Co-funded by the European Union

HPLT: High Performance Language Technologies

Software for cleaning data sets

Deliverable number: 3.1

Version 1.0

Funded by the European Union's Horizon Europe search and innovation programme under grant agreement No 101070350 and from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant number 10052546] programme

Project details

Project Acronym:	HPLT
Project Full Title:	HPLT: High Performance Language Technologies
Year of the Call:	2021
Type of Action:	HORIZON-IA (Innovation Action)
Grant Number:	101070350
Project URL:	https://hplt-project.org

Report details

Software for cleaning datasets	
Lead author:	Gema Ramírez (Prompsit)
Contributing authors:	Jaume Zaragoza-Bernabeu (Prompsit) Marta Bañón (Prompsit)
Internal reviewers:	Andrey Kutuzov (UiO) Stephan Oepen (UiO)
Deliverable number:	3.1, OTHER (software)
Dissemination level:	Public (PU)
Contractual Delivery Date:	Feb 29, 2024
Actual Delivery Date:	Feb 29, 2024
Number of pages:	17

Document history

Version	Date	Changes
1.0	Feb 29, 2024	Original Submission

Abstract

This report provides a description of deliverable D3.1 – the software developed to clean monolingual and bilingual datasets in the HPLT project.

Contents

1 Executive summary **2**

1.1 Brief summary of the HPLT project 2

2 Introduction **3**

3 Monolingual Cleaning **4**

3.1 Software and pipeline to clean monolingual datasets 4

3.1.1 Monocleaner models 5

3.1.2 Filtering 6

3.2 Impact of cleaning on monolingual datasets 7

3.3 Monolingual cleaning code and future work 9

4 Bilingual Cleaning **10**

4.1 Software and pipeline to clean bilingual datasets 10

4.1.1 Bicleaner AI Models 11

4.2 Impact of cleaning on bilingual datasets 12

4.3 Bilingual cleaning code and future work 12

5 OpusCleaner **13**

1 Executive summary

Deliverable D3.1 describes the software used to clean HPLT datasets for the first data release. A different set of tools for monolingual and bilingual cleaning has been produced resulting in two different pipelines. All tools and pipelines have been made publicly available and will be enhanced for subsequent data releases. Support for 75 languages has been implemented in the monolingual cleaning pipeline and support for 9 new languages has been added to the existing 36 in the bilingual cleaning pipeline. Impact on both monolingual and bilingual datasets is huge as the raw data extracted from web crawls tends to be very repetitive and noisy.

1.1 Brief summary of the HPLT project

The EU-funded HPLT project applies high-performance computing to scale and advance language technologies. Taking advantage of recent advances in machine learning and astonishing storage capacities, it will create and process huge language data sets and produce language and translation models in a large number of languages. The resulting models will be tested from various angles to ensure smooth integration, high accuracy, and regulatory compliance concerning privacy, unwanted biases and ethical issues. The models and data sets will be a game changer in the language service market in the EU and beyond. The resulting models will be open, free and available from established language repositories for anyone interested in pursuing research or innovation projects.

The project, coordinated by the Charles University in Prague (CUNI), gathers partners from 5 different universities 2 HPC centers and a private NLP company from all around Europe.

CHARLES UNIVERSITY	UNIVERSITY OF OSLO	UNIVERSITY OF EDINBURGH	UNIVERSITY OF TURKU
UNIVERSITY OF HELSINKI	PROMPSIT	CESNET	SIGMA2

2 Introduction

HPLT data is derived from petabytes of web crawled data which tends to be very noisy, repetitive and full of unwanted content like explicit or machine generated content. Before becoming useful for translation or language model training, specific cleaning for monolingual and bilingual raw data needs to be applied, usually looking for a compromise between size and content quality.

Deliverable D3.1 belongs to work package 3 (WP3) and focuses on the software used to clean HPLT raw monolingual and parallel data. The report explains the cleaning pipelines and their impact on the HPLT data release. The pipeline used to clean monolingual data is described in section 3 while section 4 details the pipeline to clean parallel data. Both sections provide details on HPLT data before and after cleaning.

Cleaning datasets is also a common task as a pre-processing step to train machine translation and language models. Special software, OpusCleaner, is being produced to ease and address this task as part of this project. It includes most of the cleaning tools described in this report. Section 4.3 briefly describes this piece of software which is still under development.

The partners involved in this deliverable and the underlying tasks are:

- **Prompsit** for tasks 3.1 and 3.2 with responsibilities for the data pipelines, tool setups and also for the actual cleaning of data
- **UH** for tasks 3.1 and 3.2 mainly involved in shaping and applying parallel data filtering.
- **UiO, CUNI and UTU** for tasks 3.2 mainly involved in monolingual filtering and testing.

