A Comparison of Data Filtering Methods for Neural Machine Translation

Fred Bane fbane@translations.com
Celia Soler Uguet csuguet@transperfect.com
Wiktor Stribiżew wstribizew@translations.com
Anna Zaretskaya azaretskaya@translations.com
Transperfect Translations, Barcelona, Spain

Abstract

With the increasing availability of large-scale parallel corpora derived from web crawling and bilingual text mining, data filtering is becoming an increasingly important step in neural machine translation (NMT) pipelines. This paper applies several available tools to the task of data filtration, and compares their performance in filtering out different types of noisy data. We also study the effect of filtration with each tool on model performance in the downstream task of NMT by creating a dataset containing a combination of clean and noisy data, filtering the data with each tool, and training NMT engines using the resulting filtered corpora. We evaluate the performance of each engine with a combination of MQM-based human evaluation and automated metrics. Our results show that cross-entropy filtering substantially outperforms the other tested methods for the types of noise we studied, and also leads to better NMT models. Our best results are obtained by training for a short time on all available data then filtering the corpus with cross-entropy filtering and training until convergence.

1 Introduction

Large-scale, publicly available bilingual corpora are an excellent resource for training neural machine translation (NMT) models. Performance in the NMT task improves as the size of the training data increases (Koehn and Knowles, 2017), and with datasets like CC Matrix (Schwenk et al., 2019), tens or even hundreds of millions of sentence pairs are freely available for many language pairs. However, these corpora are known to be noisy (Kreutzer et al., 2022), and NMT models are quite sensitive to noisy training data (Khayrallah and Koehn, 2018a). Thus, tools to filter noisy data are becoming an important step in NMT training pipelines.

In this paper, we compare the performance of several available tools in the task of data filtering, breaking down the results by different types of noise. We then train MT engines with different filtered versions of the same corpus to compare the effects of data filtering on the downstream task of translation.

2 Related Research

Cleaning noisy data with the purpose of using them for MT training has been a major topic in research. Since neural MT performance has shown to be highly dependent on the size of the training data (Koehn and Knowles, 2017) as well as their quality (Khayrallah and Koehn,

2018b), several large-scale initiatives for crawling and cleaning data from the web appeared, such as Paracrawl (Bañón et al., 2020) and CCMatrix (Schwenk et al., 2019).

For this reason, most works in this area focus on filtering this type of data, i.e. noisy data collected from the web. One of the earlier works proposed an unsupervised method, in particular using an outlier detection algorithm to filter a parallel corpus (Taghipour et al., 2011), which led to an increased performance of the SMT system trained on these cleaned data. Another unsupervised method consisted of a graph-based random walk algorithm and extracted phrase-pair scores to weigh the phrase translation probabilities to bias towards more trustworthy ones (Cui et al., 2013). The method is based on the observation that better sentence pairs often lead to better phrase extraction and vice versa.

Several subsequent works treated the data filtering task as a classification problem. An example of this is the method proposed in Xu and Koehn (2017), which is based on generating synthetic noisy data (inadequate and non-fluent translations) and using these data to train a classifier to identify good sentence pairs in a noisy corpus. Another classification approach was proposed within the 2020 task on parallel data filtering (Koehn et al., 2020). In this approach, the authors used an end-to-end classifier that learns to distinguish clean parallel data from misaligned sentence pairs. The system first uses a Transformer model to obtain sentence representations, followed either by a classifier (Siamese network) or additional layers that are fine-tuned (Açarçiçek et al., 2020).

Another popular approach is based on utilizing cross-entropy. In the 2018 edition of the shared task on data filtering, the winning system used neural MT models in both directions trained on clean data to score sentence pairs with dual cross-entropy (Junczys-Dowmunt, 2018). The divergent cross-entropies are penalized and the penalty is weighed by the average cross-entropy of the two NMT models. Another winning system in the 2020 shared task enhanced this approach by combining a dual cross-entropy from two NMT models with a number of other features: a bilingual GPT-2 model trained on source-target language pairs as well as a monolingual GPT-2 model for each of the languages, and statistical word translation model scores (Lu et al., 2020).

