How Should Markup Tags Be Translated?

Greg Hanneman and **Georgiana Dinu**
Amazon AI
{ghannema, gddinu}@amazon.com

Abstract

The ability of machine translation (MT) models to correctly place markup is crucial to generating high-quality translations of formatted input. This paper compares two commonly used methods of representing markup tags and tests the ability of MT models to learn tag placement via training data augmentation. We study the interactions of tag representation, data augmentation size, tag complexity, and language pair to show the drawbacks and benefits of each method. We construct and release new test sets containing tagged data for three language pairs of varying difficulty.

1 Introduction

The quality of machine translation (MT) has drastically improved in recent years, making MT technology more widely used than ever before in applications ranging from financial services (Nunziatini, 2019) to fashion, social media, and other user-generated content (Kosmaczewska and Train, 2019; Birch et al., 2019; Michel and Neubig, 2018), among other tasks.

A large amount of content requiring translation is not isolated plain text. Rather, it originates in the context of structured documents, using document format specifications such as HTML, Microsoft Word, PDF, etc.

Currently, the translation industry addresses the translation of structured documents by dividing the task between a translation management system (TMS) and an underlying MT system (e.g. Federico et al. (2014)). Figure 1 shows a schematic of the process. The TMS parses, manipulates, and validates the higher-level document structure. It is responsible for finding the translatable portions of the input document, performing sentence segmentation on the content, sending it to an underlying MT system for translation, and placing the result back into the document structure. For example, the TMS may pass over material contained in HTML `<script>...</script>` tags, while sending to MT the contents of `<p>...</p>` tags and the string values of `<img alt="..."/>` attributes.

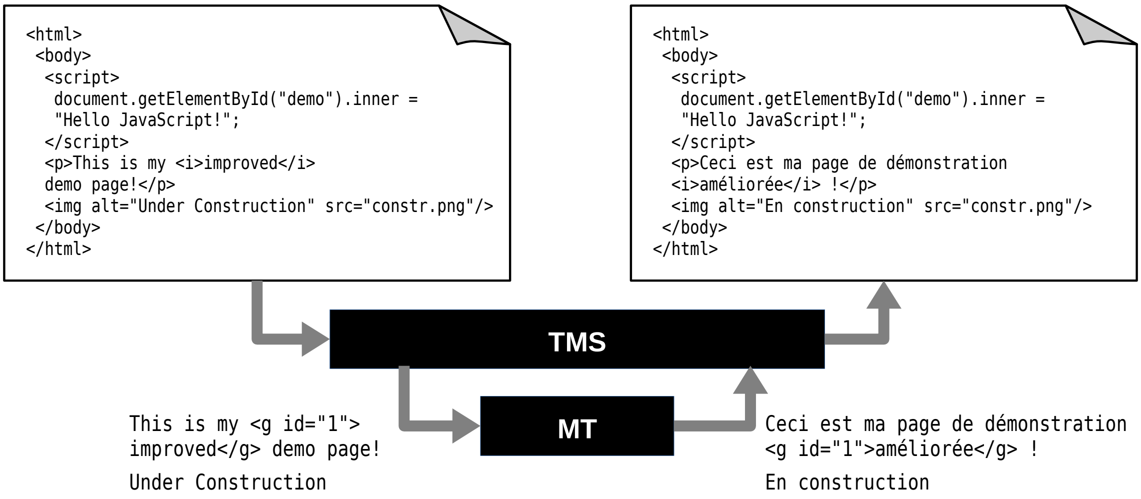
Properly transferring formatting tags *within* the translatable content (bold, italic, hyperlink, superscript, etc.) remains the responsibility of the MT process rather than the TMS. The correct preservation and transfer of inline markup from source to target thus forms a crucial component of the overall quality of the MT system. An accurate placement within the segment of inline markup provides a more readable and usable document in its raw MT form, in addition to saving human time and effort if post-editing is used.

Despite the key role of the MT system in processing structured documents — both in terms of TMS expectations and in lowering translation costs — the setup of translation in the context of structure is rarely addressed in the standard MT evaluation benchmarks. They typically focus on the pure-content, string-based tasks (Joanis et al., 2013; Müller, 2017).

In this paper, we study the question of how inline markup should be represented in and processed by the MT system in order to result in the highest placement accuracy. Given the deep semantic representations and generalization abilities provided by modern neural MT systems, we design a series of experiments to test their capabilities specifically on the problem of markup transfer within the translation process. The paper makes the following contributions:

(1) We propose a technique to augment any parallel corpus with inline tags, addressing the scarcity of high-quality parallel data containing markup tags (Section 3). We show that the method results in highly accurate tag placement and can improve the accuracy of tag placement of MT models when used to augment the training data.

Proceedings of the 5th Conference on Machine Translation (WMT), pages 1160–1173
Online, November 19–20, 2020. ©2020 Association for Computational Linguistics



```
<html>
 <body>
  <script>
   document.getElementById("demo").inner =
   "Hello JavaScript!";
  </script>
  <p>This is my <i>improved</i>
  demo page!</p>
  <img alt="Under Construction" src="constr.png"/>
 </body>
</html>
```

```
<html>
 <body>
  <script>
   document.getElementById("demo").inner =
   "Hello JavaScript!";
  </script>
  <p>Ceci est ma page de démonstration
  <i>améliorée</i> !</p>
  <img alt="En construction" src="constr.png"/>
 </body>
</html>
```

```
This is my <g id="1">
improved</g> demo page!
Under Construction
```

```
Ceci est ma page de démonstration
<g id="1">améliorée</g> !
En construction
```

Figure 1: Translation of a structured document. A translation management system (TMS) is responsible for higher-level document structure, while the machine translation (MT) system transfers inline markup.

(2) We provide a comprehensive evaluation of several tag-handling methods (Section 4). We test the ability of neural networks to jointly learn to translate content and transfer tags from automatically generated tagged data. Within this approach, we experiment with two ways of representing tags and compare these with a baseline detag-and-project method, on language pairs of varying data sizes and difficulty (English–French, English–German, and English–Hungarian).

