TransType: a Computer-Aided Translation Typing System

Philippe Langlais and **George Foster** and **Guy Lapalme**
RALI/DIRO — Université de Montréal
C.P. 6128, succursale Centre-ville
H3C 3J7 Montréal, Canada
Phone: +1 (514) 343-2145
Fax: +1 (514) 343-5834
email: {felipe,foster,lapalme}@iro.umontreal.ca

Abstract

This paper describes the embedding of a statistical translation system within a text editor to produce TransType, a system that watches over the user as he or she types a translation and repeatedly suggests *completions* for the text already entered. This innovative Embedded Machine Translation system is thus a specialized means of helping produce high quality translations.

1 Introduction

TransType is a project set up to explore an appealing solution to the problem of using *Interactive Machine Translation* (IMT) as a tool for professional or other highly-skilled translators. IMT first appeared as part of Kay's MIND system (Kay, 1973), where the user's role was to help the computer analyze the source text by answering questions about word sense, ellipsis, phrasal attachments, etc. Most later work on IMT, eg (Blanchon, 1991; Brown and Nirenburg, 1990; Maruyama and Watanabe, 1990; Whitelock et al., 1986), has followed in this vein, concentrating on improving the question/answer process by having less questions, more friendly ones, etc. Despite progress in these endeavors, systems of this sort are generally unsuitable as tools for skilled translators because the user serves only as an advisor, with the MT components keeping overall control over the translation process.

TransType originated from the conviction that a better approach to IMT for competent translators would be to shift the focus of interaction from the *meaning* of the source text to the *form* of the target text. This would relieve the translator of the burden of having to provide explicit analyses of the source text and allow him to translate naturally, assisted by the machine whenever possible.

In this approach, a translation emerges from a series of alternating contributions by human and machine. The machine's contributions are basically proposals for parts of the target text, while the translator's can take many forms, including pieces of target text, corrections to a previous machine contribution, hints about the nature of the desired translation, etc. In all cases, the translator remains directly in control of the process: the machine must respect the constraints implicit in his contributions, and he or she is free to accept, modify, or completely ignore its proposals.

So TransType is a specialized text editor with an embedded Machine translation engine as one of its components. In this project we had to address the following problems: how to interact with the user and how to find appropriate multi-word units for suggestions that can be computed in real time.

2 The TransType model

2.1 User Viewpoint

Our interactive translation system is illustrated in figure 1 for an English to French translation. It works as follows: a translator selects a sentence and begins typing its translation. After each character typed by the translator, the system displays a proposed completion, which may either be accepted using a special key or rejected by continuing to type. This interface is simple and its performance may be measured by the proportion of characters or keystrokes saved in typing a translation. Note that, throughout this process, the translator remains in control, and the machine must continually adapt its suggestions to the translator's input. This differs from the usual machine translation set-ups where it is the machine that produces the first draft which

