SMT-CAT integration in a Technical Domain: Handling XML Markup Using Pre & Post-processing Methods

Arda Tezcan
Katholike Universiteit Leuven
Leuven, Belgium
ardatezcan@gmail.com

Vincent Vandeghinste
Centre for Computational Linguistics
Katholike Universiteit Leuven
Leuven, Belgium
vincent@ccl.kuleuven.be

Abstract

The increasing use of eXtensible Markup Language (XML) is bringing additional challenges to statistical machine translation (SMT) and computer assisted translation (CAT) workflow integration in the translation industry. This paper analyzes the need to handle XML markup as a part of the translation material in a technical domain. It explores different ways of handling such markup by applying transducers in pre and post-processing steps. A series of experiments indicates that XML markup needs a specific treatment in certain scenarios. One of the proposed methods not only satisfies the SMT-CAT integration need, but also provides slightly improved translation results on English-to-Spanish and English-to-French translations, compared to having no additional pre or post-processing steps.

1 Introduction

Although it took decades for machine translation (MT) to find its way into the translation business, and although the goal of perfect translation performed by machines alone seems to be unfeasible, MT systems are increasingly being integrated into the translation and localization workflows. A translation memory (TM) is one of the primary components of current CAT tools in today's translation and localization needs (Garcia, 2005). After being commercialized, TMs were basically used as a database of translation units, which are able to retrieve existing translations for the sentences that need to be translated again, increasing the efficiency and productivity of a translation task.

Driven by competition, the translation industry integrated new MT systems with the widely used TM technology. Today, as one of several approaches, SMT systems prove to be successful, especially when they are integrated in post-editing workflows and are trained with TM data (He et al., 2010).

While the translation industry follows the scientific developments of MT closely, it faces its own specific problems. Although there is much effort put into scientific research topics in the field of SMT (Yamada and Knight, 2000; Och and Ney, 2002; Koehn et al., 2003; Koehn et al., 2007; Koehn, 2010) this paper introduces the XML markup problem in SMT-CAT integration and proposes practical solutions. Section 2 presents background information about XML in translation and post-editing workflows, and explains the challenges XML markup brings. Section 3 refers to related work and motivates this paper. Section 4 introduces several methods to handle the XML markup. Section 5 reports and analyzes the experiments that were conducted. Section 6 concludes, looking into the possibilities for future work.

2 XML in Translation Workflows

An important challenge in SMT-CAT integration is the use of a TM database for the creation of the necessary corpora for an SMT system. The parallel text stored in a TM can be used after it is exported. Although exporting can be done in different formats depending on the tool that is used, there is an XML based standard defined for this purpose by the Localization Industry Standards Association (LISA)[1]. TMX (Translation Memory eXchange) is a vendor-neutral open XML standard for the exchange of TM data and can be created directly within the TM software. It is evident that this file format can be used for other circumstances than data exchange purposes such

1 http://www.lisa.org/fileadmin/standards/tmx1.4/tmx.htm

Mikel L. Forcada, Heidi Depraetere, Vincent Vandeghinste (eds.)
Proceedings of the 15th Conference of the European Association for Machine Translation, p. 55–62
Leuven, Belgium, May 2011

as for the creation of training corpora for SMT systems. TMX contains its own XML structure as a first layer of markup. A second layer, consisting of the translation units that are stored in TMs can also contain XML, HTML, SGML and/or RTF markup. This second layer of XML markup and the challenges it brings are the main focus of this paper.

There are several reasons why it is possible to have XML tags inside the sentences, such as protecting text from being translated or for automatic replacements. When XML markup is involved in post-editing, the SMT system should be able to satisfy additional requirements to keep this process as efficient as possible.

First of all, XML markup should appear in the translation without any loss of (meta) data, keeping its well-formedness. A TMX including non-well-formed XML tags can be imported into a TM without any warnings, if the correct TMX markup is provided.

Besides the integrity of the structure, the system should also be able to preserve the content of the tags, as an output lacking the correct content of tags risks passing the post-editing stage undetected. As part of a commercial business, such systems cannot afford to make mistakes on the correctness of the content of XML elements in the absence of extra checking mechanisms that work on the meta-data level.

