Does Sentence Segmentation Matter for Machine Translation?

Rachel Wicks[1] and **Matt Post**[1,2]
[1]Center for Language and Speech Processing
[2]Human Language Technology Center of Excellence
Johns Hopkins University
{rewicks@,post@cs.}jhu.edu

Abstract

For the most part, NLP applications operate at the sentence level. Since sentences occur most naturally not on their own but embedded in documents, they must be extracted and segmented via the use of a segmenter, of which there are a handful of options. There has been some work evaluating the performance of segmenters on intrinsic metrics, that look at their ability to recover human-segmented sentence boundaries, but there has been no work looking at the effect of segmenters on downstream tasks. We ask the question, "does segmentation matter?" and attempt to answer it on the task of machine translation. We consider two settings: the inference scenario, where sentences are passed into a black-box system whose training segmentation is mostly unknown, and the training setting, where researchers have full control over the process. We find that the choice of segmenter largely does not matter, so long as its behavior is not one of extreme under- or over-segmentation. For such settings, we provide some qualitative analysis examining their harms, and point the way towards document-level processing.

1 Introduction

Contemporary machine translation assumes a sentence-level paradigm. However, data doesn't exist naturally at the sentence level, requiring the use of automatic segmenters to split the data at both training and inference time. Training data is prepared with the use of sentence segmenters,[1] which are preprocessing steps that occur prior to alignment and bitext creation. At test time, deployed models also require the use of a segmenter. Many times, for downloaded models, especially, this inference-time application must be made without knowing what segmenter was used to train the model, introducing a potential misalignment or discrepancy and resulting performance degradation.

[1]Sometimes called *sentence breakers*.

Sentence segmentation itself has received only a little attention in the research literature, although there has been a recent uptick (Moore, 2021; Wicks and Post, 2021). But to our knowledge, no work has been done investigating the effects of segmentation on machine translation. In fact, most research papers do not deal with the question at all, relying as they do on pre-segmented parallel data for both training and test time. This is a practical problem for deployment scenarios, where segmentation must be considered. It is also a deeper problem, since segmentation is ultimately a modeling decision that should be noted and made available with any published models, such as is done for other modeling decisions affecting input text, such as normalization, tokenization, and subword processing.

To understand whether and to what extent segmentation matters, we ask a series of questions: (1) What segmenter is best used at inference time? (2) When training a model, how important is the choice of segmenter? We break down this last question into two settings: (i) the standard training procedure in which sentences from parallel documents are aligned (Gale and Church, 1993), and (ii) more recent "mining" approaches, which use sentence representations to find sentence pairs without regard for document boundaries.

We find that

- for two black-box models trained with unknown segmentation, inference-time segmentation largely does not matter;
- when training new models, more aggressive segmentation generally produces better models, but these models are less robust to training-/inference-time segmentation mismatch;
- Global bitext mining approaches generally outperform document-based alignment tech-

Proceedings of the Seventh Conference on Machine Translation (WMT), pages 843–854
Abu Dhabi, December 7–8, 2022.

niques, but the latter is more robust to under-segmented data at inference.

2 Evaluation

Our research questions address two scenarios. In the first, a researcher has downloaded a shared model and wishes to use it to translate new data. In many instances—perhaps most—the providers of the model have neither shared nor reported what segmenter they used. Likely the model was trained on "standard" provided datasets such as those from WMT. We would like to have some understanding of the effect of different segmentations when we don't have control over the training segmentation.

Alternatively, we have a "glass box" model. In this setting, we are training the model, and have full flexibility over the choice of segmentation. By reconstructing the entire NMT pipeline with segmentation as the first step of dataset preprocessing, the researcher has complete control over the resulting model. This settings provides us with a more granular look at the effect of segmenter choice.

2.1 Metric Settings

In order to evaluate in either of these settings, we need to address a difficulty: automatic metrics for machine translation, whether source-based or reference-based, compare the machine translation output for sentences on a *pre-segmented*. For example, the WMT20 `en-de` test set (Barrault et al., 2020) has 1,418 pre-segmented sentences,[2]. In order to evaluate the effect of segmenters, we need to run three steps:

1. Remove the provided segmentation
2. Re-segment and translate
3. Align the translation outputs to the original references

This alignment step is necessary because metric scores cannot be compared across different reference segmentations. And it is complicated because we have no guarantee that the new segmentation will line up cleanly with the existing one.

We address this problem with three different alignment approaches.

Preserve keeps the provided segmentation, skipping step (1) above. Segmenters are applied to each sentence separately. It is easy to restore the original segmentation by simply keeping track of the number of sub-splits that were created with each line. On the downside, it does not allow the segmenter its full flexibility.