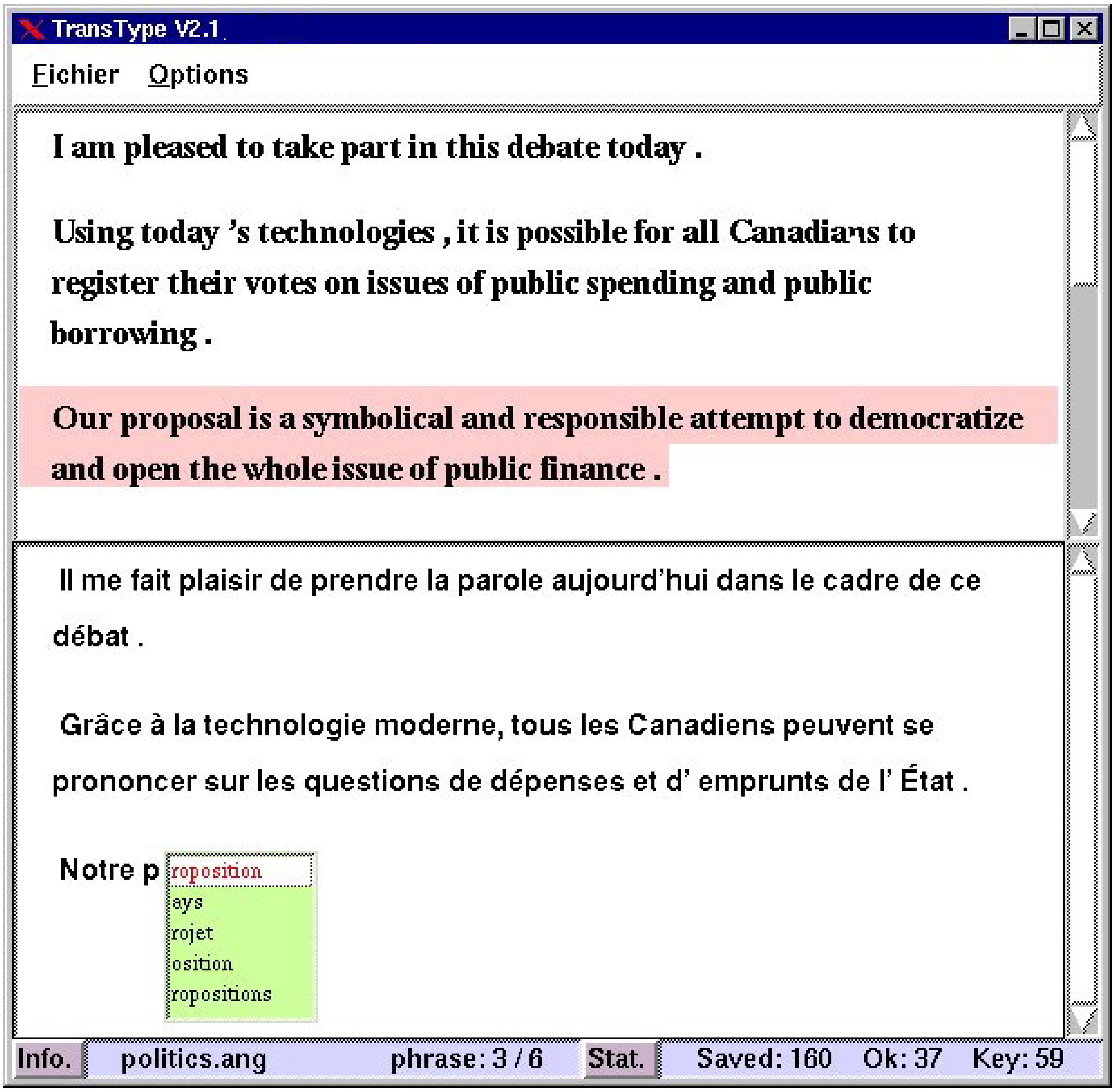

Figure 1: Example of an interaction in TransType with the source text in the top half of the screen. The target text is typed in the bottom half with suggestions given by the menu at the insertion point.

then has to be corrected by the translator.

The first version of TransType (Foster et al., 1997) only proposed completions for the current word. We are now working on predictions which extend to the next several words in the text. The potential gain from multiple-word predictions (Langlais et al., 2000) can be appreciated in the one-sentence translation task reported in table 1, where a hypothetical user saves over 60% of the keystrokes needed to produce a translation in a word completion scenario, and about 75% in a "unit" completion scenario

2.2 System Viewpoint

The core of TransType is a completion engine which comprises two main parts: an *evaluator* which assigns probabilistic scores to completion

