Building a Custom Machine Translation Engine as part of a Postgraduate University Course: a Case Study

Michael Farrell
IULM University
Milan, Italy

Abstract

In 2015, I was asked to design a postgraduate course on machine translation (MT) and post-editing. Following a preliminary theoretical part, the module concentrated on the building and practical use of custom machine translation (CMT) engines. This was a particularly ambitious proposition since it was not certain that students with undergraduate degrees in languages, translation and interpreting, without particular knowledge of computer science or computational linguistics, would succeed in assembling the necessary corpora and building a CMT engine. This paper looks at how the task was successfully achieved using KantanMT to build the CMT engines and Wordfast Anywhere to convert and align the training data.

The course was clearly a success since all students were able to train a working CMT engine and assess its output. The majority agreed their raw CMT engine output was better than Google Translate's for the kinds of text it was trained for, and better than the raw output (pre-translation) from a translation memory tool.

There was some initial scepticism among the students regarding the effective usefulness of MT, but the mood clearly changed at the end of the course with virtually all students agreeing that post-edited MT has a legitimate role to play.

1 Introduction

After teaching an undergraduate course on Computer Tools for Translators and Interpreters for six years at the International University of Languages and Media (IULM), Milan, Italy, I was asked to design a postgraduate course module specifically aimed at teaching the use of machine translation and post-editing as part of a Master's Degree in Specialist Translation and Conference Interpreting[1]. The course module began with a brief summary of the history of machine translation from its early stages, full of optimism, to the slow-down in the 1960s (ALPAC report), and on to today's more realistic and pragmatic application. It then went on to a simplified discussion of the theoretical aspects of rule-based and statistical machine translation systems, and a brief outline of neural machine translation. It also laid out the concept and goals of post-editing, illustrated the benefits of pre-editing and controlled language authoring, and explained some machine translation quality assessment techniques. In addition there were practical exercises on pre-editing, controlled language authoring and post-editing. Once this preliminary part was out of the way, after the first semester, the course moved on to the practical use of custom machine translation (CMT) engines. This was a particularly ambitious and challenging proposition since it was not at all certain that a group of students with undergraduate degrees in languages, translation and interpreting, without particular knowledge of computer science or computational linguistics, would succeed in putting together the necessary corpora and building a CMT engine. Another aim was to keep the cost to the university and to the students as low as possible.

[1] Machine Translation and Post Editing, Course Module Syllabus, International University of Languages and Media (IULM), Milan, Italy: http://bit.ly/2wxitJZ

Proceedings of the 39th Conference Translating and the Computer, pages 35–39,
London, UK, November 16-17, 2017.

2 Methods

After comparing various commercial programs and platforms for the building of custom machine translation engines (notably including Slate Desktop [2] and Lilt [3]), I opted for KantanMT[4]. The deciding factors were:

- KantanMT is cloud-based, and can therefore be used by students at home;
- KantanMT provides Library data, in case the bilingual corpora produced by the students do not reach the critical mass required to get meaningful output from the engine built:
- KantanMT's generous Academic Partner Programme.

The Academic Partner Programme provides access and use of the platform free of charge for students and lecturers for the duration of the course module, and one-to-one online training for lecturers to help create lesson plans. Besides allowing students to build custom machine translation engines, the platform also gives them a feel for the automatically generated evaluation metrics (Bilingual Evaluation Understudy [BLEU], F-Measure and Translation Edit Rate [TER]).

In order to create the corpora needed to train our Italian to English CMT engines, we contacted several companies, all of which freely publish user manuals for their products on the Internet in several languages. We asked permission to use their data for teaching purposes, and two firms replied: Philips[5] and Smeg (Smalterie Metallurgiche Emiliane Guastalla)[6]. Both companies market products in fairly limited domains, thus making their manuals ideal for building bilingual corpora to train CMT engines. One of our aims was precisely to restrict the domain sufficiently to reduce post-editing requirements to a bare minimum.

