TransType: Text Prediction for Translators

George Foster Philippe Langlais Guy Lapalme

RALI, Université de Montréal

{foster,felipe,lapalme}@iro.umontreal.ca

ABSTRACT

Text prediction is a novel form of interactive machine translation that is well suited to skilled translators. It has the potential to assist in several ways: speeding typing, suggesting possible translations, and averting translator errors. However, recent evaluations of a prototype prediction system showed that predictions can also distract and hinder translators if made indiscriminately. We demonstrate an experimental prototype intended to address this problem by selecting the prediction that has maximal expected benefit to the user in any given context. This leads it to make longer predictions where it is more certain and shorter ones—or none at all—in contexts where it is less certain.

Keywords

text prediction, interactive machine translation, user modeling, statistical translation models

1. INTRODUCTION

The idea of using text prediction as a tool for translators was first introduced by Church and Hovy as one of many possible applications for "crummy" machine translation technology [2]. Text prediction can be seen as a form of interactive MT that is well suited to skilled translators. Compared to the traditional form of IMT based on Kay's original work [5]—in which the user's role is to help disambiguate the source text—prediction is less obtrusive and more natural, allowing the translator to focus on and directly control the contents of the target text. Predictions can benefit a translator in several ways: by accelerating typing, by suggesting translations, and by serving as an implicit check against errors.

The TransType project at the Université de Montréal [6] was set up to explore the idea of a predictive tool for translators. It resulted in a prototype system for English to French translation that displays a short pop-up menu of completions for the next word or lexicalized multi-word unit after each character the translator types, as shown in figure 1. Although TransType is capable of correctly anticipating over 70% of the characters in a freely-typed translation, this does not mean that users can translate in 70% less time when using the tool. In fact, in a trial with skilled translators, the users' rate of text production *declined* by an average of 17% as a result of using the tool [7]. There are two main reasons for this. First, it takes time to read the system's proposals, so that in cases where they are wrong or too short, the net effect will be to slow the translator down. Second, translators do not always act "rationally" when confronted with a proposal; that is, they do not always accept correct proposals and they occasionally accept incorrect ones.

The system we are demonstrating is intended to address these problems by making proposals that maximize the expected benefit to the user in each context. Benefit is estimated from two components: a statistical translation model that gives the probability that a candidate prediction will be correct or incorrect, and a user model that determines the benefit to the translator in either case. The user model takes into account the cost of reading a proposal, as well as the random nature of the decision to accept it or not. This approach results in longer predictions in contexts where the translation model is confident, shorter ones where it is less so, and no predictions at all where it is very uncertain. In addition to taking user benefit into account, our new prototype also improves on the previous one by using more accurate translation models, and by using a decoder to predict arbitrary amounts of upcoming text, rather than just the next word or lexicalized unit. The user interface remains the same, except that at most one prediction is made in any context, rather than a menu of alternate predictions.

As implied above, the predictive tool consists of two main components: a graphical user interface to make predictions available to the translator, and a predictor to generate them. These are described in the following sections.

2. USER INTERFACE

As shown in figure 1, the user interface is similar to that of a commercial translator's workstation, with a split screen for simultaneous display of source and target texts, and standard editing capabilities. The current interface is implemented in TCL/TK. To use it, a translator selects a sentence in the source text and begins typing its translation. After each character is typed, the system may display a prediction for one or more upcoming words (only one proposal, not a menu as shown in figure 1). These suggestions are continually adapted to conform to the translator's input. At any point, the translator has the option of incorporating a suggestion into the target text with a special keystroke or mouse action, or ignoring the proposals and continuing to type normally.

.

