Transferring Markup Tags in Statistical Machine Translation: A Two-Stream Approach

Eric Joanis, Darlene Stewart, Samuel Larkin and **Roland Kuhn**
National Research Council Canada
1200 Montreal Road, Ottawa ON, Canada, K1A 0R6
First.Last@nrc-cnrc.gc.ca

Abstract

Translation agencies are introducing statistical machine translation (SMT) into the work flow of human translators. Typically, SMT produces a first-draft translation, which is then post-edited by a person. SMT has met much resistance from translators, partly because of professional conservatism, but partly because the SMT community has often neglected some practical aspects of translation. Our paper discusses one of these: transferring formatting tags such as **bold** or *italic* from the source to the target document with a low error rate, thus freeing the post-editor from having to reformat SMT-generated text. In our "two-stream" approach, tags are stripped from the input to the decoder, then reinserted into the resulting target-language text. Tag transfer has been tackled by other SMT teams, but only a few have published descriptions of their work. This paper contributes to understanding tag transfer by explaining our approach in detail.

1 Introduction

Increasingly, translation agencies are incorporating machine translation (MT) in the work flow of their translators, to make them more productive. The usual scenario is a variant of post-editing, in which an initial translation generated by MT is manually corrected by a human translator (who is now called a "post-editor" instead). Green et al. (2013) show that SMT followed by post-editing can improve translator productivity, and even translation quality. Some interesting questions arise. For instance, how should MT interact with other productivity tools used by translators, such as terminology databases and translation memories? Koehn and Senellart (2010) and Du et al. (2010) discuss these issues.

Many translators resist using MT. Green et al. (2013) write bluntly: "Translators often show intense dislike for working with MT output." This is confirmed by our own experience. Why does this particular productivity tool, unlike others mentioned above, attract so much hostility? Perhaps it is because of status anxiety: a translator who becomes a post-editor may perceive him/herself as having been proletarianized, going from being the machine's master to its slave.

However, there are also practical objections to putting out-of-the-box MT into the translation workflow. This paper looks at one of these, the need to transfer markup tags from the source to the target document. Tags are omnipresent in real-life documents: they appear in almost every file format (e.g., Word, HTML, Excel, PowerPoint, PDF) and are used to encode everything that is not plain text (fonts, footnotes, table cells, hyperlinks, etc.). In the translation memories of our clients, 10–40% of segments have at least one tag. In sentences having tags, the average number of tags was about 3 per sentence.

In this paper, we will only discuss tag transfer for language pairs with similar orthographies, especially European languages (e.g., it's not clear what the equivalent of "italic" would be in Chinese text). We will only discuss tag transfer for statistical MT (SMT) systems: rule-based MT systems became translators' tools decades ago, and their designers worked out then how to handle this problem. We are unfamiliar with their solutions (finding out details about commercial rule-based systems tends to be difficult). By contrast, SMT researchers often give themselves the luxury of pretending that only pure text matters (but see "Related Work"). Unfortunately, this is not the world in which professional translators

Sharon O'Brien, Michel Simard and Lucia Specia (eds.)
Proceedings of MT Summit XIV Workshop on Post-editing Technology and Practice, Nice, September 2, 2013, p. 73–81.

live. For them, post-editing SMT output without the formatting information found in the source may represent a serious loss of productivity.

There are two possible approaches to the problem. One might consider formatting to be an intrinsic part of the source text, carrying important information that should help determine the choice of words and word order in the translation. In that case, tags should be part of the information available to the SMT decoder. This is the "one-stream" approach. By contrast, in the pure "two-stream" approach we adopted, all text fed to the decoder has been stripped of formatting information; once a target-language translation has been produced, the tags are reinserted into it. **Section 3** below discusses how previous work by other authors fits into this classification. An original aspect of our implementation is that the tag re-insertion module has information not only about phrase pair alignments, but also about word alignments within each phrase pair.