Recently, there has been a new direction in parallel data filtering research consisting of using multilingual language models, which create sentence representations in a multilingual vector space. Then, two parallel sentences are identified by taking the nearest neighbor of each source sentence in the target side according to cosine similarity, and filtering those below a fixed threshold (Schwenk, 2018). Another work improves on these results suggesting an alternative scoring method that uses the margin between the similarity of a given candidate and that of its k nearest neighbors (Artetxe and Schwenk, 2019).

As demonstrated in a recent work (Herold et al., 2021), the performance of a given parallel data cleaning method can vary significantly depending on the data conditions and the task definitions. In one attempt to clean mostly well-aligned bilingual data (Carpuat et al., 2017), the authors investigate the problem of filtering out semantically divergent sentences from a parallel corpus. Some sentence pairs considered "parallel" present source and target sentence that do not convey exactly the same meaning, which is quite a common phenomenon in curated parallel corpora originating from translation memories. In our experiment, we use several multilingual language models, a method based on cross-entropy and a pre-trained model for MT evaluation with the goal of identifying the methods that can be most successfully applied to our use case of filtering corpora to train MT systems.

3 Materials and Methods

For this study, we selected two language pairs: German>English (abbreviated below as 'DE>EN') and Japanese>English (abbreviated below as 'JA>EN'). These language pairs were

chosen with consideration to their linguistic properties (diverse source languages with different scripts, differing levels of linguistic distance from English, and quite different linguistic characteristics), the demand for these language pairs in translation, and the availability of data and tools for the experiment.

3.1 Part I

In Part I of the study we created datasets for each language to be used in the experiments. We randomly sampled 5,000,000 sentence pairs for each language pair from the CC Matrix data set. Then, we synthesized 1,000,000 segments representing ten different types of noise and injected them into the CC Matrix data. We scored these 6 million sentences with each tool and retained the top 50% of sentences for each tool to be used as the training set for an NMT engine. We then trained engines with each data set and compared their performance after ten training epochs on a common test set sampled from the same distribution as the training data. We used the same arbitrary threshold for each tool and each language to minimize experimental complexity. The 50% threshold was chosen to account for the noise we introduced as well as the fact that we expect CC Matrix to contain significant amounts of native noise. Using the mean scores from the validation and test sets as the cutoff values was also considered, but the number of included segments was quite similar to using a fixed threshold, so we chose the simpler of the two options.

3.1.1 Collection and Synthesis of Noisy Data

With reference to Khayrallah and Koehn (2018a), we introduced 100,000 segments for each of the following types of noise:

1. **Word order permutations in target**: we introduced errors in an iterative way (i.e., starting from one error in the first 20,000 segments and adding one additional error every 20,000 segments until obtaining 100,000 segments);
2. **Spelling permutations in target**: in the same way as above, we added a number of spelling permutations which increased every 20,000 segments until we arrived at 100,000 segments;
3. **Untranslated segments**: to simulate untranslated segments, we copied the source segment and used it as the target;
4. **Third language in source**: we chose segments for each language pair that contained a different source language than German and Japanese. We tried to choose one language that was relatively close to the original and one that was linguistically distant from the original. For DE>EN we chose 50,000 segments with Dutch as source and 50,000 segments with Russian as source. In the case of JA>EN, we selected 50,000 segments with Chinese as source and 50,000 segments with German as source. In each case, the English target was a correct translation of the source;
5. **Third language in target**: in this case, we followed the same approach as the previous type of noise, but replacing the target instead of the source. In the case of DE>EN, we chose 50,000 segments with Dutch as a target language and 50,000 segments with Russian as target. For JA>EN, we chose 50,000 segments with Chinese as target and 50,000 segments with German as target.
6. **Missing content in source**: we deleted between 5%-50% of the words in source. The number of words deleted grew by 5% increments every 10,000 segments until we reached 100,000 segments. To create this type of noise, we used only sentences with more than 20 words in the source. We used Fugashi (McCann, 2020) to perform word segmentation in Japanese;

