

SORBONNE UNIVERSITÉ

ECOLE DOCTORALE INFORMATIQUE, TÉLÉCOMMUNICATIONS ET ÉLECTRONIQUE - ED130

INRIA DE PARIS / ÉQUIPE ALMANACH

THÈSE DE DOCTORAT

Discipline : Informatique

Présentée par

Pedro ORTIZ SUAREZ

Dirigée par

Laurent ROMARY et Benoît SAGOT

Pour obtenir le grade universitaire de

DOCTEUR de SORBONNE UNIVERSITÉ

On Language Modeling and its Applications for Contemporary and Historical French

Présentée et soutenue publiquement le 30 avril 2022 devant le jury composé de :

Francis BACH	Inria - SIERRA	Examineur
Alexander GEYKEN	Berlin Academy	Examineur
Sebastian PADÓ	University of Stuttgart	Examineur
Barbara PLANK	IT University of Copenhagen	Rapporteur
Laurent ROMARY	Inria - ALMANACH	Directeur
Benoît SAGOT	Inria - ALMANACH	Directeur
Holger SCHWENK	Facebook AI Research	Rapporteur
Achim STEIN	University of Stuttgart	Examineur

ABSTRACT

Scientific documents often use \LaTeX for typesetting. While numerous packages and templates exist, it makes sense to create a new one. Just because.

CONTENTS

1	INTRODUCTION	1
1.1	Why?	1
1.2	How?	1
1.3	Features	2
1.3.1	Typesetting mathematics	2
1.3.2	Typesetting text	3
1.4	Changing things	3
I	DATA	5
2	OSCAR	7
2.1	goclassy	7
2.2	Introduction	8
2.3	Related Work	9
2.4	Common Crawl	10
2.5	fastText’s Pipeline	11
2.6	Asynchronous pipeline	12
2.7	Benchmarks	14
2.8	OSCAR	15
2.9	Conclusions	16
2.10	Introduction	18
2.11	Related Work	18
2.12	Multilingual Corpora	19
2.13	Auditing Data Quality	21
2.13.1	Auditing Process	21
2.13.2	Human Audit Results	23
2.13.3	Automatic Filtering	25
2.14	Dataset Mis-labeling	27
2.15	Risks of Low-Quality Data	27
2.16	Future Work	28
2.17	Conclusion & Recommendations	28
2.18	Details on Language Code Issues	29
2.19	Complete Error Taxonomy and Instructions	30
2.20	Non-proficient Rater Evaluation	37

2.21	Not-So-Parallel Data	38
2.22	Quality vs Size	38
2.23	Methodological Notes	38
2.24	Aspirational Error Taxonomy	39
2.25	Complete Tables	41
2.26	Limitations of the OSCAR Corpus and its Generation Pipeline	47
2.26.1	OSCAR	47
2.26.2	goclassy	49
2.27	Building a new OSCAR-like corpus	49
2.27.1	Ungoliant	49
2.27.2	Iterating on the goclassy pipeline	50
2.27.3	Characteristics of our new backward compatible OSCAR-like corpus	53
2.27.4	License	56
2.28	Conclusion	56
3	MODERN FRENCH DATA	59
3.1	LEM17	59
3.2	presto max	59
3.3	presto gold	59
4	ANCIENT/MEDIEVAL FRENCH DATA	61
4.1	BERTrade Corpus	61
5	OTHER DATA	63
II	MODELS	65
6	CAMEMBERT	67
7	FrELMo	69
8	D’ALEMBERT	71
9	BERTRADE	73
III	DOWNSTREAM TASKS	75
10	PARSING	77
11	POS TAGGING	79

12 NAMED-ENTITY RECOGNITION	81
13 TEXT NORMALIZATION	83
14 DOCUMENT STRUCTURATION	85
IV REAL WORLD APPLICATION	87
15 BASNUM	89
16 NAMED-ENTITY RECOGNITION CORPORA	91

1 INTRODUCTION

In which the reasons for doing this Ph.D. are laid bare for the whole world to see and we encounter some answers to questions in which, frankly, only an extremely small number of people were interested in the first place.

This package contains a minimal, modern template for writing your thesis. While originally meant to be used for a Ph. D. thesis, you can equally well use it for your honour thesis, bachelor thesis, and so on—some adjustments may be necessary, though.

