Computer-Aided Translation Backed by Machine Translation

Ondřej Odcházel and Ondřej Bojar

Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University, Prague, Czech Republic

October 30, 2009

Abstract

A number of tools to support translators (computer-aided translation, CAT) exist, as there are many systems of machine translation (MT). So far, the integration of the two system types was little or none. The aim of this paper is to examine a tighter coupling of MT and CAT.

We introduce our web-based CAT tool implemented using the modern AJAX technology that communicates with Moses MT system on the server side to provide the translator with suggested translations of individual phrases of the source sentence as well as several options of the complete continuation of the output sentence. The suggested continuation is based on what has been already translated and what the user has already written as the output. Hopefully, the proposed user interface and the MT system at the back end will accelerate and simplify the process of translation.

Keywords: Computer-Aided Translation, Machine Translation

1 Motivation

There are many software solutions for computer-aided translation (CAT) and the term itself is very broad. In many cases, CAT implies as little as some filters for input/output formats (like HTML, ODF, Microsoft Word etc.) with no support for the translation itself.

More sophisticated systems facilitate the use of translation memories (TM). Sentences (usually referred to as segments or translation units) that have a translation available in the TM are offered for an update only.

Machine translation (MT) systems on the other hand focus on providing a single-best output for a given input sentence.

We attempt to combine the two complementary approaches in a tighter fashion. The structure of the paper is as follows: Section 2 provides examples of current solutions of CAT tools, Section 3 briefly summarizes the phrase-based MT model and an open-source implementation in the system Moses. Section 4 describes the contribution of this paper, our system AJAX CAT.

2 Current CAT Solutions

Out of the many CAT software solutions, we briefly describe three examples most related to our project.

2.1 OmegaT

A fine example of a CAT tool is OmegaT[1]. It is a free desktop software, that allows translators to import custom glossaries and translation memories. It is intended to be used by professional translators. During the translation, the user is provided with segments from translation memory. These advices do not need to match exactly to the currently translated segment, because OmegaT also implements fuzzy matching to items available in the translation memory.

OmegaT has no access to any MT system and the fuzzy-matching segments available in TM are only provided for reference, not anyhow recombined to propose the translation.

2.2 Google Translator Toolkit

Recently, Google revealed its CAT tool. Google Translator Toolkit (GTT[2]) is a tool based on Google Translate MT system. When starting the translation in GTT, the source text is translated by Google MT and the user's task is to post-edit the MT output. Additionally, the user is allowed to import custom translation memories or glossaries and share them with other users. All text submitted to GTT is probably planned to extend the training data of Google MT.[3]

GTT is a nice example of an AJAX application, i.e. a lightweight web-based solution that seamlessly communicates with an underlying server.

2.3 Caitra

Caitra[4] [2] is an experimental translation tool developed by the Machine Translation Group at the University of Edinburgh. Caitra illustrates a fine-grained assistance that an MT system can offer to human translators. Unlike TMs that suggest usually translations of whole sentences, caitra presents multiple translation options of short subsequences of words of the input sentence. The user (translator) is then allowed to construct the translation by both typing as well as by clicking on the selected translation options. We use a similar principle in our solution, see Section 4 below.

3 Phrase-Based Machine Translation

Current state-of-the-art generic (i.e. more or less language independent) MT systems are statistical and phrase-based (PBMT, see e.g. [1, 3]). The "phrases" are simply any contiguous sequences of words with a known translation, as extracted automatically from a parallel corpus. Often, the phrases span across the boundaries of linguistically motivated constituents of the sentence, e.g. *as segments or* is a valid "phrase" but not a constituent in most grammars.

[1] `http://www.omegat.org/`

[2] `http://translate.google.com/toolkit/`

[3] Specifically, the terms of service say: By submitting your content through the Service, you grant Google the permission to use your content permanently to promote, improve or offer the Services.

[4] `http://www.caitra.org/` or `http://tool.statmt.org/`

While TM-based approaches to CAT use complete sentences as translation units (and employ some fuzzy matching to improve the recall), phrases in phrase-based MT are hardly ever longer than a few words and usually an arbitrary limit at e.g. 10 words is set when automatically extracting the phrases from the corpus.