Document is possible when the sentences of a test set are grouped into documents. In this setting, step (1) above is done, but only at the document level. The segmenter is applied to the sentences in each document. Step (3) is undertaken by treating each document as a single line. Because of this, the number of references changes, and numbers computed from this approach cannot be compared to the other two.

Realign provides full flexibility to each segmenter. For step (3), we concatenate all outputs, and then align its words to the original reference segmentation using a search algorithm described in Section 2.2.

As many of the test sets originate from news articles and include header information (which typically includes a designated line break), we additionally insert sentence-final punctuation where it is not provided. This allows all segmenters to recover this segmentation.

2.2 Aligning outputs to references

Assume a source sequence (S) comprising tokens $(s_1, s_2, s_3, ...s_n)$, which aligns to reference r_i and a subsequent source sequence (T) comprised of $(t_1, t_2, t_3, ...t_m)$, which aligns to reference r_j. In the released test set, there exists an explicit segmentation between s_n and t_1. If we maintain this segmentation, the realignment of the translated tokens is obvious: any sub-sequence spawned from S aligns to r_i and we can re-concatenate the translations for scoring.

However, in production, there isn't an explicit segmentation between these tokens. Therefore, to give the segmenters the full degree-of-freedom that one would find in production, we must remove these segmentations. In this scenario, a segmenter may create the subsequences of (s_1, s_2, s_3), $(s_4, ...s_n, t_1, t_2)$, and $(t_3, t_4, ...t_m)$. Translation can also re-order tokens which makes the realignment non-obvious.

The realignment can be reduced to a search problem. To limit the search, we can impose hard constraints on the alignment based on subsequence matching. The field of Biomedical Engineering has a similar problem when trying to align two similar (but not identical) DNA sequences. We

[2] In `en-de` the "sentences" are typically several sentences to promote document translation

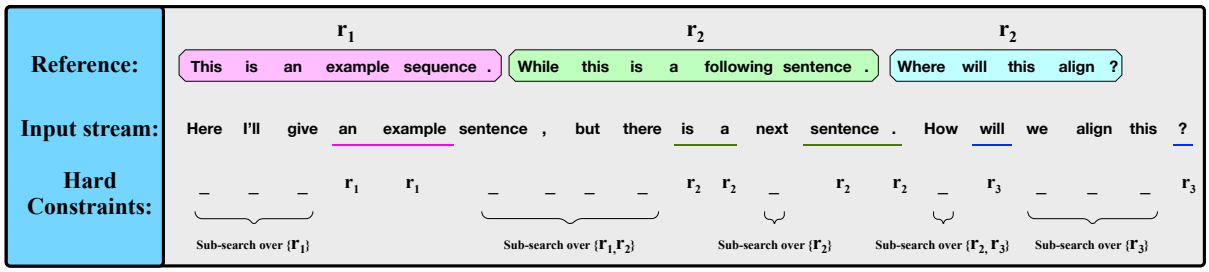

Figure 1: Example of the realignment method. Top row indicates the reference with grouped tokens each belonging to r_1, r_2, and r_3. The hard constraints are determined via longest subsequence matching (indicated with underlines). Note that not all matching surface forms may be determined as hard constraints based on token ordering. These hard constraints fix certain alignment points so the search algorithm (described in Section 2.2) has a limited reference set.

use an off-the-shelf capability[3] which maximizes subsequence matching length. This aligns some tokens to references so we search *between* the already aligned tokens. This is further illustrated in Figure 1.

Between a start and end token (t_i and t_j respectively) that are aligned to two references (r_x and r_y), we search for the best alignment of all intermediate tokens (t_k such that $i < k < j$) to a reference (r_z such that $x < z < y$). Plainly, this maintains a monotonicity: subsequent tokens can only be aligned to the same or a future reference.

We additionally require alignments to be consecutive sequences–no produced alignment to a reference can be a subsequence of an alignment to a different reference.

We optimize the alignment via the following costs:

- **Length-Ratio:** An optimal alignment should be the same length as the reference. This feature is the ratio of the shorter sequence to the longer sequence.
- **Final Punctuation:** A binary feature that determines if the aligned sentence and the original reference both end in punctuation.
- **N-gram Probability:** For unigrams and bigrams, the $p(t_k|r_z)$ or $p(t_{k-1}, t_k|r_z)$, respectively.
- **Start Word:** A binary feature that determines if the aligned sentence and the original reference both start with the same word.
- **End Word:** A binary feature that determines if the aligned sentence and the original reference both end with the same word.
- **Initial Capitalization:** A binary feature that determines if the aligned sentence and the original reference both start with a capitalized word.

We let the alignment cost be:

$$a_k = \sum_{i=0}^{k} a_i + \sum_j w_j * f_j(\text{alignment of } t_k)$$

where w_j is an associated weight on feature f_j. We perform a beam search with these features, expanding with each token t_k.