1.1 WHY?

I was not satisfied with the available templates for L^AT_EX and wanted to heed the style advice given by people such as Robert Bringhurst or Edward R. Tufte . While there *are* some packages out there that attempt to emulate these styles, I found them to be either too bloated, too playful, or too constraining. This template attempts to produce a beautiful look without having to resort to any sort of hacks. I hope you like it.

1.2 How?

The package tries to be easy to use. If you are satisfied with the default settings, just add

```
\documentclass{mimosis}
```

at the beginning of your document. This is sufficient to use the class. It is possible to build your document using either L^AT_EX, X_YL^AT_EX, or LuaL^AT_EX. I personally prefer one of the latter two because they make it easier to select proper fonts.

Package	Purpose
<code>amsmath</code>	Basic mathematical typography
<code>amsthm</code>	Basic mathematical environments for proofs etc.
<code>booktabs</code>	Typographically light rules for tables
<code>bookmarks</code>	Bookmarks in the resulting PDF
<code>dsfont</code>	Double-stroke font for mathematical concepts
<code>graphicx</code>	Graphics
<code>hyperref</code>	Hyperlinks
<code>multirow</code>	Permits table content to span multiple rows or columns
<code>paralist</code>	Paragraph ('in-line') lists and compact enumerations
<code>scrlayer-scrpage</code>	Page headings
<code>setspace</code>	Line spacing
<code>siunitx</code>	Proper typesetting of units
<code>subcaption</code>	Proper sub-captions for figures

Table 1.1: A list of the most relevant packages required (and automatically imported) by this template.

1.3 FEATURES

The template automatically imports numerous convenience packages that aid in your typesetting process. [Table 1.1](#) lists the most important ones. Let's briefly discuss some examples below. Please refer to the source code for more demonstrations.

1.3.1 TYPESETTING MATHEMATICS

This template uses `amsmath` and `amssymb`, which are the de-facto standard for typesetting mathematics. Use numbered equations using the `equation` environment. If you want to show multiple equations and align them, use the `align` environment:

$$V := \{1, 2, \dots\} \tag{1.1}$$

$$E := \{(u, v) \mid \text{dist}(p_u, p_v) \leq \epsilon\} \tag{1.2}$$

Define new mathematical operators using `\DeclareMathOperator`. Some operators are already pre-defined by the template, such as the distance between two objects. Please see the template for some examples. Moreover, this template contains a correct differential operator. Use `\diff` to typeset the differential of integrals:

$$f(u) := \int_{v \in \mathbb{D}} \text{dist}(u, v) \, \mathrm{d}v \tag{1.3}$$

You can see that, as a courtesy towards most mathematicians, this template gives you the possibility to refer to the real numbers \mathbb{R} and the domain \mathbb{D} of some function.

Take a look at the source for more examples. By the way, the template comes with spacing fixes for the automated placement of brackets.

1.3.2 TYPESETTING TEXT

Along with the standard environments, this template offers `paralist` for lists within paragraphs. Here's a quick example: The American constitution speaks, among others, of (i) life (ii) liberty (iii) the pursuit of happiness. These should be added in equal measure to your own conduct. To typeset units correctly, use the `siunitx` package. For example, you might want to restrict your daily intake of liberty to 750 mg.

Likewise, as a small pet peeve of mine, I offer specific operators for *ordinals*. Use `\th` to typeset things like July 4th correctly. Or, if you are referring to the 2nd edition of a book, please use `\nd`. Likewise, if you came in 3rd in a marathon, use `\rd`. This is my 1st rule.

1.4 CHANGING THINGS

Since this class heavily relies on the `scrbook` class, you can use *their* styling commands in order to change the look of things. For example, if you want to change the text in sections to **bold** you can just use

```
\setkomafont{sectioning}{\normalfont\bfseries}
```

at the end of the document preamble—you don't have to modify the class file for this. Please consult the source code for more information.

PART I

DATA

2 OSCAR

2.1 GOCLASSY

2.2 INTRODUCTION

In recent years neural methods for Natural Language Processing (NLP) have consistently and repeatedly improved the state-of-the-art in a wide variety of NLP tasks such as parsing, PoS-tagging, named entity recognition, machine translation, text classification and reading comprehension among others. Probably the main contributing factor in this steady improvement for NLP models is the raise in usage of *transfer learning* techniques in the field. These methods normally consist of taking a pre-trained model and reusing it, with little to no retraining, to solve a different task from the original one it was intended to solve; in other words, one *transfers* the *knowledge* from one task to another.