(3) Finally, testing of tag placement accuracy poses difficulties in terms of both data-set availability and quality metrics. For this reason, we assemble new test sets of natively tagged structured documents for the three targeted language pairs. We enhance these test sets by adding synthetically generated tagged data according to human-quality glossary entries (Section 5), and we release these test sets in order to facilitate further research on the topic. Evaluation is carried out according to standard automatic metrics, automatic detection of obvious tagging errors, and human assessments of tag placement accuracy (Section 6).

2 Related Work

Various works have addressed the inline markup problem in the context of statistical MT systems. Joanis et al. (2013) extensively summarize many such methods. Their "two-stream" approach is quite similar to our detag-and-project baseline, relying on phrase and word alignments and carefully designed tag transfer rules to re-insert inline markup into the translated content. Though the accuracy of this approach is tested in a small human evaluation, it is explicitly not compared to any other method of tag representation.

Müller (2017) addresses the same question, comparing five different varieties of detag-and-project and mask-based approaches. The work concludes that the detag-and-project approach of Joanis et al. (2013) performs the best in complicated tagging scenarios, while masking can be a strong approach in simplistic cases.

Moving to neural MT, Hashimoto et al. (2019) experiment with "raw" tags that are left unmodified in the input. The authors test a constrained beam search that restricts the decoder to outputting all and only the tags actually present in the source while maintaining XML well-formedness. They also introduce a pointer mechanism (See et al., 2017) to promote copy-through of tags and other non-translatable content. These restrictions are shown to improve output, but no other tag representation aside from raw tags is tested.

We thus position our work as spanning and unifying elements of these previous studies, directly comparing the major tag representation methods with a minimum of other modeling changes in a state-of-the-art neural MT setting. Additionally we introduce a data augmentation technique that removes the dependence on pre-existing tagged training data assumed by the approaches proposed by e.g. Hashimoto et al. (2019).

In this context, we also note complementary work on the implicit learning of structure by sequence-to-sequence models. Target-side structure expressed as content, without special annotations, has been used in string-to-tree syntax-based MT by Nădejde et al. (2017) and Aharoni and Goldberg (2017); the latter group notes that well-formed parse trees are generated 99.5% of the time. The masking method of passing abstracted information between input and output has been applied in MT to non-translatables such as numbers, URLs, named entities, etc. (Crego et al., 2016; Post et al., 2019). It has been shown to improve automatic metric scores, with the masks correctly appearing in the MT output a very high percentage of the time (Murakami et al., 2019; Bérard et al., 2019). Source factors, or additional input streams provided in lockstep with the content to translate (Sennrich and Haddow, 2016), are one mechanism for signaling structural information in the input while leaving it accessible to the model. It has been used to augment an MT system with terminology constraints (Dinu et al., 2019), and the same mechanism could be in principle applied to tags as well.

Training data augmentation is a popular way to increase model learning and robustness for specific phenomena. Of particular interest is the finding by Karpukhin et al. (2019) showing that a "balanced diet" of training data synthetically augmented with spelling errors improved model performance on natural errors, without harming performance on clean text. Our notion regarding synthetically injected and naturally occurring tags is similar.

3 Data Augmentation via Tag Injection

The vast majority of data available for MT research appears as plain text only.[1] While tagged training corpora, when available, will inevitably be limited in size, domains covered, and language availability, we propose as an alternative a general technique for injecting markup tags of controlled complexity into any parallel corpus with high accuracy.

Although not always adhering to the strict syntax of a well-defined markup language, most inline tags follow the XML standard and create a hierarchical structure within a segment by introducing either paired tags (opening and closing) or self-closing unary tags. This structure is expected to be preserved by the translation, with paired tags surrounding corresponding text fragments. In Figure 1, the italicized fragment *improved* transfers to italicized *améliorée*. We propose a method to identify such corresponding fragments in the source and target sides of parallel data and to automatically inject tags based on them.

3.1 Tag Injection

We identify corresponding fragments using the hypothesis that, if the out-of-context translation of a sentence fragment is found in the target sentence, then those text fragments are aligned. More formally, assume source segment s and target segment t are each decomposed in three substrings $(s,\ t) = (a\,b\,c,\ x\,y\,z)$, where b and y are not empty. We hypothesize that if an MT model translates b into y in isolation, then b and y are corresponding fragments and we can inject the following tag structure:

$$a \texttt{ <t>}b\texttt{</t> } c$$
$$x \texttt{ <t>}y\texttt{</t> } z$$

In our implementation, the search over candidate n-grams b proceeds in random order, subject to a parameter controlling the maximum span. Our preliminary experiments showed that natively tagged text is not always well-formed. For this reason we introduce a "pair damage fraction" parameter, which sets the probability that one half of the injected tag pair will be skipped. This produces the patterns a <t>b c or a b</t> c, with analogous results on the target side. We also incorporate a fixed probability that the tag will be injected as self-closing rather than a pair, i.e. as a <t/>b c or a b<t/> c. Finally, we allow for the injection of multiple tags into the same parallel segment via a parameter that specifies the maximum number of tag pairs to insert per segment.

This method can be adapted to any markup schema choice. For the experiments described in this paper, we choose XLIFF, a translation industry standard that uses a reduced vocabulary of tag and attribute names as an abstraction over an original document's markup language. We inject four tags in accordance with the XLIFF 1.2 specification:[2] <g>...</g> for a paired tag, <x/> for a self-closing tag, <bx/> for a tag that opens without closing, and <ex/> for a tag that closes without opening. In all cases we include a numerical id attribute, with the value starting at 1 for the leftmost tag injected in each source segment and incrementing by one for each successive tag in left-to-right

[1] A recent exception is the 17-language XML-tagged parallel corpus described by Hashimoto et al. (2019), released publicly in July 2020 concurrently with our work.

[2] http://docs.oasis-open.org/xliff/v1.2/cs02/xliff-core.html#Specs_Elem_Inline

Language	Tags	Cor	Inc	Imp	Unc
EN–DE	532	484	22	8	18
		501	17	7	7
		501	17	4	10
EN–FR	560	504	29	7	20
		504	31	5	20
		508	38	3	11
EN–HU	540	438	47	49	6
		497	32	6	5
		463	20	47	10

Table 1: Human evaluation of tag injection accuracy, with the count of tags judged as **Cor**rect, **Inc**orrect, **Imp**ossible, or **Unc**lear by each of three annotators.

source order.