The software and tools reported in this deliverable are released through GitHub in several repositories which will be mentioned in the corresponding sections of this deliverable.

3 Monolingual Cleaning

3.1 Software and pipeline to clean monolingual datasets

This section describes the software used to clean HPLT *monolingual* datasets. Software specific for cleaning *bilingual* datasets is described in Chapter 4.

HPLT data is produced in several steps. The software described in this section is applied after text is extracted from crawled websites,[1] along with some metadata, particularly document and segment[2] boundaries and language identification at document level. The extracted text is subsequently processed with a set of cleaning tools aiming at: 1) performing fixes, mainly at character level, to improve text quality, 2) enriching documents with additional metadata which will be used to perform further filtering and 3) removing exact and near-duplicate documents.

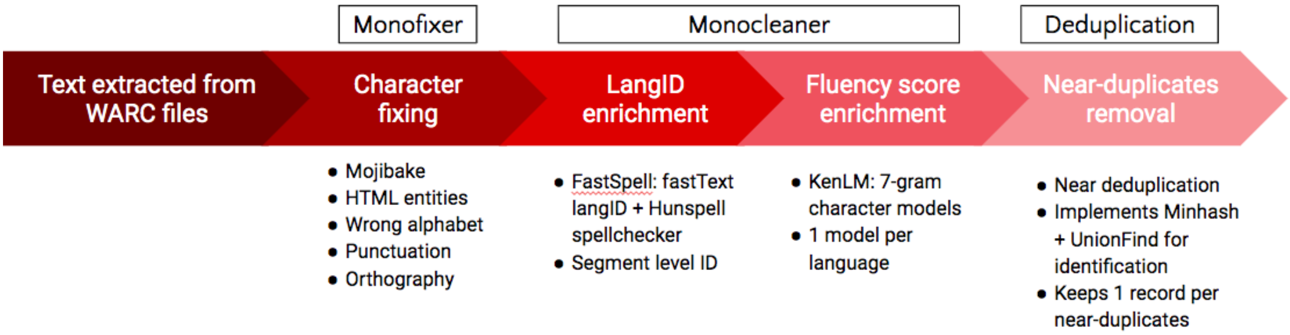

Figure 3.1: Monolingual cleaning pipeline flux diagram.

These tools, included in the HPLT monolingual cleaning pipeline (see figure 3.1), proceed as follows:

- Character and encoding fixes are applied using **Monofixer**.[3] For every paragraph in each document, Monofixer will try to fix issues in the content by:
  - Fixing mojibake (encoding errors).
  - Unescaping HTML entities.
  - Removing HTML tags.
  - Fixing common orthographic errors for Danish, German, English, Spanish, Dutch, Norwegian, Portuguese and Turkish. This is a default feature that will be removed in future releases.
- Metadata enrichment for language identification at paragraph level is produced by **FastSpell**.[4] FastSpell implements fastText for language identification which is then refined using Hunspell dictionaries for improved precision. This refinement consists in checking spelling errors with each Hunspell dictionary in a list of similar[5] languages to the one identified by fastText. The language

[1] HPLT extracts text from WARC files, a standard storage format for web crawled pages
[2] Segment roughly corresponds to HTML `<p> tags`
[3] https://github.com/bitextor/bifixer/tree/v0.8.8
[4] https://github.com/mbanon/fastspell/tree/v0.8
[5] https://github.com/mbanon/fastspell/blob/main/src/fastspell/config/similar.yaml

among the similar ones whose dictionary produces less spelling errors, is the final prediction.

- Metadata enrichment for fluency scores is computed by **Monocleaner**.[6] Monocleaner gets fluency scores computed with a 7-gram modified Knesser-Ney character language model. Fluency scores computed by Monocleaner can be used to estimate the 'quality' of paragraphs in the document, allowing to filter out noise which is detrimental for training language models.
- Near-duplicate documents are detected and removed using the **MinHash** algorithm [1]. There are two main steps in our deduplication algorithm:
  1. Identifying duplicate documents: Each JSON document is tokenized before computing MinHash signatures. Hashes are inserted in an index and clustered computing UnionFind. Clusters array is saved in a file.
  2. Removing duplicate documents: the clusters array is loaded into memory and then, documents are read sequentially. Those that belong to a cluster or are not the parent of a cluster, are discarded. Conversely, all documents that do not have near-duplicates and one document for each near-duplicates cluster, are kept.

  In our pipeline, the MinHash algorithm is configured to detect near-duplicate documents that have 0.8 Jaccard similarity or higher.

