of TMS based on this idea. It allows translators to interactively query a large database of past translations for specific terms, expressions, or any sequence of words. If current usage statistics are an argument in favor of exploiting TM's at a sub-sentential level, the system has been online[3] for almost 5 years, and currently processes over 50 000 queries per month (March 2001).

Another argument in support of this idea comes in the form of recent work in the closely related field of example-based machine translation (EBMT): the *Pangloss* multi-engine machine-translation system comprises an

---

[1] http://www.google.com

[2] Note that even for large segments of text such as sentences and paragraphs, the general re-usability of past translations is being seriously questioned by some translators; see Bédard (2001), for example.

[3] http://www-rali.iro.umontreal.ca/TransSearch

EBMT component based on a very simple mechanism (Brown, 1996). The system's database of examples consists in a large collection of aligned sentences (in other words, a standard translation memory). Given a new sentence to translate, the system looks up all possible sub-sequences of words of this sentence in the database. It then relies on a simple word-alignment mechanism to locate the translation of each matching sequence within the retrieved pairs of sentences. These TL sequences are then added to a pool, from which a distinct generation component will select the sequences that make up the final translation. Using this approach, the author reports covering over 70% of an input of unrestricted Spanish newswire.

In the context of fully automated translation, the succes of the whole enterprise rests crucially on that of each of the individual steps described above. The constraints are much less stringent in the context of machine-assisted human translation: since the goal is to propose partial translations to the user, it is not essential to cover the source text in its entirety, because the translator can provide for missing segments, and we don't have to worry about combining segments, because that is left entirely to the translator. Success is then more a question of *usefulness*: to what extent can translations extracted from a translation memory in this way be of use to a human translator?

In this article, we examine the potential of a type of TMS based precisely on this type of search mechanism. We first describe how such a *generalized TMS* departs from existing systems, and then report on some experiments that were carried out in order to quantify the gains.

Generalized Translation Memory System

In this section, we present a *generalized translation memory system* (GTMS). The fundamental difference between this and existing TMS's is its ability to operate at a *sub-sentential* level: in other words, a GTMS will propose TL sequences that may be of use to translate sub-sequences of a source sentence.

As in existing systems, we assume that the couples that make up the translation memory are pairs (s,t), of mutually translated sentences (because sentences don't always translate one-to-one, either s or t can occasionally consist in more than a single sentence).

To simplify our notation, we use x_i^j to represent the sequence of word tokens $x_i, \ldots x_j$. By convention, $e = e_1^k$ designates the sentence the translator is currently working on. At any time, we assume that the GTMS "knows" e, i.e. it is aware of the part of the text that is the focus of the user's attention.

We focus here only on those components responsible for automatically proposing to the translator TL segments extracted from the translation memory. We can view this process as being handled by three distinct components, namely the **search mechanism**, the **couple selector** and the **target-text generator**.

Search mechanism

The search mechanism ranks each couple in the TM with regard to its similarity with the sentence to be translated. Most existing systems use variants of the *edit distance* metric (Planas and Furuse, 1999) i.e. SL sentences are compared to e on a character-per-character basis.

The search mechanism of the GTMS operates on sub-sequences e_i^j of source sentence e. It ranks each couple (s,t) in the TM with regard to the longest common sub-sequence of words between e and s. In other words, (s,t) matches e if there exists i, j, k and l such that $e_i^j = s_k^l$, and matches are ranked with regard to the length of e_i^j. Figure 1 shows all matching sub-sequences of two or more words on an example sentence.

Source Sentence: *Will he table the recommendations made by both firms?*
Matching sub-sequences:
<will he>, <will he table>, <will he table the>,
<he table>, <he table the>, <table the>,
<the recommendations>,
<the recommendations made>
<the recommendations made by>
<recommendations made>,
<recommendations made by>
<recommendations made by both>,
<made by>, <made by both>,
<by both>, <by both firms>, <both firms>

Figure 1: Matching queries of two or more words for the given sentence

In practice, a GTMS would examine all possible sub-sequences of e, and retrieve all matching couples. In a series of experiments with this type of search mechanism (Langlais & Simard, 2001), we found that concentrating on "linguistically motivated" subsequences was more productive than considering all possible sub-sequences: intuitively, segments with a clear syntactic status (e.g. a simple nominal compound) stand a much better chance than arbitrary segments of having a clearly identifiable translation, and therefore of being useful to the translator.

Based on this idea, we implemented a search mechanism based on a *text chunker*, i.e. a procedure that identifies simple surface syntactic constructs ("chunks") in the source-text e (Langlais, 2001). It considers only those sub-sequences of e whose beginning and end coincide with chunk boundaries. Figure 2 shows an example.