This bill is very similar to its companion bill which we dealt with yesterday in the house of commons				
	word-completion task.		unit-completion task	
	pref.	completions	pref.	completions
ce	ce+	/loi · c/'	c+	/loi · c/e projet de loi
projet	p+	/est · p/rojet	-	
de	d+	/très · d/e	-	
loi	l+	/très · l/oi	-	
est	e+	/de · e/st	e+	/de · e/st
très	t+	/de · t/rès	t+	/de · t/rès
semblable	se+	/de · s/es · se/mblable	se+	/de · s/es · se/mblable
au	au+	/loi · a/vec	a+	/loi · a/u projet de loi sur
projet	p+	/loi · p/rojet	-	
de	d+	/loi · d/e	-	
loi	l+	/nous · l/oi	-	
que	qu+	/nous · q/ui · qu/e	qu+	/nous · q/ui · qu/e
nous	+	/nous	+	/nous
avons	av+	/nous · a/vec · av/ons	av+	/nous · a/vec · av/ons
examiné	ex+	/hier · e/n · ex/aminé	exa+	/à la chambre des communes e/n · ex/istence · exa/miné
hier	+	/hier	h+	/à la chambre des communes h/ier
à la	à+	/hier · à/ la	+	/à la chambre des communes
chambre	+	/chambre	-	
des	de+	/communes · d/e · de/s	-	
communes	+	/communes	-	
106 char.	23	20 accept. **43 keystrokes**	14	11 accept. + 1 correc. **26 keystrokes**

Table 1: A one-sentence session illustrating the word- and unit- completion tasks. The first column indicates the target words the user is expected to produce. The next two columns indicate respectively the prefixes typed by the user and the completions made by the system under a word-completion task. The last two columns provide the same information for the unit-completion task. The total number of keystrokes for both tasks is reported in the last line. + indicates the acceptance key typed by the user. A Completion is denoted by α/β where α is the typed prefix and β the completed part. Completions for different prefixes are separated by · .

hypotheses and a *generator* which uses the evaluation function to select the best candidate for completion.

2.2.1 The evaluator

The evaluator is a function $p(t|t', s)$ which assigns to each target-text unit t an estimate of its probability given a source text s and the tokens t' which precede t in the current translation of s. Our approach to modeling this distribution is based to a large extent on that of the IBM group (Brown et al., 1993), but it differs in one significant aspect: whereas the IBM model involves a "noisy channel" decomposition, we use a linear combination of separate predictions from a language model $p(t|t')$ and a translation model $p(t|s)$. Although the noisy channel technique is powerful, it has the disadvantage that $p(s|t', t)$ is more expensive to compute than $p(t|s)$ when using IBM-style translation models. Since speed is crucial for our application, we chose to forego it in the work described here. Our linear combination model is fully described in (Langlais and Foster, 2000) but can be seen as follows:

$$p(t|t', s) = \underbrace{p(t|t')\ \lambda(\Theta(t', s))}_{\text{language}} + \underbrace{p(t|s)\ [1 - \lambda(\Theta(t', s))]}_{\text{translation}} \quad (1)$$

where $\lambda(\Theta(t', s)) \in [0, 1]$ are context-dependent interpolation coefficients. $\Theta(t', s)$ stands for any function which maps t',s into a set of equivalence classes. Intuitively, $\lambda(\Theta(t', s))$ should be high when s is more informative than t' and low otherwise. For example, the translation model could have a higher weight at the start of sentence but the contribution of the language model can become more important in the middle or the end of the sentence.

2.2.2 The language model

We experimented with various simple linear combinations of four different French language models: a cache model, similar to the cache component in Kuhn's model (Kuhn and Mori, 1990); a unigram model; a triclass model (Derouault and Merialdo, 1986); and an interpolated trigram (Jelinek, 1990).

We opted for the trigram, which gave significantly better results than the other three models. The trigram was trained on the Hansard corpus (about 50 million words), with 75% of the corpus used for relative-frequency parameter estimates, and 25% used to reestimate interpolation coefficients.

2.2.3 The translation model

Our translation model is based on the linear interpolation given in equation 2 which combines predictions of two translation models — M_s and M_u — both based on an IBM-like model 2 (see equation 3). M_s was trained on single words and M_u was trained on both words and units.

$$p(t|s) = \underbrace{\beta.p_s(t|s)}_{\text{word}} + \underbrace{(1-\beta).p_u(t|\mathcal{G}(s))}_{\text{unit}} \quad (2)$$

where p_s and p_u stand for the probabilities given respectively by M_s and M_u. $\mathcal{G}(s)$ represents the new sequence of tokens obtained after grouping the tokens of s into units.