  To be able to fit in memory very large amounts of hashes, deduplication can be performed with a distributed index using multiple jobs. This allows to store and near-deduplicate the largest of the produced datasets, English and Chinese (12 and 7 billion documents respectively), before deduplication.

3.1.1 Monocleaner models

As explained above, for monolingual data enrichment, we compute fluency scores which rely on language-specific language models.

One language model was trained for each of the 75 languages in the HPLT collection. These **75 language models** were trained on samples of about 200,000 sentences, mostly coming from the monolingual part of OPUS corpora [2]. Samples were selected to match two criteria: not coming from web-crawls, and not having been automatically language identified. For languages with insufficient OPUS monolingual data, we added data from Wikipedia dumps. Given the simplicity of training these models and the availability of data, other languages can be easily added. All the models were released as part of the Monocleaner language data package.[7]

In order to obtain fluency scores for each paragraph, the computed perplexity of the paragraph is normalized. This normalization is achieved by taking three values into account:

- Upper limit: clean text average perplexity plus standard deviation.
- Lower limit: noisy text average perplexity minus standard deviation.
- Middle point: perplexity value in the middle between noisy and clean averages.

[6] https://github.com/bitextor/monocleaner/tree/v1.3.0
[7] https://github.com/bitextor/monocleaner-data/releases/tag/v1.0

This 'clean text' is text from the training set, while 'noisy text' is the same text but with scrambled characters. Then, during processing, each paragraph gets a perplexity value normalized according to this formula:

$$score = \begin{cases} 0, & perplexity > upper \\ \frac{perplexity}{(upper-middle)}, & perplexity > middle \\ \frac{perplexity}{(middle-lower)}, & perplexiti \leq middle \\ 1, & perplexity < lower \end{cases}$$

3.1.2 Filtering

As a result of the cleaning processing, two version of the corpus are obtained: the `raw` version, which contains all the original documents but fixed and enriched with metadata coming from the pre-processing and cleaning pipelines, and the `deduplicated` version, which is the same as the `raw` one but without the duplicate and near-duplicate documents.

Additionally, having in mind potential users that might take advantage of a conservative but cleaner version of the corpus, the so-called `cleaned` version is created (see the details of available versions in the HPLT website in Figure 3.2. We apply supplementary filtering to the `deduplicated` version. For this, we use the metadata computed by the monolingual cleaning pipeline and also apply other common practices in the production of monolingual corpora. This filtering step removes documents matching any of the criteria below:

- URL is in UT1 blacklist of adult sites[8]
- contains less than 200 characters
- contains less than 5 segments (paragraphs)
- average words per segment is less than 5
- less than 20% of its segments match the language identified at document level

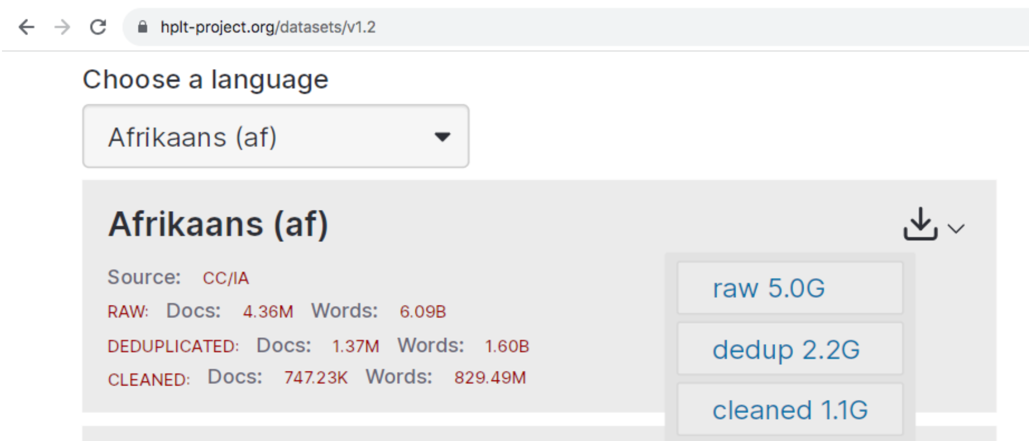

Figure 3.2: Details of the three versions produced for each language as part of the first release of the HPLT monolingual dataset.