To sum up, the existence of XML markup inside sentences brings some challenges to the SMT system, such as: 1) preserving the validity and the content of the XML tags in the SMT output; 2) Poor coverage of unseen data, considering that the attributes of such elements can create a big vocabulary of a set of alphanumeric characters; 3) Poor word alignments, and; 4) Poor word reordering, due to a significantly increased number of tokens in the sentences. Seeing the importance and the challenges that XML tags can add to translation workflows, it is essential to pay extra attention to handling these tags in the SMT system to be integrated within the CAT workflows.

3 Related Work and Motivation

Although one of the options of handling XML tags in a corpus and an SMT system might simply be removing them from the data (and reconstructing them afterwards), this paper will focus on approaches that will preserve the XML markup as part of the training and translation material due to the word-like use inside sentences. Unlike the TMX markup, the tags that are used inside sentences can be more than placeholders, containing/representing words or phrases.

An increasing amount of work is being invested in producing successful SMT-CAT integration and several approaches improve integration and translation quality. Vogel et al. (2000) present a hierarchical TM system, in which the bilingual corpus is converted into a set of patterns by applying transducers to convert words to category labels, and recursively, to convert sets of labels to more complex expressions. The translation takes place with this modified TM, followed by the use of a complete cascade of transducers to recursively convert complex expressions to sets of labels and finally produces text from labels. This approach is reported to provide good coverage for unseen data.

Du et al. (2010) propose different methods for treating TMX markup, in a scenario of SMT-TM integration. This study focuses on the "first layer" of XML markup, and suggests that an SMT system can handle such markup well, when the markup is kept as a part of the training data.

Leplus et al. (2004) show that TM data is more successful as training material for an MT system when simple alterations are made on numbers, names of days, etc. This study proposes adding pre and post-processing steps to the actual translation process, altering the training material and the input for translation so that all numbers are represented with an "__INT__" token, resulting in an output containing the same tokens. The post-processing step finalizes the translation by replacing the token with the correct number.

A similar idea is used by Eck et al. (2004) to improve SMT results in the Medical Domain. They use the semantic type information that is defined in the UMLS Semantic Network[2] to generalize the training data. A set of transducers is used in pre-processing to alter the words like "head", "arm" and "knee" to "@BODYPART" tokens. After translation, these dummy words are changed back into the actual word in the target language.

This paper goes one step further than the existing studies that are directly or indirectly related to SMT-CAT integration. We provide methods that can directly be applied within the current workflows, facing the challenges of XML markup that are explained in section 2. From a more general perspective this work will also help

2 http://www.nlm.nih.gov/research/umls/

the translation industry, considering that current academic research is more focused on general purpose approaches and not on specific domains, specific types of data, or specific types of problems that occur in the day-to-day real life translation process.

4 Handling XML Markup in SMT

We propose four different methods for treating XML markup inside the sentences of the TM database, and an additional method for solving a problem that can be introduced while handling XML markup. All the methods have been tested on TM data, which was collected from the TMX and cleaned from the TMX markup. This cleaning process is beyond the scope of this paper. All methods assume that the data does not include any TMX markup.

4.1 Method 1: Full Tokenization

This approach represents our baseline and consists of the default tokenizer script of Moses (Koehn et al., 2007). This method produces an SMT system where all the meta-data is treated as plain text. This is a "dirty" approach since all the XML characters are tokenized, increasing the token size of the sentences dramatically in some cases.

This baseline is interesting as it shows how well Moses can handle the translation of the actual material and the ordering of the XML reserved characters (or entity references). Any slight change in the order of such characters may result in a non-well-formed XML structure. This method provides a wide range of results, from showing that no special treatment is needed for XML markup to how important it is to properly treat such markup. Moreover, as a consequence of tokenizing the XML elements in preprocessing, this method requires a post-processing stage to reconstruct the XML tags from the separate tokens. This process is essential to provide the same XML structure as in the source segments.

4.2 Method 2: Full Markup Normalization

As mentioned in section 3, normalizing certain words and/or numbers has been used in the past to improve SMT results. Considering that an XML tag in a TM can contain up to 50 tokens as a result of "full tokenization" (Method 1), a similar normalization of XML markup solves most of the problems of that approach.