3.1 Translation and Language Models

Phrase-based machine translation is built upon the assumption that any sentence can be translated by splitting it to a sequence of (multi-word) phrases, translating the phrases more or less independently and concatenating the translations, possibly with some reordering of the phrases. Local word reordering often happens at no computational cost within phrases.

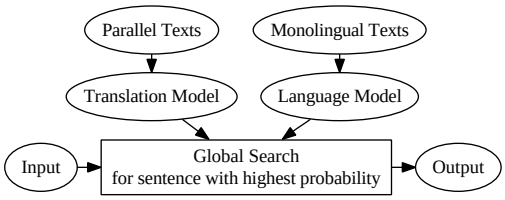

Figure 1: The architecture of a phrase-based MT system.

The architecture of a PBMT system is depicted in Figure 1: The "translation model" is constructed automatically from a parallel corpus and contains the mapping of source phrases to target phrases, associated with a probability (score, cost) of the translation. The "language model" captures frequent sequences of words in the target language only, regardless any segmentation into phrases. The language model can thus be collected from monolingual target-side data only, which are often available in larger quantities. The main component of a PBMT is the search described in the next section.

The quality of the translation depends on the language pair (e.g. languages with a different word order or richer morphology are harder to capture for the phrase-based model) and on the appropriateness of the training parallel and monolingual corpora. The larger the corpora and the more they match the domain of the test set, the better the translation quality.

3.2 Search in PBMT

PBMT systems translate sentences in two steps. In the first step, all possible translation options (see Figure 3 below) matching portions of the input sentence are found in the translation model. In the second step, a beam search is used to choose the best combination and order of translation options covering each input word exactly once. As there are exponentially many possible reorderings, the search has to be severely restricted (only reorderings deviating little from the monotone one are considered) and efficient data structures must be used.

During the search, many competing partial hypotheses are gradually added to a common data structure called "lattice" or "word graph", see Figure 2. Each node of the graph corresponds to one partial hypothesis and represents the output sequence so far, the input words covered (see the stars and dashes in Figure 2) and the score of the hypothesis. Starting from a blank hypothesis, partial hypotheses are expanded by extending the output sequence and covering some more input words (from anywhere in the sentence, thus choosing the reordering). Low scoring partial hypotheses are not expanded any further. The highest scoring hypothesis

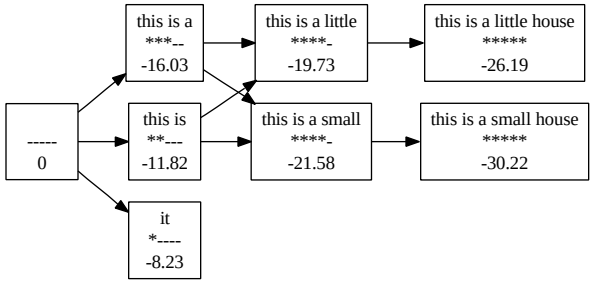

Figure 2: A sample lattice of (partial) output hypotheses for the input sentence *Das ist ein kleines Haus.*

that covers all input words is finally selected as the single best output, i.e. *This is a little house* in our example. We also see that some paths may lead to the same output. The example in Figure 2 is obviously small, usually there are a few thousands of nodes in the lattice.

4 Our Proposal: AJAX CAT

Despite the recent great progress in statistical machine translation, these systems will probably never be able to generate perfect translations on their own. However, significant speedups as well as improvements in translation quality are achieved if human translators are provided with some assistance from MT, see [2] for an evaluation.

The online application we propose is a CAT system, that will combine the force of human and machine translation. We will offer the translator an opportunity to explore a larger set of candidate translations as suggested by the MT system and the option to easily select the preferred translation by clicking or typing the beginnings of phrases in an auto-complete fashion. The automatic completion should be as little obtrusive as possible in case the user prefers to write the translation without the aid of MT.

The proposal goes clearly beyond plain TM-backed CAT systems because the full-fledged MT in the background has the ability to combine parts of items from TMs to translate unseen sentences. We also differ from GTT (Section 2.2) as we will provide the user with a variety of translation options, not a single-best output of the MT. Finally caitra, the closest tool similar to the one we propose, has a rather limited capabilities when proposing the continuation: only a single candidate for the next word is offered.