[8] https://dsi.ut-capitole.fr/blacklists/index_en.php

The three versions of the corpora (raw, deduplicated and cleaned) are delivered in exactly the same JSON-lines (JSONL) format, having each document serialized as a JSON entity that contains the following fields as metadata (see an example in Listing 3.1)

- Document identification number.
- Document language, as identified by CLD2[9] during the WARC extraction process.
- Document URL.
- Collection name.
- All the paragraphs of the document, joined together with end line/paragraph separators and processed by Monofixer.
- For each paragraph, the language identified by FastSpell.
- For each paragraph, the fluency score obtained with Monocleaner.

Certain applications or users might have different requirements regarding monolingual corpora, so further cleaning and/or filtering can be applied to any of these 3 curated versions of the HPLT release. By releasing these three versions with permissive licenses, as well as free/open-source cleaning tools, we maximize the reproducibility of the processing carried out by the HPLT consortium, while maintaining the data as similar as possible to the original data. This way, we hope to encourage efforts that might lead to better processing in the future.

```

{"id":1, "document_lang":"en",
    "scores":[0.76,0.78,0.79],
    "langs":["en","en","en"],
    "text":"this is paragraph 1\nthis is paragraph 2\nthis is paragraph 3",
    "url":"url1", "collection":"collection-1"
}
{"id":2, "document_lang":"en",
    "scores":[0.65,...],
    "langs":["en",...],
    "text":"this is another paragraph\n...",
     ...
}
...
```

Listing 3.1: JSONL example

3.2 Impact of cleaning on monolingual datasets

The cleaning pipeline reduces the size of the raw data to almost one sixth of its original size on average. Deduplication reduces most languages to a third of its size and filtering downsizes the deduplicated version also to a half.

[9] https://github.com/CLD2Owners/cld2

Language	Raw		De-duplicated		Filtered	
Code	# Words	# Docs	# Words	# Docs	# Words	# Docs
af	6.1G	4.4M	1.7G	1.4M	830M	748K
ar	217G	198M	50G	47M	32G	27M
az	11G	11M	2.9G	3.0M	1.2G	1.1M
be	4.8G	4.0M	2.0G	1.3M	395M	357K
bg	67G	59M	16G	14M	8.8G	6.5M
bn	47G	24M	4.9G	6.0M	2.8G	2.9M
ca	25G	25M	7.9G	7.8M	5.8G	4.6M
cs	193G	185M	37G	39M	20G	17M
cy	624M	726K	235M	286K	125M	112K
da	140G	687M	23G	24M	9.4G	8.2M
de	900G	1.1G	191G	227M	111G	102M
el	249G	135M	50G	31M	34G	16M
en	11T	13G	2.9T	1.8G	2.4T	1.1G
eo	296M	401K	153M	178K	102M	68K
es	870G	672M	240G	202M	182G	130M
et	22G	22M	6.6G	5.9M	1.8G	1.5M
eu	1.6G	2.3M	661M	1.1M	325M	344K
fa	319G	190M	58G	43M	48G	31M
fi	81G	90M	20G	20M	9.1G	7.2M
fr	792G	661M	174G	176M	123G	100M
ga	1.5G	2.8M	520M	932K	131M	116K
gl	5.1G	4.6M	1.3G	1.8M	848M	732K
gu	1.1G	916K	431M	455K	304M	265K
hbs	70G	61M	18G	18M	11G	8.7M
he	62G	47M	15G	12M	7.5G	5.0M
hi	42G	34M	15G	12M	7.6G	5.8M
hu	138G	138M	29G	29M	15G	12M
hy	4.1G	4.0M	1.3G	1.4M	590M	622K
id	209G	126M	55G	46M	43G	32M
is	3.5G	3.9M	1.6G	1.5M	563M	482K
it	406G	338M	116G	97M	75G	54M
ja	306G	680M	78G	219M	64G	191M
ka	7.2G	6.5M	1.7G	1.7M	574M	534K
kk	2.6G	3.5M	1.1G	1.5M	472M	407K
kn	2.2G	2.0M	493M	558K	236M	229K
ko	162G	249M	35G	45M	26G	32M
ky	264M	334K	153M	189K	102M	89K
la	15G	21M	3.9G	4.9M	295M	302K
lt	34G	33M	7.4G	7.4M	3.0G	2.8M
lv	28G	22M	5.9G	5.2M	1.6G	1.6M
mk	3.3G	3.5M	1.1G	1.3M	737M	735K
ml	2.7G	2.2M	918M	1.2M	518M	470K
mn	2.5G	2.6M	1.1G	1.1M	804M	595K
mr	2.0G	1.7M	813M	858K	520M	454K
ms	57G	29M	14G	8.4M	9.1G	4.9M
mt	1.4G	927K	819M	485K	103M	112K
my	4.5G	2.5M	1.2G	827K	358M	240K
nb	56G	62M	17G	15M	8.4G	6.2M
ne	1.7G	2.2M	967M	1.4M	695M	864K
nl	251G	235M	56G	67M	34G	32M
nn	1.7G	1.9M	616M	753K	299M	229K
pa	1.4G	2.4M	524M	889K	185M	153K
pl	367G	347M	77G	83M	45G	40M
ps	357M	314K	173M	143K	114M	89K
pt	608G	449M	122G	104M	82G	59M
ro	145G	104M	29G	25M	20G	15M