At the time of the course KantanMT could only be used to build statistical machine translation (SMT) engines. The neural machine translation version was not available to Academic Partners. To build an SMT engine, you ideally need a monolingual corpus (language model) and a bilingual corpus (translation model). However we were only able to put together bilingual corpora since we did not know for certain which the original source language was. Using translated material to build the language model would probably lead to a defective model since it is very often possible to identify the source language in medium-length translations[7]. Moreover, there was virtually always an English language version of every manual in Italian, so it made more sense to use all the material available to maximize the amount of bilingual training data. In any case, a slight *stink* of translation, due to the lack of a language model, is not particularly important for the type of material we were training our CMT engines to translate (user manuals for household appliances), so the absence of this model was unlikely to be a big issue.

Unfortunately the manuals we downloaded from the Internet were in PDF format, and unaligned. To convert and align the files, I prescribed the use of the on-line translation environment tool Wordfast Anywhere[8]. This tool was chosen for three main reasons:

- it is cloud-based, so students can use it from home;
- it is free to use;

[2] Slate Desktop, https://slate.rocks
[3] Lilt, https://lilt.com
[4] KantanMT, https://www.kantanmt.com
[5] Philips, https://www.philips.com
[6] Smeg, http://www.smeg.com
[7] Hans van Halteren, 2008. Source Language Markers in EUROPARL Translations.
[8] Wordfast Anywhere, https://www.freetm.com

- in tests carried out before the start of the course, I was impressed by the high quality both of the PDF conversion feature and Wordfast Autoaligner (the alignment function).

Wordfast Anywhere converts PDF files to Microsoft Word doc format, and the Autoaligner only worked if one of the two languages being aligned was English. This restriction, which was not an issue in our case, has since been lifted.

The 42 students in the class were first divided into two groups (Smeg and Philips) to download as many manuals as they could from the Internet. They then worked together in pairs within their groups to convert the files and carry out the alignment. One student in each pair dealt with the source language files (Italian) and the other with the target language ones (English). It was decided to work this way because Wordfast Anywhere assumes the PDF file is in the source language of the active memory. Wordfast Anywhere creates an empty memory file when a project is set up since it expects to be used as a translation memory tool, and not simply as a PDF converter. Obviously the language settings can be reversed, but it is less time consuming to leave things as they are and convert PDFs written in one language only. Each member of the pair then gave half their files to the other and began the alignment process. Wordfast Autoaligner produces three types of aligned file: Translation Memory eXchange (TMX), plain text (TXT) and Microsoft Excel (XLS). All students chose to use TMX format.

The students pooled all the data they aligned with the other group members, although they did not necessarily all use the same data to create their corpora. After the alignment was complete, the students formed smaller groups to build CMT engines with KantanMT. Several students chose to work alone.

To assess their engines, besides considering the automatic metrics generated by KantanMT (BLEU, F-Measure and TER), the students carried out a series of comparisons. They took a manual, for which there was an existing translation which had not been used as training data for the CMT engine, and used it as input in three different tools:

- Their KantanMT CMT engine.
- Google Translate[9].
- A classic translation environment tool set up using the CMT engine training data corpus as a translation memory and only using the translation memory system features of the tool.

The raw output from each was then compared with the *official* existing version published by Philips or Smeg on their websites.

Everyone chose to use SDL Trados Studio[10] as translation environment tool, except one student who used OmegaT (freeware)[11]. To put all the aligned files together into one TMX memory file for the translation memory system, the students used Heartsome TMX Editor (freeware)[12].

Moreover the students compared the time required to produce an *unaided human translation* of part of the same manual with how long it took to post-edit the raw output from their CMT engine. In order not to remain influenced by one task when performing the other, the student who did the *unaided human translation* was always different from the student who post-edited the raw output. They also assessed the degree of similarity of these two versions to the *official* translation.