We add pre and post-processing steps to the SMT flow. The pre-processing step transforms all the different types of tags (and their contents) into a general token "@tag@". The corpus is transformed prior to training and the input files are similarly modified. The content of the tag is then injected into the output, replacing the corresponding "@tag@" token during post-processing.

Besides possibly solving most of the problems that are subject to SMT-TM integration, such as poor coverage, poor alignments and preserving the content and the structure of the XML, this method causes an additional challenge: the alignment task of the "correct" XML tags in the input segment with the tags in the output, in the case that the output order of the tags is different from the input order. In the technical data subject to our experiments 10% of the sentences in English-Spanish (En-Sp) and 9% in English-French (En-Fr) included at least one XML tag. 16% (En-Sp) and 20% (En-Fr) of these sentences (with at least one tag) contained multiple tags. We focus on retrieving the tag contents and the tag alignment problem in section 4.5.

4.3 Method 3: Role-Based Markup Normalization

When altering the XML markup to "@tag@" tokens we are decreasing the vocabulary size noticeably. However, we might actually get a poorer translation system by normalizing different tags (that contain different contextual information, words and phrases) to one single type of tag and create an overgeneralization. As an alternative method, role-based markup normalization modifies the tags, based on the element names. As different tags are used in different contexts, just like_words, this method helps to distinguish certain contextual differences while creating better phrase and word alignments.

Different pre and post-processing steps are applied to alter the XML tags to tokens based on their names (in the format "*@tag-name@*"), and to take the corresponding tag content from the source after the input file is translated. In our data 16 different tags were converted to different tokens before the segments were passed to the Moses decoder.

Although the risk of having problems with the alignment of tags is reduced, it still exists.

4.4 Method 4: Role-Based Markup Normalization – XML Input

Unlike the previous methods, this method avoids introducing additional problems by ex-

perimenting with a feature of Moses, as an addition to the method described in section 4.3.

Moses offers an "`-xml-input`" flag, which can be turned on during the decoding process[3]. The decoder has an XML markup scheme that allows the specification of translations for parts of the sentence. This is used to plug translations into the decoder, without changing the model. There are four different values that are associated with this flag (`exclusive`, `inclusive`, `pass through-`, and `ignore`). During these experiments we use the "`exclusive`" value, which only uses the translation specified in the XML structure of the input phrase.

The specified translation is treated just like any other translation, being scored with the language model (LM)[4].

Instead of only using the tokens in the input file, the translation of the actual content of these tags is forced by wrapping the tokens with the Moses XML markup, keeping the actual tag available at decoding time so it is plugged into the translation. This method requires only a change in the treatment of the input file, compared to the method in section 4.3.

Although the idea is rather simple, Moses still refuses such an XML tag (including XML reserved characters) when wrapped with Moses markup as this results in non-wellformed XML. Therefore, we add another step to pre-processing, to convert the XML reserved characters of the data-specific-tags to entity references or other tokens (for example "<" could be replaced by "@arro@", to represent the character "arrow opening"), so that they are treated as text by Moses. As a result, the post-processing step requires converting these entity references or tokens back to XML reserved characters.

We test this method with two different LMs, one (same as in Method 3) keeping the tokens that represent the tags (as Method 4a), and the other keeping the tags in fully tokenized format (same as in Method 1) in which the XML-specific characters are replaced by additional tokens (as Method 4b). The reason for this additional experiment is to make sure which type of LMs yields better results, considering that in this case, the translation (the XML tags) would already have been plugged into the decoder (with the use of the "`-xml-input`" flag) and that it would still be scored by the LM. A final overview of the four different methods is shown in Figure 1.

4.5 Retrieving the Content of Tags and Reordering

In the translations produced with Methods 2 and 3, if the order of (multiple) tags is different in the input and output, the necessary reordering should be externally applied to the tokens in the output, as normalization cuts down all the connections with the tokens, the corresponding tags, and their contents. One of the additional steps that can be applied for this purpose is the "`-report-segmentation`"[5] functionality of Moses, to report phrase segmentation in the output. Figure 2 illustrates a sample output of Moses[6] with this functionality turned on during the decoding, in which the translation "a" was generated from the German word "ein" (0-0). A similar interpretation applies to the other words in the translation.

```
echo 'ein haus ist das' | moses -f
moses.ini -t -d 0
> this |3-3| is |2-2| a |0-0| house |1-1|
```

Figure 2: An example translation of Moses using the "`-report-segmentation`" flag.