Language	Raw		De-duplicated		Filtered	
Code	# Words	# Docs	# Words	# Docs	# Words	# Docs
ru	1.8T	1.6G	414G	398M	285G	225M
si	2.4G	1.4M	735M	564K	569M	323K
sk	95G	91M	15G	14M	5.0G	4.7M
sl	23G	20M	6.8G	5.9M	2.6G	2.2M
so	545M	677K	254M	375K	212M	284K
sq	12G	9.2M	3.7G	3.3M	1.4G	1.3M
sv	96G	97M	30G	30M	17G	14M
sw	2.2G	2.2M	831M	984K	669M	699K
ta	9.3G	5.5M	3.0G	2.5M	2.0G	1.3M
te	2.5G	3.6M	1.1G	1.7M	438M	416K
th	79G	96M	17G	30M	4.4G	8.2M
tl	7.1G	5.0M	1.7G	1.3M	912M	586K
tr	239G	216M	65G	60M	43G	28M
tt	262M	369K	135M	173K	75M	66K
uk	53G	48M	19G	18M	11G	9.4M
ur	6.6G	6.1M	2.1G	2.3M	1.5G	1.5M
uz	1.2G	1.4M	557M	634K	368M	291K
vi	288G	175M	60G	41M	50G	32M
zh	1.8T	7.0G	483G	1.3G	433G	1.1G

Table 3.1: Statistics on the extracted bitexts without filtering (Raw), after de-duplication (De-duplicated) and after cleaning (Filtered).

3.3 Monolingual cleaning code and future work

Links to the code and models for the individual tools involved in monolingual cleaning have been provided in the previous sections. These are organised in a pipeline that has also been published on Github.[10] In the pipeline, Monofixer + FastSpell + Fluency scores steps belong to the `10.processing` script, near-deduplication to `20.dedup` and cleaning filters to `30.clean`.

After producing the first release of HPLT corpora, the major issues observed in the obtained corpora will be addressed before running a second iteration: in pre-processing, for example, better language identification and boilerplate removal are being explored. Thanks to this, we expect to have an improved input for the cleaning pipeline which we aim to improve with two main objectives:

- Getting further metadata at different content levels (document, paragraph) to be able to better determine the document quality.
- Exploring a more aggressive filtering for the `cleaned` version to produce ready-to-use subsets of the datasets.

The second iteration will include at least three times more data and an increased number of languages. Further adjustments to the tools and the pipeline are expected to cope with scalability and language coverage needs.

[10] https://github.com/hplt-project/monotextor-slurm/tree/v1.0

4 Bilingual Cleaning

4.1 Software and pipeline to clean bilingual datasets

In this section, we address the tools and pipeline used to clean the parallel datasets produced in HPLT.

The input for the bilingual cleaning tools, derived from a complex pipeline described in Deliverable 2.1,[1] is parallel sentences. These are pairs of sentences in two different languages, one being potentially a translation of the other. These sentences are stored with additional metadata, such as the source URLs (the URL to which each sentence belongs) and the collection containing these URLs.

Bilingual cleaning involves the following tools or steps, as presented in the cleaning pipeline:

1. **Bifixer** [3]: it fixes encoding and orthographic issues, similar to Monofixer (for monolingual text data, described in 3). It also computes hashes for sentence pairs and scores the duplicate ones according to textual quality after ignoring punctuation.
2. **Bicleaner-hardrules** [3]: it removes noisy sentence pairs by looking for obvious noise based on rules, poor language identification (by using FastSpell as a second-opinion classifier) and vulgar language (based on specific language modelling).
3. **Bicleaner AI** [4]: it gives sentence pairs a score that indicates the likelihood of its sentences to be mutual translations (with a value near to 1) or not (with a value near to 0). We keep sentence pairs that obtained a Bicleaner score above 0.5.
4. **De-duplication and TMX Formatting**: the final step generates a TMX file[2]. In this step, the sentence pairs are de-duplicated, ignoring differences in punctuation. The source URLs are retained and joined so that a single sentence pair can have multiple URLs, identifying all the documents where it occurred.
5. **Biroamer** [3]: it ROAMizes (Random, Omit, Anonymize and Mix) parallel corpora, although for this project only the anonymization component is used. It also removes all parallel sentences in which personal identifiable information (PII) has been found. This includes the 'PER' (person) entity as identified by a named entity recognition system on the English side and e-mails, IP addresses or phone numbers detected in one of both sides using regular expressions.