Tag injection may fail in specific instances for several reasons, such as inability to find a $(b,\, y)$ pair, or matched phrases overlapping with each other and blocking intended injections. We do not explicitly remove or attempt to repair any of these cases, using them instead as additional sources of variety and noise in the final tag-injected corpus.

3.2 Injection Evaluation

We test our hypothesis that the proposed procedure leads to accurate data by performing a human evaluation on the accuracy of tag placement. This experiment is carried out on a random selection of 200 sentence pairs from our training data in each of three language pairs used in our subsequent experiments: English–German (EN–DE), English–French (EN–FR), and English–Hungarian (EN–HU). (See Section 5.1 for details on the training data itself.)

We ask bilingual speakers in the relevant languages to judge whether each individual tag is correctly placed in the output, incorrectly placed (and a proper location for it can be identified), impossible to place (because no correct location exists given the target content), or too unclear to evaluate (e.g. because a placement decision depends on the semantics of the tag). Each set of sentences is independently evaluated by three different people who are not aware of how the tagged data was created.

The results of the evaluation (Table 1) show that the tag injection is accurate: using all the judgements combined, tags are correctly placed in 93.1% of cases in EN–DE, 90.2% in EN–FR, and 86.3% in EN–HU. The rates of actually wrongly placed tags are 3.5%, 5.8%, and 6.1%, respectively, with the remaining tags being judged as impossible to place or as unclear.

Pairwise inter-annotator agreement varies: judges clearly disagree (correct vs. incorrect) no more than 2.3% of the time in EN–DE and EN–FR, though up to 5.9% in EN–HU. Because of the large class imbalance, we also examine Fleiss's κ metric with all three annotators. The overall κ scores are 0.69 for EN–DE and 0.72 for EN–FR, but only 0.39 for EN–HU. In Hungarian, a larger number of tags are judged as correctly placed by one annotator but impossible to place by another, underscoring the difficulty of markup transfer between morphologically and grammatically distant languages.

4 Tag Representations

4.1 Baseline: Detag/Project

Our baseline setup does not model inline tags in any way. Instead, all markup is removed from the run-time input and reinserted into the MT output by a post-processing step. Reinsertion is carried out via *tag projection*: it uses the position of a tag in the input, a subword-level alignment model between source and target, and a set of projection heuristics to determine the analogous position of the tag in the MT output.

The necessary alignment model is built from the MT system's own training data. We use BPE (Sennrich et al., 2016) for subword creation and FastAlign (Dyer et al., 2013) for training the alignment model. At run time, we force-align the detagged BPE-level MT input to the BPE-level MT output, then convert the source-side subword indexes back to their token-level equivalents.[3] This provides a mapping between input tokens and output BPE pieces. A tag or tag pair is transferred from input to output according to this map and several hardcoded rules. Let s_i and t_i represent the ith source token or target BPE piece, respectively, in a segment of length I and J, and let $A(s_i) = \{j\}$ be the set of target indexes j of alignments induced for s_i. Then the most important projection rules are:

- A tag pair <x>...</x> spanning $s_a ... s_b$ encompasses alignments $\ell = \bigcup_{i=a}^{b} A(s_i)$ and is projected to span $t_{\min(\ell)} ... t_{\max(\ell)}$.
- A self-closing tag <x/> appearing before s_a follows alignments $\ell = A(s_a)$ and is projected to appear before $t_{\min(\ell)}$.

[3] An alternative is building and applying the alignment model on tokens instead of BPE pieces. As a practical concern, we prefer the BPE level for ease in handling non-whitespace languages and for its substantially smaller model size.

Original	`<bx id="1"/>As regards <g id="2">exports</g> of <g id="3">radioactive waste</g> from the Community to third countries, six Member States issued a total of 13 authorisations, representing 35 shipments.`
Masked	`__XLF_BX,1__ As regards __XLF_OPENG,1__ exports __XLF_CLOSEG,1__ of __XLF_OPENG,2__ radioactive waste __XLF_CLOSEG,2__ from the Community to third countries , six Member States issued a total of 13 authoris@@ ations , representing 35 shipments .`
Raw	`<bx id="1"/> As regards <g id="2"> exports </g> of <g id="@@ 3"> radioactive waste </g> from the Community to third countries , six Member States issued a total of 13 authoris@@ ations , representing 35 shipments .`

Figure 2: Example of inline markup tag representation when it is included in the MT input, either by masking or in raw form.

- A tag pair with zero span or a self-closing tag appearing before s_0 is projected to appear before t_0. The same appearing after s_I is projected to appear after t_J.

Additional heuristics specify behavior for cases when the relevant alignment set is empty, when the nesting of the input tags is malformed, when an unpaired tag <x> or </x> appears without its other half, etc.

4.2 Masked and Raw Representations

We compare the above approach to two alternatives where some representation of inline markup is provided to the MT system. Tags are either masked to generic placeholder tokens or else left raw in the input. Figure 2 gives an example of each.

Aside from reducing vocabulary, masking protects the original content from subword splits and mutilations during the MT process. In our implementation, if a mask is present in the source but fails to appear in the MT output, we forcibly add it back at the end of the output; spuriously generated masks are likewise removed. We use a different placeholder name for each of the five XLIFF tag names present in our data; this treats <g> and </g> independently. We also include a sequence number in the mask token, so that the original content can be matched with the correct placeholder in the target side. Similar to the XLIFF `id` parameter, these sequence numbers start with 1 in each sentence pair and increase left to right in the source sentence. Unlike in XLIFF, our placeholder sequence numbers are incremented individually for each of the five placeholder names.

Our other alternative approach leaves the XLIFF tags raw in the input, trusting the MT system to learn their correct formatting as well as placement.

Both the masking and the raw approach rely on tags appearing often enough in the training data — not only for the MT system to learn their transfer and placement, but also to be recognized as tokens as part of the system's subword vocabulary. Since the masks are single tokens, adding self-translating examples as additional parallel data and exempting the mask tokens from BPE application is sufficient. For the raw approach, we include the same number of self-translation examples to boost the frequency count of XLIFF tag tokens above the minimum BPE frequency cutoff. As Figure 2 illustrates, however, this does not necessarily ensure that all possible tag tokens appear often enough in the training data to be recombined by BPE into full tokens: lower-numbered tags still occur more frequently than higher-numbered examples.

