# The Challenge of Finding Robust and Efficient Strategies for Training Machine Translation Models with Noisy Data

**Mikko Aulamo**[1] and **Sami Virpioja**[1] and **Yves Scherrer**[1,2] and **Jörg Tiedemann**[1]
[1]University of Helsinki, Dept. of Digital Humanities
[2]University of Oslo, Dept. of Informatics
[1]firstname.lastname@helsinki.fi
[2]firstname.lastname@ifi.uio.no

## Abstract

Most machine translation datasets come with a certain level of noise, and strategies for handling such data need to be robust and efficient. Data selection and filtering are challenging and may depend on expensive language-specific tools that are not necessarily available, especially for low-resource languages. This paper looks at training strategies that combine cheap heuristic filters with curriculum learning to implement iterative procedures that robustly operate on raw noisy data without expensive prior preprocessing and data selection. The intuition is that we can cluster data into buckets with varying noise levels and use different sets of buckets at different stages of MT model training. We test various strategies and compare them to pre-filtering approaches for a diverse set of low-resource languages and conclude that curriculum learning can improve robustness but does not necessarily lead to improved translation performance. Overall, the experiments demonstrate the importance of proper experimental workflows, which cannot easily generalize from one language pair and scenario to another.

## 1 Introduction

Neural machine translation (NMT) models are dependent on clean parallel training data in order to produce high quality translations (Khayrallah and Koehn, 2018). Available parallel datasets – especially those based on web crawls (Bañón et al., 2020; de Gibert et al., 2024) – can contain significant amounts of noise. Therefore, a common preprocessing step in NMT training pipelines is a cleaning phase which removes noisy and low-quality sentence pairs (Koehn et al., 2018; Koehn et al., 2019; Koehn et al., 2020).

Simple data cleaning can be conducted with heuristic methods than only use inexpensive CPU processing (Koehn et al., 2007; Sánchez-Cartagena et al., 2018; Aulamo et al., 2020), but state-of-the-art cleaning methods rely on neural classifiers that require significant computational resources and often have to be adapted to specific language pairs (Junczys-Dowmunt, 2018; Artetxe and Schwenk, 2019; Zaragoza-Bernabeu et al., 2022).

In this work, we evaluate different data cleaning methods and find that neural methods are not always better than heuristic methods. We also propose a data cleaning approach that combines the simplicity and efficiency of heuristic cleaning methods with *curriculum learning* to select training data points that are the most useful for improving translation quality.

Curriculum learning is a method of training machine learning models that feeds different types of training data examples to the model at different time steps (Bengio et al., 2009). Usually, the model is first trained with the "easiest" data and incrementally more difficult data is included in later training stages (Wang et al., 2021). The "difficulty" of the data can be measured in different ways depending on the task. In language-related tasks, sentence length or the amount of rare words have been used as proxies for difficulty (Kocmi and Bojar, 2017; Platanios et al., 2019).

Similarly, our core idea is to divide the training data into buckets that represent different types

and levels of noise. We then train NMT models using these buckets in a multi-stage curriculum learning process, selecting different buckets in different training stages, following the Baby Steps method (Spitkovsky et al., 2010). The bucket creation relies on scores produced by cheap heuristic filters and simple K-means clustering and therefore uses a fraction of the computational resources that are used in GPU-based cleaning methods. Additionally, the proposed method does not make permanent binary decisions of filtering data but, instead, the method enables NMT models to benefit from different levels of noisy data at different stages of training. Furthermore, our curriculum learning based cleaning method is unsupervised, as it requires no training on clean and noisy sentence pairs. This is in contrast to state-of-the-art parallel data cleaning models, such as Bicleaner AI (Zaragoza-Bernabeu et al., 2022), which require clean sentence pairs for a given language pair for training. The code for our method is available at https://github.com/Helsinki-NLP/OpusFilter/tree/curriculum-learning.

## 2 Related work

The simplest methods for cleaning parallel text data involve heuristics that are inexpensive to run. The Moses phrase-based translation framework (Koehn et al., 2007) includes a script for removing too long and empty sentences, as well as sentence pairs with highly dissimilar source and target sentence lengths.[1] In addition to length-based rules similar to the ones above, Bicleaner (Sánchez-Cartagena et al., 2018) implements a number of hard rule filters based on proportions of different types of characters, number and script inconsistencies, and language identification.[2] Another toolbox, OpusFilter (Aulamo et al., 2020) is a configurable parallel data cleaning tool that implements filtering rules based on segment lengths, script and language identification, special characters, and sentence similarity. While OpusFilter comes with default threshold values for each rule, it also includes a method for adjusting the threshold values automatically based on the properties of a given training set (Aulamo et al., 2023).

Other state-of-the-art data cleaning methods are based on neural networks. Bicleaner AI (Zaragoza-Bernabeu et al., 2022) is a binary classifier that distinguishes noisy from clean segments. It is trained by finetuning XLM-R (Conneau et al., 2020) using synthetically created noisy data. Dual conditional cross-entropy is another noise classification method. It compares the cross-entropy scores from two inverse translation models (Junczys-Dowmunt, 2018). Another way to measure the quality of a sentence pair is to compute the similarity of the sentence embeddings. Different sentence embedding frameworks can be used for this purpose, for example LASER (Schwenk and Douze, 2017; Artetxe and Schwenk, 2019), which has been trained in a machine translation context, or LaBSE (Feng et al., 2022), which are language agnostic sentence embeddings learned with BERT (Devlin et al., 2019).

Curriculum learning is a method of training a machine learning system that feeds training data in a specific order, usually starting with easy data and gradually including harder data (Bengio et al., 2009). Curriculum learning has also been applied in the context of training MT systems. Kocmi and Bojar (2017) organize their training data into mini-batches, with each mini-batch containing similar sentences based on sentence length or the number of given part-of-speech tags in the sentence. Platanios et al. (2019) sort their data based on sentence length and word rarity and apply a continuous competence function to determine which part of the data to use at which time step. Zhao et al. (2020) improve the performance of pre-trained NMT models using their original training data by applying reinforcement curriculum learning. As a part of their episodic NMT training procedure, Chen et al. (2024) use curriculum learning for domain adaptation by starting the training with general data and adding domain specific data at later stages.