This set of tools and pipeline produces three versions of the bilingual datasets: the `raw` version, the deduplicated and filtered version in two formats (`tmx` and `moses`), and the anonymized version (`roam`) in TMX format. The `raw` contains the source URLs, the Bifixed sentence pairs and their corresponding Bifixer hashes and scores, along with Bicleaner scores and collection name.[4] The `tmx` deduplicated and filtered version (see figure 4.1) contains the same information as the raw file for the remaining sentence pairs after removing near-duplicates, but omitting Bifixer hashes and scores. Some additional information regarding sentence pair quality, particularly length ratio and number mismatching, is also present following the recommendations of the European Language Resource Coordination (ELRC)

[1] https://hplt-project.org/HPLT_D2_1___Initial_release_of_monolingual_and_parallel_data_sets-1.pdf
[2] https://xml.coverpages.org/tmxSpec971212.html
[3] https://github.com/bitextor/biroamer/releases/tag/v2.1.0
[4] Bicleaner scores are missing from HPLT 1.1 raw version due to time constraints, but the current pipeline adds them to the raw file

guidelines.[5] The `moses` versions contain the same sentences as the `tmx` versions, but omitting all metadata and splitting parallel sentences in two files (one per language). These two files are parallel, that is, they have the same number of lines, and each sentence in one of the files is parallel to the sentence in the same line of the other file. The `roam` version contains the same sentences as the filtered versions but without sentences where PII was detected during the anonymization procedure. This file is provided in the same TMX format as the `tmx` file, but containing only the remaining sentences and without any metadata.


```xml
<tu tuid="154" datatype="Text">
 <prop type="collection">wide15</prop>
 <prop type="collection">wide17</prop>
 <prop type="score-bicleaner">0.947</prop>
 <prop type="type">1:1</prop>
 <tuv xml:lang="en">
  <prop type="source-document">http://www.laerdalferiepark.com/en/press-and-awards/presse-og-omtaler</prop>
  <prop type="source-document">http://www.laerdalferiepark.com/en/press-and-awards/presse-og-omtaler-2</prop>
  <prop type="source-document">http://www.laerdalferiepark.com/en/press-and-awards</prop>
  <prop type="source-document">http://www.laerdalferiepark.com/en/press-and-awards/2</prop>
  <prop type="checksum-seg">d05a2604449da1ad</prop>
  <seg>We always aim to be better at what we do!</seg>
 </tuv>
 <tuv xml:lang="nn">
  <prop type="source-document">http://www.laerdalferiepark.com/presse-og-omtaler/2</prop>
  <prop type="source-document">http://www.laerdalferiepark.com/presse-og-omtaler/presse-og-omtaler-2</prop>
  <prop type="source-document">http://www.laerdalferiepark.com/presse-og-omtaler</prop>
  <prop type="source-document">http://www.laerdalferiepark.com/presse-og-omtaler/presse-og-omtaler</prop>
  <prop type="checksum-seg">4f44fc6456abe7c5</prop>
  <seg>Vårt mål er alltid å bli betre!</seg>
 </tuv>
```


Figure 4.1: Example of a sentence pair in the deduplicated and filtered bilingual TMX file.

4.1.1 Bicleaner AI Models

As explained in the previous section, for bilingual data filtering we use Bicleaner AI, which relies on per-language-pair trained classification models [4].

Although there were Bicleaner models already available for most of the languages covered by the HPLT v1.2 release, we trained new Bicleaner models for the following languages (paired with English): `ar`, `ca`, `eu`, `gl`, `he`, `hi`, `ja`, `sw`, `uk`, `vi`, and `zh`. We have, therefore, increased the total amount of available language pairs in Bicleaner AI from 36 to 45,[6]. Training the new models implied many changes and improvements to the tool since its development for the ParaCrawl project[7]. All the newly trained Bicleaner AI models are available for download.[8]

[5] ELRC Guidelines are described in section 5.1 on the following document: http://www.elra.info/media/filer_public/2017/09/27/europeanlanguageresourcecoordinationelrcfinalreportapril2015-april2017.pdf
[6] https://huggingface.co/models?other=bicleaner-ai
[7] https://github.com/bitextor/bicleaner-ai/blob/v2.3.2/CHANGELOG.md
[8] https://github.com/bitextor/bicleaner-ai#download-a-model

4.2 Impact of cleaning on bilingual datasets

The bilingual cleaning pipeline reduces considerably the number of sentence pair candidates that get into it. As shown in table 4.1, cleaning with Bicleaner-hardrules and filtering by 0.5 Bicleaner AI scores reduces the total number of raw sentence pair candidates to 12% and deduplication to 8%. Reduction varies a lot depending on languages as some have noisier sentence pair candidates and other much more duplicates.