5 Experimental Setup

5.1 Training Data

Our training data is sourced from the Conference on Machine Translation (WMT) series of shared tasks. Since our focus is on inline tag handling rather than corpus filtering or new state-of-the-art translation quality, in some cases we have ignored especially large or noisy data sets.

For EN–DE, we begin with the training data released by the WMT 2020 news task, ignoring the Common Crawl and Paracrawl corpora and heavily filtering WikiMatrix. Our EN–FR training data comes from the 2014 news translation task; we use only Europarl, News Commentary, and UN Docs. For EN–HU, we use the single available training corpus from the 2009 translation task.[4] Our final training data comprises 5.7 million lines for EN–DE, 14.5 million lines for EN–FR, and 1.5 million

[4] Data available from http://www.statmt.org/wmt20/translation-task.html (EN–DE), http://www.statmt.org/wmt14/translation-task.html (EN–FR), and http://www.statmt.org/wmt09/translation-task.html (EN–HU).

lines for EN–HU. See Appendix A for the exact enumeration of component corpora, line counts, and a description of our filtering process.

On top of this baseline data, we experiment with various amounts of tag-injected data augmentation. We sample 1%, 2%, 5%, 10%, or 15% of the baseline training data and inject XLIFF tags into it as described in Section 3. We attempt to inject up to two pairs of tags per segment, with a max span of six tokens, pair damage fraction of 0.10, and self-closing fraction of 0.27. Candidate n-grams for tag injection are identified by looking for equivalent translations with the publicly available version of Amazon Translate as of April 2020. Lines that completely fail tag injection are discarded, not counted as part of the goal percentage, and replaced with successful lines. Each successive augmentation percentage is a strict superset of the ones before it: e.g. all the data present in the 2% corpus is also present in the 5%, 10%, and 15% settings.

5.2 Tagged Dev and Test Sets

Although we believe the tag injection technique achieves sufficiently high accuracy to be used in training data, we prefer our development and test sets to represent — as much as possible — naturally occurring tags in the source and human-quality placements for them in the target. This section describes the test sets created. They are also publicly available at https://github.com/amazon-research/mt-markup-tags.

EUR-Lex EUR-Lex[5] is the European Union's online repository of legal documents, which are provided synchronously in several structured formats and in the union's 24 official languages. We select a cohesive block of documents in Microsoft Word format, from CELEX numbers 52019DC0601 through 52019DC0680, to serve as the base of our dev and test sets. Each document is available as monolingual downloads in English, German, French, or Hungarian; several processing steps are needed to create a sentence-aligned tagged parallel corpus.

For a set of four monolingual documents, we first extract each one from Word to XLIFF format using the open-source Okapi Tikal document filter.[6] Aside from performing automatic paragraph and sentence segmentation according to pre-defined rules, the Okapi filter also converts the inline Microsoft markup to XLIFF 1.2 tags. We then check the extracted XLIFF documents for parallelism at several levels. Any document set that does not contain the same number of paragraphs across all four languages is entirely rejected. Any paragraph that does not have the same number of sentences across languages is skipped; any sentence that does not have the same set of XLIFF tags across all languages is likewise skipped. The surviving sentences form a four-way parallel corpus with inline markup tags. Each successfully extracted document is then assigned to either the dev or the test set. Sets assembled in this way are finally deduplicated to unique sentence pairs. The EUR-Lex dev set contains 1888 lines; the test set 1450.

We find, surprisingly, that a significant number of otherwise parallel segments do not contain the same inline tags across all languages. One side effect of enforcing this restriction is that the tags that are indeed parallel are biased towards trivial cases, such as an opening tag at the beginning of the sentence and a closing tag at the end. Only 11% of the sentences extracted above contain line-medial tags, and only 4% contain more than two tags per line. We mitigate this problem via the construction of two additional sets.

EUR-Lex mono We return to the unfiltered monolingual English documents assigned to the EUR-Lex test set. Without parallelism restrictions, this collection forms a much more diverse test set: after deduplication, 26% of lines contain medial tags and 13% hold more than two tags. While we are unable to use this for MT evaluation metrics that require a reference, we employ this 2525-line test set for other types of automatic and human evaluation.

Glossary We use additional EUR-Lex documents (CELEX numbers 52019DC0520 to 52019DC0599) to construct dev and test sets with synthetically introduced tags. In contrast to tag injection, however, the markup is inserted using human-curated translation glossaries. We extract and filter each set of monolingual Word documents as before, with the additional step of removing all the inline tags to obtain plain text. Given four-way parallel segments, we then search within each line for a synchronous occurrence of entries from our glossaries for EN–DE, EN–FR, and EN–HU. If

[5] https://eur-lex.europa.eu/homepage.html

[6] https://okapiframework.org/wiki/index.php/Tikal

found, the terms are surrounded by a pair of identical XLIFF `<g>...</g>` tags in each language. The synchronous glossary restriction reduces heavily the amount of successfully extracted and tagged sentences. On the other hand, it provides in practice 100% examples with line-medial tags, as well as an improved 22% of lines containing more than two tags. The final sizes are 286 lines for dev and 289 for test.

The complete dev set for our MT systems is a concatenation of three different sources. First is the official WMT dev set that corresponds to the training data: newstest2018 for EN–DE, newstest2013 for EN–FR, and newsdev2009 for EN–HU. To this we add the EUR-Lex and glossary dev sets described above. Tags are removed from these sets when they are used in the baseline system, which is not trained on any tagged data. Test sets are kept separated by data source. In addition to tagged and detagged versions of the EUR-Lex and glossary test sets described above, we include the EUR-Lex mono test set and the WMT official test sets: newstest2019 for EN–DE, newstest2014 for EN–FR, and newstest2009 for EN–HU. See Appendix A for the complete itemization of dev and test sets.