There are tradeoffs between the one- and two-stream approaches. The two-stream approach pools information in the SMT system's training data more efficiently: e.g., the word sequences "he never wins", "he **never** wins", and "he never wins" will look exactly the same in the training data, instead of differing because of the tags around the central word. (Data pooling could be achieved in the one-stream approach too, but at the cost of complicating decoding). An advantage and a disadvantage of the two-stream approach is that it permits the decoder to break apart words found together inside a paired tag: the decoder has more freedom to choose a good translation, but that may make deciding where to put the tag in the target text more difficult.

In the two-stream approach, the decoder may initially translate "**Hang up** the phone!" into German as "Lege das Telefon auf!" Depending on how the tag reinsertion rules are written, the final translation might be "**Lege** das Telefon **auf**!", "**Lege das Telefon auf**!", or even "Lege das Telefon **auf**!" In the one-stream approach, we can easily tell the decoder not to break up a contiguously-tagged word sequence. The decoder would probably produce "**Lege auf** das Telefon!", which exactly reproduces the formatting of the original, but has unidiomatic word order.

This paper gives a detailed description of our implementation of the pure two-stream approach, to clarify the issues and to help other people who might wish to implement it. We do not carry out an experimental comparison between the one-stream and two-stream approaches, though this would certainly be a worthwhile next step.

2 Background

2.1 Tags and Translator Productivity

Consider translating this sentence into French:

> **Acknowledgement** section should go as a last section immediately *before the references*.

An old-fashioned translator without access to a translation memory might "translate by replacement": put French text into a copy of the source, progressively deleting English words. This transfers the format, because word processing programs typically transfer formatting to adjacent characters. The translator might begin by typing the French word "Remerciements" next to its English equivalent, "**Acknowledgement"**, so the text may briefly look like this:

> **AcknowledgementRemerciements** section should go as a last section immediately *before the references*.

Next, the translator will delete "**Acknowledgement**" and proceed to type other French words in place. Virtually no productivity is lost by coping with the **bold** font of "**Acknowledgement"** (nor, later on, with the *italics* of "*before the references*").

Often, users of translation memories get format transfer for free: if words they keep in the target sentence retrieved from the memory are formatted, perhaps that formatting is still appropriate. This is not guaranteed to happen, but can occur in cases where translations are regularly updated (e.g., with Web pages).

If the translator uses a translation provided by vanilla SMT, neither of these tag transfer shortcuts is possible. For example, suppose the source sentence above yields the following translation (taken from vanilla Google Translate):

> Section de reconnaissance doit aller une dernière section immédiatement avant les références.

Here, not only must the post-editor fix up the text of the translation, but he/she may also need to change the typeface for the whole French sentence, put part of it into **bold**, and put another part into *italic*. Each manipulation represents a further loss of productivity.

In the example just given, we would like the tag transfer module to modify the output into something like this (before post-editing):

Section de **reconnaissance** doit aller une dernière section immédiatement *avant les références*.

2.2 Markup Formats and Wrappers

In SMT applications for translation agencies, both the source sentence and the training data for the SMT system are typically in markup formats used by translation memories. Two common open-standard formats for these are Translation Memory eXchange (TMX) (LISA, 2005) and XML Localization Interchange File Format (XLIFF) (OASIS, 2008), XML standards for storing and transferring translation memory contents, and documents to be or already translated, between applications.

A translation work package is a unit of work a manager would give a translator—typically, all sentences that need translating in a document from the client, but sometimes part of a document or several documents—and the matching sentences in the other language (where available). XLIFF was designed to transfer a document with its translations (localized versions), while TMX was designed to transfer whole translation memory contents (which could be thousands of documents from hundreds of translators). Though both formats can be used to store a translation work package, XLIFF is better suited for that purpose because it was designed with that end in mind. It is establishing itself as the standard for translation work packages, both in proprietary software (e.g., the SDL Trados suite) and in open source translation memory software (e.g., the Okapi Framework).