We use this methodology to re-align translations to references when the original segmentations are not maintained. We also note that this technique and toolkit can be used to reproduce alignments in other fields when the model's segmentation is not identical to that of the test sets as one might see in speech translation.

3 Experimental Setup

We focus on our investigation on English and German. We make this choice because this language pair has sufficient document-level information in datasets released by WMT. For many language pairs, datasets with true document pairs do not exist. A wider consideration of language pairs is not possible without further work to cultivate document-pair datasets.

Given document pairs in German and English, we extract sentences with a variety of segmenters and apply a typical document-based aligner to create bitext. Each segmenter creates a unique training set which we use to train a neural machine translation model.

Traditional alignment methodologies assume true document pairs. A search through both the source and target assumes alignments will be found

[3] https://biopython.org

in roughly sentence order. Vecalign (Thompson and Koehn, 2019) is an example of one of these document-based aligners. This method has the benefit of being capable of recovering erroneous segmentation because over-segmented sequences will still be consecutive during the search.

The growing field of bitext alignment has created new trends that search for sentence pairs outside the context of a document. One method of extracting sentences from all documents and searching globally for a sentence pair has created massive datasets such as CCMatrix (Schwenk et al., 2021b), WikiMatrix (Schwenk et al., 2021a), and CCAligned (El-Kishky et al., 2020). To compare the effects of segmentation in conjunction with both alignment techniques, we train models on data produced from all segmenters using both a document-based alignment method and a global, context-less based aligner.

3.1 Data

German–English has three datasets that preserve document-level boundaries in German-English—Europarl v10.[4] News Commentary v16.[5] and DGT[6] available through OPUS (Tiedemann, 2012). We find these datasets sufficient to train NMT models without other supplementary data.

Europarl comes from proceedings of the European Parliament. Aligned sentences are released as well as document IDs. The aligned sentences are roughly sentence-level. News Commentary is similarly produced from news articles.

DGT is a set of manually produced translations released by the European Commission's Directorate-General for Translation (DGT) from their translation memory. This dataset has substantial over-segmentation where one clause or phrase may be segmented onto its own line.

We use the Workshop on Machine Translation 2020 (WMT20) news task test sets and sacreBLEU[7] (Post, 2018) to score.

The sizes of the data before and after segmentation are available in Table 6 in the Appendix.

3.2 Segmentation models

We compare the following segmenters:

- **ORIGINAL:** The provided segmentations.
- **ALWAYS:** An over-segmentation approach that treats every piece of potentially sentence-ending punctuation as unambiguous.
- **ERSATZ:** A neural model that uses context windows to produce segmentations (Wicks and Post, 2021).
- **MOSES:** Always splits on punctuation, unless the previous token is in a pre-defined list of acronyms and abbreviations (Koehn et al., 2007).
- **PUNKT:** An unsupervised approach that uses thresholding to produce segmentations based on features such as casing, token length, and word frequency (Kiss and Strunk, 2006).
- **SPACY:** A "Rule-based" technique that varies on language.[8]
- **PAIRS:** The DGT dataset is oversegmented, and many lines contain less than one whole sentence. Lines must be merged in order to have complete sentences. In this setting, we merge the original bitext (instead of inserting segmentations). To implement, we simply combine every two lines together and treat as one "sentence." This merging also adds many-to-many sentence alignments for training in the Europarl and News Commentary datasets. *We only consider this "segmentation" method at training as the test data is sufficiently undersegmented.*

4 Segmentation at Inference with a Black Box System

In order to replicate a real-world use-case, we use an off-the-shelf pre-trained model. Datasets used to train these models are reported, but for the most-part, segmentation is unknown. We consider test sets in a variety of language pairs[9] for comprehensiveness. For model consistency, we chose a multilingual NMT model. We use the Prism (Thompson and Post, 2020) as a blackbox translation model,

After applying the segmentation methods described in Section 2, we translate with Prism.

[4] https://www.statmt.org/europarl/v10/training/

[5] https://data.statmt.org/news-commentary/v16/training/

[6] https://opus.nlpl.eu/DGT.php

[7] We considered COMET (Rei et al., 2020) as an alternative but did not find significant differences in trends

[8] v2.3.5, https://spacy.io

[9] cs-en, de-en, en-cs, en-de, en-pl, en-ru, en-zh, pl-en, ru-en, zh-en

	PRESERVE PRISM	REALIGN PRISM	DOCUMENT PRISM
ORIGINAL	27.1	27.5	28.9
ALWAYS	28.7	29.1	30.5
ERSATZ	29.0	29.4	30.8
MOSES	29.0	29.5	30.8
PUNKT	29.0	29.4	30.8
SPACY	28.7	29.3	30.6

Table 1: PRESERVE maintains original segmentations before applying the segmenter and aligns all produced sentences to the original corresponding reference. DOCUMENT removes segmentations from the original source before applying segmenter and aligns all translations to a single reference sentence (the entire document). REALIGN removes original segmentations and applies the alignment technique in Section 2.2 before scoring. Note that columns are not directly comparable. Differences between segmenters are not statistically significant.