5.3 MT Systems

All our experiments are carried out using the Sockeye neural MT toolkit (Hieber et al., 2017). We use the Transformer architecture (Vaswani et al., 2017), with a hidden layer size of 512, an encoder of 20 layers, and a decoder of 2 layers: Hieber et al. (2020) report improved WMT results for such a configuration. For training, we set the batch size to 8192 tokens and the checkpoint interval to 2000 batches. Optimization is carried out with Sockeye's implementation of the Adam algorithm (Kingma and Ba, 2014). The learning rate, from an initial value of 0.0002, is multiplied by a factor of 0.9 every time eight training checkpoints pass without any improvement in dev-set perplexity. Training is stopped when there is no improvement after 32 checkpoints. Following convergence, the parameters from the eight best checkpoints are averaged.

We present a total of 33 experimental configurations: the detag-and-project baseline, plus the cross product of $\{1, 2, 5, 10, 15\}\%$ tag-injected data augmentation with {masked, raw} tag representation, for each of our three language pairs.

6 Results

6.1 Evaluation Metrics

Depending on the test set, we use a variety of evaluation metrics to judge performance.

We compute case-sensitive BLEU scores according to the SacreBLEU implementation (Post, 2018).[7] We distinguish untagged BLEU scores computed on test sets with no source-side tags (or for which the source-side tags have been removed) from tagged BLEU scores, where the tags are tokenized and treated as content by the built-in SacreBLEU tokenizer. Evaluation occurs only after any masked placeholders have been converted back to literal output. For each system, we also compute statistical significance relative to the baseline using stratified approximate randomization (Yeh, 2000).

Tagged test sets are also evaluated according to specifically designed "flagrant failure" metrics, whose goal is to detect obviously erroneous tag placement in MT output. Automatic evaluation of tag placement becomes difficult as the MT output diverges from a tagged reference translation's word choice, sentence structure, etc. Still, certain errors can be reliably detected regardless of language or context. We define flagrant-failure metrics to count occurrences of dropped, added, or mutilated tags, along with tags that become improperly nested relative to the source. In XLIFF, we distinguish a change of index — from e.g. `<g id="2">` to `<g id="4">` — as its own type of failure, rather than as independent drop and add mistakes.

Finally, we conduct a full human evaluation of tag placement accuracy, similar to the one introduced in Section 3.2. Due to the large amount of data involved, in this case we collect judgements from only a single annotator. Each tag is evaluated independently for whether its placement in the target is correct, incorrect, impossible given the MT output, missing, duplicated, or unclear.

6.2 Evaluation of Untagged Input

Our first concern is to ensure that augmenting the training data with tag-injected content does not harm translation of untagged inputs. We validate this claim on untagged versions of our WMT, EUR-Lex, and glossary test sets, comparing the BLEU scores of the tag-augmented systems with the score of the baseline, which was trained without tags.

[7] BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.4.3

Lang	System	Set	ΔBLEU	*p*
EN–FR	Raw 10%	WMT	–0.4	0.02
EN–HU	Mask 15%	WMT	–0.3	0.03
EN–HU	Mask 5%	WMT	–0.3	0.04
EN–HU	Raw 10%	WMT	–0.3	0.05

Table 2: Experimental conditions with the most significant untagged BLEU differences relative to the baseline. Tag representation has no consistent effect on translation quality of untagged content.

As hoped, we observe no clear pattern of degradation by tag representation, augmentation percentage, or language pair. Out of the 90 experimental cases, only four reach statistical difference with the corresponding baseline at $p < 0.05$. These scattered instances are shown in Table 2. See Appendix B for the full results.

6.3 Automatic Evaluation of Tag Placement

We have two proxy metrics at our disposal for automatically evaluating tag placement accuracy: tagged BLEU scores and flagrant-failure counts.

The previous section indicated that training-data augmentation did not have a strong effect on translation quality of *content*. We now turn to tagged BLEU scores to see if they provide any signal as to the translation quality of *markup*. As shown in Table 3, we observe a small positive improvement for the masking approach: statistical difference in 17 of 30 cases, and +0.79 BLEU on average when compared to the detag-and-project baseline. This effect is strongest for French and for 15% augmentation. At 15% masked augmentation, four of six results achieve statistical significance, and the average improvement is 1.00 BLEU. The best results for raw XLIFF systems appear scattered: the highest BLEU gains at 10% augmentation, the most consistent at 2%, and statistical significance achieved at least once everywhere except at 5% augmentation and in EN–DE.

We cross-check these conclusions by examining the flagrant failure rates. This analysis shows an extreme variability by test set. No configuration reaches more than a 3.0% failure rate on the EUR-Lex test set or 5.2% on the glossary test set. However, the increased diversity of tag indexes and tag patterns in the EUR-Lex mono test set provides for a much higher rate: up to 24.7% in the worst case. The full count of flagrant failures on the mono test set is displayed in Table 4.

By hard-coded design, the detag-and-project approach is not capable of changing, dropping, mutilating, or adding tags — they are placed without modification as a post-process after translation. We find that this technique also does not commit any flagrant errors of tag nesting on our test sets. Similar hard-coded limits affect the masking approach; the one instance of a dropped tag that we recorded is due to a tokenization error. Masking does, however, commit a certain number of nesting mistakes.[8] The raw-tag approach is susceptible to all kinds of flagrant failures. We note that, surprisingly, the number of errors tends to *increase* as more tagged examples are added to the training data in German and French, while Hungarian (with much smaller training data) does not show a clear pattern in any direction.

	`<g`	`id="2">`
Raw 1%	183,475	74,055
Raw 15%	2,606,016	1,098,275

	`<g`	`id="@@`	`3">`
Raw 1%	183,475	3,150	1,300
Raw 15%	2,606,016	2,550	1,300

Figure 3: Training-data token frequencies for the BPE pieces involved in EN–FR translation of two tags. Indexes above 2 are seldom seen.

Increased errors in German are primarily due to more tags being generated with incorrect `id` parameters. Recall that our settings for tag injection (Section 5.1) introduce no more than two tag pairs into any sentence pair, resulting in a maximum `id` value of 2. The only examples of indexes beyond 2 in the training data come from the addition of tag self-translations (Section 4), which we include for indexes up to 20. These higher indexes never occur in the context of any content, and their relative prominence in the training data decreases as more tag-injected data is added, so the MT system may become less and less sure how to "translate" them in practice. The failures for German confirm this pattern. At 1% augmentation, all 30 tags with their IDs incorrectly changed have values beyond 2, but the MT system produces a different value beyond 2 in five cases. At 15% augmentation, the system does not propose a value beyond 2 in any of the 206 failure cases.