Curriculum learning can also be considered a denoising process when the data presented to the system are sorted from clean to noisy. For example, Wang et al. (2019) produce scores for sentence pairs with language models and apply curriculum learning to conduct data selection and domain adaptation. Mohiuddin et al. (2022) propose two data selection methods based on curriculum learning for NMT. In the first method, they order the training data using pre-calculated cleanness scores from LASER, dual conditional cross-entropy, or a language model cross-entropy based

[1] http://www2.statmt.org/moses/?n=Moses.Baseline

[2] https://github.com/bitextor/bicleaner-hardrules for full list of filters.

| | en-eu | en-ga | en-mk | en-nn | en-sw |
|---|---|---|---|---|---|
| Raw | 20,830,243 | 101,001,090 | 91,293,129 | 28,701,601 | 247,557,313 |
| Deduplicated | 4,005,529 | 10,159,720 | 8,017,670 | 5,262,996 | 34,734,871 |
| Near-deduplicated | 3,088,483 | 6,335,623 | 5,051,038 | 3,136,567 | 21,576,709 |
| Bicleaner AI | 610,694 | 994,756 | 1,139,063 | 132,540 | 1,710,223 |
| OF Default | 880,873 | 2,241,556 | 993,807 | 73,480 | 1,744,039 |
| OF Autogen | 155,395 | 1,749,767 | 5,042,721 | 3,131,545 | 1,502,216 |
| OF Adapt 1ep (en-xx) | 998,577 | 4,696,352 | 3,156,487 | 2,523,075 | 4,620,352 |
| OF Adapt 1ep (xx-en) | 2,813,541 | 2,864,631 | 4,245,247 | 3,135,013 | 14,962,742 |
| OF Adapt Conv (en-xx) | 1,461,191 | 3,625,025 | 2,748,101 | 1,114,502 | 2,774,207 |
| OF Adapt Conv (xx-en) | 2,090,293 | 4,684,891 | 3,051,952 | 2,417,068 | 14,610,340 |

**Table 1:** Number of sentence pairs in each dataset. The deduplication and cleaning methods are described in Sections 3 and 4. The bottom four rows show the total number of sentence pairs used for training for each translation model trained with our curriculum learning method. The number is calculated by taking the sum of the sizes of the buckets that were accepted into the training set during any bucket evaluation phase.

score and they feed training data to the model using a continuous scheduler. In the second method, they score the training data with the translation model that is being trained and use a selection window to choose the most beneficial training data for the model at a given time step.

Our method generalizes the second method of Mohiuddin et al. (2022). Their approach of scoring all training samples with a translation model is feasible for their small in-domain corpus, but the computational cost of scoring a large, noisy general-domain corpus with a translation model is prohibitive. Instead of scoring the entire corpus, we first cluster the sentence pairs based on cleanness scores obtained from OpusFilter and score samples of each cluster, assuming that the usefulness of all sentence pairs in a cluster is approximately the same.

Our method is based on heuristic scores that are produced using CPU computation, which is much less resource intensive than training and running the GPU-heavy state-of-the-art parallel data cleaning methods. Moreover, using the heuristic scores in combination with K-means clustering and curriculum learning makes our method unsupervised. This is another difference to state-of-the-art cleaning models, which typically require labeled sentence pairs for training.

## 3 Data

In our experiments, we use the raw HPLT v1.2 parallel datasets[3] (de Gibert et al., 2024), which are crawled from public websites excluding adult content. We focus on translation for five language pairs in both directions where English (`en`) is paired with Basque (`eu`), Irish (`ga`), Macedonian (`mk`), Nynorsk (`nn`) and Swahili (`sw`). These languages represent a variety of language families and are relatively low-resourced.

As a pre-filtering step, we remove exact duplicate sentence pairs from the dataset. After removing exact duplicates from the dataset, we noticed that there are still a fair number of near-duplicate training examples. In particular, the dataset contains sentences that describe products and include product identifier tags consisting of numeral and alphabetic characters. We observed large sets of sentences that differ from each other only by the product identifier tag but are otherwise identical. To remove such near-duplicates, we rerun deduplication ignoring words that contain non-alphabetic characters.[4]

The HPLT dataset also contains a version that has been cleaned with Bicleaner AI (Zaragoza-Bernabeu et al., 2022), which we use as our baseline of a neural data cleaning method. Table 1 shows the data sizes for each language pair.

## 4 Methodology and experiments

The main goal of our experiments is to investigate the impact of data cleaning methods on machine translation quality. All translation models are trained with MarianNMT (Junczys-Dowmunt

[3]https://hplt-project.org/datasets/v1.2

[4]In order to separate alphabetic words from punctuation marks, the sentences are tokenized into words using the Moses tokenizer via OpusFilter: https://helsinki-nlp.github.io/OpusFilter/preprocessors/tokenizer.html Tokenization is only applied on the fly during the deduplication process, and the resulting dataset remains untokenized.

et al., 2018) using the Transformer base architecture (Vaswani et al., 2017). Training parameters are specified in Appendix A. We use Flores-200 (NLLB Team et al., 2024) as development and test sets.

We experiment with the following data cleaning methods:

- **Near-deduplicated**: No cleaning, only near-duplicate removal,
- **Bicleaner AI**: The cleaned HPLT datasets obtained with Bicleaner AI,
- **OF Default**: OpusFilter with default threshold values (cf. Section 4.1),
- **OF Autogen**: OpusFilter with automatic threshold generation (cf. Section 4.1),
- **OF Adapt 1ep** and **Conv**: Curriculum learning methods relying on the OpusFilter scores (cf. Sections 4.2 and 4.3).

OpusFilter and curriculum learning cleaning methods are explained in more detail in the following subsections.

### 4.1 OpusFilter

OpusFilter operates on a set of parallel data filters. Each filter in OpusFilter produces a score for every sentence pair in a dataset. If the score exceeds a threshold value, the sentence pair is considered clean. The threshold values can be set manually, but there is also a default value for each filter. In the **OF Default** setting, we use the same set of filters as in Aulamo et al. (2023) and their corresponding default threshold values, with one addition: SimilarityFilter. We noticed that the training sets contain a significant number of sentence pairs consisting of identical source and target sentences which can be detected and filtered by SimilarityFilter. The full set of filters used for the OF Default setup is the following:

- AlphabetRatioFilter: The proportion of alphabetical characters in a sentence.
- CharacterScoreFilter: The proportion of characters in a given script in a sentence.
- LanguageIdFilter: Confidence score from the fasttext language identification model[5] (Joulin et al., 2016; Joulin et al., 2017).
- LengthRatioFilter: The ratio between the lengths of the source and target sentences. There are two length ratio scores for each sentence pairs: one based on characters and one based on words.
- NonZeroNumeralsFilter: Similarity of numerals between source and target sentences ignoring the leading zeros (Vázquez et al., 2019).
- SimilarityFilter: Levenshtein distance between source and target sentence.
- TerminalPunctuationFilter: Similarity of sentence-ending punctuation mark between source and target sentences (Vázquez et al., 2019).

OpusFilter includes a method for automatically generating optimal threshold values for each filter based on clustering and feature importance analysis for a given dataset (Aulamo et al., 2023). We refer to this cleaning method as **OF Autogen**. It clusters the sentence pairs into two groups, clean and noisy, and uses the values of the noisy cluster center as threshold values. We obtain specific threshold values for each language pair with this method and produce new datasets with these thresholds.

### 4.2 Clustering the data according to noise level

All existing cleaning methods are essentially binary classifiers: they remove the sentence pairs that they consider noisy, and keep those that they consider clean. However, the border between clean and noisy is often not as clear-cut as it may seem, and it may be argued that MT models may still benefit from data that is slightly noisy. Moreover, the OF Autogen method can fail if it tries to force a clean-noisy division when an input dataset contains no noise, or if some of the filters do not work properly and assign low scores for obviously clean examples. To address these issues, we (1) cluster the data into many clusters instead of just two and (2) use curriculum learning strategies to select the most useful data clusters.