In principle, the markup format is irrelevant to the work described in this paper. We used files exported by SDL Trados, the dominant commercial provider of translation memories. When Trados exports TMX and XLIFF files (or files in any one of over 70 formats), it hides the markup details in a wrapper layer (SDL International, 2011). Since TMX tags are still complex, we rewrap them in another simplifying layer. From our system's perspective, there are two types of tags: isolated tags and paired tags—it has no notion of italic, bold, typeface, etc. Isolated tags occur when the underlying document has a point tag, or when it has a tag pair that starts in one sentence and ends in another: in that case we see an isolated tag in each sentence.

In practice, though our system handles TMX and XLIFF files in the same way, the formatting of a given sentence is often more economical in XLIFF than in TMX. For instance, in our XLIFF files, tags that are common to a sequence of words tend to be factored out more than in TMX files.[1] Thus, our system often generates better output from XLIFF than from TMX, even though both are in wrapped formats, because it gets confused by the TMX's verbosity.

Below is a real example of TMX and XLIFF tags for the same text. The id numbers after the "=" point to the formatting details (e.g., **bold**, *italic*, etc.):

Original text: National CH_4 and N_2O emissions.

TMX: <op id="6"/><op id="7"/>National CH
<cl id="7"/><cl id="6"/><op id="8"/>4
<cl id="8"/><op id="9"/> <op id="10"/>and
N<cl id="10"/><cl id="9"/><op id="11"/>2
<cl id="11"/><op id="12"/>O emissions.
<cl id="12"/>

NOTE: <op…> and <cl…> stand for our wrappers around the actual "open" (<bpt>…</bpt>) and "close" (<ept>…</ept>) tags in the TMX.

XLIFF: National CH<g id="3173">4</g> and
N<g id="3174">2</g>O emissions.

NOTE: Here <g id=…> and </g> are the "open" and "close" tags in the XLIFF file.

3 Related Work

Koehn and Senellart (2010) discuss the integration of SMT into translation memories. They also discuss XML markup, but not because they are interested in tag transfer from source to target. Instead, they use XML to mark untranslated parts of a source sentence, after most of it has been matched in the translation memory; the marked-up portions are passed to SMT.

Du et al. (2010) focus on the central topic of the current paper: how to handle markup in an SMT system whose output will be post-edited. They compare the performance of Moses (Koehn et al., 2007) using three different tag transfer

[1] The differences observed between our TMX and XLIFF files may be explained by the version of the software used to create them: SDL Trados 2007 for TMX files, SDL Trados Studio 2011 for XLIFF. However, our aim is to handle any TMX or XLIFF file submitted to our software.

methods on Symantec French and English data in TMX format: “Complete Tokenisation”, “Partial Tokenisation”, and “Markup Transformation”.

These methods lie along a spectrum between the one-stream and two-stream approaches. “Complete Tokenisation” is a pure one-stream method that treats tags as normal input, tokenizing them just like regular text. This leads to much longer sentences and diminishes the ability of the phrase table and the language model (LM) to learn useful patterns: the phrase table may have more noise in it because of bad word alignments, and the LM will be based on much sparser statistics. Worst of all, tags may be reordered during SMT, leading to syntactically incorrect TMX in the output, so a tidying post-processing stage is needed.

“Partial Tokenisation” has the same data flow as “Complete Tokenisation”, but handles tags specially. It deals with data sparsity by grouping frequent tag sequences into a single token, and also by grouping tags and tag sequences into categories, each assigned a single symbol. Thus, sentences are about 50% shorter than in “Complete Tokenisation”. This method can handle tags not seen in the training data.

“Markup Transformation” might be called a “1.7-stream” approach. As in the pure two-stream approach, markup is stripped from the input prior to SMT and then reinserted in the resulting translation. However, the decoder is not given complete freedom to reorder words: when markup is stripped from a region of input text, that region is marked as a Moses “zone”. During SMT, words may be reordered within the zone, and the zone can be moved around, but there may be no movement of words between the inside and the outside of the zone. Thus, no tricky cases can arise where word reordering causes words to cross tag region boundaries (see **Section 4.2** below); however, the translation quality may suffer.