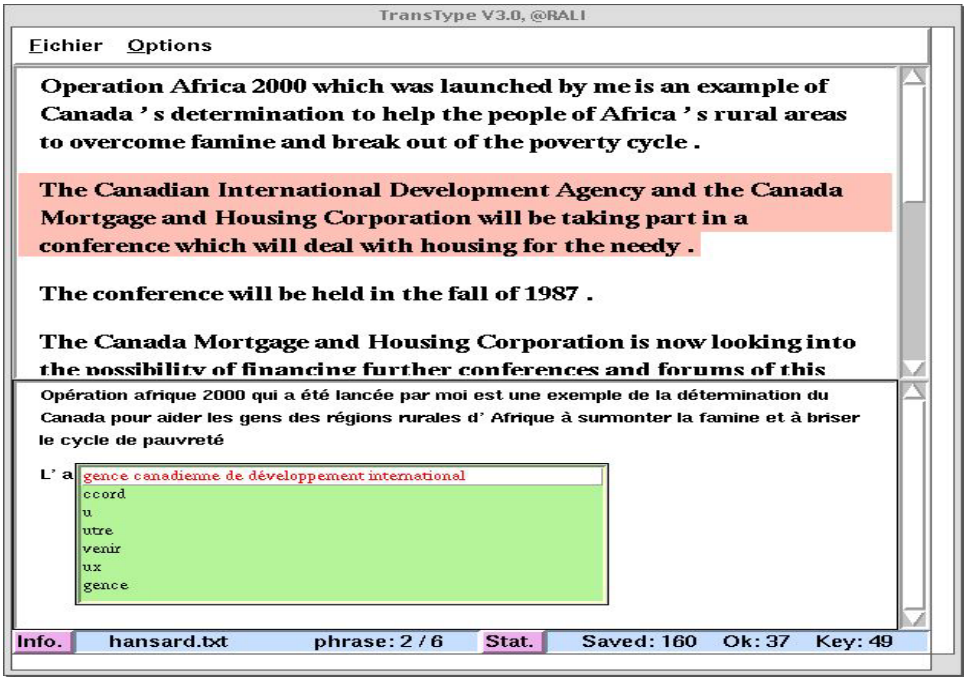

Figure 1: Screen dump for the TransType prototype. The source text is shown in the top half of the screen, and the target text is typed in the bottom half, with suggestions given by the menu at the cursor position.

3. PREDICTOR

In this section, we briefly outline how the predictor works; further details are available in [4]. The basic task is to find the prediction $\hat{\mathbf{x}}$ that maximizes the expected benefit to the user:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmax}}\ B(\mathbf{x}, \mathbf{h}, \mathbf{s}), \tag{1}$$

where $B(\mathbf{x}, \mathbf{h}, \mathbf{s})$ measures typing time saved, $\mathbf{s}$ is a source sentence, and $\mathbf{h}$ is a prefix of its translation (the target text before the current cursor position). Figure 2 gives an example.

To model $B(\mathbf{x}, \mathbf{h}, \mathbf{s})$, we begin with the major simplifying assumption that the user edits only by erasing wrong characters from the end of a proposal. Given a TransType-style interface where acceptance places the cursor at the end of a proposal, this is the most common editing method, and it gives a conservative estimate of the cost attainable by other methods. With this assumption, the key determinant of benefit is the length of the correct prefix of $\mathbf{x}$, so the expected benefit can be written as:

$$B(\mathbf{x}, \mathbf{h}, \mathbf{s}) = \sum_{k=0}^{l} p(k|\mathbf{x}, \mathbf{h}, \mathbf{s})\ B(\mathbf{x}, \mathbf{h}, \mathbf{s}, k), \tag{2}$$

where $p(k|\mathbf{x}, \mathbf{h}, \mathbf{s})$ is the probability that exactly k characters from the beginning of $\mathbf{x}$ are correct, l is the length of $\mathbf{x}$, and $B(\mathbf{x}, \mathbf{h}, \mathbf{s}, k)$ is the benefit to the user given that the first k characters of $\mathbf{x}$ are correct.

Equations (1) and (2) define three main problems: estimating the prefix probabilities $p(k|\mathbf{x}, \mathbf{h}, \mathbf{s})$, estimating the user benefit function $B(\mathbf{x}, \mathbf{h}, \mathbf{s}, k)$, and searching for $\hat{\mathbf{x}}$. The following three sections describe our solutions to these.

3.1 Translation Model

The prefix probabilities $p(k|\mathbf{x}, \mathbf{h}, \mathbf{s})$ come from a maximum-entropy translation model for $p(w|\mathbf{h}, \mathbf{s})$, where w is a word that follows $\mathbf{h}$ in the translation of $\mathbf{s}$. This model is essentially a maximum-entropy analog of a linear combination of a trigram language model for $p(w|\mathbf{h})$ and the classical IBM model 2 [1] for $p(w|\mathbf{s})$. It was shown in [3] to have significantly lower test corpus perplexity than an equivalent linear combination used in the previous TransType prototype. It also supports very rapid searches for upcoming text, chiefly by virtue of the fact that it completely ignores the relation between $\mathbf{h}$ and $\mathbf{s}$.

$\mathbf{s}$: Let us return to serious matters.

$\mathbf{t}$: On va r$\overbrace{\text{evenir}}^{\mathbf{h}}\ \overbrace{\text{aux choses sérieuses.}}^{\mathbf{x}^*}$

$\mathbf{x}$: *evenir à*

Figure 2: Example of a prediction for English to French translation. $\mathbf{s}$ is the source sentence, $\mathbf{h}$ is the part of its translation that has already been typed, $\mathbf{x}^*$ is what the translator wants to type, and $\mathbf{x}$ is the prediction.

The technique for deriving the required probabilities $p(k|\mathbf{x}, \mathbf{h}, \mathbf{s})$ from the model $p(w|\mathbf{h}, \mathbf{s})$ is straightforward but rather tedious to describe. Briefly, it involves first expressing $p(k|\mathbf{x}, \mathbf{h}, \mathbf{s})$ in terms of string probabilities $p(\mathbf{x}'|\mathbf{h}, \mathbf{s})$, then calculating the string probabilities by summing over all compatible token sequences. Further details are given in [4].