	Raw		Filtered		Deduplicated	
Language Pair	**# Segments**	**# Tokens**	**# Segments**	**# Tokens**	**# Segments**	**# Tokens**
Norwegian (nn)	29M	497M	0.7M	7M	0.2M	2.1M
Bosnian* (bs)	27M	522M	1.5M	13M	0.3M	2.8M
Basque (eu)	21M	401M	3.1M	32M	0.7M	10.0M
Maltese (mt)	136M	2,821M	9.2M	134M	0.9M	18.9M
Gaelic (ga)	102M	2,014M	15.7M	145M	1.0M	16.4M
Galician (gl)	57M	1,016M	5.8M	50M	1.1M	14.0M
Macedonian (mk)	92M	1,869M	20.5M	222M	1.2M	18.6M
Albanian (sq)	254M	5,820M	16.8M	145M	1.7M	25.9M
Swahili (sw)	248M	5,747M	24.5M	210M	1.8M	20.1M
Icelandic (is)	171M	3,267M	28.2M	263M	2.2M	29.5M
Serbian (sr)	755M	14,250M	60.5M	587M	4.7M	67.1M
Chinese (zh)	531M	9,163M	47.9M	511M	5.4M	83.9M
Estonian (et)	866M	15,477M	73.0M	753M	6.1M	96.0M
Catalan (ca)	403M	8,035M	88.5M	883M	9.0M	141.9M
Croatian* (hr)	896M	16,566M	128.2M	1,166M	9.4M	138.4M
Hindi (hi)	1,044M	19,247M	117.4M	997M	12.1M	165.2M
Arabic (ar)	1,546M	33,200M	277.9M	2,308M	14.7M	239.4M
Finnish (fi)	3,827M	65,313M	495.4M	4,187M	25.2M	338.1M
Total	10,996M	205,214M	1,414.0M	12,605M	96.7M	1,427.4M

Table 4.1: Statistics on the extracted bitexts without filtering (Raw), after cleaning (Filtered) and after de-duplication (De-duplicated) ordered by available clean de-duplicated segments. All statistics are measured from the English side of each language pair. The symbol * indicates that a joint Bicleaner AI model has been used for processing those languages.

4.3 Bilingual cleaning code and future work

Code for the bilingual cleaning tools used to produce HPLT datasets can be found in the Bitextor[9] project repositories on Github:

- Bifixer: `https://github.com/bitextor/bifixer/releases/tag/v0.8.9`
- Bicleaner-hardrules: `https://github.com/bitextor/bicleaner-hardrules/releases/tag/v2.9.0`
- Bicleaner AI: `https://github.com/bitextor/bicleaner-ai/releases/tag/v2.3.2`
- Bicleaner AI models: `https://github.com/bitextor/bicleaner-ai/#download-a-model`

[9] `https://github.com/bitextor`

These are included in the Bitextor-based pipeline used in HPLT which has been adapted from a previous pipeline, the ParaCrawl project one, to run on the LUMI cluster. The pipeline is available on Github.[10]

Regarding future work, we are now exploring how to scale Bicleaner AI support going from a per-language based approach to a multilingual one. We are also exploring further filtering of explicit content based on not only the language-model-based classifier but also based on lists of URLs or rule-based heuristics. Finally, we are also exploring futher filters to improve the cleaning of badly aligned sentences and moving deduplication to a much earlier step of the processing pipeline to avoid expensive processing to be applied to duplicate content.

5 OpusCleaner

OpusCleaner[1] is a data downloading, cleaning, and preprocessing toolkit. It is designed to allow researchers to quickly download, visualise and preprocess datasets that come from many different sources, each of them with different quality, issues, and unique filtering/preprocessing requirements.

OpusCleaner, currently under development as part of HPLT Work Package 5 includes, as shown in Figure 5.1 most of the cleaning software tools described in this report for bilingual cleaning complemented with additional filters from OpusFilter[2] and some other original ones. Currently in a usable status for bilingual filtering, it will be extended to process monolingual data including the filters and tools already explored in HPLT when suitable.