We report results in Table 1 by averaging BLEU scores across the languages. As shown in the table, different segmenters are consistent within each alignment technique. The original test sets from some language pairs (namely `cs-en`, `en-cs`, `de-en`, and `en-de`) were undersegmented in the release to encourage document-level MT. For this reason, the average with the ORIGINAL segmentation is lower—the Prism model trained primarily on sentence-level data does not generalize as well to multiple sentence inputs. PRESERVE and REALIGN have the same number of references while DOCUMENT doesn't. PRESERVE and REALIGN are more directly comparable but REALIGN still may introduce realignment errors. DOCUMENT is used as an additional score to contextualize the performance. More about the realignment methodologies is in Section 2.

5 Segmentation at Training with a Glass Box System

Segmentation occurs at an early stage in the NMT pipeline, so it is intuitive to think it could have a large effect: Incorrect segmentations can lead to incorrect alignments; incorrect alignments lowers the quality of the training data; and low quality training data will produce worse models.

In order to study the effects on training, we recreate the NMT pipeline by segmenting documents and aligning bitext to train models. We apply each segmenter to the training data resulting in a new unique set of "sentences" for each segmenter. We can then align these sentences to create a unique dataset.

5.1 Document-based Alignment

The standard training paradigm for machine translation identifies bilingual document pairs, segments the sentences on both sides, and then aligns. The alignments are ideally one-to-one, but often many-to-one (or one-to-many) alignments are also permitted. The product of this is (ideally) tens of millions of sentence pairs that can be used to train machine translation models.

To replicate this, we take monolingual datasets that we know to contain parallel documents. The document alignment is known and labelled. Using these document alignments, we segment and manually re-align the sentences using a document-based aligner.

The document-based aligner we use is Vecalign (Thompson and Koehn, 2019) which uses LASER[10] (Artetxe and Schwenk, 2019) sentence embeddings to compute alignment and also considers many-to-one or one-to-many alignments. This system is a document aligner because it aligns within document context–considering surrounding sentences for many-to-one (or one-to-many) alignments and also constrains the search to alignments along the diagonal (i.e., sentences aligned to each other should occur within a similar placement within their documents).

The number of sentences recovered from the alignment, as well as the average length of source and target in the resulting dataset is shown in Table 2. The difference in size of the resulting datasets is important to note and likely explains the differences in models.

5.2 Global Search Alignment

Recent releases of WikiMatrix (Schwenk et al., 2021a), CCMatrix (Schwenk et al., 2021b), and CCAligned[11] (El-Kishky et al., 2020) have increased the scaling of bitext mining by doing away with the need for bilingual documents.

These techniques extract sentences from monolingual data and attempt to align them with techniques by combining sentence embeddings and clever search algorithms. In such settings, it stands to reason that proper segmentation might be even

[10] https://github.com/facebookresearch/LASER

[11] CCAligned somewhat limits the globalness of the search by aligning pseudo documents based on domains.

	Unaligned	Aligned Bitext					
		Document-Based			Global Search		
	total	sents	toks	avg.	sents	toks	avg.
ORIGINAL	7.5M 8.3M	7.3M	183.8M 182.6M	25.0 24.9	5.3M	155.1M 150.9M	29.4 28.6
ALWAYS	6.5M 5.9M	5.2M	177.1M 188.5M	33.8 35.9	4.1M	144.6M 150.1M	35.5 36.8
ERSATZ	4.9M 5.1M	4.5M	181.2M 184.5M	40.4 41.1	3.9M	155.3M 154.8M	39.8 39.7
MOSES	4.7M 5.2M	4.3M	182.2M 181.8M	41.9 41.8	3.7M	155.9M 153.1M	41.8 41.1
PUNKT	5.0M 5.3M	4.5M	181.7M 183.4M	40.1 40.5	3.9M	157.0M 155.5M	39.9 39.5
SPACY	5.8M 6.5M	5.4M	183.1M 182.3M	34.0 33.9	4.5M	154.7M 150.7M	34.0 33.1
PAIRS	3.7M 4.2M	3.6M	183.8M 185.2M	51.5 51.9	3.0M	160.1M 156.7M	53.2 52.1