Most of the transfer learning done in NLP nowadays is done in an unsupervised manner, that is, it normally consist of a *language model* that is fed unannotated plain text in a particular language; so that it *extracts* or *learns* the basic *features* and patterns of the given language, the model is subsequently used on top of an specialised architecture designed to tackle a particular NLP task. Probably the best known example of this type of model are *word embeddings* which consist of real-valued vector representations that are trained for each word on a given corpus. Some notorious examples of word embeddings are word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and fastText (Mikolov et al., 2018). All these models are *context-free*, meaning that a given word has one single vector representation that is independent of context, thus for a polysemous word like Washington, one would have one single representation that is reused for the city, the state and the US president.

In order to overcome the problem of polysemy, *contextual* models have recently appeared. Most notably ELMo (Peters et al., 2018) which produces deep contextualised word representations out of the internal states of a deep bidirectional language model in order to model word use and how the usage varies across linguistic contexts. ELMo still needs to be used alongside a specialised architecture for each given downstream task, but newer architectures that can be fine-tuned have also appear. For these, the model is first fed unannotated data, and is then fine-tuned with annotated data to a particular downstream task without relying on any other architecture. The most remarkable examples of this type of model are GPT-1, GPT-2 (Radford et al., 2018, 2019), BERT (Devlin et al., 2019) and XLNet (Yang et al., 2019); the latter being the current state-of-the-art for multiple downstream tasks. All of these models are different arrangements of the Transformer architecture (Vaswani et al., 2017) trained with different datasets, except for XLNet which is an instance of the Transformer-XL (Dai et al., 2019).

Even though these models have clear advantages, their main drawback is the amount of data that is needed to train them in order to obtain a functional and efficient model. For the first English version of word2vec, Mikolov et al. (2013) used a one billion word dataset consisting of various news articles. Later Al-Rfou' et al. (2013) and then Bojanowski et al. (2017) used the plain text from Wikipedia to train

distributions of word2vec and fastText respectively, for languages other than English. Now, the problem of obtaining large quantities of data aggravates even more for contextual models, as they normally need multiple instances of a given word in order to capture all its different uses and in order to avoid overfitting due to the large quantity of hyperparameters that these models have. Peters et al. (2018) for example use a 5.5 billion token¹ dataset comprised of crawled news articles plus the English Wikipedia in order to train ELMo, Devlin et al. (2019) use a 3.3 billion word² corpus made by merging the English Wikipedia with the BooksCorpus (Zhu et al., 2015), and Radford et al. (2019) use a 40GB English corpus created by scraping outbound links from Reddit.³

While Wikipedia is freely available, and multiple pipelines exist^{4,5} to extract plain text from it, some of the bigger corpora mentioned above are not made available by the authors either due to copyright issues or probably because of the infrastructure needed to serve and distribute such big corpora. Moreover the vast majority of both these models and the corpora they are trained with are in English, meaning that the availability of high quality NLP for other languages, specially for low-resource languages, is rather limited.

To address this problem, we choose Common Crawl⁶, which is a 20TB multilingual free to use corpus composed of crawled websites from the internet, and we propose a highly parallel multithreaded asynchronous pipeline that applies well-known concurrency patterns, to clean and classify by language the whole Common Crawl corpus to a point where it is usable for Machine Learning and in particular for neural NLP applications. We optimise the pipeline so that the process can be completed in a sensible amount of time even in infrastructures where Input/Output (I/O) speeds become the main bottleneck.

Knowing that even running our pipeline will not always be feasible, we also commit to publishing our own version of a classified by language, filtered and ready to use Common Crawl corpus upon publication of this article. We will set up an easy to use interface so that people can download a manageable amount of data on a desired target language.

2.3 RELATED WORK

Common Crawl has already been successfully used to train language models, even multilingual ones. The most notable example is probably fastText which was first trained for English using Common Crawl (Mikolov et al., 2018) and then for other 157

¹Punctuation marks are counted as tokens.

²Space separated tokens.