Arguably, the increased training focus on low-

[8] This count would include tags natively dropped by the MT system but re-added to the end of the output by rule. Masked systems produce on average 3% more line-final tags than raw systems, but essentially the same number as the baseline.

		Detag/	Masked					Raw				
Language	**Set**	**Project**	**1%**	**2%**	**5%**	**10%**	**15%**	**1%**	**2%**	**5%**	**10%**	**15%**
EN–DE	EUR-Lex	70.0	0.1	0.2	0.4	0.1	0.3	–0.2	0.2	0.2	–0.1	0.2
	Glossary	60.2	1.1	0.6	1.2	0.8	0.8	1.0	0.6	1.0	0.8	0.8
EN–FR	EUR-Lex	65.9	1.0	0.1	0.9	0.1	1.2	0.5	0.5	0.3	0.9	–0.5
	Glossary	61.7	0.9	0.9	0.9	1.0	1.7	1.1	0.6	0.7	0.7	0.2
EN–HU	EUR-Lex	61.4	0.2	0.3	0.2	0.4	0.6	–0.5	0.0	–0.1	–0.2	–0.1
	Glossary	53.5	1.2	1.6	1.4	1.8	1.5	–0.3	0.3	0.6	1.8	1.3

Table 3: BLEU scores on tagged test sets, shown as differences from the baseline (detag-and-project) system's performance on the same test set. Results in grey are statistically significant at $p < 0.05$. The masked representation tends to produce the best results.

			Masked					Raw				
Language	**Failure**	**DP**	**1%**	**2%**	**5%**	**10%**	**15%**	**1%**	**2%**	**5%**	**10%**	**15%**
EN–DE	Changed ID	0	0	0	0	0	0	30	66	177	226	206
	Dropped	0	0	0	0	0	0	92	105	128	162	131
	Mutilated	0	0	0	0	0	0	17	3	85	91	129
	Badly Nested	0	12	14	14	15	10	26	25	33	28	31
	Added	0	0	0	0	0	0	22	11	14	10	6
	Total	0	12	14	14	15	10	187	210	437	517	503
EN–FR	Changed ID	0	0	0	0	0	0	5	112	325	349	95
	Dropped	0	0	0	0	0	0	34	81	82	115	423
	Mutilated	0	0	0	0	0	0	1	59	232	249	257
	Badly Nested	0	9	16	9	20	16	10	16	16	28	162
	Added	0	0	0	0	0	0	10	21	17	18	36
	Total	0	9	16	9	20	16	60	289	672	759	973
EN–HU	Changed ID	0	0	0	0	0	0	337	357	319	358	306
	Dropped	0	0	1	1	0	1	265	243	257	221	272
	Mutilated	0	0	0	0	0	0	24	19	57	4	45
	Badly Nested	0	26	10	8	16	17	39	28	35	37	33
	Added	0	0	0	0	0	0	37	44	43	28	18
	Total	0	26	11	9	16	18	702	691	711	648	674

Table 4: Flagrant failure counts on the EUR-Lex mono tagged test set. ("DP" = detag and project.) Tag translation failures increase rapidly as the training data is augmented with more raw tags.

index tags should be equally true of the mask-based systems. A key difference is illustrated by the rise in mutilated tags for French. Masked tags are always expressed as single tokens; if the self-translated examples are enough to induce a copy-through behavior for them, the behavior can be correctly applied in any content segment regardless of context. However, raw tags are expressed as at least two and sometimes more tokens (cf. Figure 2), which turn out to have wildly different frequencies in the training data as augmentation increases.

Figure 3 illustrates the BPE pieces involved in translating the raw tags `<g id="2">` and `<g id="3">` from English to French, along with the observed frequencies of those tokens in the training data. For a low-index tag, the tokens are quite common and, moreover, have relatively balanced counts: the opening token is seen roughly 2.4 times as often as the closing token no matter the data augmentation percentage. The situation is markedly different for a higher-index tag. In this case, increasing amounts of tag injection heavily shift vocabulary mass toward the opening token: it already appears 58 times more frequently than the tail at 1% augmentation, a ratio that grows to 1022 at 15%.

This extreme imbalance may be responsible for the 15% model's tag mutilation behavior. On our mono test set, this model never produces the first token of a tag without the tail, nor does it mutilate any tags for indexes up to 2. All the mutilation failures are caused by producing the tail alone for higher indexes, e.g. `id="3">` — exactly corresponding to the string of low-frequency tokens where a copy-through behavior can be easily learned.

6.4 Human Evaluation of Tag Placement

We conduct a human evaluation of tag placement accuracy in order to get a more complete picture of errors than afforded by tagged BLEU scores and flagrant-failure counts. Since it is impractical to

(a) Lines with indexes 1 and 2 only

Lang	System	Good	Bad	Residual
EN–DE	detag/project	97.3%	2.7%	0.0%
	masked (15%)	98.6%	1.4%	0.0%
	raw (1%)	98.4%	1.6%	0.0%
EN–FR	detag/project	92.9%	4.0%	3.1%
	masked (15%)	97.0%	0.5%	2.5%
	raw (1%)	96.1%	1.4%	2.5%
EN–HU	detag/project	93.8%	4.9%	1.3%
	masked (15%)	96.9%	2.5%	0.9%
	raw (10%)	96.1%	3.0%	0.9%

(b) Lines with indexes 3 and above

Lang	System	Good	Bad	Residual
EN–DE	detag/project	87.3%	12.7%	0.0%
	masked (15%)	84.2%	15.8%	0.0%
	raw (1%)	83.6%	16.0%	0.4%
EN–FR	detag/project	77.7%	9.8%	12.5%
	masked (15%)	78.7%	9.0%	12.3%
	raw (1%)	80.8%	6.2%	12.9%
EN–HU	detag/project	77.9%	19.3%	2.7%
	masked (15%)	70.9%	19.5%	9.6%
	raw (10%)	33.0%	63.8%	3.2%

Table 5: Summarized human evaluation of tag placement accuracy.

collect judgements on full test sets for all 33 experiments, we restrict this study to subsets of both. In terms of systems, we evaluate the detag-and-project baseline, the 15% masked augmentation, and either the 1% raw (EN–DE, EN–FR) or 10% raw (EN–HU) augmentation. For data, we use the entire glossary test set (289 lines), all the lines in the EUR-Lex monolingual test set containing tag indexes 3 and above (283 lines), and an equal number of randomly sampled lines from the same test set containing indexes 1 and 2 only.