The additional information stored in the output shows the alignment of source and target tokens. Using this information, a post-processing step can be applied to transfer the actual tags to the output, in the correct order.

An alternative approach can be constructed by running two decoding processes in parallel with two different input files. Besides the input file that is used in Method 3, we translate another file, modified additionally with the same use of the "`-xml-input`" functionality that has been discussed in Method 4. With this parallel translation, we can align the two output files and transfer the tags from the second output file as in this file the XML tags will be present explicitly, due to the use of the "`-xml-input`" flag. Figure 3 shows the workflow of the two different approaches to give a better overview.

3 http://www.statmt.org/moses/

4 Moses Support, 2010: http://www.mail-archive.com/moses-support@mit.edu/msg02618.html

5 Further information at : http://www.statmt.org/moses/?n=Moses.Tutorial

6 Example taken from Moses tutorial page: http://www.statmt.org/moses/?n=Moses.Tutorial

Method 1: Full tokenization
install the transfer (See page < xref attribute = " at01 " href = " AZE0033XSZLM " / >) .
reposer la boîte de transfert (cf. page < xref attribute = " at01 " href = " aze0033xszlm " / >) .

Method 2: Full markup normalization
install the transfer (see page @tag@) .
reposer la boîte de transfert (cf. page @tag@) .

Method 3: Role based markup normalization
install the transfer (See page @xref@) .
reposer la boîte de transfert (cf. page @xref@) .

Method 4a and 4b: Role based markup normalization – XML input
install the transfer (see page <np translation="@arro@ xref label = @dbq@ at01 @dbq@ href = @dbq@ aze0033xszlm @dbq@ / @arrc@">@xref@</np>) .
reposer la boîte de transfert (cf. page @arro@ xref attribute = @dbq@ at01 @dbq@ href = @dbq@ aze0033xszlm @dbq@ / @arrc@) .

Figure 1: Sample input and output segments, when a Moses system is built and ran in four different ways.

5 Experiments and Analysis

In the experiments we use two TM exports (TMX) from the automotive domain for the language pairs English-Spanish and English-French. These TMs include domain specific data and are heavily tagged with XML. 41.145 segments out of 400.912 (En-Sp) included one or more tags. 36.540 segments out of 400.360 (En-Fr) included one or more tags.

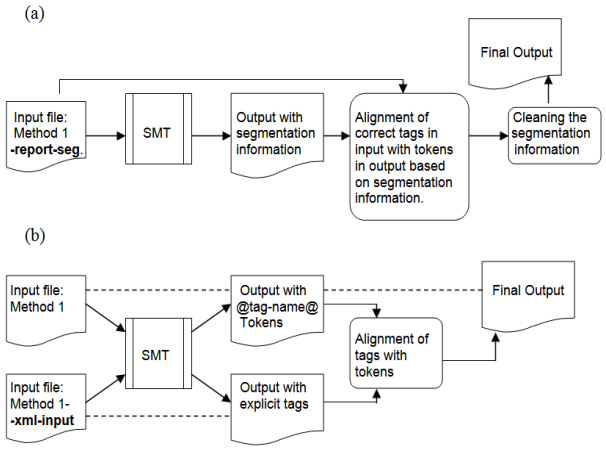

Figure 3: Representation of tag content retrieval and reordering methods.

5.1 Corpora, System and Evaluation

From this data 912 and 871 pairs of segments are extracted respectively as test sets, leaving 400.000 and 399.489 fragments as training data for the SMT system. As these TMs do not contain any duplicate translation pairs, there is no overlap between the test set and the training set. The TMX exports are cleaned from TMX markup prior to the training, leaving two aligned files per TM (on the sentence level), for source and target segments.

For the SMT system, we use the Moses toolkit consisting of Moses, GIZA++ (Och and Ney, 2003) and SRILM (Stolcke, 2002). The LMs were trained with five-grams, applying interpolation, Kneser-Ney discounting and the phrase based translation models with a maximum phrase length of seven.