³<https://www.reddit.com/>

⁴<https://github.com/attardi/wikiextractor>

⁵<https://github.com/hghodrati/wikifil>

⁶<http://commoncrawl.org/>

different languages (Grave et al., 2018). In fact Grave et al. (2018) proposed a pipeline to filter, clean and classify Common Crawl, which we shall call the “fastText pre-processing pipeline.” They used the fastText linear classifier (Joulin et al., 2016; Joulin et al., 2017) to classify each line of Common Crawl by language, and downloaded the initial corpus and schedule the I/O using some simple Bash scripts. Their solution, however, proved to be a synchronous blocking pipeline that works well on infrastructures having the necessary hardware to assure high I/O speeds even when storing tens of terabytes of data at a time. But that downscales poorly to medium-low resource infrastructures that rely on more traditional cost-effective electromechanical mediums in order to store this amount of data.

Concerning contextual models, Baevski et al. (2019) trained a BERT-like bi-directional Transformer for English using Common Crawl. They followed the “fastText pre-processing pipeline” but they removed all copies of Wikipedia inside Common Crawl. They also trained their model using News Crawl (Bojar et al., 2018) and using Wikipedia + BooksCorpus, they compared three models and showed that Common Crawl gives the best performance out of the three corpora.

The XLNet model was trained for English by joining the BookCorpus, English Wikipedia, Giga5 (Parker et al., 2011), ClueWeb 2012-B (Callan et al., 2009) and Common Crawl. Particularly for Common Crawl, Yang et al. (2019) say they use “heuristics to aggressively filter out short or low-quality articles” from Common Crawl, however they don’t give any detail about these “heuristics” nor about the pipeline they use to classify and extract the English part of Common Crawl.

It is important to note that none of these projects distributed their classified, filtered and cleaned versions of Common Crawl, making it difficult in general to faithfully reproduce their results.

2.4 COMMON CRAWL

Common Crawl is a non-profit foundation which produces and maintains an open repository of web crawled data that is both accessible and analysable.⁷ Common Crawl’s complete web archive consists of petabytes of data collected over 8 years of web crawling. The repository contains raw web page HTML data (WARC files), metadata extracts (WAT files) and plain text extracts (WET files). The organisation’s crawlers has always respected `nofollow`⁸ and `robots.txt`⁹ policies.

Each monthly Common Crawl snapshot is in itself a massive multilingual corpus, where every single file contains data coming from multiple web pages written in a large variety of languages and covering all possible types of topics. Thus, in order to effectively use this corpus for the previously mentioned Natural Language Processing

⁷<http://commoncrawl.org/about/>

⁸<http://microformats.org/wiki/rel-nofollow>

⁹<https://www.robotstxt.org/>

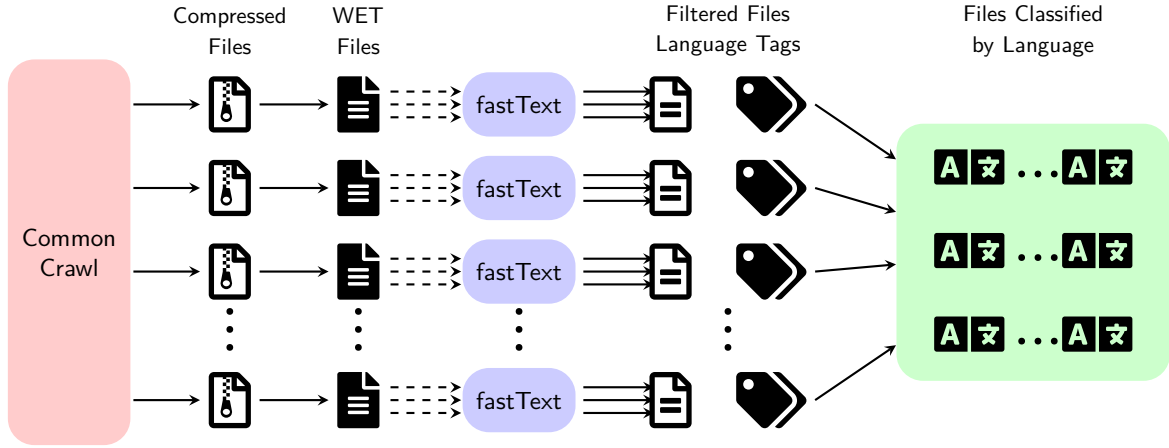








Figure 2.1: A scheme of the *goclassy* pipeline. The red square represents the Compressed WET files stored on Amazon Web Services. The  icons represent the gzip files stored locally, the  represent one of the 50K WET files. The  represents the filtered file and the  represents a file of language tags, one tag per line in . The  represents one of the 166 classified files. Each arrow represents an asynchronous non blocking worker and dotted arrows represent a line filtering process.

and Machine Learning applications, one has first to extract, filter, clean and classify the data in the snapshot by language.