Unlike the pipeline style followed by the current approach to cleaning, OpusCleaner is conceived as a web interface that to ease the selection, cleaning and scheduling of data for training machine translation models and language models. It will be further described in future deliverables.

[10] https://github.com/paracrawl/cirrus-scripts
[1] https://github.com/hplt-project/OpusCleaner
[2] https://github.com/Helsinki-NLP/OpusFilter

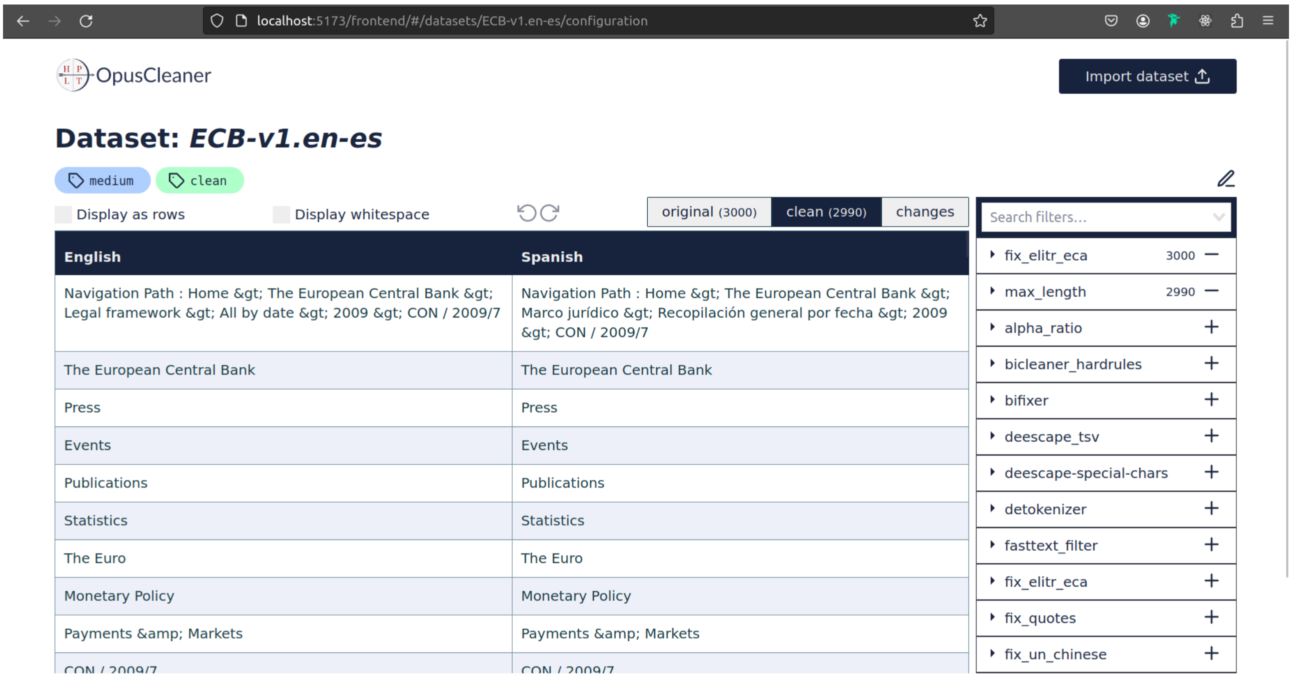

Figure 5.1: Example of the OpusCleaner filtering interface including cleaning tools such as Bifixer or Bicleaner-hardrules, among others.

Bibliography

[1] A. Z. Broder, "Identifying and filtering near-duplicate documents," in *Annual Symposium on Combinatorial Pattern Matching*, 2000. [Online]. Available: https://api.semanticscholar.org/CorpusID:2865406

[2] J. Tiedemann, "Parallel data, tools and interfaces in OPUS," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, N. C. C. Chair), K. Choukri, T. Declerck, M. U. Dogan, B. Maegaard, J. Mariani, J. Odijk, and S. Piperidis, Eds. Istanbul, Turkey: European Language Resources Association (ELRA), may 2012.

[3] G. Ramírez-Sánchez, J. Zaragoza-Bernabeu, M. Bañón, and S. Ortiz-Rojas, "Bifixer and bicleaner: two open-source tools to clean your parallel data." in *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*. Lisboa, Portugal: European Association for Machine Translation, November 2020, pp. 291–298.

[4] J. Zaragoza-Bernabeu, G. Ramírez-Sánchez, M. Bañón, and S. Ortiz Rojas, "Bicleaner AI: Bicleaner goes neural," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, Jun. 2022, pp. 824–831. [Online]. Available: https://aclanthology.org/2022.lrec-1.87