[9] Google Translate, https://translate.google.com
[10] SDL Trados Studio, http://www.sdl.com
[11] OmegaT, http://omegat.org
[12] Heartsome TMX Editor, https://github.com/heartsome/tmxeditor8

3 Results

All the students were able to build at least one working CMT engine (a total of 26 engines).

The BLEU scores for the students' engines reported by KantanMT ranged from 32% to 79% (mean: 64%). F-measure went from 52% to 85% (mean: 75%) and TER from 14% to 66% (mean: 34%). In most cases these are truly remarkable results also considering that no one had to use KantanMT's Library data. The majority of students agreed, on the basis of their human quality assessments, that their raw KantanMT CMT engine output was better than Google Translate's raw output for the kinds of text it was trained for (35/36 = 97%) and better than the raw output (pre-translation) obtained using the TM features of a translation environment tool (22/32 = 69%). In reality, in some cases, there was not much difference in quality between the raw translation produced by the translation environment tool and the raw CMT engine output, but several students observed that it would be quicker in practice to post-edit the CMT engine output since there is an editable proposal for every segment; translation memory systems leave the segment blank when no useful match is found. Unfortunately none of the students actually ran tests to verify this.

A couple of students made the interesting observation that, for a few segments, their KantanMT CMT engine had produced a translation which was better than the *official* version stating that it *sounded better*.

Almost everyone reported that it took less time to post-edit their raw CMT engine output than it did to produce an *unaided human translation* (27/28 = 96%). Only one person said the post-editing had taken slightly longer (1/28 = 4%). More than one student preferred their post-edited versions, defining the style as more *manual-like*. This of course could be due to the fact that students are not professional translators specialized in translating manuals.

Another important goal was to keep the cost to the university and to the students as low as possible. This was successfully achieved, by exploiting the Kantan Academic Partner Programme, freeware tools (Wordfast Anywhere, Google Translate, Heartsome TMX Editor and OmegaT), and existing software licences (SDL Trados Studio).

4 Discussion

Although I clearly laid out the aims, chose the tools, and suggested possible evaluation methods, I gave the students complete freedom to organize themselves, choose the files to include in their corpora, and establish their own human assessment criteria; some worked in teams, some in pairs and many alone, which explains why 42 students produced only 26 CMT engines. In addition, some of the students only reported part of the data according to what they found most interesting. All this unfortunately means that it is absolutely impossible to analyse their human evaluation data to produce aggregate scores. I have no intention of remedying this in future editions of the course module, since that would mean imposing rigid scoring models and the choice of material for the corpora. Given the degree of cynicism some of the students showed towards MT at the outset, such impositions risk giving grounds to accusations of *result rigging*.

Seven students also managed to find time to experiment with Lilt (fourteen-day free trial), and five of them (5/7 = 71%) were very enthusiastic about it. Lilt also allows users to build CMT engines, and has the look and feel of a highly simplified on-line translation environment tool. I did not choose Lilt as primary tool for the course mainly because it does not generate any standard evaluation metrics (BLEU, TER, etc.). Since Lilt's MT system is adaptive and interactive, the output changes while the translator works in the application. For this reason, existing *static* evaluation metrics are not suited to it. In future editions of the course module, I will encourage more students to try Lilt out.

5 Conclusion

The proposition was quite evidently a success since all the students were able to build at least one working CMT engine, try it out, and assess its output. At the beginning of the course, there was a certain amount of scepticism among the students regarding the effective usefulness of machine translation and post-editing. Although I did not aim to evangelize, there was a clear mood change in the end with all students except one stating – some perhaps still a little begrudgingly – that post-edited machine translation has a legitimate role to play in the translation industry (41/42 = 98%). The dissenter wrote: "Of all the systems used during the course, I remain of the opinion that the best translation is manual, albeit more laborious and slower, because it requires less [pre-editing and post-editing] than any other translation system."

Acknowledgements

All trademarks and trade names are the property of their respective owners.

References

Hans van Halteren, 2008. Source Language Markers in EUROPARL Translations. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 937–944.