In order to divide the training data based on different types of noise, we first score the sentence pairs using the same set of filters as described in Subsection 4.1 and use the numeric scores directly without applying any thresholds. Next, we use the score from each filter as an individual feature to

[5] https://fasttext.cc/docs/en/language-identification.html

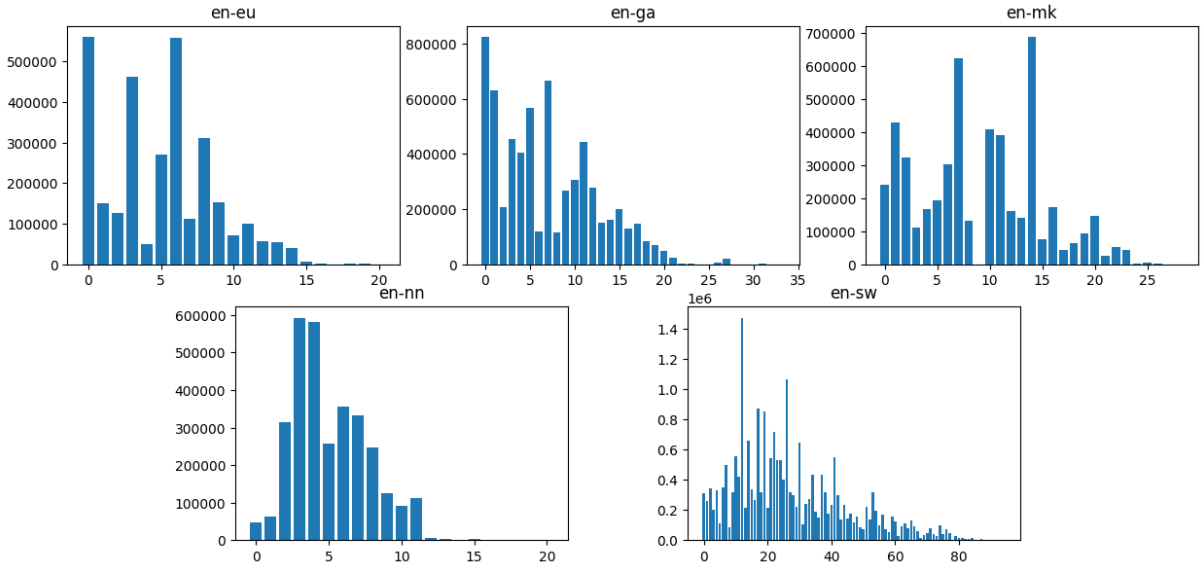


**Figure 1:** Bucket sizes after clustering. The buckets are in the order of expected cleanness.

create sentence pair vectors. We standardize the features and cluster them with K-means. We set the number of clusters to depend on the dataset size using the formula $10 + (d - b)/b$, where $d$ is the number of sentence pairs. We set $b$ to be 250,000, because we found 250,000 to be an effective bucket size in our early experiments. The formula ensures that even small datasets are divided into at least 10 buckets. We train K-means on a sample of 100,000 sentence pairs and then classify the rest of the training data into the found clusters. This gives us buckets containing different types of noise. Next, we sort the buckets from clean to noisy based on the geometric mean of the values in the cluster centers. Figure 1 shows the amount of data in each bucket for all language pairs after clustering and ordering the buckets. There is a high degree of variance in the bucket sizes, although most of the data is concentrated in the cleaner buckets. Figure 2 shows the different types of noise present in each bucket of the English-Nynorsk dataset. Buckets 0-4 have low noise values overall, while buckets 5-11 are also fairly clean with slight noise peaks in AlphabetRatioFilter, LanguageIDFilters, SimilarityFilters, NonZeroNumeralsFilter or TerminalPunctuationFilters. Buckets 12-15 have high and buckets 16-20 have very high noise values in CharacterScoreFilters and LengthRatioFilters.

### 4.3 Curriculum learning strategies

There are various ways to train MT systems on the bucketed training data. While an obvious approach starts with the cleanest buckets and gradually adds noisier ones, it is not clear (a) how long to train on each set of buckets and (b) if the cleanness scores obtained during clustering are good approximations of the usefulness of the bucket. Therefore, we conducted a number of preliminary experiments to find a curriculum learning strategy that optimizes both translation quality and training time. These experiments only considered translation from English to Irish.

**OF Baby Steps Conv** We start our experimentation with a method inspired by Baby Steps (Spitkovsky et al., 2010). We initialize NMT training with the cleanest bucket as the training set and train until convergence. Next, we add the second-cleanest bucket to the training set and again train until convergence. We continue this way of iterative training until all buckets are added to the training set.

**OF Baby Steps 1ep** The method above will likely result in very long training runs because it needs to converge for every bucket. We experiment with a variation of the above Baby Steps method, where we train only for one epoch each time a new bucket is added to the training data.

**OF Self-paced 1ep** The cleanness order of the buckets is based on average scores from heuristic filters, and it is not always perfect. Thus, we might encounter some low-quality buckets earlier than some high-quality buckets. For this reason, we move to self-paced learning, similar to Mohiuddin et al. (2022), in which the current transla-