Both models are based on IBM translation model 2 (Brown et al., 1993) which has the property that it generates tokens independently. The total probability of the ith target-text token t_i is just the average of the probabilities with which it is generated by each source text token s_j; this is a weighted average that takes the distance from the generating token into account:

$$p(t_i|s) = \sum_{j=0}^{|s|} p(t_i|s_j)\ a(j|i, |s|) \quad (3)$$

where $p(t_i|s_j)$ is a word-for-word translation probability, $|s|$ is the length (counted in tokens) of the source segment s under translation, and $a(j|i, |s|)$ is the *a priori* alignment probability that the target-text token at position i will be generated by the source text token at position j; this is equal to a constant value of $1/(|s|+1)$ for model 1. This formula follows the convention of (Brown et al., 1993) in letting s_0 designate the null state. We modified IBM model 2 to account for invariant entities such as English forms that almost invariably translate into French either verbatim or after having undergone a predictable transformation e.g. numbers or dates. These forms are very frequent in the Hansard corpus.

2.3 The Generator

The task of the generator is to identify units matching the current prefix typed by the user, and pick the best candidate using the evaluation function. Given the real time constraints of an IMT system, we divided the French vocabulary into two parts: a small *active* component whose contents are always searched for a match to the current prefix, and a much larger *passive* part which comes into play only when no candidates are found in the active vocabulary. Both vocabularies are coded as tries.

The passive vocabulary is a large dictionary containing over 380,000 word forms. The active part is computed dynamically when a new sentence is selected by the translator. It relies on the fact that a small number of words account for most of the tokens in a text. It is composed of a few entities (tokens and units) that are likely to appear in the translation. In practice, we found that keeping 500 words and 50 units yields good performance.

3 Implementation

From an implementation point of view, the core of TransType relies on a flexible object oriented architecture, which facilitates the integration of any model that can predict units (words or sequence of words) from what has been already typed and the source text being translated. This part is written in C++. Statistical translation and language models have been integrated among others into this architecture (Lapalme et al., 2000).

The graphical user interface is implemented in Tcl/Tk, a multi-platform script language well suited to interfacing problems. It offers all the classical functions for text edition plus a pop-up menu which contains the more probable words or sequences of words that may complete the ongoing translation. The proposed completions are updated after each keystroke the translator enters.

4 Evaluation

We have conducted a theoretical evaluation of TransType on a word completion task, which assumes that a translator carefully observes each completion proposed by the system and accepts it as soon as it is correct. Under these optimistic conditions, we have shown that TransType allows for the production of a translation typing less than a third of its characters.

In order to better grasp the usefulness of TRANSTYPE, we also performed a more practical evaluation by asking ten translators to use the prototype for about one hour to translate isolated sentences. We first asked them to translate without any help from TRANSTYPE and then we compared their typing speed with TRANSTYPE suggestions turned on. Overall, translators liked the concept and found it very useful; they all liked the suggestions although it seemed to induce a literal style of translation. We also asked them if they thought that TRANSTYPE improved their typing speed and the majority of them said so; unfortunately the figures showed that none of them did so ... The typing rates are nevertheless quite good, given that the users were new to this environment and this style of looking at suggestions while translating. But interestingly this practical evaluation confirmed our theoretical evaluation that a translation can be produced with TRANSTYPE by typing less than 40% of the characters of a translation. Results of this evaluation and comparisons with our theoretical figures are further described in (Foster et al., 2000).