The authors found that human evaluators do not have a strong preference for any one of these methods. Automatic metrics (NIST, BLEU, TER, and MTR), however, overwhelmingly prefer the two “Tokenisation” methods. There is some evidence that the “zone” restriction reduces the quality of the output.

Tezcan and Vandeghinste (2011) apply the one-stream approach to TMX data for English-to-French and English-to-Spanish data, also using Moses. They look at four methods. One of these methods is a pure one-stream implementation roughly equivalent to “Complete Tokenisation” in Du et al. (2010). The other three methods can be seen as variants of “Partial Tokenisation” as described in Du et al. (2010), differing in the degree of generalization of tags and in the mechanisms for placing tags in the target text. The experiments in this paper seem to show a slight advantage for “Role-Based Markup Normalization”, in which tags are grouped into categories based on their roles, with each category given its own token. This improves the quality of the output compared to that from complete tokenization to an extent equivalent to the training data being increased by 50–100%.

Zhechev and van Genabith (2010) describe a system that matches chunks of the input sentence with chunks found in a translation memory where possible, then uses SMT decoding to fill in the unmatched pieces. Although tag transfer is not the main subject of the paper, the system described does transfer tags from the source to the target. As in the “Partial Tokenization” method above, tags are replaced by a simplified representation. Reading between the lines, one infers that the tag transfer approach here is probably “one-stream” with the simplified tags left in the training data, not stripped out prior to decoding.

The work closest in spirit to our own approach is described in Hudík and Ruopp (2011), as part of a description of the ongoing Moses for Localization (M4Loc) project undertaken by the open-source Moses community. The overall pipeline described by Hudík and Ruopp (2011) is identical to ours, but some important details differ. For instance, the authors use only phrase alignment information to transfer tags, but not word alignment information, inserting tags only at phrase boundaries in the target text, which can easily result in misplaced tags. In our system, we use the word alignment from the phrase table to correctly place tags within segments (in addition to phrase pair information). We handle both TMX and XLIFF formats, while they handle only XLIFF (a defensible decision, as XLIFF is replacing TMX). The software described in Hudík and Ruopp (2011) has been released (M4Loc, 2012), but has undergone some subsequent changes.

Finally, our work generalizes software mechanisms implemented by our colleague George Foster for transferring simple markup in Canadian parliamentary data, such as the political affili-

ations of people speaking in a debate, from one language to another (Foster et al., 2010).

4 Our Approach

4.1 Data Flow

We implemented two-stream tag transfer in our in-house system, a phrase-based SMT system resembling Moses (and which is licensed to translation agencies). As in Moses, the most frequent word alignment is stored with each phrase pair. **Figure 1** illustrates the data flow:

- **XML file:** This is the input file in XML format (either TMX or XLIFF).
- **Extract:** From XML files, extract the list of input sentences to translate, including their formatting tags. For XLIFF, tags are kept as is: <g id="*i*"> words… </g> for paired tags, <x id="*i*"/> for isolated tags. TMX tags are more complicated, so they are wrapped in <open_wrap id="*i*">, <close_wrap id="*i*"> or <tag_wrap> tags that are designed to be parsed by simple regular expressions.
- **Tokenization:** We use our standard tokenizer, customized to be aware of XLIFF and wrapped TMX tags. We tokenize the actual text, not tags, but the tags inform some choices. E.g., the tokenizer normally splits tokens at tag boundaries, but if a token contains an open/close tag pair, we keep it together, so 31^{st}, CO_2, etc., are kept as single tokens, despite the superscript and subscript tags they contain.
- **Q.tags.tok file:** This file contains tokenized text with the tags still embedded.
- **Strip tags:** By stripping out the tags, we get standard tokenized text of the kind our decoder normally works with.
- **Decoder pipeline:** This is our normal SMT pipeline: lowercase, apply rules (e.g., date and number parsing), decode, truecase.
- **P.tok file:** This is the decoder pipeline's main output – tokenized truecased text.
- **P.trace file:** The decoder trace includes phrase segmentation and alignment from the decoding process, and word alignments associated with each phrase pair.