7. **Missing content in target**: we followed the same approach as in the previous type of noise, but this time we applied it to target segment;
8. **Mismatching numbers**: we searched for matching numbers in the source and target and increased the first number by a random integer between 1 and 1000. We changed numbers in 50,000 source segments and in 50,000 target segments, but we make no distinction in our analysis based on which number was modified;
9. **Complete misalignment**: we took a properly aligned corpus and intentionally moved several of the target segments from the head of the corpus to the end. In this way, we ended up with a misaligned corpus and sampled 100,000 random segments from it;
10. **Unbalanced [sic] tags**: This type of noise is possibly unique to our use case as a commercial translation provider with human translated data. But we find that unbalanced [sic] tags (i.e. which appear in only one of the source or target but not both) can introduce a systemic bias to the corpus and can cause hallucinations in an MT system if they are not removed prior to training. To create this type of segment, we searched for pairs of sentences that contained [sic] tags in the target but not in the source, but given that the CC Matrix corpus did not contain enough of these segments, we created them by inserting a [sic] tag after a random word in a total of 100,000 segments ;

3.1.2 Data Filtering

For the next step of the process, we concatenated the clean data with the noisy data and used the following tools to score each sentence pair in the combined dataset: XLM-R (Conneau et al., 2019), MUSE (Conneau et al., 2017) and LASER (Schwenk and Douze, 2017) - create sentence representations in an aligned multilingual vector space; COMET (Rei et al., 2020) - pre-trained model for MT evaluation; Marian-scorer (Junczys-Dowmunt et al., 2018) - part of the MarianNMT toolkit, computes cross-entropy.

For XLM-R, MUSE and LASER we used the open-source models available and computed cosine similarity between the resulting embeddings. For COMET, we used the wmt-20-qe-da model for Quality Estimation and Direct Assessment. And finally, for Marian-scorer, we used our company's existing Marian models (which were not trained using CC Matrix data) for the various language directions.

Having calculated scores for each sentence with each tool, we proceeded to filter the data to create datasets for each tool, retaining the top 50% of segments as scored by that tool (i.e., 3 million segments).

3.1.3 Engine Training

After filtering, we trained the following systems:

- One system for each of the training-sets generated by each scoring method;
- One system using the unfiltered dataset containing 5,000,000 clean segments and 1,000,000 noisy segments.

The engines were trained for 35,000 training steps each, and each training was repeated three times with different random seeds to control for the effects of random weight initialization. All other training parameters were held fixed across all runs, and used the base transformer configuration with tied embeddings and a shared sub-word vocabulary of 32,000.

3.1.4 Evaluation

For the engines in Part I, performance in the machine translation task was evaluated using the automated metrics BLEU, TER, and chrF2 obtained using the Sacrebleu package (Post,

2018). Statistical significance for automated metrics was calculated using the paired bootstrap comparison. We used common validation and test sets which were partitioned prior to noise injection and scoring. We report the scores from an ensemble translation with all three models for each tool.

3.2 Part II

3.2.1 Engine Training

In Part II of the study we continued training from some of the baseline models created in Part I using different conditions. For each language pair, we continued training the best performing individual model and one model trained on the unfiltered data set. To test if there are benefits to beginning training with all available data and continuing with a cleaner dataset after a small number of training epochs, we also continued training the best performing model trained on the full dataset using the dataset filtered by the best performing tool. We were also curious to see if a model trained on such data could be used to score and filter its own training data, so we used the best performing model trained on the unfiltered dataset to score and filter its training set, retaining the top 75% of sentences, and continued training using this newly filtered dataset. The engines were allowed to train for 170,000 training steps or until early stopping criteria were met (defined as no improvement in validation perplexity for 5 consecutive checkpoints, or 15,000 training steps).

3.2.2 Evaluation

Once these were trained, sample translations for an in-domain test set and an out-of-domain test set (WMT 2020) were obtained from each model. The translations were scored using the automated metrics BLEU, TER, and chrF2 (with statistical significance determined in the same way as in Part I), and a subset of the test set translations were sent for human annotation. We used an MQM-based annotation method, which, as demonstrated by Freitag et al. (2021a), is more accurate than the previously widely used direct assessment method, and is now the standard in the WMT shared tasks (Freitag et al., 2021b). We used the error types and severity levels, as well as the weights calculation described by Freitag et al. (2021a).

Sentences were selected for human review using different criteria: the most different translations (using Levenshtein distance), the five worst COMET scores from each engine, longest sentences, shortest sentences, and translations containing different numbers of brackets or whose numbers did not match. Out of the total of 200 source sentences per language, 100 were drawn from the in-domain test set, and the remaining 100 came from the out-of-domain test set.