For our purposes we use the WET files which contain the extracted plain texts from the websites mostly converted to UTF-8, as well as headers containing the metadata of each crawled document. Each WET file comes compressed in gzip format¹⁰ and is stored on Amazon Web Services. We use the November 2018 snapshot which surpasses 20TB of uncompressed data and contains more than 50 thousand plain text files where each file consists of the plain text from multiple websites along its metadata header. From now on, when we mention the “Common Crawl” corpus, we refer to this particular November 2018 snapshot.

2.5 FASTTEXT’S PIPELINE

In order to download, extract, filter, clean and classify Common Crawl we base ourselves on the “fastText pre-processing pipeline” used by Grave et al. (2018). Their pipeline first launches multiple process, preferably as many as available cores. Each of these processes first downloads one Common Crawl WET file which then proceeds to decompress after the download is over. After decompressing, an instance of the fastText linear classifier (Joulin et al., 2016; Joulin et al., 2017) is launched, the classifier processes each WET file line by line, generating a language tag for each

¹⁰<https://www.gnu.org/software/gzip/>

line. The tags are then stored in a tag file which holds a one-to-one correspondence between lines of the WET file and its corresponding language tag. The WET file and the tag files are read sequentially and each on the WET file line holding the condition of being longer than 100 bytes is appended to a language file containing only plain text (tags are discarded). Finally the tag file and the WET files are deleted.

Only when one of these processes finishes another can be launched. This means that one can at most process and download as many files as cores the machine has. That is, if for example a machine has 24 cores, only 24 WET files can be downloaded and processed simultaneously, moreover, the 25th file won't be downloaded until one of the previous 24 files is completely processed.

When all the WET files are classified, one would normally get around 160 language files, each file holding just plain text written in its corresponding language. These files still need to be filtered in order to get rid of all files containing invalid UTF-8 characters, so again a number of processes are launched, this time depending on the amount of memory of the machine. Each process reads a language file, first filters for invalid UTF-8 characters and then performs deduplication. A simple non-collision resistant hashing algorithm is used to deduplicate the files.

The fastText linear classifier works by representing sentences for classification as Bags of Words (BoW) and training a linear classifier. A weight matrix A is used as a look-up table over the words and the word representations are then averaged into a text representation which is fed to the linear classifier. The architecture is in general similar to the CBoW model of Mikolov et al. (2013) but the middle word is replaced by a label. They use a softmax function f to compute the probability distribution over the classes. For a set of N documents, the model is trained to minimise the negative log-likelihood over the classes:

$$-\frac{1}{N} \sum_{n=1}^N y_n \log(f(BAx_n)),$$

where x_n is the normalised bag of features of the n -th document, y_n is the n -th label, and A, B are the weight matrices. The pre-trained fastText model for language recognition (Grave et al., 2018) is capable of recognising around 176 different languages and was trained using 400 million tokens from Wikipedia as well as sentences from the Tatoeba website¹¹.

2.6 ASYNCHRONOUS PIPELINE

We propose a new pipeline derived from the fastText one which we call `goclassy`, we reuse the fastText linear classifier (Joulin et al., 2016; Joulin et al., 2017) and the

¹¹<https://tatoeba.org/>

	10 files			100 files			200 files		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
<i>real</i>									
fastText	2m50s	6m45s	3m31s	13m46s	38m38s	17m39s	26m20s	47m48s	31m4s
goclassy	1m23s	3m12s	1m42s	7m42s	12m43s	9m8s	15m3s	15m47s	15m16s
<i>user</i>									
fastText	26m45s	27m2s	26m53s	4h21m	4h24m	4h23m	8h42m	8h48m	8h45m
goclassy	10m26s	12m53s	11m0s	1h46m	1h54m	1h49m	3h37m	3h40m	3h38m
<i>sys</i>									
fastText	40.14s	40.85s	40.56s	6m14s	6m17s	6m15s	12m26s	12m45s	12m31s
goclassy	37.34s	45.98s	39.67s	5m7s	5m34s	5m16s	9m57s	10m14s	10m5s