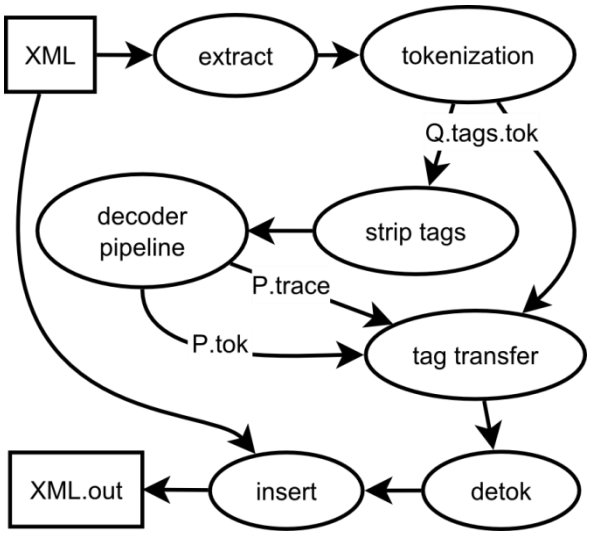

Figure 1. Two-Stream Data Flow.

- **Tag transfer:** This is the core module for this paper. It uses segmentation and word-alignment information from P.trace to insert the tags from Q.tags.tok correctly in P.tok.
- **Detok**: Detokenization is done using our standard detokenizer, again customized to be aware of tags.
- **Insert**: The translated detokenized sentences including their formatting tags are re-inserted into the original XML.
- **XML.out file**: This is the final TMX or XLIFF output file containing the source and translated text, including tags.

4.2 Tag transfer rules

The success of two-stream approaches depends on the **tag transfer** rules. These rules know about the phrase pair alignments used by the decoder to generate **P.tok** and the word alignments within each phrase pair.

Let "source tag region" (STR) refer to a contiguous sequence of source words that is enclosed by matched "open" and "close" tags, let "target covering phrases" (TCP) be the target-language phrases used to decode the source tag region, and let the "source phrase region" (SPR) be the words in the source covered by the source-language phrases corresponding to the TCP. The nature of decoding ensures that the SPR is a contiguous word sequence, but it may extend beyond STR, whereas TCP need not be contiguous.

The core transfer rules for paired tags are:

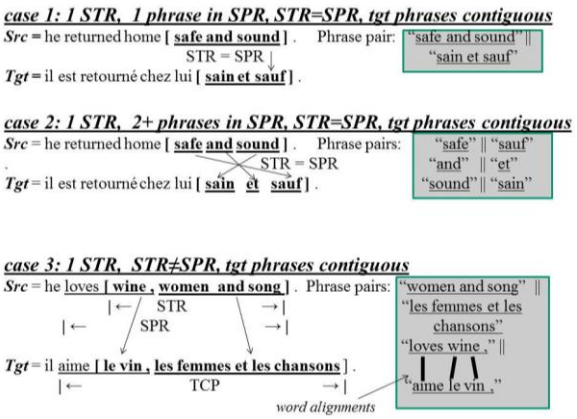

Figure 2. Some Special Cases.

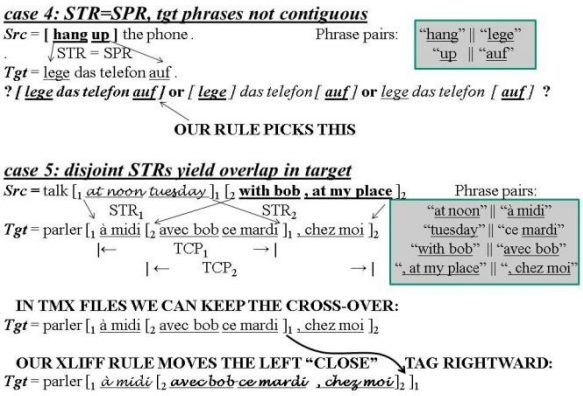

Figure 3. More Complex Cases.