4 Results

Below we present the results of the two parts of our study. The results of Part I show that cross-entropy filtering is significantly better for removing the types of noise we studied. The automated metrics from engine training reinforce this conclusion. The results from Part II are less clear cut, with filtering having a comparatively stronger beneficial effect for the JA>EN direction than the DE>EN direction.

4.1 Part I

4.1.1 Data Filtering Results

With few exceptions, marian-scorer was the clear winner in filtering out noisy data, allowing an order of magnitude fewer noisy segments than the next runner-up in multiple categories. The number of corrupt sentence pairs of each type included in the datasets for each tool are shown

in Tables 1 and 2 below. A detailed breakdown of the performance of each tool on different types of noise is provided in Appendix A.

Examining the data in these tables, a few noteworthy observations present themselves:

- While third-language data is a common form of noise in parallel bilingual datasets, all of the tools we tested except marian-scorer are language-agnostic, and thus cannot be used for filtering this kind of noise;
- COMET was the only tool to fail to filter out all completely misaligned segments, but this tool excelled at filtering segments with word order or spelling permutations;
- COMET was much more sensitive to missing target content than to missing source content, while marian-scorer showed the opposite trend. In fact, the amount of missing text apparently made little difference in the scores from these tools (Figure 1). Other tools demonstrated more or less similar performance on these two types of noisy data;
- LASER and COMET did not do well in filtering out segments with mismatching numbers, while other tools generally did well.

4.1.2 First-Step Training Results

After filtration, the resulting datasets were used to train NMT engines. Each training was repeated three times with different random seeds to control for differences resulting from weight initialization. Translation of the common test set was obtained using an ensemble of the three models for each tool. After ten epochs, the models trained on data filtered by Marian performed the best for both languages, significantly outperforming the model trained with unfiltered data. Automated metrics for these translations are reported in Tables 3 and 4.

4.2 Part II

Given its superior performance in the initial training step, we selected Marian-scorer as the tool to use in the second part of the experiment. For each language pair, we trained three test models and one control model. The three test models included one trained to convergence using the dataset filtered by marian-scorer (referred to below as "Marian"), one which was trained on the unfiltered dataset for ten epochs then trained until convergence with the dataset filtered with Marian ("Marian from no filter"), and one which was trained on the unfiltered dataset for ten epochs then used to score and filter its own training data before training until convergence on

Table 1: Number of corrupt sentence pairs of each type included in each DE>EN data set.

Type of Corruption	**MUSE**	**Marian-scorer**	**XLM-R**	**LASER**	**COMET**
Word order permutations	39,369	**370**	15,876	7,072	873
Spelling permutations	9,435	**296**	5,073	8,008	1,270
Untranslated segments	100,000	**646**	100,000	99,972	86,588
Third language src	45,483	**375**	33,628	29,362	37,190
Third language tgt	29,930	**10**	55,091	52,279	58,280
Missing content src	8,102	**6,131**	13,126	12,574	33,549
Missing content tgt	9,908	11,056	10,155	**5,165**	9,569
Mismatching numbers	12,462	11,618	**4,797**	22,675	47,611
Complete misalignment	**0**	**0**	**0**	**0**	1,903
Unbalanced *sic* tags	43,009	**9,716**	48,468	20,117	31,116
TOTAL	297,968	**40,218**	286,412	257,224	307,994

Table 2: Number of corrupt sentence pairs of each type included in each JA>EN data set.

Type of Corruption	**MUSE**	**Marian-scorer**	**XLM-R**	**LASER**	**COMET**
Word order permutations	52,222	1,169	28,235	11,151	**367**
Spelling permutations	20,546	**503**	4,939	9,758	5,840
Untranslated segments	100,000	**269**	100,000	42,570	23,031
Third language src	79,446	**810**	38,550	34,708	24,898
Third language tgt	53,331	**30**	56,078	36,367	18,462
Missing content src	**24,948**	26,923	26,042	28,153	37,178
Missing content tgt	24,212	13,165	12,574	5,537	**4,837**
Mismatching numbers	32,241	20,410	**12,241**	27,737	25,532
Complete misalignment	**0**	**0**	**0**	**0**	21,914
Unbalanced *sic* tags	49,389	29,791	47,419	21,050	**13,826**
TOTAL	436,335	**93,070**	326,078	217,031	170,629

Table 3: Automated comparison of Ensemble translations for DE>EN.