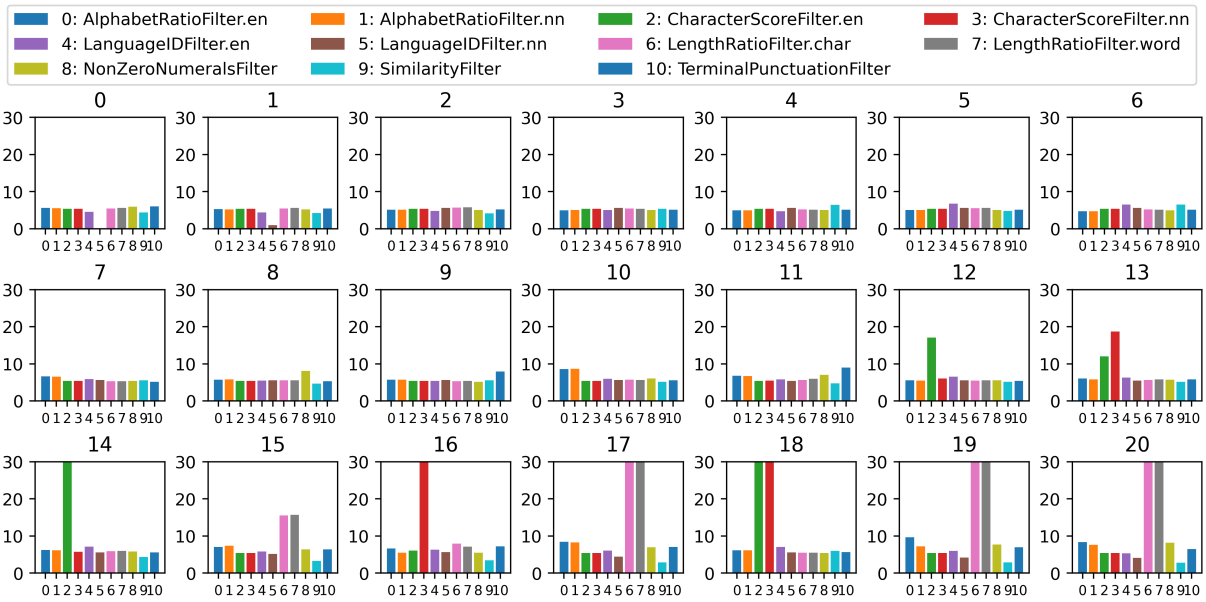


**Figure 2:** Standardized and normalized noise values for each bucket of the English-Nynorsk dataset.

tion model is used to evaluate the training data. OF Self-paced 1ep is otherwise the same as Baby Steps, but each time a new bucket is added to the training data and the model is trained for one epoch, the model is evaluated against the development set using BLEU (Papineni et al., 2002). If the evaluation score is lower than the previous evaluation score, the current bucket is removed from the training set before the next bucket is added.

**OF Adapt Conv** OF Self-paced 1ep permanently discards noisy buckets, while we hypothesize that data with some degree of noise could still be beneficial for the NMT model at a different stage of training. Therefore, in the final method, OF Adapt Conv, we use adaptive self-paced learning. We train the translation model by alternating between two phases: in the first phase, we find the most beneficial buckets of training data for the model at the current time step, and in the second phase, we train the model using the selected buckets for one epoch. This process is illustrated in Figure 3.

During the bucket selection phases, the model is trained for 100 updates with each bucket starting from the cleanest bucket and moving on to noisier ones. We measure the evaluation score after training with each bucket. If the score improves from the previous best score, the current bucket is included in the training set. If the score does not improve, the current bucket is not included in the training set, and the translation model is restored to the state with the previous best score. In the training phase, the NMT model is trained until convergence with the selected buckets. The training process ends once no beneficial buckets are found during the bucket selection phase. The resulting translation model is the one with the highest evaluation score.

We assume that some of the data buckets are highly noisy and do not need to be evaluated during all bucket selection phases. Therefore, during the first selection phase, we stop evaluating buckets after encountering five consecutive harmful buckets. The buckets that were evaluated in the first bucket selection phase form the set of buckets that will be evaluated in all subsequent selection phases. During the first selection phase, we use cross-entropy instead of BLEU as the evaluation score because BLEU did not produce meaningful scores during the early stages of training.

**OF Adapt 1ep** OF Adapt 1ep is a variation of OF Adapt Conv, where the model is trained for one epoch instead of training until convergence during each training phase. We evaluate this variation to study how its translation performance compares to OF Adapt Conv's while having a shorter training time.

### 4.4 Evaluation of preliminary experiments

Table 2 shows the training times and BLEU scores measured on the development set for all preliminary curriculum learning experiments, and for tra-

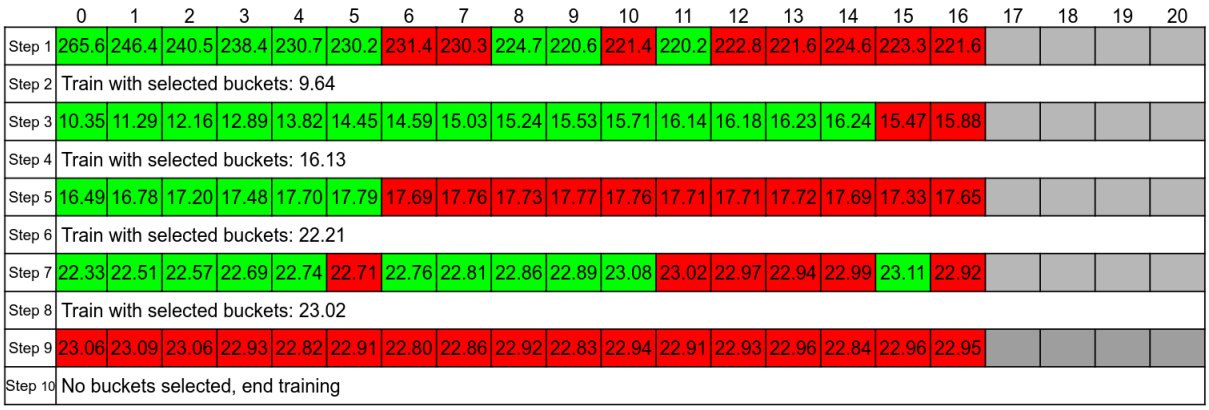


**Figure 3:** Visualization of training a Nynorsk to English model with OF Adapt 1ep. The odd numbered steps are bucket evaluation phases and the number in each cell represent the evaluation score for each bucket. The even numbered steps show the best evaluation scores achieved during the training phases. The scores in Step 1 are cross-entropy and all scores in later steps are BLEU. The grey cells represent buckets that are not considered for evaluation since they occur after five consecutive harmful buckets in Step 1. The training process stops at Step 10 because no buckets were selected during Step 9. The resulting final translation model is restored from Step 7 after training with bucket 15, since the model achieved the best evaluation score in that state.

| | BLEU | Time |
|---|---|---|
| OF Default | 30.1 | 9:22 |
| OF Baby Steps Conv | 28.5 | 12:53 |
| OF Baby Steps 1ep | 17.5 | 2:31 |
| OF Self-paced 1ep | 27.9 | 22:06 |
| OF Adapt Conv | 28.3 | 9:40 |
| OF Adapt 1ep | 15.1 | 3:06 |

**Table 2:** Curriculum learning strategy experiments. BLEU scores are calculated on the development set and are not comparable to those in Table 6. Time is training time for the translation model (h:min).

ditional NMT training with the English-Irish data cleaned with default filtering thresholds (OF Default) for comparison. The OF Baby Steps Conv method performs decently well, but it does achieve a lower BLEU score than OF Default, and it is also slower to train. OF Baby Steps 1ep has a dramatically lower BLEU score, which suggests that training for only one epoch after each added bucket might be insufficient for capturing adequate training signals. However, OF Self-paced 1ep raises the BLEU score again to a decent level, which shows that excluding the harmful buckets from the training set has a significant positive effect. The downside of OF Self-paced 1ep is that the longer the training process continues, the larger the training set grows and the longer it takes to evaluate each new bucket.

OF Adapt Conv solves this problem by separating the bucket evaluation and model training phases, and by assessing the harmfulness of each bucket individually without adding them to the training set first. OF Adapt Conv reaches a similar training time to OF Default, but the BLEU score remains 1.8 points lower. We still go forward with assessing this method for additional language pairs as it is our most promising approach and its performance could vary in different settings. OF Adapt 1ep achieves a dramatically lower BLEU score than OF Adapt Conv, but we also include this method in further experiments to see how it performs for other language pairs.