Besides the first set of experiments, focusing on the proposed methods in Section 4, a second set of experiments was conducted with half and quarter size of the initial corpora consisting of randomly selected sentences from the original training set, to see possible changes in the results. Table 1 shows the numbers of sentence pairs present in different sets of training data.

	full	half	quarter
ENG-SPA	400.000	200.000	100.000
ENG-FRE	399.489	197.745	99.873

Table 1: Different training sets represented with number of sentence pairs used for both language pairs.

To evaluate the SMT results, we use automatic evaluation metrics such as BLEU (Papineni et al., 2000), NIST (Doddington, 2002) and METEOR (Banerjee and Lavie, 2005) and a human translator for judging the MT outputs for tag reordering. The training data, the input, the output, and the reference files are all tokenized (with the Moses tokenizer) and all the tags in all outputs of different methods are normalized to "@tag@", to avoid possible score differences caused by comparing different type of output and reference translations regarding the XML tags.

5.2 Results

Table 2 shows the scores obtained by the proposed methods for the two language pairs.

ENG-SPA	BLEU	NIST	METE
Method 1 - FT	67,93	9,94	54,68
Method 2 - FMN	68,20	9,95	55,01
Method 3 - RBMN	**68,63**	**10,00**	**55,25**
Method 4a - RBMN xml - LM tags	68,12	9,95	54,66
Method 4b - RBMN xml - LM tokens	67,48	9,91	54,08
ENG-FRE			
Method 1 - FT	70,94	10,12	47,32
Method 2 - FMN	71,02	10,12	47,19
Method 3 - RBMN	**71,82**	**10,19**	**48,05**
Method 4a - RBMN xml - LM tags	70,50	10,05	46,65
Method 4b - RBMN xml - LM tokens	70,54	10,06	46,74

Table 2: Automatic evaluation scores for English-Spanish and English-French.

The most striking outcome is how the tags were handled by the "full tokenization" method. The systems actually handle the XML tags quite well, making no mistakes on the XML structure itself. However, when the tags included words or phrases, the systems provide "unnecessary" translations (for phrases of three or more tokens).

These "unnecessary" translations damage the integrity of some of the tags in general, proving the potential risks of this method for translation tasks that are sensitive to such errors.

We also observe how the role-based normalization of tags improves the scores slightly, compared to a strong baseline, by relative values of 1% and 1,2% on BLEU, 0.06%, 1% and 1,5% on METEOR for Spanish and French translations respectively. For a more in-depth analysis we score the translations once more, after dividing the test sets into two. In one set we keep only the sentences that include at least one tag and in the second set we keep only the sentences without tags (forming almost equal sizes of sub test tests). Table 3 shows the BLEU results of the divided test sets for both language pairs.

From these scores, it is clear that when the sentences include tag(s), the results improve more (relative to baseline, by 1,5% for Spanish and 1,6% for French on BLEU) compared to when the sentences do not include any tags at all, showing that the improvement is minimal for the segments without tags. These results indicate that "overgeneralization" of tags (Method 2) actually decreases the quality of non-tagged segment translations.

ENG-SPA	Without Tags	With Tags
Method 1 - FT	63,80	70,50
Method 2 - FMN	62,95	71,51
Method 3 - RBMN	**63,94**	**71,58**
Method 4a - RBMN xml - LM tags	63,94	70,73
Method 4b - RBMN xml - LM tokens	63,94	69,48
ENG-FRE		
Method 1 - FT	68,58	64,54
Method 2 - FMN	68,13	64,76
Method 3 - RBMN	**68,62**	**65,62**
Method 4a - RBMN xml - LM tags	68,62	63,89
Method 4b - RBMN xml - LM tokens	68,62	64,04

Table 3: BLEU results for the split test sets.

For Method 4, we use two different LMs to score the translations, where "LM tags" represents the LM that was used in the baseline method, with full tokenization (XML characters were additionally converted to tokens to match the output of Moses), and "LM tokens" represents the LM that was used in Method 3. As shown in tables 1 and 2, the systems handling the tokens with the "`-xml-input`" flag score worse than Method 3 in both cases. Although this flag helps us protect the XML structure, as all the tags are scored with the LM in both cases, we can still expect a poor coverage and mismatches between both LMs and the output. However, considering which LM performs best, the results are not conclusive, although they point to an improvement in Spanish translation quality when using "LM tags".