System	**BLEU** ($\mu 95\% CI$)	**chrF2** ($\mu 95\% CI$)	**TER** ($\mu 95\% CI$)
No filter (Baseline)	47.6 (47.6 ± 1.3)	70.0 (70.0 ± 0.9)	36.4 (36.4 ± 1.2)
COMET	46.7 (46.7 ± 1.3)	69.1 (69.1 ± 0.9)	37.3 (37.3 ± 1.2)
LASER	48.1 (48.1 ± 1.4)	**70.4** (70.4 ± 0.9)*	**36.0** (36.0 ± 1.2)
Marian	**48.2** (48.2 ± 1.3)*	**70.4** (70.4 ± 0.9)*	**36.0** (36.1 ± 1.2)
MUSE	46.1 (46.0 ± 1.4)*	68.7 (68.7 ± 0.9)*	37.8 (37.8 ± 1.2)*
XLMR	47.6 (47.6 ± 1.4)	69.7 (69.7 ± 0.9)	36.5 (36.5 ± 1.2)

* Indicates the result is a statistically significant (p <0.05) improvement over the unfiltered baseline

Table 4: Automated comparison of Ensemble translations for JA>EN.

System	**BLEU** ($\mu 95\% CI$)	**chrF2** ($\mu 95\% CI$)	**TER** ($\mu 95\% CI$)
No filter (Baseline)	25.1 (25.1 ± 1.9)	52.8 (52.8 ± 1.1)	63.4 (63.4 ± 2.4)
COMET	30.3 (30.3 ± 1.9)*	56.1 (56.1 ± 1.3)*	55.1 (55.1 ± 1.7)*
LASER	34.1 (34.0 ± 2.0)*	59.0 (58.9 ± 1.3)*	52.2 (52.2 ± 1.7)*
Marian	**35.2** (35.1 ± 1.9)*	**59.3** (59.3 ± 1.3)*	**51.8** (51.8 ± 1.8)*
MUSE	31.5 (31.5 ± 2.1)*	56.7 (56.7 ± 1.3)*	54.9 (54.9 ± 1.7)*
XLMR	32.7 (32.7 ± 2.0)*	57.5 (57.5 ± 1.3)*	53.7 (53.7 ± 1.8)*

* Indicates the result is a statistically significant (p <0.05) improvement over the unfiltered baseline

Table 5: Automated comparison of DE>EN models on in-/out-of-domain test data.

System	BLEU	chrF2	TER
No filter (Baseline)	**51.4**/34.4	**72.2/62.9**	**33.4/52.2**
Marian	50.7/33.3	71.9/61.5	34.0/53.9
Marian from no filter	51.0/33.8	72.1/61.8	34.0/53.5
Train then filter	50.8/**34.6**	72.1/62.6	33.9/52.3

* Indicates the result is a statistically significant (p <0.05) improvement over the unfiltered baseline

Table 6: Automated comparison of JA>EN models on in-/out-of-domain test data.

System	BLEU	chrF2	TER
No filter (Baseline)	39.6/19.1	62.8/**50.6**	47.4/70.3
Marian	39.0/19.4	62.3/50.4	47.9/70.7
Marian from no filter	39.2/19.2	62.5/50.6	47.6/70.8
Train then filter	**40.5***/**19.6***	**63.5***/49.9	**46.2***/**70.1**

* Indicates the result is a statistically significant (p <0.05) improvement over the unfiltered baseline

the newly filtered data ("Train then filter"). The control model was trained on the unfiltered dataset ("No filter").

After training, we obtained translations of an in-domain test set and out-of-domain test set (WMT 2020) for each model and evaluated the translations with automated metrics and performed human evaluation.

4.2.1 Automated Metrics

For JA>EN, the "Train then filter" approach achieved results on the in-domain test set that were significantly better than any other model. It also achieved the best BLEU score on the out-of-domain test set. For the DE>EN language direction, the "No filter" baseline achieved the best scores for both test sets. Overall, scores were higher for the DE>EN models than for the JA>EN models. In Tables 5 and 6 below we report automated metrics for each system divided by language pair and domain.