1. If the boundaries of the STR and the SPR coincide exactly, and the TCP are contiguous in the output target-language sentence, then the tag pair surrounding the STR is copied into the output so it surrounds the TCP. This rule is shown in **Figure 2**, **cases 1** and **2**. In **case 2**, note that phrase reordering during translation does not affect application of the rule.

2. If the boundaries of the SPR extend beyond the STR, even if the TCP are contiguous in the output target-language sentence, there may be target words from the TCP that are word-aligned with source words outside the STR. This is shown in **Figure 2**, **case 3**: "aime" comes from the phrase pair ("loves wine ,", "aime le vin ,") whose source side is partly inside, partly outside the STR; "aime" is a word aligned with a word inside the SPR ("loves") but outside the STR. This is where word-alignment information helps us. Here, we copy the tag pair surrounding the STR into the output so that it surrounds every target word that is word-aligned with a word in the STR. So "aime" is not **bold** even though it is in a phrase that is partially bold.

3. If the TCP are not contiguous in the output, we copy the tag pair from the STR into the output in such a way that all target phrases in the TCP are surrounded by the tag pair, thus extending the tag to apply to intervening phrases too. (We may also apply rule 2 if the SPR extends beyond the STR, to decide which target words to include in the phrases at the boundaries of the TCP). **Figure 3**, **case 4** shows this along with two alternative rules that could have been applied.

4. Sometimes, two or more disjoint STRs may generate overlapping TCPs, as in **Figure 3**, **case 5**. Here, STR_1 applies Lucida Handwriting typeface to words in its scope, STR_2 applies bold Times New Roman typeface, and the TCPs overlap. The above rules will yield tags that cross over in the output. This is seldom desirable, but there is no good rule to fix it automatically. TMX format allows its <bpt> and <ept> tag pairs to cross over, so we leave this situation as is in TMX files, letting the translator fix it by hand. However, XLIFF does not permit cross-over of its <g>/</g> tag pairs, so we must remove the cross-overs from the output. We arbitrarily chose to move the first close tag past the second close tag, creating a nested but legal structure for post-edition.

The rules for transferring isolated tags are:

1. If an isolated tag is a point tag in the document, it is considered to be attached to the next source word, and is placed before the target word aligned to that source word.

2. If an isolated tag is the close or open tag of a tag pair that starts or ends outside the current segment in the underlying document, then it is treated using rules 1 to 3 above, as if it was paired with an open tag at the beginning of the segment, or a close tag at the end of the segment, respectively.

In addition to these core rules, some care is taken to preserve the source order when multiple tags end up between the same two target-language words. Note that both cases in **Figure 3** might have been handled better by a one-stream system, which could have kept target words orig-

inating from the same STR together. However, this might have made the translation itself worse.

5 Mini-Evaluation

We manually evaluated the accuracy of tag placement in the output of our system on one document. This is a "mini-evaluation" because it involved only one document, and it did not look at post-editing effort, just tag placement. The goal was to understand what's going on, rather than to draw statistically defensible conclusions.

We trained our in-house phrase-based SMT system on 384K English-French sentence pairs collected during Nov. 2007 – Feb. 2008 from the Environment Canada (EC) web site of Environment Canada (www.ec.gc.ca), with 5.1M English and 6.1M French words in total. The EC website constitutes a nice parallel corpus because it has varied types of formatted content published in French and English simultaneously. Our system obtains BLEU scores in the low fifties on randomly sampled dev and test sets of 1500 sentences (trained/tested on text stripped of tags; one reference for dev and test).

We tested tag transfer on a more recent EC document, cutting/pasting the HTML text into MS Word. The TMX test file was created by extracting segments from the document with SDL Trados 2007. It had 367 segments, most with multiple sentences. There were 298 open/close tag pairs and 161 isolated tags for a total of 757 tags, with on average 2.1 tags per segment (tag pairs count as two tags). 56% of segments had no tags, 22% had one tag, while the remaining 22% had two or more; 6% had ten or more. The segment with the most tags had 24.