4.1 Table of Translation Options

As mentioned in Section 3, the first step in PBMT is the collection of phrasal translations available in the translation model. Traditionally, they are called "translation options" and can be presented in a compact table, clearly indicating which span of words in the source sentence they correspond to, see Figure 3.

In caitra, translation options are color-coded to indicate the score of competing translation options. Apart from the color-coding, we also consider implementing various filtering techniques of the translation options. For instance, proficient translators may wish to see only translation options for technical terms.

Clicking on translation options inserts them to the output text box so if the translation model contains relevant phrases, the output translation can be constructed simply by clicking.

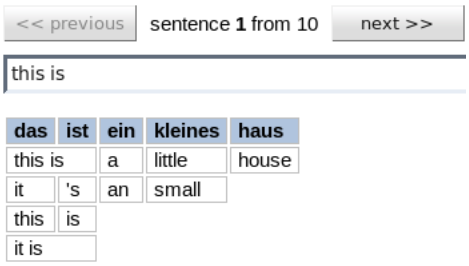

Figure 3: The table of translation options suggests translations for various input phrases (spans) in the input sentence *Das is ein kleines Haus.*

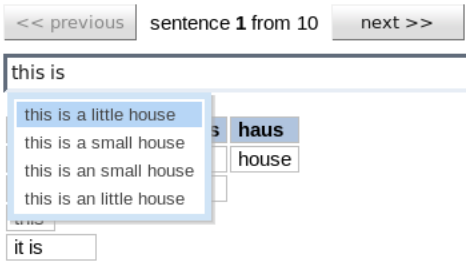

Figure 4: The suggestion box provides probable continuations of the output sentence.

4.2 Suggestion Box

The main feature of our CAT tool is the suggestion box aimed at accelerating the typing. The suggestion box provides a list of possible continuations of the output sentence, see Figure 4. The suggestions are selected from the lattice described in Section 3.2. There are two main issues with the suggestion box:

First, the list of suggestions has to be short, diverse and relevant. The lattice of output hypotheses constructed by Moses includes scores of the individual paths. We can sort the suggestions based on the score but this does not prevent us from flooding the user with too many options differing too little. The set of suggestions can be restricted by considering only a few more words beyond what has been already written instead of complete hypotheses. Note however, that even these partial suggestions should be sorted based on the *final* score because it is quite common that less promising partial hypotheses turn out as better scoring further in the search. We also consider increasing the diversity of the suggestion box by further shortening the lookahead of some suggestions if they would explode to too many options at the next word.

Second, the continuations must not contain words that have been already translated. While we know for sure which source words were covered if the user clicked on a translation option, we have to guess in cases where she preferred typing. The guess is based on the translation options table: we search for translation options matching parts of the output so far and mark the corresponding source spans as covered. In order to allow for a manual correction of the guess e.g. when the user has used a completely different word, we allow the user to click the source words in the translation options table to toggle their status.

5 Implementation Details

Our solution–AJAX CAT–is an online application. The architecture of such applications can be usually divided into the server and client sides. In our case, the main component of the server side is Moses used to generate detailed MT output (translation options tables and lattices). The client side is an AJAX application presenting the information to the user and handling his or her actions.

5.1 AJAX CAT Server

Given an input sentence, the server side of AJAX CAT uses Moses to prepare the translation options table and the lattice of hypotheses. The lattice serves then as the basis of our suggestion box.

The construction of the translation options table is straightforward: Moses prints it when launched in a more verbose mode. The planned filtering of the translation options table will happen at the server side of AJAX-CAT to avoid unnecessary network traffic. Moreover, the translation options table can be pre-computed for all sentences of the input document so the user does not have to wait for the MT system.