4.2.2 Human Evaluation

Human evaluation results are mostly in line with the automatic metrics. Overall, judging by these results, we did not observe any statistically significant improvement over the "No filter" baseline thanks to data filtering (we used the Student *t*-test for statistical significance). In Table 7, we show the average scores for each model for both languages pairs. A score of 0 indicates a perfect translation, while 25 indicates the lowest possible quality. For the DE>EN language pair, the best result was achieved with the baseline method for out-of-domain data (which is is in line with most of the automatic metrics), while the "Train and then filter" method had the best score for the in-domain data set (although the difference was minimal). For the JA>EN language pair, we observed the best scores with the "Train and then filter" method, which, again, is in line with most of the automatic metrics.

5 Discussion

In this paper we explored the relative performance of different methods of filtering noise from natural language training data, and the effect of filtering on the downstream task of machine translation. We found that cross-entropy filtering using models trained for the translation task

Table 7: Average human MQM evaluation scores on in-/out-of-domain test data.

System	DE>EN	JA>EN
No filter (Baseline)	0.73/**1.32**	1.77/8.14
Marian	0.72/1.67	1.97/7.40
Marian from no filter	0.83/1.51	2.10/7.89
Train then filter	**0.71**/1.58	**1.67/7.00**

performed better than multilingual alternatives such as LASER or COMET at identifying the types of noise we introduced across almost all noise types in both the DE>EN and JA>EN language directions. Language agnostic models have another disadvantage, which is that they cannot be used to identify wrong language data, a common source of noise in bilingual corpora.

However, the clear superiority of cross-entropy filtering did not unambiguously extend to the downstream translation task, where a model trained on the unfiltered dataset performed the best in DE>EN translation, and no model achieved a statistically significant improvement over the baseline in the human evaluation. This suggests that in the regime of a few million sentences, the advantages of having more data volume or more diverse data can outweigh the costs incurred by significant noise present in the dataset.

Our results suggest that in situations where the quality of training data is uncertain, fair results can be obtained by training for a short time on all the available data, filtering the training data with LASER or cross-entropy scores, and then continuing to train on a cleaner subset of the data. Given that LASER is language-agnostic, an additional filtering step based on language-identification may be required when using this tool.

In this study we generally followed the noise taxonomy found in Khayrallah and Koehn (2018a), but other ways of categorizing noise also exist. We are also interested to investigate how these tools perform with noisy data categorized in linguistic terms, such as problems of fluency vs. adequacy. Does data filtration with these tools introduce systemic bias of some sort, such as by preferentially removing sentences with numerous acronyms, shorter sentences, or sentences with lots of punctuation marks? Would the same results be obtained with lower resource languages? We hope to pursue these questions in future research.

A Appendix A

In Figure 1 below, we provide a detailed breakdown of the number of sentences with different types of corruption included in the datasets for each engine, grouped by the degree of corruption. For word and spelling permutations, we included 20,000 sentences with 1 permutation, 20,000 sentences with 2 permutations, and so on up to 5 permutations. For missing source and missing target content, we removed 5% of the words in the first 10,000 sentences, 10% of the words in the second 10,000 sentences, and so on up to 50% of the words. For sentence pairs with a third-language source or target, for half the sentences we used a more similar language (Chinese for Japanese, Dutch for German), and for the other half we used a more distant language (German for Japanese, and Russian for German).