A final observation can be made about the tag reordering capabilities of the different systems, when the tags are normalized. First of all, both TMs are analyzed to see how often the tag order is changed in the translations compared to the source segments. In eight translations of the English-Spanish TM, and in 14 translations of the English-French TM (less than 0.001% of the number of sentences stored in these TMs), the order of tags in translation is different than in the source. This low number indicates that an additional reordering task is almost never necessary for this specific data. Still, we remove five of these translations from each database, retrain the systems and translate the source segments, using Methods 1 and 3. As a result, all translations fail a potentially necessary reordering. Although this result might indicate incorrect translations, a human translator judges that all translations are still correct as a whole. We have to mention that the result could potentially be different for "less similar" language pairs, or even for the same language pairs in another scenario, due to the difference in the use of tags.

5.3 Adding data vs. RBMN

The aim of a second set of experiments is to compare the improvement using role-based markup normalization, while adding data. This is another straightforward way of improving the translation results in the case of SMT-TM integration. For this purpose, the tests are repeated (using Methods 1 and 3) with "half" and "quarter" size data. Creating two more systems per language pair, the BLEU scores are shown in Table 4.

It is accepted that more data implies better translations (Fraser and Marcu, 2007) and increasing the corpus size results in a decreasing growth rate in the quality of an SMT system (Koehn, 2002). The systems that are subject to these experiments are no exception, as it can

ENG-SPA			
Method 1 – FULL	67,93	**Method 3 – FULL**	**68,63**
Method 1 – HALF	65,76	Method 3 – HALF	66,34
Method 1 – QUARTER	63,40	Method 3 – QUARTER	63,69
ENG-FRE			
Method 1 – FULL	70,94	**Method 3 – FULL**	**71,82**
Method 1 – HALF	69,31	Method 3 – HALF	69,71
Method 1 – QUARTER	66,82	Method 3 – QUARTER	66,77

Table 4: Comparison of BLEU scores using Method 1 and Method 3, for different sizes of data.

clearly be seen that reducing the size of the corpus by half and three quarters (by removing random sentences), decreases the translation quality similarly. The most interesting part of this experiment is to see what size of additional data is necessary to improve the system as much as the (role-based) normalization of tags. This additional information enables us to see the scale of improvement in a more practical point of view in translation business, compared to analyzing the improvement purely on metric scores. When the translations of Method 3 are compared to the translations of Method 1 for different sizes of data, it can be seen that the improvement made by such normalization becomes greater as the size of the data increases. This can be considered an important aspect when larger sizes of data are subject to such a method, considering that the rate of improvement on the translation quality would decrease as the data size increases. Table 5 and Table 6 show the rate of improvement in BLEU scores that Method 3 provides on top of the baseline and comparing these results with the effect of doubling the size of our data.

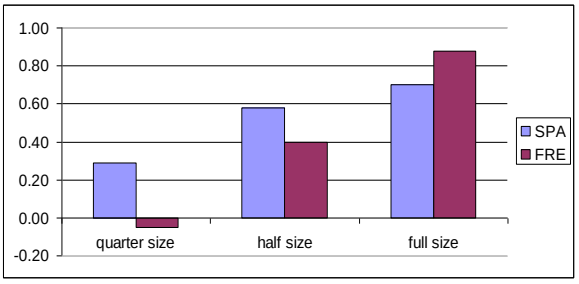

Table 5: The improvement in BLEU points of Method 3 over the baseline system, for different sizes of data.

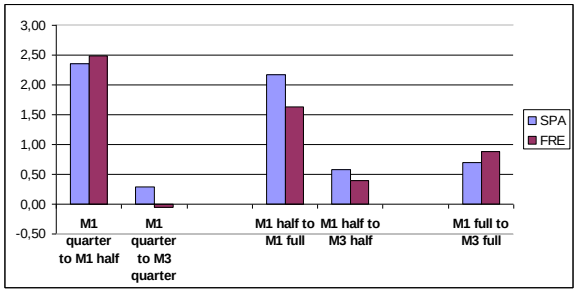

Table 6: Comparison of the improvements provided by doubling the size of training data and by applying Method 3 instead. Although the improvement on translation is expected to decrease when size of the full data is doubled, the exact amount of improvement remains unknown.