In summarizing the results, we collapse the eight annotation types into three categories. "Good" tags are those placed correctly in the output, including if they were correctly deleted or duplicated. "Bad" tags are incorrectly placed, incorrectly dropped or duplicated, hallucinated, or mutilated in the output. "Residual" tags are those judged as impossible or unclear to place. Given the marked difference in flagrant failures observed for tag indexes 1 and 2 versus 3 and beyond, we report human judgements separately by whether the input included indexes beyond 2 or not. These summarized results appear in Table 5.

Placement for low-index tags present in the augmented training data is learned quite well: in all language pairs, the masked and raw-tag systems outperform the detag-and-project baseline. Humans also find the annotation of inputs with few tags to be a straightforward task, as very few tags are marked as awkward to place.

Results are less clear-cut on high-index tags that appear in training only as self-translated examples. Word-alignment-based projection works best in EN–DE and EN–HU. Translating raw tags does well in EN–DE and EN–FR but is unusable in EN–HU. The masking approach performs consistently in second place. Especially in French, the human task of judging placement accuracy has become notably harder, an effect that could also significantly affect the good/bad results of any method.

7 Conclusion

We have performed a comprehensive evaluation of several tag representation methods and proposed a data-augmentation technique that allows MT models to jointly learn content translation and inline tag placement.

Results show that representing tags as masks, together with data augmentation, leads to equivalent or improved performance over a detag-and-project approach: placement accuracy is higher for tags frequent in the training data, while it may vary for tags never observed in context. In practice, it may be preferable to rely on the MT model's ability to learn mask placement — even with some variability in accuracy — than to implement, debug, and maintain the baseline's more complicated projection rules and the required alignment model.

Raw tags, on the other hand, fail our generalization tests. Though placement accuracy is again baseline-beating for commonly observed tags, raw models seem unable to copy rare tags into the output without a significant number of mutilations, deletions, and duplications: an unacceptable result for the goal of obtaining well-structured output.

Several changes to our setup may improve the transfer of raw tags. Injecting XLIFF tags with a wider variety of `id` values is needed to expose the model to them in context instead of merely in self-translation. Explicitly identifying tag tokens via input factors (Dinu et al., 2019), or constraining/promoting the output of complete tags (Hashimoto et al., 2019), would also be helpful for reducing the rate of malformed output.

Acknowledgements

We thank Yaser Al-Onaizan, Marcello Federico, Stanislas Lauly, and Prashant Mathur for early discussions regarding the tag-injection technique.

References

Roee Aharoni and Yoav Goldberg. 2017. Towards string-to-tree neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 132–140, Vancouver, Canada. Association for Computational Linguistics.

Alexandre Bérard, Ioan Calapodescu, and Claude Roux. 2019. Naver Labs Europe's systems for the WMT19 machine translation robustness task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 526–532, Florence, Italy. Association for Computational Linguistics.

Alexandra Birch, Barry Haddow, Ivan Tito, Antonio Valerio Miceli Barone, Rachel Bawden, Felipe Sánchez-Martínez, Mikel L. Forcada, Miquel Esplà-Gomis, Víctor Sánchez-Cartagena, Juan Antonio Pérez-Ortiz, Wilker Aziz, Andrew Secker, and Peggy van der Kreeft. 2019. Global under-resourced media translation (GoURMET). In *Proceedings of Machine Translation Summit XVII Volume 2: Translator, Project and User Tracks*, pages 122–122, Dublin, Ireland. European Association for Machine Translation.

Josep Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart, Egor Akhanov, Patrice Brunelle, Aurelien Coquard, Yongchao Deng, Satoshi Enoue, Chiyo Geiss, Joshua Johanson, Ardas Khalsa, Raoum Khiari, Byeongil Ko, Catherine Kobus, Jean Lorieux, Leidiana Martins, Dang-Chuan Nguyen, Alexandra Priori, Thomas Riccardi, Natalia Segal, Christophe Servan, Cyril Tiquet, Bo Wang, Jin Yang, Dakun Zhang, Jing Zhou, and Peter Zoldan. 2016. SYSTRAN's pure neural machine translation systems. *Computing Research Repository*, arXiv:1610.05540. Version 1.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM Model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Marcello Federico, Nicola Bertoldi, Mauro Cettolo, Matteo Negri, Marco Turchi, Marco Trombetti, Alessandro Cattelan, Antonio Farina, Domenico Lupinetti, Andrea Martines, Alberto Massidda, Holger Schwenk, Loïc Barrault, Frederic Blain, Philipp Koehn, Christian Buck, and Ulrich Germann. 2014. The MateCat tool. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: System Demonstrations*, pages 129–132, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Kazuma Hashimoto, Raffaella Buschiazzo, James Bradbury, Teresa Marshall, Richard Socher, and Caiming Xiong. 2019. A high-quality multilingual dataset for structured documentation translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 116–127, Florence, Italy. Association for Computational Linguistics.

Felix Hieber, Tobias Domhan, Michael Denkowski, and David Vilar. 2020. Sockeye 2: A toolkit for neural machine translation. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 457–458, Lisbon, Portugal.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A toolkit for neural machine translation. *Computing Research Repository*, arXiv:1712.05690. Version 2.

Eric Joanis, Darlene Stewart, Samuel Larkin, and Roland Kuhn. 2013. Transferring markup tags in statistical machine translation: A two-stream approach. In *Proceedings of MT Summit XIV Workshop on Post-editing Technology and Practice*, pages 73–81, Nice, France.

Vladimir Karpukhin, Omer Levy, Jacob Eisenstein, and Marjan Ghazvininejad. 2019. Training on synthetic noise improves robustness to natural noise in machine translation. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 42–47, Hong Kong, China. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *Computing Research Repository*, arXiv:1412.6980. Version 9.

Kasia Kosmaczewska and Matt Train. 2019. Application of post-edited machine translation in fashion eCommerce. In *Proceedings of Machine Translation Summit XVII Volume 2: Translator, Project and User Tracks*, pages 167–173, Dublin, Ireland. European Association for Machine Translation.