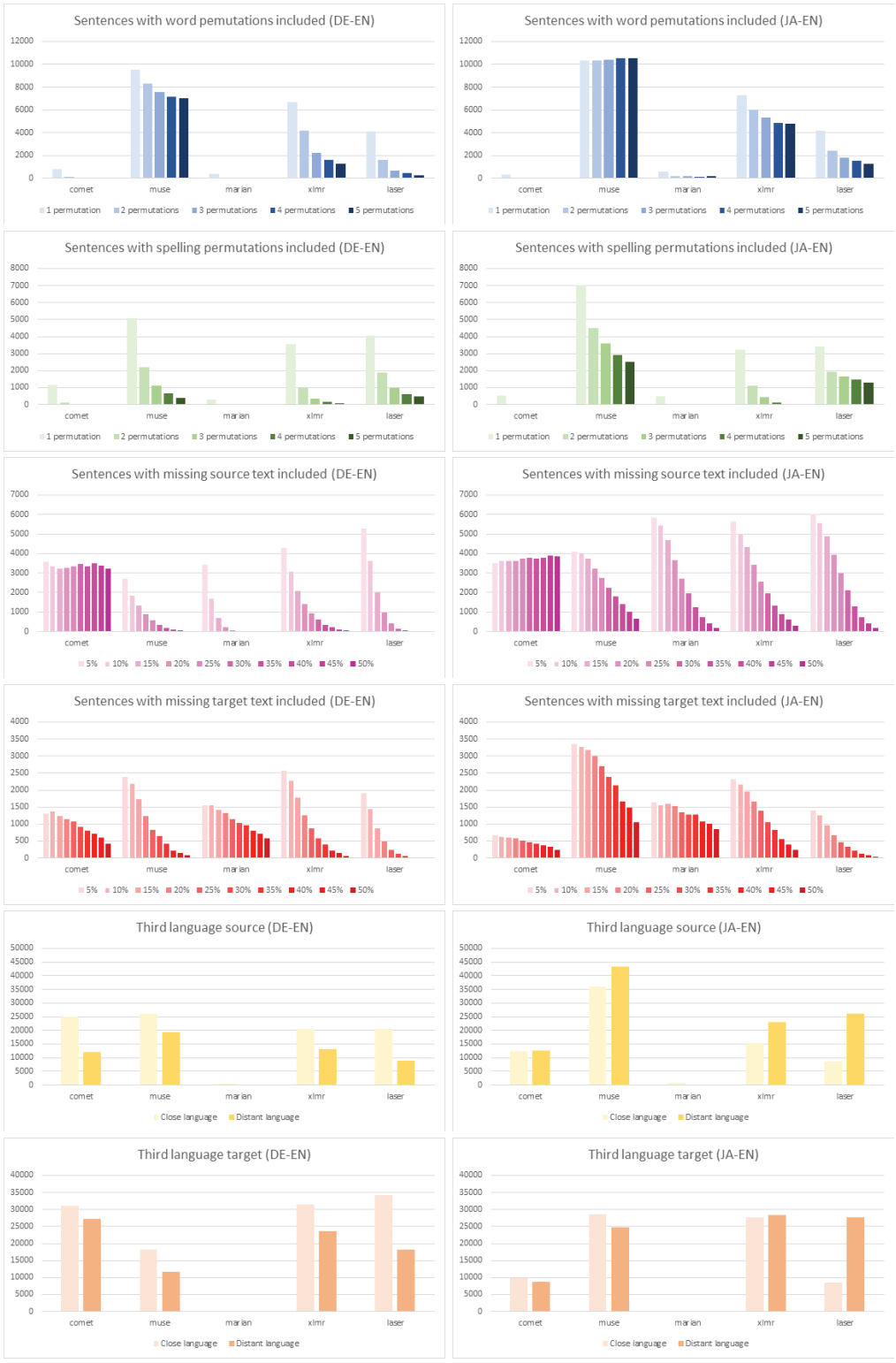

Figure 1: Comparison of filtering performance of different tools on different types of noise

References

Açarçiçek, H., Çolakoğlu, T., Aktan Hatipoğlu, P. E., Huang, C. H., and Peng, W. (2020). Filtering noisy parallel corpus using transformers with proxy task learning. In *Proceedings of the Fifth Conference on Machine Translation*, pages 940–946, Online. Association for Computational Linguistics.

Artetxe, M. and Schwenk, H. (2019). Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203, Florence, Italy. Association for Computational Linguistics.

Bañón, M., Chen, P., Haddow, B., Heafield, K., Hoang, H., Esplà-Gomis, M., Forcada, M. L., Kamran, A., Kirefu, F., Koehn, P., Ortiz Rojas, S., Pla Sempere, L., Ramírez-Sánchez, G., Sarrías, E., Strelec, M., Thompson, B., Waites, W., Wiggins, D., and Zaragoza, J. (2020). ParaCrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online. Association for Computational Linguistics.

Carpuat, M., Vyas, Y., and Niu, X. (2017). Detecting cross-lingual semantic divergence for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 69–79, Vancouver. Association for Computational Linguistics.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Conneau, A., Lample, G., Ranzato, M., Denoyer, L., and Jégou, H. (2017). Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Cui, L., Zhang, D., Liu, S., Li, M., and Zhou, M. (2013). Bilingual data cleaning for SMT using graph-based random walk. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 340–345, Sofia, Bulgaria. Association for Computational Linguistics.