## 5 Results

We evaluate the translation experiments by measuring COMET (Rei et al., 2020), BLEU (Papineni et al., 2002) and chrF (Popović, 2015) scores on the Flores-200 devtest set (NLLB Team et al., 2024). COMET scores are measured with the `unbabel-comet` Python package[6] using the `wmt22-comet-da` model. BLEU and chrF scores are measured with sacrebleu (Post, 2018).

We train each model six times, initializing the NMT model with a random seed each time. For OF Autogen, OF Adapt 1ep and OF Adapt Conv, we also randomly initialize data clustering each time.

Curriculum learning methods generally lead to faster convergence during model training (Wang et al., 2021). We report the training times for all our experiments to examine how our proposed curriculum learning approach compares to traditional training.

[6]https://github.com/Unbabel/COMET

| | en-eu | eu-en | en-ga | ga-en | en-mk | mk-en | en-nn | nn-en | en-sw | sw-en |
|---|---|---|---|---|---|---|---|---|---|---|
| Near-dedup. | 61.8±0.4 | 78.3±1.0 | 42.5±2.1 | 73.7±0.9 | 59.2±1.0 | 81.3±0.9 | 68.4±0.6 | **75.7**±1.0 | 56.0±0.1 | 64.6±6.0 |
| Bicleaner AI | **80.8**±0.4 | **81.0**±0.2 | **76.3**±0.2 | **77.0**±0.4 | **84.8**±0.1 | **83.8**±0.2 | 71.4±0.8 | 71.9±0.6 | **77.2**±0.6 | 75.4±0.6 |
| OF Default | 79.0±0.4 | **81.0**±0.2 | 74.8±0.3 | 75.6±0.8 | 83.9±0.3 | 83.2±0.3 | 46.8±0.6 | 45.8±0.3 | 75.6±0.4 | **76.1**±0.3 |
| OF Autogen | 57.2±3.7 | 68.8±10.3 | 74.3±1.0 | 74.4±1.8 | 80.2±9.0 | 82.9±0.6 | 66.8±8.6 | 67.3±9.9 | 72.0±7.1 | 74.0±3.8 |
| OF Adapt 1ep | 41.9±15.3 | 74.0±2.2 | 68.7±2.7 | 62.8±0.7 | 71.7±8.7 | 77.3±3.1 | 48.0±17.8 | 64.0±1.5 | 71.3±0.7 | 63.2±4.1 |
| OF Adapt Conv | 60.5±20.3 | 75.8±0.7 | 73.0±0.6 | 69.0±3.8 | 80.4±0.8 | 80.0±1.4 | **74.7**±0.8 | 71.5±4.1 | 64.8±16.1 | 61.6±5.6 |

**Table 3:** COMET scores (mean±std) for the translation tasks using near-deduplicated data, Bicleaner AI, and OpusFilter (OF) based methods. The best scores per language pair are bolded.

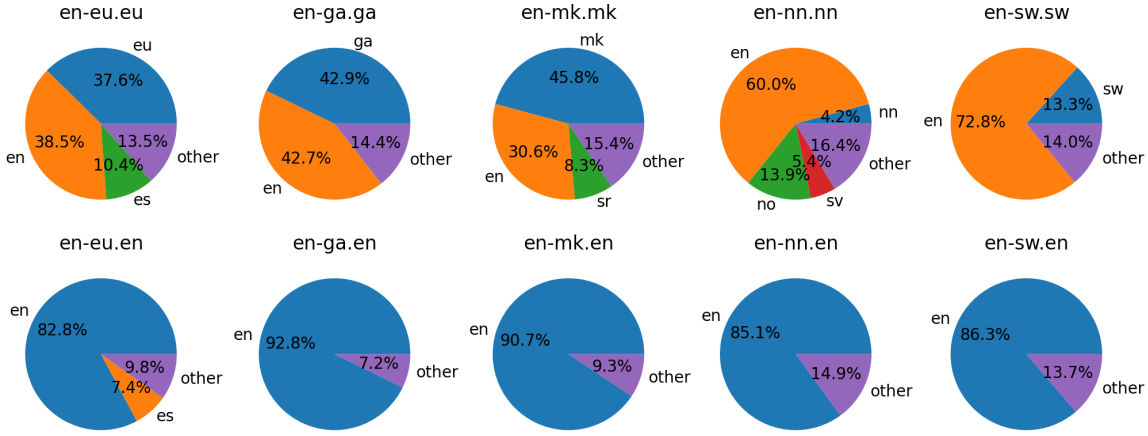


**Figure 4:** Distributions of language labels for each near-deduplicated training set predicted with fasttext.

### 5.1 Scores

Table 3 shows the COMET scores and Tables 6 and 7 in Appendix B show the BLEU and chrF scores for all translation experiments. Somewhat surprisingly, the near-deduplicated datasets without removing any noise achieve competitive scores when translating into English and the highest score for `nn-en`. In the other translation directions, the scores are low, as is expected for highly noisy data. In particular, we noticed that the `en-xx` translation models produce mainly English text. To investigate this issue further, we ran language identification with fasttext on all near-deduplicated datasets. Figure 4 indeed shows that there is a high amount of English on both sides of the training data, which turns the translation task into an English auto-encoder task. This makes the `en-xx` models generate English, but also explains why the `xx-en` models are able to generate fairly high quality English output.

Bicleaner AI achieves the highest scores for most language pairs. OF Default is usually not far behind, and it outperforms Bicleaner AI for `sw-en` according to all metrics. This suggests that simple heuristic data cleaning methods can be as effective as more sophisticated neural cleaning methods in many cases.

However, OF Default completely fails in translating to and from Nynorsk while Bicleaner AI performs well. OF Default filters data based on hard threshold values and one of the filters is language identification using fasttext. Figure 4 reveals that fasttext annotates a large part of the Nynorsk (`nn`) dataset with the label `no` for Norwegian.[7] OF Default removes all sentence pairs labeled as `no`, which – in combination with false positives from other filters – makes the OF Default training set tiny (73,480) and leads to poor translation quality.