3.2 User Model

The purpose of the user model is to determine the expected benefit $B(\mathbf{x}, \mathbf{h}, \mathbf{s}, k)$ to the translator of a prediction $\mathbf{x}$ whose first k characters match the text that the translator wishes to type. This will depend heavily on whether the translator decides to accept or

reject the prediction, so the first step in our model is the following expansion:

$$B(\mathbf{x}, \mathbf{h}, \mathbf{s}, k) = \sum_{a \in \{0,1\}} p(a|\mathbf{x}, \mathbf{h}, \mathbf{s}, k)\ B(\mathbf{x}, \mathbf{h}, \mathbf{s}, k, a),$$

where $p(a|\mathbf{x}, \mathbf{h}, \mathbf{s}, k)$ is the probability that the translator accepts or rejects $\mathbf{x}$, $B(\mathbf{x}, \mathbf{h}, \mathbf{s}, k, a)$ is the benefit they derive from doing so, and a is a random variable that takes on the values 1 for acceptance and 0 for rejection. The first two quantities are the main elements in the user model. The parameters of both were estimated from data collected during the TransType trial described in [7], which involved nine accomplished translators using a prototype prediction tool for approximately half an hour each. In both cases, estimates were made by pooling the data for all nine translators, so as to reflect as accurately as possible the characteristics of an average translator.

3.3 Search

Searching directly through all character strings $\mathbf{x}$ in order to find $\hat{\mathbf{x}}$ according to equation (1) would be very expensive. The fact that $B(\mathbf{x}, \mathbf{h}, \mathbf{s})$ is non-monotonic in the length of $\mathbf{x}$ makes it difficult to organize efficient dynamic-programming search techniques or use heuristics to prune partial hypotheses. Because of this, we adopted a fairly radical search strategy that involves first finding the most likely sequence of words of each length, then calculating the benefit of each of these sequences to determine the best proposal. The algorithm is:

1. For each length $m = 1 \ldots M$, find the best word sequence of that length:

$$\hat{\mathbf{w}}_m = \operatorname*{argmax}_{\mathbf{w}_m} p(\mathbf{w}_m|\mathbf{h}, \mathbf{s}),$$

2. Convert each $\hat{\mathbf{w}}_m$ to a corresponding character string $\hat{\mathbf{x}}_m$.

3. Output $\hat{\mathbf{x}} = \text{argmax}_m\ B(\hat{\mathbf{x}}_m, \mathbf{h}, \mathbf{s})$, or the empty string if all $B(\hat{\mathbf{x}}_m, \mathbf{h}, \mathbf{s})$ are non-positive.

Step 1 is carried out using a Viterbi beam search with the translation model $p(w|\mathbf{h}, \mathbf{s})$. To speed this up, the search is limited to an *active vocabulary* of target words likely to appear in translations of $\mathbf{s}$, defined as the set of all words connected by some word-pair feature in our translation model to some word in $\mathbf{s}$. Step 2 is a trivial deterministic procedure that mainly involves deciding whether or not to introduce blanks between adjacent words (eg yes in the case of *la* + *vie*, no in the case of *l'* + *an*). Step 3 involves a straightforward evaluation of m strings according to equation (2). With this algorithm, the average time to make predictions with $M = 5$ is less than .02 seconds on a 1.2GHz processor. This does not cause a perceptible delay to users of our prototype.

4. FUTURE WORK

Each of the components of the predictor described in the previous sections offers some fairly obvious possibilities for improvement. The translation model could be made to capture some of the dependence between $\mathbf{h}$ and $\mathbf{s}$, ideally in such a way as to preserve its efficient search properties. It could also be made to adapt to the translator's previous input, which should be a valuable source of information about what translations are appropriate in the current context. The user model could also be made adaptive, to conform to the preferences and characteristics of the current user. Finally, there are numerous possibilities for improving the search for $\hat{\mathbf{x}}$, for instance using the n best word sequences of each length (rather than just one) as input to steps 2 and 3 of the algorithm above.

The user interface could also be improved in various ways. One obvious possibility is to have a passive completion mode in which the user would request predictions only when desired rather than having them be made automatically. An extension to this would be to make proposals when the user paused, signalling possible receptiveness to input from the tool.

We look forward to pursuing these and other research avenues in the TransType 2 project, due to begin this year under the auspices of the European Commission's 5th Framework programme.

5. REFERENCES

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

[2] Kenneth W. Church and Eduard H. Hovy. Good applications for crummy machine translation. *Machine Translation*, 8:239–258, 1993.

[3] George Foster. A Maximum Entropy / Minimum Divergence translation model. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, Hong Kong, October 2000.

[4] George Foster, Philippe Langlais, and Guy Lapalme. User-friendly text prediction for translators. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, PA, 2002. Submitted.

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

[6] Philippe Langlais, George Foster, and Guy Lapalme. Unit completion for a computer-aided translation typing system. *Machine Translation*, 15(4):267–294, December 2000.

[7] Philippe Langlais, Guy Lapalme, and Marie Loranger. Transtype: From an idea to a system. *Machine Translation*, 2002. To Appear.