Freitag, M., Foster, G., Grangier, D., Ratnakar, V., Tan, Q., and Macherey, W. (2021a). Experts, errors, and context: A large-scale study of human evaluation for machine translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Freitag, M., Rei, R., Mathur, N., Lo, C.-k., Stewart, C., Foster, G., Lavie, A., and Bojar, O. (2021b). Results of the WMT21 metrics shared task: Evaluating metrics with expert-based human evaluations on TED and news domain. In *Proceedings of the Sixth Conference on Machine Translation*, pages 733–774, Online. Association for Computational Linguistics.

Herold, C., Rosendahl, J., Vanvinckenroye, J., and Ney, H. (2021). Data filtering using cross-lingual word embeddings. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 162–172, Online. Association for Computational Linguistics.

Junczys-Dowmunt, M. (2018). Dual conditional cross-entropy filtering of noisy parallel corpora. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 888–895, Belgium, Brussels. Association for Computational Linguistics.

Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., Seide, F., Germann, U., Fikri Aji, A., Bogoychev, N., Martins, A. F. T., and Birch, A. (2018). Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Khayrallah, H. and Koehn, P. (2018a). On the impact of various types of noise on neural machine translation. *arXiv preprint arXiv:1805.12282*.

Khayrallah, H. and Koehn, P. (2018b). On the impact of various types of noise on neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia. Association for Computational Linguistics.

Koehn, P., Chaudhary, V., El-Kishky, A., Goyal, N., Chen, P.-J., and Guzmán, F. (2020). Findings of the WMT 2020 shared task on parallel corpus filtering and alignment. In *Proceedings of the Fifth Conference on Machine Translation*, pages 726–742, Online. Association for Computational Linguistics.

Koehn, P. and Knowles, R. (2017). Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Kreutzer, J., Caswell, I., Wang, L., Wahab, A., van Esch, D., Ulzii-Orshikh, N., Tapo, A., Subramani, N., Sokolov, A., Sikasote, C., Setyawan, M., Sarin, S., Samb, S., Sagot, B., Rivera, C., Rios, A., Papadimitriou, I., Osei, S., Suarez, P. O., Orife, I., Ogueji, K., Rubungo, A. N., Nguyen, T. Q., Müller, M., Müller, A., Muhammad, S. H., Muhammad, N., Mnyakeni, A., Mirzakhalov, J., Matangira, T., Leong, C., Lawson, N., Kudugunta, S., Jernite, Y., Jenny, M., Firat, O., Dossou, B. F. P., Dlamini, S., de Silva, N., Ballı, S. Ç., Biderman, S., Battisti, A., Baruwa, A., Bapna, A., Baljekar, P., Azime, I. A., Awokoya, A., Ataman, D., Ahia, O., Ahia, O., Agrawal, S., and Adeyemi, M. (2022). Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Lu, J., Ge, X., Shi, Y., and Zhang, Y. (2020). Alibaba submission to the WMT20 parallel corpus filtering task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 979–984, Online. Association for Computational Linguistics.

McCann, P. (2020). fugashi, a tool for tokenizing Japanese in python. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 44–51, Online. Association for Computational Linguistics.

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

Rei, R., Stewart, C., Farinha, A. C., and Lavie, A. (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.

Schwenk, H. (2018). Filtering and mining parallel data in a joint multilingual space. *CoRR*, abs/1805.09822.

Schwenk, H. and Douze, M. (2017). Learning joint multilingual sentence representations with neural machine translation. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 157–167, Vancouver, Canada. Association for Computational Linguistics.

Schwenk, H., Wenzek, G., Edunov, S., Grave, E., and Joulin, A. (2019). Ccmatrix: Mining billions of high-quality parallel sentences on the web.

Taghipour, K., Khadivi, S., and Xu, J. (2011). Parallel corpus refinement as an outlier detection algorithm. In *MT Summit XIII. Machine Translation Summit (MT Summit-11), 13., September 19-23, Xiamen, China*. NA.

Xu, H. and Koehn, P. (2017). Zipporah: a fast and scalable data cleaning system for noisy web-crawled parallel corpora. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2945–2950, Copenhagen, Denmark. Association for Computational Linguistics.