Paul Michel and Graham Neubig. 2018. MTNT: A testbed for machine translation of noisy text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 543–553, Brussels, Belgium. Association for Computational Linguistics.

Mathias Müller. 2017. Treatment of markup in statistical machine translation. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 36–46, Copenhagen, Denmark. Association for Computational Linguistics.

Soichiro Murakami, Makoto Morishita, Tsutomu Hirao, and Masaaki Nagata. 2019. NTT's machine translation systems for WMT19 robustness task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 544–551, Florence, Italy. Association for Computational Linguistics.

Maria Nădejde, Siva Reddy, Rico Sennrich, Tomasz Dwojak, Marcin Junczys-Dowmunt, Philipp Koehn, and Alexandra Birch. 2017. Predicting target language CCG supertags improves neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 68–79, Copenhagen, Denmark. Association for Computational Linguistics.

Mara Nunziatini. 2019. Machine translation in the financial services industry: A case study. In *Proceedings of Machine Translation Summit XVII Volume 2: Translator, Project and User Tracks*, pages 57–63, Dublin, Ireland. European Association for Machine Translation.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Matt Post, Shuoyang Ding, Marianna Martindale, and Winston Wu. 2019. An exploration of placeholding in neural machine translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 182–192, Dublin, Ireland. European Association for Machine Translation.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2019. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. *Computing Research Repository*, arXiv:1907.05791. Version 2.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *COLING 2000 Volume 2: The 18th International Conference on Computational Linguistics*.

A Train, Dev, and Test Corpora

Table 6 gives a more detailed account of our training resources, showing the amount of material sourced from each standard WMT corpora we use. Following Figure 2 of Schwenk et al. (2019), we reduce the EN–DE WikiMatrix corpus to only those lines with a margin threshold of 1.05 or higher *and* where the WMT-provided language detection registered English on the source and German on the target. This is in addition to ignoring Common Crawl and Paracrawl entirely. For EN–FR, we discard the Common Crawl and Giga-FrEn corpora.

Clean-up on the training data is limited to removing lines where the source and/or target side is blank, removing lines whose source side is contained in one of our dev or test sets, tokenization, byte-pair encoding (Sennrich et al., 2016) using 32,000 operations, and length-based filtering. In this final step, we remove sentence pairs of more than 95 tokens on either side, containing tokens with more than 100 characters, or where the length ratio between source and target is too unbalanced.

Language	**Corpus**	**Lines**
EN–DE	Europarl	1,828,521
	News Commentary	371,225
	Rapid	1,631,639
	WikiMatrix (filtered)	916,242
	WikiTitles	1,382,687
	Total	6,130,314
	After filtering	5,746,433
EN–FR	Europarl	2,007,723
	News Commentary	183,251
	UN Docs	12,886,831
	Total	15,077,805
	After filtering	14,467,303
EN–HU	Hung-Train	1,517,584
	Total	1,517,584
	After filtering	1,465,919

Table 6: Line counts of baseline training data.

Language	Set	Source	Lines
EN–DE	Dev	WMT (nt2018)	2,998
		EUR-Lex	1,888
		Glossary	286
	Test	WMT (nt2019)	1,997
		EUR-Lex	1,450
		EUR-Lex mono	2,525
		Glossary	289
EN–FR	Dev	WMT (nt2013)	3,000
		EUR-Lex	1,888
		Glossary	286
	Test	WMT (nt2014)	3,003
		EUR-Lex	1,450
		EUR-Lex mono	2,525
		Glossary	289
EN–HU	Dev	WMT (nd2009)	2,051
		EUR-Lex	1,888
		Glossary	286
	Test	WMT (nt2009)	3,027
		EUR-Lex	1,450
		EUR-Lex mono	2,525
		Glossary	289

Table 7: Line counts of the dev and test sets.

Table 7 shows the sizes of our final dev and test sets, including WMT "newsdev" (nd) and "newstest" (nt) releases.

B Additional Results

Table 8 (on the next page) shows complete results on translating *untagged* test sets, to ensure that adding masked or raw tags to our training data does not adversely affect the translation of plain content. BLEU scores are computed according to SacreBLEU (Post, 2018), while statistical significance uses 1000 trials of stratified approximate randomization (Yeh, 2000). The small glossary test set shows the highest BLEU variance, but only once to statistical significance. Meanwhile, the few significant differences are scattered across language pairs, test sets, tag representations, and augmentation percentages.

			Masked					Raw				
Language	**Set**	**Baseline**	**1%**	**2%**	**5%**	**10%**	**15%**	**1%**	**2%**	**5%**	**10%**	**15%**
EN–DE	WMT	38.6	0.1	0.3	0.3	–0.1	0.0	0.1	0.2	0.1	–0.2	–0.3
	EUR-Lex	44.4	–0.5	–0.1	0.6	–0.3	–0.4	–0.9	0.5	0.3	0.1	0.1
	Glossary	39.9	0.6	–0.2	0.0	1.0	1.0	0.5	–0.5	0.2	0.0	0.5
EN–FR	WMT	37.5	–0.2	–0.1	–0.1	–0.1	0.2	0.0	–0.2	–0.1	–0.4	0.0
	EUR-Lex	43.0	–0.1	–0.3	0.4	–0.1	0.5	0.5	–0.1	0.0	0.5	–0.3
	Glossary	45.7	0.3	–0.2	0.1	0.3	0.9	0.7	0.7	0.3	0.7	–0.2
EN–HU	WMT	12.9	0.0	–0.1	–0.3	0.1	–0.3	0.0	0.0	–0.3	–0.3	0.0
	EUR-Lex	27.6	–0.4	–0.3	–0.5	–0.2	0.3	–0.2	0.1	–0.6	–0.1	0.0
	Glossary	27.4	–0.2	–0.3	–1.0	0.4	–0.7	–0.8	–0.4	–0.6	1.1	–0.1

Table 8: BLEU scores on untagged test sets, shown as differences from the baseline system's performance on the same test set. Cells in light grey are statistically significant at $p < 0.10$; dark grey indicates $p < 0.05$. Tag representation has no consistent effect on translation quality of untagged content.