Table 2: Training data sizes before (left, Unaligned) and after (right, Aligned Bitext) segmentation and alignment. For each row, the top number denotes the source (de) size while the bottom denotes the target (en) size. For each segmentation method, displays the total number of retrieved sentence pairs, the total number of tokens (based on white-space), and the average number of tokens in a sentence.

more important, since all alignments are one-to-one. The scaling potential of this technique allows for massive datasets with billions of aligned sentences to produced in many languages. We use the same toolkit used to produce these datasets which uses LASER embeddings and FAISS indexing for quick retrieval.[12]

5.3 Experimental Details

We train a 32,000 joint unigram subword vocabulary using SentencePiece[13] (Kudo, 2018; Kudo and Richardson, 2018) using the original data. We use a Transformer (Vaswani et al., 2017) architecture with 6 encoder and 6 decoder layers. We train with a batch size of 16k tokens validating at the end of each epoch and stopping if the validation has not improved after 10 validations. We validate on WMT19 test sets (with original segmentations). For a comprehensive list of hyperparameters, please Table 7 in the Appendix.

5.4 Results

The amount of data produced by each segmenter and alignment method varied significantly. Data quantity after segmentation and alignment is displayed in Table 2. Vecalign is fairly consistent in the amount of data aligned—roughly 180M tokens with ALWAYS creating the highest variance. Vecalign also produces more data in terms of number of sentences compared to the alternative global search method. The global search also varies more significantly with a 10M token difference between the smallest and largest datasets (excluding ALWAYS).[14]

We compute the full cross-product of segmentations at training and inference. Results are reported in Table 3. Once again, we find that within a given model, performance is relatively consistent at inference regardless of segmentation. The exception is the ORIGINAL row as these inputs are under-segmented. This strong mismatch between training and testing points to hallucinations which are further explored in Section 6.

Generally, we see more variation in model performance based on training data segmentation rather than inference segmentation. One of the best performing models was the model trained on the ORIGINAL data—made by preserving the original segmentations. The prominent feature of its training data was the prevalence of sub-sentence segmentations. We hypothesize this helped in two ways: 1) it was not reliant on a strong end-of-sentence signal (§ 6) and 2) the true alignments were more likely to exist in the training set. If errors in segmentation make alignment difficult, it is beneficial to have segments that are *guaranteed* to correctly align to something. Because the DGT dataset was transla-

[12] https://github.com/facebookresearch/LASER/blob/main/source/mine_bitexts.py

[13] https://github.com/google/sentencepiece

[14] The default `mine_bitexts.py` setting was used for LASER. The parameters for Vecalign are listed in Table 8 in the Appendix.

	ORIGINAL		ALWAYS		ERSATZ		MOSES		PUNKT		SPACY		PAIRS	
	Vec.	Global	Vec.	Global	Vec.	Global	Vec.	Global	Vec.	Global	Vec.	Global	Vec.	Global
ORIGINAL	25.6	24.9	24.7	16.4	25.0	9.4	22.1	9.8	25.0	11.6	23.8	8.7	28.2	**28.9**
ALWAYS	31.2	**31.7**	30.8	**31.2**	30.4	**31.3**	30.8	30.9	31.3	31.0	30.2	29.8	30.6	**31.4**
ERSATZ	31.3	**31.9**	30.9	31.3	30.6	**31.4**	31.0	31.1	31.3	31.2	30.4	29.9	30.7	**31.5**
MOSES	31.4	**31.9**	30.9	31.3	30.6	31.4	31.0	**31.1**	31.3	31.2	30.4	29.9	30.8	31.6
PUNKT	31.3	**31.9**	30.9	31.3	30.6	**31.5**	31.0	31.1	31.3	31.2	30.4	29.9	30.7	**31.5**
SPACY	**31.4**	**31.8**	30.9	**31.3**	30.7	**31.4**	30.9	31.1	31.4	31.3	30.8	31.2	30.5	**31.6**

Table 3: German–English (`de-en`) results. The rows denote the segmenter used at *inference* while the columns denote the segmenter used to create the *training data*. The diagonal, thus, has a matching segmenter for both training and inference. The LASER global search alignment method was used to create bitext. Bold denotes significance ($p < 0.05$) run by paired bootstrapping with sacreBLEU.

tion memory, most segments had a true alignment.

In the ORIGINAL inference-time setting, models trained with Vecalign-produced bitext performed better than their Global counterparts. We hypothesize this is because Vecalign was able to recover many-to-one or one-to-many alignments where the Global aligner was not. This made the models more robust to many-sentence inputs and outputs.

Lastly, we note that the choice of segmenter *does* affect the training data, and thus the final trained model. The ORIGINAL model often had the highest BLEU score across inference-time segmentations. The differences between the ORIGINAL model, and the ERSATZ and PAIRS models were not statistically significant in most cases. Models trained on data created by PUNKT, SPACY, or MOSES (often used to create MT datasets) were not as competitive.