The XLIFF test file was created by extracting the segments from the document with SDL Trados Studio 2011. It had 1026 segments, most containing a single sentence. There were 194 open/close tag pairs and only 2 isolated tags for a total of 390 tags, with an average of 0.38 tag per segment. 91% of the segments had no tags, 3% had two tags (one tag pair), 6% had more; only 0.6% of segments had more than ten. The highest tag count for a segment was 14.

The TMX and XLIFF statistics are not strictly comparable because the XLIFF data are segmented into sentences, unlike the TMX data. However, there are fewer tags in the XLIFF, which makes tag transfer easier. In particular, the near absence of isolated tags in XLIFF shows that it abstracts more formatting information out of the segments.

We gave each tag in each segment produced by our system one of the following annotations (we evaluated only tag placement, not translation quality): **Good** (tag is well placed); **Reasonable** (tag needs to be moved, but is next to text that also needs to be moved, so these can be moved together); **Wrong** (tag is wrongly placed); **No chance** (the decoder's output is so bad there is no correct place for the tag). We added **Spacing error** if space before or after the tag should be deleted or inserted.

	TMX	XLIFF
Total no. of tags	757	390
Good	704 (93%)	351 (90%)
Reasonable	23 (3%)	8 (2%)
Wrong	27 (4%)	30 (8%)
"No chance"	3 (.4%)	1 (.3%)
Spacing error	165 (22%)	59 (15%)

Table 1. Mini-evaluation results by tags.

Table 1 shows tag-by-tag results. Most tags are placed correctly, reducing post-editing effort compared to a system with no tag transfer. A number of reasonable or wrong tags still need to be moved, though most of the remaining post-editing will fix spacing before or after tags. Scoring of spacing errors was strict: e.g., a space moved from before a tag to after it was counted as two errors (1 deletion + 1 insertion). Some of these spacing issues might not matter in practice.

	TMX	XLIFF
Segments with at least one tag	161	88
All tags good, no spacing errors	88 (55%)	33 (38%)
All tags good	143 (89%)	66 (75%)
All tags good or reasonable	144 (89%)	67 (76%)
At least one tag wrong or no chance	17 (11%)	21 (24%)
At least one spacing error	70 (43%)	41 (47%)

Table 2. Evaluation results by segments.

Table 2 shows our results by whole segments. Many segments have all tags placed correctly (89% / 75%) and a lower but still large proportion have completely correct spacing around tags (55% / 38%), and thus require no post-editing.

We also tried two methods for obtaining the word alignment within phrase pairs. The first,

"heuristic word alignment", uses heuristics developed by Foster et al. (2010) to recover a possible word alignment for a translation, given only its phrase pairs; the heuristics employ cognate information combined with a bias towards linear alignment. The second, "original word alignment" applies the original word alignment used to create the phrase table and stored in it. The two methods produce almost the same output; the number of differences is too small to draw any firm conclusions. Although a software bug affected some of our results, it is clear that the two methods perform at about the same level. This may be because the word alignment is not often required to place tags correctly, or perhaps because the heuristics of Foster et al. (2010) are well suited to the task at hand.

6 Discussion

We thought of, but did not try, some intriguing alternative rules for tricky tag transfer situations. In future, we'd like to try:

- Biasing the system to favour the original tag order whenever possible. Our current rules only require an "open" tag to precede the matching "close" tag, and require the XML in the target to be valid.
- An extreme version of this: a hard rule that considers contiguous sequences of tags as if they were just one tag. The translators/post-editors we observed working in SDL Trados TagEditor seemed to do this, and shortcuts in the interface support it.
- Better rules for handling spacing. During the mini-evaluation, we noticed that the detokenizer often left unwanted spaces around tag sequences. This may be due to a design error: the tag transfer module makes spacing decisions based on tokenized input. Then the detokenizer removes unnecessary spaces, but without knowledge of the source text. The system could use raw input, before tokenization, to decide what spaces need to be kept.