Chunks:
will [$_{VP}$ he table] [$_{NP}$ the recommendations] [$_{VP}$ made by] [$_{NP}$ both firms] ?
Matching sub-sequences:
<will he table>, <he table>, <the recommendations>,
<the recommendations made by>, <made by>,
<both firms>

Figure 2: Chunking of the sentence of Figure 1 and the corresponding matching sub-sequences.

Couple selection

Using the above ranking, the system selects the best matching couple(s). Because a GTMS deals with couples corresponding to multiple sub-sequences of e, it does not make sense to propose just a single match. On the other hand, we do not want to swarm the user with large amounts of information. A more sensible approach is to identify, for each sub-sequence of *e* for which matches have been found in the TM, those couples most likely to be useful. Also, because the user is unlikely to make use of TL proposals corresponding to overlapping portions of *e*, we can restrict the selection by considering only a subset of the matching sub-sequences of *e*, chosen so as to cover as much of *e* as possible without overlapping. Figure 3 below show an example of this type of *source-cover.*

Source-cover:
<will he table> <the recommendations made by>
<both firms> ?

Figure 3: Optimal SL text cover using sub-sequences of Figure 2.

In our experiments, we found this kind of strategy to be an effective way of selecting the most relevant among all matching couples (evaluation issues are discussed further below). In practice, we try to cover as much of *e* as possible, using the largest available sub-sequences. Adding this last constraint follows the intuition (fundamental in existing TMS's) that larger segments are more likely to be useful to the translator.

Once we have identified sub-sequences of interest in this way, we may still have to deal with multiple couples matching each sub-sequence of *e*. Here again, to limit the quantity of information we present to the user, it is necessary to make a selection. One possibility is simply to keep the first match(es) found. A more effective method is to use a statistical translation model to measure the strength of the association between the SL and TL sub-sequences. Stronger associations usually correspond to more frequent ones, and we can assume these to be the ones most likely to be useful. However, this requires identifying matching SL and TL sub-sequences within couples. We discuss this topic below.

Target-language Text Generation

Using the selected couples, the system must produce the TL text to be proposed to the user. Since each selected couple (*s,t*) may correspond to only a fraction of the source sentence *e*, large segments of the TL sentences *t* will usually not be relevant to the translation of *e*. Limiting the amount of information presented to the user requires identifying some specific portion in *t* that best corresponds to the common sub-sequence s_i^j between *e* and *s*.

To perform this operation, we have experimented with a word-alignment procedure adapted from Wu's *statistical inversion transduction grammars* (SITG – Wu, 1997). Briefly, this procedure takes as input the pair of sentences *s* and *t*, and recursively segments both texts in parallel, identifying at each step the most probable alignment between sub-sequences. By forcing this procedure to "stop" around the matching sub-sequence s_i^j in *s*, we can locate the sub-sequence t_k^l of *t* to which it most likely corresponds in this "parallel derivation" of *s* and *t*. Figure 4 below shows an example.

SL sub-sequence : <*the recommendations made by*>

Matching couple :
SL : « What we find in this bill are things that are directly from *the recommendations made by* these groups. »
TL : « Ce qu' on trouve dans ce projet de loi , ce sont des choses qui émanent directement des *recommandations faites par* ces groupes. »

TL sub-sequence : <*recommandations faites par*>

Figure 4: TL segment identified by word-alignment procedure within the given couple.

Experiments

Evaluation Protocol

We implemented a prototype GTMS "core engine", to evaluate the potential of the approach. To make this evaluation as independent as possible of implementation decisions and available resources, we designed an evaluation protocol resorting to a test bitext *B*, i.e. a set of pairs of translated sentences (*s*, *t*) distinct from the translation memory.

Essentially, the idea was to simulate a scenario where a human translator is translating each source sentences *s* of *B*, and assess to what extent the GTMS proposals would help him to produce the "oracle" translation *t*.

This was done by measuring how much of the oracle translations in *B* could be "covered", using the proposals of the GTMS. For this measure to be meaningful, however, it must be contrasted with the quantity of information that the GTMS provides the user with. Clearly, with a large enough translation memory, we could cover just about any TL segment of text, simply by proposing all available TL words. Of course, this would not be very useful, because it would mean swarming the user with useless information.

This suggested resorting to the notions of *recall* and *precision* to measure the usefulness of the system. In this context, *recall* refers to the proportion (in terms of word tokens) of the oracle translations in *B* that can be covered by the system's proposals, while *precision* refers to the proportion of these proposals that were actually useful.

The experiments were carried out as follows: the SL part *s* of each pair in the test bitexts were submitted to our GTMS prototype. For each of these, the system made a number of TL proposals. Using arbitrary sub-sequences of these proposals, we then computed an optimal cover of the oracle translation *t*, – this corresponds to a scenario where the user can "cut and paste" at will in the proposed sequences to construct his translation. As with the source-cover, this target-cover uses the largest possible sub-sequences. We also assumed that the user does not go as far as copying isolated words out of the proposals: the minimum sub-sequence is two words long.