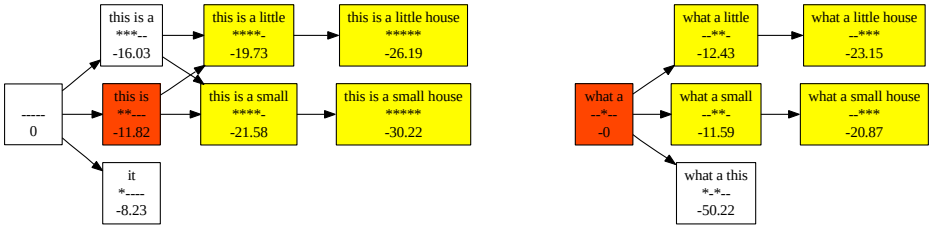

Figure 5: Selecting paths from the lattice for the suggestion box. In the left hand side picture, the user's output so far matched a path in the pre-computed lattice. In the right hand side picture, the user has started differently and the lattice had to be constructed from scratch.

The lattice of possible translations is pre-computed as well. However, if the translator deviates from the paths explored enough in the lattice (remember, not all partial hypotheses are expanded), we have to run Moses again to obtain the missing paths. To facilitate this, we will modify Moses in order to support translating from a non-empty hypothesis.

Both situations are depicted in Figure 5. The first situation illustrates how the pre-computed lattice is used even after the user has provided some output, namely the words *This is*. Because the pre-computed lattice contains enough matching paths, we move to the node corresponding to the output so far (highlighted in orange). The yellow highlighting then indicates paths selected for the suggestion box, with the hypothesis *This is a little house* scoring higher and thus appearing first in the suggestion box.

The second situation occurs, if the user decides to translate the sentence differently, e.g. to start with the words *What a*. None of the paths in the pre-computed lattice can be followed and we ask Moses to provide us with a new lattice based on the given beginning. Because the word *a* appears as the translation option of (still untranslated) *das*, we assume that the user has covered *das* with her output. Again, the colors indicate which paths will be selected for the suggestion box.

5.2 AJAX CAT Client Side

The client side of AJAX CAT is still very much work in progress. The current version implements just enough to demonstrate the MT integration but core functions like providing custom input text are still missing. Once finished, AJAX CAT will allow users to log in and upload source texts. The texts will be automatically segmented into sentences and offered for translation. The output could be later downloaded back from the server.

Another equally promising option is to include AJAX CAT specific features to existing CAT tools. We plan to keep all the important code at the server side, possibly allowing even the integration of our suggestion box to desktop CAT tools (provided there is an Internet connection to the server side).

To briefly summarize client-side implementation details, AJAX CAT user interface is developed using Google Web Toolkit (GWT [5]). In GWT, applications are coded in the Java programming language and compiled to JavaScript (later interpreted in web browsers). GWT isolates the programmer from individual browser quirks and generates more or less universal JavaScript, following the AJAX (Asynchronous JavaScript and XML) principle where only relevant pieces of the web page are modified and redrawn. The end user experience is then rather close to a regular desktop application: immediate response to user actions and the complete context preserved between the actions.

6 Conclusion and Future Research

We presented an experimental user interface for a CAT tool that closely cooperates with an MT system. The key new feature of our AJAX CAT is the suggestion box based on a set of candidate translations provided by the MT system.

At the current stage of the development, AJAX CAT should be seen as a mere prototype of the user interface. Future versions will include the inevitable functionality of uploading custom source texts etc. Alternatively, we may choose to integrate AJAX CAT features to any other CAT system, relying on AJAX CAT server as implemented.

In future research, we would like to evaluate the utility of the suggestion box in a small-scale experiment with human translators. We also consider examining, whether the choices of a particular item in the suggestion box could be used for model adaptation on the fly.

7 Acknowledgment

The work on this project has been supported by the grants FP7-ICT-2007-3-231720 (EuroMatrix Plus) and MSM 0021620838.

References

[1] Philipp Koehn. Pharaoh: A Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In Robert E. Frederking and Kathryn Taylor, editors, *AMTA*, volume 3265 of *Lecture Notes in Computer Science*, pages 115–124. Springer, 2004.

[2] Philipp Koehn and Barry Haddow. Interactive Assistance to Human Translators using Statistical Machine Translation Methods. In *MT Summit XII*, 2009.

[5] `http://code.google.com/webtoolkit/`

[3] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.