6 Qualitative Analysis

Hallucinations, or addition of content during translation, and deletions are common in neural machine translation. These models are no exception. Qualitative analysis reveals two types of errors that are worth investigating further: 1) seemingly arbitrary deletion of content when the input is unsegmented 2) addition of content without a true signal in the source. We suspect the explanations for these behaviors are 1) a lack of many-sentence inputs occuring in training data and 2) incorrect segmentations leading to poorly aligned data. We display some examples in Table 4.

6.1 Deletion

Almost all models fail when given unsegmented data at inference (the ORIGINAL row). Upon inspection of these translations, it is obvious the reasons for these scores. In Table 4, we show an instance of this in the first column. The source input has three sentences. Some models trained with segmenters (ERSATZ, MOSES, PUNKT, and SPACY) drop the majority of these sentences. The models trained with the ORIGINAL segmentations and the ALWAYS segmentation method incorporate information across sentences and hallucinate new conjunction methods (inserting “with" or using commas). The model trained with the PAIRS setting does a combination. As this setting often has two sentences per line in the bitext, this translation also is limited to two lines and similar, to PUNKT and ORIGINAL, hallucinates ways to combine these sentences. We can infer the reason for the drop in BLEU scores in the unsegmented setting is because most models are *deleting* content. In order to report the prevalence of deletion, we report how many sentences were deleted during translation.

There are 785 lines in the test data but most of the lines contain more than one sentence. We can use Moses (one of the segmenters that is quite conservative—prone to undersegmenting) to count how many sentences occur in the source input as well as how many sentences occur in the translated outputs. In Table 5, we display this information.

The fact that PAIRS translates more sentences is logical as its training data often had *pairs* of sentences in the training data. The ORIGINAL setting translating more sentences than other models seems counterintuitive as the training data was shorter on average (see Table 2). We suspect that the reason for this is more related to the fact that many training examples in the ORIGINAL setting *did not end in punctuation* since they were below the sentence level. In the ERSATZ training data, for example, 98% of training example’s target sequences end with a period. Conversely, 62% of the ORIGINAL data ended with a period. We reason that the model was not highly likely to end the sequence after de-

	DELETION	ADDITION
SOURCE	Für Online-Händler sind viele zurückgeschickte Pakete verlorene Ware. Rund 20 Millionen Retouren landen so auf den Müll. Doch gibt es eine Alternative.	Der Premier droht damit, das Land am 31. [Oktober ohne Abkommen aus der EU zu führen...]
ORIGINAL	For online traders, many returned packages are lost commodities, with around 20 million retours pouring into the rubbish, but there is an alternative.	The Prime Minister is threatening the country on 31 December.
ALWAYS	For online traders, many returned packages are lost, but there is an alternative.	The Prime Minister is threatening to leave the country on 31 December.
ERSATZ	Some 20 million retours thus end up in the rubbish.	The Prime Minister is threatening to hold the country on 31 May.
MOSES	For online traders, many returned packages are lost goods.	The Prime Minister is threatening to do so, the country on 31 December.
PUNKT	For online dealers, many returned packages are lost goods.	The Prime Minister is threatening to see the country on the 31st day of the month.
SPACY	For online traders, many returned packages are lost goods.	The Prime Minister is threatening the country on 31 December.
PAIRS	For online traders, many returned packages are lost goods, with some 20 million retours ending up in rubbish. But there is an alternative.	The Prime Minister is threatening the country on 31 December.

Table 4: Examples of differences in translations. The SOURCE denotes input to the model. Content in square brackets was not part of input but has been included for context to reader. The DELETION column shows examples of different models deleting content during translation due to unsegmented input. The ADDITION column shows models hallucinating content when an incomplete input was given.

MODEL	SENTENCES
REFERENCE	1959
ORIGINAL	1009
ALWAYS	785
ERSATZ	793
MOSES	790
PUNKT	830
SPACY	790
PAIRS	1399

Table 5: Number of sentences (as counted by Moses segmenter) generated on the WMT20 de-en test set. The model is each translation system trained on data segmented by the specified segmenter.

coding a period due to these trends.

All segmenter models were affected by this behavior, but SPACY had more issues. SPACY, in a manner different to the other segmenters, also includes punctuation such as ':' as final punctuation meaning it oversegments in many scenarios. In sentences including colons, we see similar deletion from SPACY. For instance, the SPACY model translates ***"Katastrophe abgewendet: Großbrand in französischem Chemiewerk gelöscht."*** simply as ***"Avoiding disaster:"***

Of the 157 sentences in the test data that included a colon, SPACY translated *only* the first segment in 78 of them; in 52 it translated only the second segment; in 27 it translated parts of both.