From there, it was possible to compute recall and precision figures for the test bitexts. Figure 5 shows some of the proposed target sequences for our example sentence, as well as the resulting optimal target-cover. In this example, the proposed sequences were used to produce 6 words out of the oracle translation's 11, for a recall of 6/11=0.54. These 6 words were extracted from the GTMS's proposals, which totaled 22 words, for a precision of 6/22=0.27.

SL sub-sequence	**2 best-scoring TL sub-sequences**	
will he table	va-t-il déposer	*déposera-t-il* à
the recommendations made by	*recommandations faites par* le	des recommandations faites par le
both firms	les entreprises	deux sociétés

Oracle translation:
Déposera-t-il les recommandations faites par les deux agences?
Optimal Target Cover:
[Déposera-t-il] les [recommandations faites par] les deux agences?

Figure 5: Proposed TL sub-sequences and optimal cover of the oracle translation.

Test Material

The translation memory we used for the tests was constructed from a corpus of proceedings of the Canadian parliamentary debates (Hansard). This corpus covers 15 years of debates (from 1986 to 2000 inclusively) and totals over 100 million words of each language. All pairs of documents were automatically segmented into sentences, which were then aligned using the *SFIAL* program (an improved implementation of the alignment method proposed by Simard et al., 1992), thus producing over 5 million pairs of segments.
Two different test bitexts were used for our experiments, each consisting in 100 pairs of sentences, randomly selected from two quite different documents: The *Hansard* bitext comes from a parliamentary debate outside our translation memory corpus, while the *Verne* bitext was extracted from Jules Verne's novel "De la terre à la lune". In both bitexts, the alignments were verified by hand.

Basic Results

The initial objective of our work was to determine to what extent it was possible to improve the performance of existing TMS's in terms of recall. In practice, we found that this could easily be done by allowing a system to search for sub-sequences of the text to translate, rather than complete sentences. The challenge was then to retain these gains in recall while maintaining precision at an acceptable level. We figured that for a GTMS to be viable, it should not propose much more TL text than the size of the text to be translated. In terms of our experimental results, this meant that precision should always exceed recall.
In this regard, our GTMS performed best when the following constraints were applied:

- For each sub-sequence of SL text in the optimal source-cover, propose only the single TL sequence with the best association score, as computed with our statistical translation model.
- Do not propose TL sequences of less than two words.

The performance of the system in this configuration is presented in Table 1 below.

Bitext	Precision	Recall
Hansard	37.14	28.09
Verne	22.27	11.27

Table 1: GTMS performance in "best" configuration

What these results mean is that for the *Hansard* test bitext, over a third of the TL sub-sequences proposed by the GTMS were useful to reconstruct 28% of the oracle translations.
Performance on the *Verne* bitext is much lower. Yet this shows that the systems still displays potential for translating texts that are completely unrelated to the TM.

TL Sequence Size

Obviously, a system that proposes TL sequences that individually cover large portions of the oracle translations is more likely to be deemed useful by translators, because it means that a translation can be pieced together with less manipulations. Conversely, whatever the actual details of the user interface, it is unlikely that cutting and pasting small portions of TL text will result in great savings for the translator.
With this in mind, we conducted a number of experiments to measure the effect of TL sequence sizes. This was done by filtering the output of the GTMS, so as to block the output of TL sequences below a given minimal length N (measured in words), and then never using sub-sequences of less than N words in the target-cover. Table 2 below presents the results of these experiments on the Hansard bitext.

N	Precision	Recall
2	37.14	28.09
3	26.62	17.55
4	18.71	9.22
5	14.79	4.81

Table 2: GTMS performance for different minimum TL sequence sizes N.

As can be seen, recall falls rapidly as N increases. However, precision remains significantly higher than recall, which indicates that the user is not swarmed by information. In the end, this parameter could be adjusted by the user according to his own preferences.

User Manipulations

Cutting and pasting text from a GTMS's TL proposals to piece together a new translation can be a laborious activity, and whether users would find it easier just to type the text instead is not clear. To evaluate this, we measured the performance of a GTMS under 2 alternative scenarios. In the first of these, the user picks only those TL proposals that he can use "as is" in his new translation, without cutting. In the second scenario, the user also considers proposals whose *prefix* is usable as is. The intention of this last scenario is to evaluate the potential of a "typing completion" mechanism, such as that proposed in the *TransType* interactive MT project (Foster et al., 1997). The results of these experiments on the *Hansard* bitext appear in Table 3 below.