This experiment made us realize that this concept of real-time suggestions depends very much on the usability of the prototype; we had first developed a much simpler editor but its limitations were such that the translators found it unusable. So we are convinced that the user-interface aspects of this prototype should be thoroughly studied. But the TRANSTYPE approach would be much more useful if it was combined with other text editing tasks related to translation: for example TRANSTYPE could format the translation in the same way as the source text, this would be especially useful for titles and tables; it would also be possible to localize automatically specific entities such as dates, numbers and amounts of money. It would also be possible to check that some translations given by the user are correct with respect with some normative usage of words or terminological coherence; these facilities are already part of TRANSCHECK, another computer aided translation tool prototype developed in our laboratory (Jutras, 2000).

5 Conclusion

We have presented an innovative way of embedding machine translation by means of a prototype which implements an appealing interactive machine translation scenario where the interaction is mediated via the target text under production. Among other advantages, this approach relieves the translator of the burden of source analyses, and gives him or her direct control over the final translation without having to resort to post-edition.

Acknowledgements

TRANSTYPE is a project funded by the Natural Sciences and Engineering Research Council of Canada. We are greatly indebted to Elliott Macklovitch and Pierre Isabelle for the fruitful orientations they gave to this work.

References

Hervé Blanchon. 1991. Problèmes de désambiguïsation interactive et TAO personnelle. In *L'environnement Traductionnel*, Journées scientifiques du Réseau thématique de recherche “Lexicologie, terminologie, traduction”, pages 31–48, Mons, April.

Ralf D. Brown and Sergei Nirenburg. 1990. Human-computer interaction for semantic disambiguation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 42–47, Helsinki, Finland, August.

Peter F. Brown, Stephen A. Della Pietra, Vincent Della J. Pietra, and Robert L. Mercer. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312, June.

A.-M. Derouault and B. Merialdo. 1986. Natural language modeling for phoneme-to-text transcription. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 8(6):742–749, November.

George Foster, Pierre Isabelle, and Pierre Plamondon. 1997. Target-text Mediated Interactive Machine Translation. *Machine Translation*, 12:175–194.

George Foster, Philippe Langlais, Guy Lapalme, Dominique Letarte, Elliott Macklovitch, and Sébastien Sauvé. 2000. Evaluation of transtype, a computer-aided translation typing system: A comparison of a theoretical- and a user- oriented evaluation procedures. In *Conference on Language Resources and Evaluation (LREC)*, page 8 pages, Athens, Greece, June.

Frederick Jelinek. 1990. Self-organized language modeling for speech recognition. In A. Waibel and K. Lee, editors, *Readings in Speech Recognition*, pages 450–506. Morgan Kaufmann, San Mateo, California.

Jean-Marc Jutras. 2000. An automatic reviser: The TransCheck system. In *Applied Natural Language Processing 2000*, page 10 pages, Seattle, Washington, May.

Martin Kay. 1973. The MIND system. In R. Rustin, editor, *Natural Language Processing*, pages 155–188. Algorithmics Press, New York.

Roland Kuhn and Renato De Mori. 1990. A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 12(6):570–583, June.

Philippe Langlais and George Foster. 2000. Using context-dependent interpolation to combine statistical language and translation models for interactive machine translation. In *Computer-Assisted Information Retrieval*, Paris, April.

Philippe Langlais, George Foster, and Guy Lapalme. 2000. Unit completion for a computer-aided translation typing system. In *Applied Natural Language Processing 2000*, page 10 pages, Seattle, Washington, May.

Guy Lapalme, George Foster, and Philippe Langlais. 2000. La programmation orientée-objet pour le développement de modèles de langages. In Christophe Dony and Houari A. Sahraoui, editors, *LMO'00 – Langages et modèles à objets*, pages 139–147, Mont St-Hilaire, Québec, 27 Janvier. Hermes Science. Conférence invitée.

Hiroshi Maruyama and Hideo Watanabe. 1990. An interactive Japanese parser for machine translation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 257–262, Helsinki, Finland, August.

P. J. Whitelock, M. McGee Wood, B. J. Chandler, N. Holden, and H. J. Horsfall. 1986. Strategies for interactive machine translation: the experience and implications of the UMIST Japanese project. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 329–334, Bonn, West Germany.