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

June 30, 2009

1 Introduction

The aim of our project is to develop an online application to support translators (computer-aided translation, CAT) based on the modern AJAX technology on the client side and Moses machine translation (MT) system on the server side. The tool should simplify and accelerate the process of text translation.

2 Motivation and Current Solutions

There are many software solutions for 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, DOC 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. A fine example of this kind of software is e.g. OmegaT.

A different approach is used in caitra, 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 to 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 correct translation options.

Recently, also Google revealed its CAT tool. Google Translator Toolkit (GTT) is a tool based on Google Translate MT system. When you start your translation in GTT, the source text is translated by MT and your task is to improve the quality of the translation. Also you are able to import translation memories or glossaries, which would help you in the process. GTT is a fine example of an AJAX aplication allowing users seamless communication with the server.

3 Our Proposal: AJAX CAT

The online application we propose is a CAT system, that will offer the translator an opportunity to explore the whole set of candidate translations as suggested by the MT system and the ability to easily select the preferred translation by clicking or typing beginnings of selected 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 do with the full-fledged MT in the background gaining the ability to combine parts of items from TM to translate unseen sentences. We also differ from GTT as we will provide the user with a variety of translation options. Finally caitra, the closest tool similar to the one we propose, does not provide automatic text completion when typing and it does not employ the lattice of complete hypotheses; all that is being presented to the user is a table of phrase translation options.

4 Server Side — Moses

On the server side of our online application, we will use Moses [1], the state-of-the-art open-source statistical phrase-based machine translation system. In Moses you can easily perform beam-search, which allows you to quickly find the most convenient translation among the exponential number of choices. But in many cases the result is not suitable and a human translator would probably choose another wording. For such cases, we will allow the user to use the client side of the application and stear the selection of the MT system. Internally, Moses constructs its best translation by choosing a path in word lattice (directed acyclic word graph) of target options. The main purpose of our work is thus to present the user with a concise representation of several paths in the lattice, collect his or her choices moving further down the paths.

The responsiveness of the user interface is crucial. The user does not want to wait even a single second for the response from the server between his choices. In order to improve the time of the response, Moses will have to work asynchronously, preparing translation of sentences in advance. In the first attempt, we will rely on a fast network connection between the server and the client but for scenarios where this can not be guaranteed even the soon-to-be explored segments may need a pre-caching on the client side.

An interesting feature would be to generate output lattices in accordance with the part of the sentence that has already been translated. During the translation of the sentence, the advices of MT could become more accurate. We leave this option for future as there are currently no means to partially constrain output in Moses.

Another way to improve the accuracy of suggestions is to use more accurate vocabularies. In the beginning of the translation, the MT will be able to see the whole text, so it can set up its vocabularies and translation instruments for different kinds of text. Again, there is no established means for domain adaptation in Moses.

5 Client Side — Google Web Toolkit

In recent years, we have seen a big development in abilities of web applications. The group of techniques which allowed this development is called AJAX (asynchronous JavaScript and XML). AJAX allows you to communicate between client and server without refreshing the whole web page, highly increasing the percieved responsiveness. In our project, we will use the Google Web Toolkit (GWT), a tool for an easy development of AJAX applications. GWT

allows to develop applications in Java programming language with full comfort of an integrated development environment. GWT then translates the Java code into JavaScript application.

Using GWT, the developer does not have to spend his time on work-arounds for browser quirks. Many widgets and panels are also available in GWT, so the developer does not need to spend a long time with the presentation layer to make reasonable design. There is also a good support for JSON data format in GWT. We would like to use JSON for communication with the server side. With GWT, you can also easily communicate with the application using keyboard arrow keys. This will help user to choose the best path through the lattice from Moses.

Although we plan to integrate Moses as the underlying MT system, other MT systems could be used instead as long as they are able to provide a set of candidate translations for a given sentence.

6 Conclusion

In our project, we will investigate methods that tightly couple and MT system with an online text editor to support the process of translation. The resulting application will provide a concise view of candidate translations as suggested by the system and respond to user input (clicking or typing) to select the preferred path or deviate from the suggested hypotheses completely.

The key difference of our proposal to other CAT tools is the ability to suggest translation of unseen sentences and/or the ability to present several candidates for translation of the complete sentence.

The claim we hope to support is that with the use of a good user interface, an MT system can accelerate and simplify the process of translation.

References

[1] 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.

Moses - http://www.statmt.org/moses/

Google Web Toolkit - http://code.google.com/intl/cs/webtoolkit/

Google Translator Toolkit - http://translate.google.com/toolkit/

OmegaT - http://www.omegat.org/

caitra - http://tool.statmt.org/