When comparing the two curriculum learning variants, OF Adapt Conv beats OF Adapt 1ep in all language pairs except `en-sw` and `sw-en`. This suggests that it is more beneficial to train until convergence after each bucket selection phase. In general, OF Adapt Conv is 3-10 COMET points behind Bicleaner AI and OF Default but for some language pairs the difference is even higher. No-

[7]The label `no` technically refers to both Norwegian writing norms Nynorsk and Bokmål. The fasttext model contains both variety-specific labels `nn`, `nb` as well as the generic label `no`, so it is unclear to which variety the `no`-labeled sentences effectively belong.

| | en-eu | eu-en | en-ga | ga-en | en-mk | mk-en | en-nn | nn-en | en-sw | sw-en |
|---|---|---|---|---|---|---|---|---|---|---|
| Near-dedup. | 5:31±1:03 | 8:41±2:43 | 1:40±0:06 | 10:00±4:20 | 5:51±0:47 | 9:14±2:03 | 3:45±1:28 | *9:01±4:23 | 2:40±0:21 | 8:52±6:03 |
| Bicleaner AI | *4:23±0:39 | *4:45±0:25 | *6:29±0:48 | *6:48±0:50 | *6:43±0:24 | *7:03±1:12 | 6:25±1:15 | 7:06±1:19 | *7:43±1:43 | 7:57±1:29 |
| OF Default | 4:37±0:36 | *5:52±0:37 | 8:34±1:02 | 9:43±1:55 | 5:31±0:54 | 6:32±0:49 | 1:45±0:15 | 1:45±0:07 | 7:39±1:43 | *7:32±1:18 |
| OF Autogen | 4:29±0:35 | 7:29±1:31 | 8:02±1:28 | 7:17±3:32 | 6:46±3:11 | 7:32±1:11 | 4:31±2:10 | 5:20±2:60 | 7:10±3:19 | 7:03±3:32 |
| OF Adapt 1ep | 3:00±1:28 | 5:44±1:15 | 6:42±2:28 | 4:19±1:03 | 4:40±1:25 | 6:13±2:42 | 4:22±2:06 | 5:37±1:31 | 6:33±1:30 | 12:20±4:54 |
| OF Adapt Conv | 3:41±1:23 | 5:40±0:46 | 8:60±0:54 | 6:37±1:54 | 4:46±0:32 | 5:58±1:36 | *6:46±1:18 | 6:02±1:25 | 4:15±1:43 | 10:40±4:25 |

**Table 4:** GPU usage during NMT model training (h:min, mean±std). Models with the best COMET scores are marked with *.

| | en-eu | en-ga | en-mk | en-nn | en-sw |
|---|---|---|---|---|---|
| OpusFilter CPUh | 0:07 | 0:13 | 0:11 | 0:07 | 0:42 |
| Bicleaner AI GPUh | 4:17 | 8:48 | 7:01 | 4:21 | 29:58 |

**Table 5:** Corpus cleaning times (h:min). OpusFilter's times include measuring scores from all filters described in Section 4.1 for all sentence pairs in the near-deduplicated datasets using a single core of an Intel Xenon Gold 6230 CPU. Bicleaner AI's times are estimated based on the information reported in Zaragoza-Bernabeu et al. (2022): 200 sentence pairs per second with an Nvidia 3080Ti GPU.

tably, for `en-eu` there is a 20.3 COMET point difference to Bicleaner AI, and there is also large variation between the randomly initialized OF Adapt Conv models. For `en-nn`, OF Adapt Conv achieves the highest score, and for `nn-en`, its score is competitive with Bicleaner AI. This result shows that our curriculum learning method does not necessarily help in achieving higher translation quality over a more straight-forward usage of heuristic filters, but it does help in avoiding some pitfalls such as the case of Nynorsk in our experiments. This can be explained by the fact that the curriculum learning strategies are able to access useful data that would have otherwise been discarded as noise. As can be seen in Table 1, the curriculum learning methods generally use larger datasets than traditional binary filtering methods.

OF Autogen achieves results that are close to the best methods for some language pairs, but it performs much worse for others. Aulamo et al. (2023) report that in all their experiments, the autogen method beats the default thresholds, but in our tests on a larger and more diverse set of language pairs, OF Autogen performs better than OF Default only for `en-nn` and `nn-en`.

### 5.2 Training and cleaning times

Table 4 shows the model training times for all translation experiments. All models are trained using a single Nvidia Volta V100 GPU and the models end training once they have converged based on early-stopping. Some of the lowest training times are due to the model converging quickly because of low-quality training data, for example OF Default with the Nynorsk dataset. Otherwise, the training times for the best performing models largely depend on the sizes of the datasets. In most cases, using our proposed curriculum learning method does not speed up NMT training, which is due to the fact that we run multiple iterations of bucket evaluation and training phases over the whole training set. In fact, in some cases our method runs multiple hours longer than traditional training using data filtered with Bicleaner-AI.

However, when comparing total GPU usage, we have to add Bicleaner AI's data cleaning time to the NMT training time while the heuristic filters are fast and run on regular CPUs. We do not have access to exact GPU hours as the data we use is pre-filtered with Bicleaner AI, but the authors report that a full Bicleaner AI model labels approximately 200 sentence pairs per second (Zaragoza-Bernabeu et al., 2022), which can give us an estimate. Table 5 shows the times it takes to clean the near-deduplicated datasets with OpusFilter and the estimated times it would take for Bicleaner AI. OpusFilter's heuristic cleaning methods are lightweight and run in 7 to 42 minutes on a CPU while the estimated GPU usage for cleaning the datasets with Bicleaner AI ranges from 4 to 30 hours.

### 5.3 Noise type analysis

In this section, we analyze which types of noise are beneficial in different stages of NMT training in the case of the Nynorsk to English OF Adapt 1ep NMT model. Figure 2 in Section 4.2 shows presence of different noise types in each bucket of the English-Nynorsk dataset, and Figure 3 in Section 4.3 illustrates the training process of the NMT model on these buckets. Buckets 0-4 are the cleanest ones, and they are included in all training phases. Buckets 5-11 are included in two or three training phases, and they are fairly

clean as well but have some lightly higher noise values from AlphabetRatioFilter, LanguageIDFilter, NonZeroNumeralsFilter and TerminalPunctuationFilter. Buckets 12-15 have high noise values from CharacterScoreFilters or from LengthRatioFilters, but the NMT model still benefits from these buckets. This indicates that, in certain stages of NMT training, it can be beneficial to include training examples even if they contain characters from different writing scripts, or if they have source and target sentences of significantly different lengths. Buckets 16-20 are very noisy according to CharacterScoreFilters and LengthRatioFilters, and they are excluded from all training phases.

## 6 Conclusion

In this paper, we compare heuristic and neural data cleaning methods and evaluate them in machine translation tasks. We also propose and evaluate an iterative self-pacing curriculum learning method for handling noisy data. We show that simple heuristics can rival neural methods while being much less resource intensive to run. Although our proposed curriculum learning method achieves lower performance in most language pairs than using heuristic filters directly, it is promising in making heuristic-based data cleaning more robust.

In the future, it would be interesting to study the proposed curriculum learning method for language pairs that are not covered by pretrained cleaning models, such as Bicleaner AI. In theory, our method should be able to adapt to any parallel dataset in any language, as it does not require examples of clean and noisy sentence pairs for training.

This work explores methods inspired by Baby Steps and self-paced learning. In future work, we would like to investigate other ways of utilizing heuristic filters in conjunction with curriculum learning. We also plan to conduct more thorough and detailed analyses of the overall energy consumption to complement the rough GPU-usage statistics presented in this paper.

## 7 Limitations

We use data that is pre-filtered with Bicleaner AI, and we can only estimate times that it would take to clean the datasets with Bicleaner AI. Thus, the total GPU usage comparisons in Section 5.2 have to be taken with a grain of salt.

Our experiments only include language pairs where English is one of the languages. Although the non-English languages we include in our study are diverse, conducting experiment on language pairs that are not English-centric could reveal new properties from the tested cleaning methods.