The major limitation of our paper is the lack of an experimental comparison between our method and alternative ones on a real post-editing task. A thorough study would compare a variety of one-stream and two-stream methods from the literature. It could employ human evaluation as in Du et al. (2010), or follow the more extensive protocol of Green et al. (2013), using eye, cursor and mouse tracking to measure the impact of different methods on post-editing. With respect to automatic metrics, evaluation of one-stream methods should involve at least two scores, one comparing tag-free output with tag-free references and one comparing tagged output with tagged references, to distinguish the impact of the methods on translation quality and their impact on formatting. Pure two-stream methods only affect formatting, so only the second score is necessary.

Another limitation is that we studied a language pair with similar word order (English-French). Language pairs with more reordering (e.g., German-English) make it harder for the pure two-stream approach to make sensible tag transfer decisions. Language pairs with different writing systems (e.g., Arabic-English or Chinese-English) pose deeper questions: e.g., what are the semantics of **bold** or *italic* in languages not written in the Roman alphabet? One might even wish to explore whether capitalization has semantic equivalents in other writing systems.

This paper describes implementation of a pure two-stream approach for transferring tags from source to target text in an SMT system. "Two-stream" means that tags are stripped from source text before the decoder sees them, and transferred along a separate route, as opposed to the "one-stream" approach where the decoder handles the tags. Our approach gives the decoder more freedom, but requires complex tag reinsertion rules. Experimental studies of both approaches in a realistic post-editing situation will be required to decide which approach is better.

Acknowledgement

We would like to thank CLS Lexi-tech Ltd. for creating our test TMX and XLIFF files. We would also like to thank the reviewers for their constructive suggestions for improving this paper.

References

Du, Jinhua, Johann Roturier and Andy Way. 2010. TMX Markup: A Challenge When Adapting SMT to the Localisation Environment. *EAMT 2010*, Saint-Raphaël, France, 253-260.

Foster, George, Pierre Isabelle and Roland Kuhn. 2010. Translating Structured Documents. *Association for Machine Translation in the Americas (AMTA 2010)*, Denver, Colorado, USA.

Green, Spence, Jeffrey Heer, and Christopher Manning. 2013. The Efficacy of Human Post-Editing

for Language Translation. *CHI '13, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.*

Hudík, Tomáš and Achim Ruopp. 2011. The Integration of Moses into Localization Industry. *Proceedings of EAMT 2011*, Leuven, Belgium.

Koehn, Philipp and Jean Senellart. 2010. Convergence of Translation Memory and Statistical Machine Translation. *AMTA Workshop on MT Research and the Translation Industry*, Denver, Colorado, USA.

Koehn, Philipp, Hieu Hoang, *et al*. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. *Proceedings of the ACL 2007 Demo and Poster Sessions*, Prague, Czech Republic.

Localization Industry Standards Association (LISA). 2005. TMX 1.4b Specification. Accessed May 17, 2013. http://www.gala-global.org/oscarStandards/tmx/tmx14b.html

M4Loc. 2012. m4loc – Moses for Localization. Accessed May 14, 2013. https://code.google.com/p/m4loc/

OASIS. 2008. XLIFF Version 1.2 Specification. Accessed May 17, 2013. http://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html

SDL International. 2011. SDL Trados Studio 2011 Languages and Filters (guide). http://www.translationzone.com/en/landing/premium-downloads/sdl-trados-studio-2011-languages-and-filters.asp

Tezcan, Arda and Vincent Vandeghinste. 2011. SMT-CAT integration in a Technical Domain: Handling XML Markup Using Pre & Post-processing Methods. *Proceedings of EAMT 2011*, Leuven, Belgium.

Zhechev, Ventsislav and Josef van Genabith. 2010. Seeding Statistical Machine Translation with Translation Memory Output through Tree-Based Structural Alignment. *4th Workshop on Syntax and Structure in Statistical Translation (SSST-4)*, Beijing, China.