6.2 Additions

When Raunak et al. (2021) studied the causes of hallucinations, they attributed hallucinations to errors in bitext alignment. It follows that segmentation, as a precursor to bitext alignment, might also affect hallucinations. The most obvious hallucination we see in the translations is surrounding dates. German often uses a format of "Freitag, 27. September 2019" for "Friday, September 27, 2019". Erroneous segmentation around the punctuation in the date causes alignment issues or bad input to the translation model. We see the effects of both.

The former, bad alignment, we see in the case of the overly-aggressive segmenter (ALWAYS). Because the data was always split on the date in this construction, the alignment suffers severely. We see examples in the training data such as:

Source: Juli 2016 an.
Target: Done at Brussels, 20 July 2016.

The training data frequently contains dates like this and the global search aligner was unable to detect that additional information appeared on the target side. The ALWAYS model memorized this

extraneous information and generated it 8 times in these experiments.

In the second case, where the input to the model has been over-segmented, we see a similar effect. When an ALWAYS segmenter is used at inference, the models struggle on the incomplete information. In the second column of Table 4, a complete sentence has been segmented erroneously into two incomplete sentences. For clarity the end of the sentence (which was segmented into a separate input) is included in square brackets. The incomplete input has a two-fold effect: 1) the models hallucinate months to attach to the date 2) ALWAYS, ERSATZ, and PUNKT hallucinate verbs (to leave, to hold, to see respectively).

7 Related Works

To the best of our knowledge, not much work has been done about the effects of segmentation on down stream tasks. Raunak et al. (2021) investigates corpus-level noise and empirically links noise patterns to types of NMT hallucinations. Other work has focused on the effects that punctuation has on neural language models (Ek et al., 2020; Karami et al., 2021). In online simultaneous speech segmentation, Wang et al. (2019) proposes an online sentence segmentation approach which improves downstream BLEU scores.

There is much more work pushing away from the sentence-level paradigm and encouraging document translation. Sun et al. (2022) has recently shown that modern neural architectures still achieve strong performance with longer, multi-sentence inputs. A spate of recent work has gone into better document evaluation metrics (Jiang et al., 2022; Vernikos et al., 2022). Document pairs contain the additional context needed to correctly translate certain discourse phenomenon such as coreference resolution and consistent lexical choices. Further, mining documents instead of sentences circumvents the error propagation from using various segmentation methodologies during bitext mining.

8 Conclusion

An NMT system trained on segmented data requires segmentation at inference; however, the exact method of segmentation at inference seems to have little quantitative effect. The larger impact of segmentation occurs during the creation of bitext. Whether the effect stems from the quality of the produced sentence pairs or the limitations of different alignment methods cannot be determined based on these results. Despite this, various segmentation and alignment method combinations create significantly different amounts of bitext to train models on–something that needs to be investigated further. The differences in the resulting data produce models that perform differently. Lastly, we note that when models are trained on segmented data, they dramatically hallucinate at inference with unsegmented data by deleting long segments. By adding some amount of unsegmented data in the training data, this effect can be mitigated to recover upwards of 4 BLEU points.

Together, we might conclude that avoiding segmentation is the path forward. When the segmentation and alignment techniques failed, half a million sentence pairs were sometimes lost or left unaligned. Additionally, we see that less-segmented bitext produces models that are more robust to unsegmented data at inference. The biggest hurdle in training document-level models is the lack of sufficient document-level annotations. If true document pairs exist in larger web-scraped corpora, most of the original document structure (and informative context) has been removed via bitext filtering and deduplication. Future work might explore potential solutions to mining document-level data, and circumvent these segmentation tools and their respective noise.

9 Limitations

Most of this work relies on interactions between the segmenters and the aligners. It's the production of training data—and the resulting quality and quantity that is causing the differences in models. We used off-the-shelf configurations for these aligners and didn't do significant hyper-parameter searching. It's possible that other toolkits or different hyperparameters might normalize the effects of erroneous segmentation.

We also noted that Vecalign was able to recover erroneous segmentation in the one-to-many and many-to-one settings while showing that the global method was not. Having a global search method does not directly preclude these recoveries, but to the best of our knowledge it hasn't been investigated.

Lastly, we limit ourselves to `de-en` as a language pair here because of the availability of document pairs. The ambiguity in puncutation surround-

ing these two languages make them interesting for segmentation. Also, German often uses a different word order than English which can make aligning erroneous segmentations difficult. These effects might be minimized or non-existant in other language pairs.

References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Adam Ek, Jean-Philippe Bernardy, and Stergios Chatzikyriakidis. 2020. How does punctuation affect neural models in natural language inference. In *Proceedings of the Probability and Meaning Conference (PaM 2020)*, pages 109–116, Gothenburg. Association for Computational Linguistics.

Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzmán, and Philipp Koehn. 2020. CCAligned: A massive collection of cross-lingual web-document pairs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5960–5969, Online. Association for Computational Linguistics.

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

Yuchen Jiang, Tianyu Liu, Shuming Ma, Dongdong Zhang, Jian Yang, Haoyang Huang, Rico Sennrich, Ryan Cotterell, Mrinmaya Sachan, and Ming Zhou. 2022. BlonDe: An automatic evaluation metric for document-level machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1550–1565, Seattle, United States. Association for Computational Linguistics.

Mansooreh Karami, Ahmadreza Mosallanezhad, Michelle V. Mancenido, and Huan Liu. 2021. "let's eat grandma": When punctuation matters in sentence representation for sentiment analysis. *CoRR*, abs/2101.03029.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Robert C. Moore. 2021. Indirectly supervised english sentence break prediction using paragraph break probability estimates. *CoRR*, abs/2109.12023.

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.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021a. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021b. CCMatrix: Mining billions of high-quality parallel sentences on the web. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6490–6500, Online. Association for Computational Linguistics.

Zewei Sun, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Shujian Huang, Jiajun Chen, and Lei Li. 2022. Rethinking document-level neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3537–3548, Dublin, Ireland. Association for Computational Linguistics.

Brian Thompson and Philipp Koehn. 2019. Vecalign: Improved sentence alignment in linear time and space. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1342–1348, Hong Kong, China. Association for Computational Linguistics.

Brian Thompson and Matt Post. 2020. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 90–121, Online. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Giorgos Vernikos, Brian Thompson, Prashant Mathur, and Marcello Federico. 2022. Embarrassingly easy document-level MT metrics: How to convert any pretrained metric into a document-level metric.

Xiaolin Wang, Masao Utiyama, and Eiichiro Sumita. 2019. Online sentence segmentation for simultaneous interpretation using multi-shifted recurrent neural network. In *Proceedings of Machine Translation Summit XVII: Research Track*, pages 1–11, Dublin, Ireland. European Association for Machine Translation.

Rachel Wicks and Matt Post. 2021. A unified approach to sentence segmentation of punctuated text in many languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3995–4007, Online. Association for Computational Linguistics.

A Appendix

A.1 Data

In Table 6 is a further breakdown if the amount of sentences extracted from the datasets via each segmenter. In most cases, a segmenter produces more segmentations than the ORIGINAL dataset. This is not true of the DGT dataset which shows how over-segmented it was. Moses is the most conservative segmenter with high-precision and lower recall.

A.2 Hyperparameters

In Table 7, the hyperparameters used to train the Fairseq NMT models are listed. When the parameters are not listed, the defaults were used. Further, we also list the settings used with the Vecalign aligner in Table 8.

	DGT		EUROPARL		NEWS			WMT20
	de	en	de	en	de	en	totals	all
ORIGINAL	5.24M	6.12M	1.83M	1.83M	0.39M	0.39M	15.80M	12517
ALWAYS	4.06M	3.66M	2.06M	1.89M	0.41M	0.39M	12.47M	17434
ERSATZ	2.62M	2.88M	1.93M	1.83M	0.40M	0.39M	10.04M	16132
MOSES	2.37M	3.05M	1.90M	1.79M	0.39M	0.39M	9.89M	15915
PUNKT	2.63M	3.01M	1.97M	1.86M	0.40M	0.38M	10.25M	16137
SPACY	3.33M	4.15M	2.06M	1.97M	0.43M	0.42M	12.35M	17342
PAIRS	2.63M	3.07M	0.92M	0.92M	0.20M	0.20M	7.94M	-

Table 6: Sizes of the source (de) and target (en) after applying segmentation techniques described in Section 2. These sizes are before alignment. To the right (WMT20), we list the sizes of the segmented test sets (all 12 languages together).

Parameter	Value
Architecture	Transformer
Encoder Layers	6
Decoder Layers	6
Embed Dim	512
FFN Dim	512
Attention Heads	8
Dropout	0.1
Attn. Dropout	0.1
ReLU Dropout	0.1
Label Smoothing	0.1
Adam Betas	(0.9, 0.98)
Clip Norm	2.0
Lr Scheduler	Inverse Sqrt
Warmup Updates	4000
Initial LR	1e-7
LR	0.0005
Min LR	1e-9
Batch Size	16k tok
Patience	10

Table 7: Values for the hyperparameters used during training. Can be traced to the Fairseq parameters. If not listed, default was used.

Parameter	Value
Overlap	6
Max Alignment	4
Embedding Model	LASER (93 langs)

Table 8: Settings used with the Vecalign alignment toolkit.