The curriculum learning methods we experimented with are limited to approaches inspired by Baby Steps and self-paced learning. Exploring other ways of combining heuristic filtering with curriculum learning could prove to be more effective.

## Carbon Impact Statement

This work contributed 122.92 kg of $CO_2$ to the atmosphere and used 1.11MWh of electricity (calculated using https://calculator.green-algorithms.org/ (Lannelongue et al., 2021)).

## Acknowledgments

This work was supported by the HPLT project, which has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101070350. The contents of this publication are the sole responsibility of its authors and do not necessarily reflect the opinion of the European Union.

This work was also supported by "The OPUS MT Factory - Efficient and Open High-Quality Machine Translation for Everyone" project, which is funded by the Research Council of Finland.

## References

Artetxe, Mikel and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In Korhonen, Anna, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203, Florence, Italy, July. Association for Computational Linguistics.

Aulamo, Mikko, Sami Virpioja, and Jörg Tiedemann. 2020. OpusFilter: A configurable parallel corpus filtering toolbox. In Celikyilmaz, Asli and Tsung-Hsien Wen, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 150–156, Online, July. Association for Computational Linguistics.

Aulamo, Mikko, Ona de Gibert, Sami Virpioja, and Jörg Tiedemann. 2023. Unsupervised feature selection for effective parallel corpus filtering. In

Nurminen, Mary, Judith Brenner, Maarit Koponen, Sirkku Latomaa, Mikhail Mikhailov, Frederike Schierl, Tharindu Ranasinghe, Eva Vanmassenhove, Sergi Alvarez Vidal, Nora Aranberri, Mara Nunziatini, Carla Parra Escartín, Mikel Forcada, Maja Popovic, Carolina Scarton, and Helena Moniz, editors, *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 31–38, Tampere, Finland, June. European Association for Machine Translation.

Bañón, Marta, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L. Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, Sergio Ortiz Rojas, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Elsa Sarrías, Marek Strelec, Brian Thompson, William Waites, Dion Wiggins, and Jaume Zaragoza. 2020. ParaCrawl: Web-scale acquisition of parallel corpora. In Jurafsky, Dan, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online, July. Association for Computational Linguistics.

Bengio, Yoshua, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Chen, Keyu, Di Zhuang, Mingchen Li, and J. Morris Chang. 2024. Epi-curriculum: Episodic curriculum learning for low-resource domain adaptation in neural machine translation. *IEEE Transactions on Artificial Intelligence*, 5(12):6095–6108.

Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In Jurafsky, Dan, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July. Association for Computational Linguistics.

de Gibert, Ona, Graeme Nail, Nikolay Arefyev, Marta Bañón, Jelmer van der Linde, Shaoxiong Ji, Jaume Zaragoza-Bernabeu, Mikko Aulamo, Gema Ramírez-Sánchez, Andrey Kutuzov, Sampo Pyysalo, Stephan Oepen, and Jörg Tiedemann. 2024. A new massive multilingual dataset for high-performance language technologies. In Calzolari, Nicoletta, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1116–1128, Torino, Italia, May. ELRA and ICCL.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, Jill, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Feng, Fangxiaoyu, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In Muresan, Smaranda, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland, May. Association for Computational Linguistics.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomás Mikolov. 2016. Fasttext.zip: Compressing text classification models. *CoRR*, abs/1612.03651.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In Lapata, Mirella, Phil Blunsom, and Alexander Koller, editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain, April. Association for Computational Linguistics.

Junczys-Dowmunt, Marcin, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In Liu, Fei and Thamar Solorio, editors, *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia, July. Association for Computational Linguistics.

Junczys-Dowmunt, Marcin. 2018. Dual conditional cross-entropy filtering of noisy parallel corpora. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 888–895, Belgium, Brussels, October. Association for Computational Linguistics.

Khayrallah, Huda and Philipp Koehn. 2018. On the impact of various types of noise on neural machine translation. In Birch, Alexandra, Andrew Finch, Thang Luong, Graham Neubig, and Yusuke Oda, editors, *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia, July. Association for Computational Linguistics.

Kocmi, Tom and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In Mitkov, Ruslan and Galia Angelova, editors, *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 379–386, Varna, Bulgaria, September. INCOMA Ltd.

Koehn, Philipp, 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 Ananiadou, Sophia, editor, *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, June. Association for Computational Linguistics.

Koehn, Philipp, Huda Khayrallah, Kenneth Heafield, and Mikel L. Forcada. 2018. Findings of the WMT 2018 shared task on parallel corpus filtering. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 726–739, Belgium, Brussels, October. Association for Computational Linguistics.

Koehn, Philipp, Francisco Guzmán, Vishrav Chaudhary, and Juan Pino. 2019. Findings of the WMT 2019 shared task on parallel corpus filtering for low-resource conditions. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 54–72, Florence, Italy, August. Association for Computational Linguistics.

Koehn, Philipp, Vishrav Chaudhary, Ahmed El-Kishky, Naman Goyal, Peng-Jen Chen, and Francisco Guzmán. 2020. Findings of the WMT 2020 shared task on parallel corpus filtering and alignment. In Barrault, Loïc, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 726–742, Online, November. Association for Computational Linguistics.

Lannelongue, Loïc, Jason Grealey, and Michael Inouye. 2021. Green algorithms: Quantifying the carbon footprint of computation. *Advanced Science*, 8(12):2100707.

Mohiuddin, Tasnim, Philipp Koehn, Vishrav Chaudhary, James Cross, Shruti Bhosale, and Shafiq Joty. 2022. Data selection curriculum for neural machine translation. In Goldberg, Yoav, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1569–1582, Abu Dhabi, United Arab Emirates, December. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In Isabelle, Pierre, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Platanios, Emmanouil Antonios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In Burstein, Jill, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Popović, Maja. 2015. chrF: character n-gram F-score for automatic MT evaluation. In Bojar, Ondřej, Rajan Chatterjee, Christian Federmann, Barry Haddow, Chris Hokamp, Matthias Huck, Varvara Logacheva, and Pavel Pecina, editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, September. Association for Computational Linguistics.

Post, Matt. 2018. A call for clarity in reporting BLEU scores. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo

Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, October. Association for Computational Linguistics.

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

Sánchez-Cartagena, Víctor M., Marta Bañón, Sergio Ortiz-Rojas, and Gema Ramírez-Sánchez. 2018. Prompsit's submission to wmt 2018 parallel corpus filtering shared task. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, Brussels, Belgium, October. Association for Computational Linguistics.

Schwenk, Holger and Matthijs Douze. 2017. Learning joint multilingual sentence representations with neural machine translation. In Blunsom, Phil, Antoine Bordes, Kyunghyun Cho, Shay Cohen, Chris Dyer, Edward Grefenstette, Karl Moritz Hermann, Laura Rimell, Jason Weston, and Scott Yih, editors, *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 157–167, Vancouver, Canada, August. Association for Computational Linguistics.