6 Conclusions and Future Work

This paper proposes four different methods for handling XML markup in the training data with pre and post-processing steps that consist of transducers. Each of these methods is evaluated using the automated MT metrics.

The first set of the experiments shows that, although Moses can handle XML markup well when basic tokenization is applied, it still occasionally fails to deliver the original content of the XML tags. However, this result is strictly related to how the tags are used in a certain TM and the type of content they have. In the case of a similar use, such as using the XML tags to protect text from being translated or to wrap text for automatic replacements, letting Moses handle tags as part of text might not be the best option for serious translation business.

The best results were obtained when the XML tags were normalized based on their roles. Although this improvement is not stunning, the results suggest that the importance of this improvement might be greater for larger data sets and that these improvements are comparable to the effects in the range of increasing the size of the data by half (En-Sp) to doubling it (En-Fr). Considering that increasing the data on this scale is rarely a realistic scenario for large TMs in the translation industry using the method of role-based normalization can lead to cost savings and increased productivity, while also ensuring the integrity of the XML elements.

Additionally, it can be observed that, although the use of the proposed methods have the potential to impose new challenges like reordering the tags, such a reordering is almost never necessary in our test data. If reordering is necessary, using

the segmentation report of Moses or performing a parallel translation, as explained in section 4.5, can provide accurate results.

Further improvement on the translation quality of an SMT system integrated with CAT tools, can possibly be achieved by including more tasks in pre and post-processing stages. If the properties of specific domains and types of TMs are analyzed carefully, they could be exploited to supply better systems. Some of these properties could be: the use of abbreviations and their explicit counterparts; existence of phrases that should not be translated; the use of alphanumeric formulas and codes for normalization methods.

Furthermore, repeating the experiments with other language pairs would help give a better overview on the results. A human evaluation would also be necessary and helpful to confirm the results that are obtained in this paper.

7 Acknowledgement

This work has been supported by ITP nv, Belgium (http://www.itp-europe.com/).

8 References

Banerjee, S., Lavie, A. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization.*

Doddington, G. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the ARPA Workshop on Human Language Technology*.

Du, J., Roturier, J., Way, A. 2010. TMX Markup: A Challenge When Adapting SMT to the Localisation Environment. In *Proceedings of the 14th Annual EAMT conference*, pp. 253-260, Saint-Raphaël, France.

Eck, M., Vogel, S., Waibel, A. 2004. Improving statistical machine translation in the medical domain using the unified medical language system. In *Coling 2004: Proceedings of the 20th international conference on Computational Linguistics*, pp. 792–798, Geneva, Switzerland.

Fraser, M., Marcu, D. 2007. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293-303.

Garcia, I. 2005. Long term memories: Trados and TM turn 20. *Journal of Specialized Translation 4*, pp. 18-31.

He, Y., Ma, Y., Way, A., van Genabith, J. 2010. Integrating N-best SMT Outputs into a TM System. In *Proceedings of Coling 2010*, pp 374–382, Beijing, China.

Koehn, P. 2010. Statistical Machine Translation. Cambridge University Press.

Koehn, P. 2002. Europarl: A Multilingual Corpus for Evaluation of Machine Translation, *Draft, Unpublished*. http://people.csail.mit.edu/~koehn/publications/europarl.ps

Koehn et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL, Demonstration Session.*

Koehn, P., Och, F. J., Marcu, D. 2003. Statistical phrase-based translation. In *Proceedings of NAACL*, pp. 48–54.

Leplus, T., Langlais, P., Lapalme, G. 2004. Weather report translation using a translation memory. In *Proceedings of the 6th AMTA*, pp. 154–163, Washington DC, USA.

Och, F. J., Ney, H. 2002. Statistical machine translation. In *Proc. of Workshop of the European Association for Machine Translation*, pp. 39-46, Ljubljana, Slovenia.

Och, F. J., Ney, H. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19-51.

Papineni et al. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 20th Annual Meeting of the ACL.*

Stolcke, A. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing, Denver, Colorado, September*.

Vogel, S., Ney, H. 2000. Construction of a Hierarchical Translation Memory. *18th International Conference on Computational Linguistics*, pp. 1131-1135, Saarbrucken, Germany.

Yamada, K., Knight, K. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on ACL*, pp. 523-530, Toulouse, France.