Scenario	Precision	Recall
cut and paste	37.14	28.09
prefix	26.01	19.66
as is	17.96	13.58

Table 3: GTMS performance under different user-editing scenarios

What this shows is that, within our current GTMS implementation, a lot of cutting and pasting would be required to take full advantage of the TM's content. Nevertheless, using only those proposals that fit "as is" is still viable.

Multiple Translation Evaluation

Using "oracle translations" to evaluate usability does not necessarily reflect the full potential of the approach, because it is assumed that for each pair (s, t) in the bitexts, t is the only valid translation for s[4]. Therefore, the results of our experiments should be viewed as a lower-bound on the re-usability of past translations.

A more realistic assessment could be obtained using a set of possible translations for each s, as proposed for example in Niessen et al. (2000). In a very limited experiment, we picked 5 source sentences from the *Hansard* test bitext, and asked 5 of our colleagues to translate each of them. We then measured the performance of the GTMS for each of these sentences, using the translation which obtained the best target cover. This corresponds to a scenario where the translator has a set of possible oracle translations to choose from, and picks the one that is most easily assembled using the TL sequences proposed by the GTMS.

In this experiment, precision and recall jumped from 15% and 9% respectively to 37% and 24%. While these results are certainly not significant given the size of the sample, they do indicate that the true potential of the system is actually higher than what our previous figures suggest.

Conclusions

The objective of our work was to evaluate the potential of a type of translation memory system capable of supplying a human translator with sub-sentential segments of target-language text. We have proposed an architecture based on a more flexible searching mechanism than found in existing TMS's.

Our experiments indicate that this strategy can produce substantial improvements in recall, while maintaining precision at reasonable levels, especially when the text to be translated is related to the content of the translation memory. As a point of reference, none of the source sentences we used for our tests could be found in the translation memory. In other words, existing TMS's would probably not have been very useful for these texts.

One important topic not discussed here is implementation. The requirements of a realistic GTMS implementation would be numerous and complex. For example, standard text-indexing procedures are not necessarily optimal for the type of sub-sequence searches we propose, and more complex structures (e.g. suffix trees) might be more appropriate. Statistical translation models are still bulky items, and much work remains to be done on sub-sequence alignment techniques before we reach an acceptable compromise between reliability and tractability.

The most important topic that this article does not address is probably the user interface. Clearly, this plays an essential role in the system and is a crucial factor of its usability. For the system to be viable, proposals must be made in such a way that translation re-use is easier than simply typing the text. This is especially important in light of the results of our experiments regarding the type of edit operations the user is allowed to perform on GTMS proposals. Efforts in this direction would possibly benefit from the work done as part of the *TransType* interactive MT project (Foster et al., 1997).

But in the end, what our research shows is that existing TMS's are extremely far from exploiting the full potential of translation memories. Finding better ways of extracting text at the sub-sentential level turns out to be a promising avenue.

Acknowledgements

Special thanks go to all members of the RALI who took part in our experiments.

References

Bédard, Claude (2001). Mémoire de traduction cherche traducteur de phrases…, to appear in *Traduire*.

Brown, Ralf D. (1996). Example-based Machine Translation in the Pangloss System. In *Proceedings of COLING 1996*, pages 169-174. Copenhague, Denmark.

Foster, George, Pierre Isabelle and Pierre Plamondon (1997). Target-text Mediated Interactive Machine Translation. *Machine Translation*, 21 (1-2).

Langlais, Philippe (2001). Combinaison de modèles markoviens pour la segmentation sous-phrastique de textes. Technical Report, RALI.

Langlais, Philippe and Michel Simard (2001). Récupération de segments sous-phrastiques dans une mémoire de traduction. In *Actes de TALN 2001.* Tours, France.

Macklovitch, Elliott and Graham Russell (2000). What's Been Forgotten in Translation Memory. In *Proceedings of AMTA 2000*. Cuernavaca, Mexico.

Macklovitch, Elliott, Michel Simard and Philippe Langlais (2000). TransSearch: A Free Translation Memory on the World Wide Web. In *Proceedings of the LREC 2000*, Athens, Greece.

Nielssen, Sonja, Franz Josef Och, Gregor Lensch and Hermann Ney (2000). An Evaluation Tool for Machine Translation: Fast Evaluation for MT Research. In *Proceedings of LREC 2000*, Athens, Greece.

Planas, Emmanuel and O. Furuse (1999). Formalizing Translation Memories. In *Proceedings of Machine Translation Summit VII,* pp 331-339.

Simard, Michel, George Foster and Pierre Isabelle (1992). Using Cognates to Align Sentences in Bilingual Corpora. In *Proceedings of TMI-92*, pp. 67-82, Montréal, Québec.

Wu, Dekai (1997). Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23 (3), pp. 377-404.

[4] Actually, the assumption is that only certain permutations of the words of t_i are valid translations of s_i.