Spitkovsky, Valentin I., Hiyan Alshawi, and Daniel Jurafsky. 2010. From baby steps to leapfrog: How "less is more" in unsupervised dependency parsing. In Kaplan, Ron, Jill Burstein, Mary Harper, and Gerald Penn, editors, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 751–759, Los Angeles, California, June. Association for Computational Linguistics.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Vázquez, Raúl, Umut Sulubacak, and Jörg Tiedemann. 2019. The University of Helsinki submission to the WMT19 parallel corpus filtering task. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 294–300, Florence, Italy, August. Association for Computational Linguistics.

Wang, Wei, Isaac Caswell, and Ciprian Chelba. 2019. Dynamically composing domain-data selection with clean-data selection by "co-curricular learning" for neural machine translation. In Korhonen, Anna, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1282–1292, Florence, Italy, July. Association for Computational Linguistics.

Wang, Xin, Yudong Chen, and Wenwu Zhu. 2021. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):4555–4576.

Zaragoza-Bernabeu, Jaume, Gema Ramírez-Sánchez, Marta Bañón, and Sergio Ortiz Rojas. 2022. Bicleaner AI: Bicleaner goes neural. In Calzolari, Nicoletta, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 824–831, Marseille, France, June. European Language Resources Association.

Zhao, Mingjun, Haijiang Wu, Di Niu, and Xiaoli Wang. 2020. Reinforced curriculum learning on pre-trained neural machine translation models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9652–9659, Apr.

## A MarianNMT training parameters

```
seed: 0
devices: [0]
workspace: 25000
mini-batch-fit: True
shuffle-in-ram: True
no-restore-corpus: True
keep-best: True
save-freq: 1000
overwrite: True
disp-freq: 1000
disp-first: 10
quiet-translation: true
early-stopping: 5
valid-freq: 1000
valid-mini-batch: 64
valid-reset-stalled: true
valid-metrics:
    - bleu
beam-size: 6
normalize: 1
exponential-smoothing: 0.0001
max-length: 150
cost-type: ce-mean-words
type: transformer
enc-depth: 6
dec-depth: 6
dim-emb: 1024
transformer-heads: 16
transformer-dim-ffn: 4096
transformer-ffn-depth: 2
transformer-ffn-activation: swish
transformer-decoder-autoreg: self-attention
transformer-dropout: 0.1
label-smoothing: 0.1
layer-normalization: True
learn-rate: 0.0003
lr-warmup: 16000
lr-decay-inv-sqrt: 16000
lr-report: True
optimizer-params:
    - 0.9
    - 0.98
    - 1e-09
clip-norm: 1
sync-sgd: true
dim-vocabs:
    - 32000
    - 32000
tied-embeddings-all: true
```

## B BLEU and ChrF scores

Tables 6 and 7 show the BLEU and chrF scores for all translation experiments. The scores are computed with sacrebleu[8] (Post, 2018).

[8]Version signatures for BLEU and chrF respectively: `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.5.1`, `nrefs:1|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.5.1`

| | en-eu | eu-en | en-ga | ga-en | en-mk | mk-en | en-nn | nn-en | en-sw | sw-en |
|---|---|---|---|---|---|---|---|---|---|---|
| Near-dedup. | 6.1±0.2 | 20.4±0.8 | 2.3±0.1 | 27.4±0.7 | 12.4±0.9 | 31.9±1.1 | 3.1±0.4 | **30.3**±0.7 | 2.2±0.0 | 19.3±3.9 |
| Bicleaner AI | **14.9**±0.1 | **22.3**±0.2 | **29.5**±0.1 | **30.6**±0.5 | **31.6**±0.3 | **35.1**±0.3 | 20.1±0.7 | 25.8±0.6 | **29.4**±0.8 | 28.2±0.7 |
| OF Default | 14.3±0.2 | 22.0±0.2 | 28.0±0.5 | 29.3±0.9 | 30.4±0.4 | 34.0±0.2 | 2.5±0.1 | 3.5±0.1 | 28.4±0.6 | **29.2**±0.5 |
| OF Autogen | 4.7±1.2 | 13.5±7.1 | 26.9±1.4 | 28.0±2.2 | 28.0±6.8 | 34.0±0.7 | 13.9±7.8 | 21.8±8.4 | 23.6±9.6 | 27.1±3.3 |
| OF Adapt 1ep | 3.1±4.3 | 17.1±1.7 | 20.4±2.9 | 19.8±0.7 | 21.9±5.3 | 28.1±2.8 | 7.9±8.4 | 20.8±1.1 | 23.1±0.8 | 18.4±2.9 |
| OF Adapt Conv | 8.3±5.3 | 18.5±0.6 | 25.2±0.9 | 24.2±2.6 | 27.5±0.6 | 30.8±1.5 | **22.2**±0.6 | 26.8±3.6 | 19.9±8.9 | 17.4±3.9 |

**Table 6:** BLEU scores (mean±std) for the translation tasks using near-deduplicated data, Bicleaner AI, and OpusFilter (OF) based methods. The best scores per language pair are bolded.

| | en-eu | eu-en | en-ga | ga-en | en-mk | mk-en | en-nn | nn-en | en-sw | sw-en |
|---|---|---|---|---|---|---|---|---|---|---|
| Near-dedup. | 33.6±0.3 | 50.2±0.8 | 18.2±0.3 | 55.8±0.7 | 25.1±1.4 | 59.4±0.8 | 25.5±0.3 | **57.1**±0.7 | 16.8±0.0 | 44.9±4.9 |
| Bicleaner AI | **53.4**±0.3 | **52.0**±0.1 | **57.2**±0.1 | **58.2**±0.4 | **61.2**±0.2 | **61.9**±0.2 | 48.0±0.8 | 53.5±0.6 | **58.0**±0.5 | 53.6±0.6 |
| OF Default | 52.9±0.2 | **52.0**±0.2 | 55.8±0.4 | 57.2±0.8 | 60.6±0.3 | 61.0±0.2 | 22.7±0.6 | 25.2±0.2 | 57.2±0.4 | **54.6**±0.4 |
| OF Autogen | 34.6±1.6 | 41.5±8.4 | 55.1±1.1 | 56.0±1.2 | 54.9±13.0 | 61.0±0.6 | 39.2±11.4 | 47.5±10.8 | 50.1±14.9 | 52.8±3.2 |
| OF Adapt 1ep | 22.3±12.9 | 46.6±1.8 | 49.0±2.9 | 47.5±0.7 | 50.7±7.5 | 56.2±2.7 | 27.3±15.1 | 48.0±1.2 | 52.9±0.7 | 44.4±3.3 |
| OF Adapt Conv | 37.7±17.1 | 48.1±0.5 | 53.8±0.7 | 52.3±2.7 | 58.1±0.5 | 58.4±1.1 | **50.3**±0.7 | 53.8±3.4 | 46.2±16.4 | 42.7±4.8 |

**Table 7:** ChrF scores (mean±std) for the translation tasks using near-deduplicated data, Bicleaned AI, and OpusFilter (OF) based methods. The best scores per language pair are bolded.