Table 2.1: Benchmarks are done using the UNIX time tool, are repeated 10 times each and are done for random samples of 10, 100 and 200 WET files. Only the classifying and filtering part are benchmarked. The table shows the minimum, maximum and mean time for the user, real and sys time over the 10 runs. Here “fastText” is used as short for the pipeline.

pre-trained fastText model for language recognition (Grave et al., 2018), but we completely rewrite and parallelise their pipeline in an asynchronous manner.

The order of operations is more or less the same as in the fastText pre-processing pipeline but instead of clustering multiple operations into a single blocking process, we launch a worker for each operation and we bound the number of possible parallel operations at a given time by the number of available threads instead of the number of CPUs. We implement goclassy using the Go programming language¹² so we let the Go runtime¹³ handle the scheduling of the processes. Thus in our pipeline we don’t have to wait for a whole WET file to download, decompress and classify in order to start downloading and processing the next one, a new file will start downloading and processing as soon as the scheduler is able to allocate a new process.

When using electromechanical mediums of storage, I/O blocking is one of the main problems one encounters. To overcome this, we introduced buffers in all our I/O operations, a feature that is not present in the fastText pre-processing pipeline. We also create, from the start, a file for each of the 176 languages that the pre-trained fastText language classifier is capable of recognising, and we always leave them open, as we find that getting a file descriptor to each time we want to write, if we wanted leave them open just when needed, introduces a big overhead.

We also do the filtering and cleaning processes at line level before feeding each line to the classifier, which makes us create a new filtered file so that we can have a correspondence with the tag file, which in turn will consume more space, but that will also reduce the amount of unnecessary classifications performed by fastText. The filtered and file tags are then read and lines are appended to its corresponding

¹²<https://golang.org/>

¹³<https://golang.org/src/runtime/mprof.go>

language file. The writing in the classification step is asynchronous, meaning that process writing a line to the filtered files does not wait for the classifier to write a tag on the tag file. Figure 2.1 shows the pipeline up to this point.

After all WET files are processed, we then use Isaac Whitfield’s deduplication tool `runiq`¹⁴ which is based on Yann Collet’s `xxhash64`¹⁵, an extremely fast non-cryptographic hash algorithm that is resistant to collisions. We finally use the Mark Adler’s `pigz`¹⁶ for data compression, as opposed to the canonical UNIX tools proposed in the original `fastText` pipeline. We add both tools to our concurrent pipeline, executing multiple instances of them in parallel, in order to ensure we use the most of our available resources at a given time.

Beyond improving the computational time required to classify this corpus, we propose a simple improvement on the cleaning scheme in the `fastText` pre-processing pipeline. This improvement allows our pipeline to better take into account the multilingual nature of Common Crawl; that is, we count UTF-8 characters instead of bytes for setting the lower admissible bound for the length of a line to be fed into the classifier. This straightforward modification on the `fastText` pre-processing pipeline assures we take into account the multiple languages present in Common Crawl that use non-ASCII encoded characters.

Given that our implementation is written in Go, we release binary distributions¹⁷ of `goclassy` for all major operating systems. Both `pigz` and `runiq` are also available for all major operating systems.

2.7 BENCHMARKS

We test both pipelines against one another in an infrastructure using traditional electromechanical storage mediums that are connected to the main processing machine via an Ethernet interface, that is, a low I/O speed environment as compared to an infrastructure where one would have an array of SSDs connected directly to the main processing machine via a high speed interface. We use a machine with an Intel® Xeon® Processor E5-2650 2.00 GHz, 20M Cache, and 203.1 GiB of RAM. We make sure that no other processes apart from the benchmark and the Linux system processes are run. We do not include downloading, decompression or deduplication in our benchmarks as downloading takes far too much time, and deduplication and compression were performed with third party tools that don’t make part of our main contribution. We are mainly interested in seeing how the way the data is fed to the classifier impacts the overall processing time.

¹⁴<https://github.com/whitfin/runiq>

¹⁵<https://github.com/Cyan4973/xxHash>

¹⁶<https://zlib.net/pigz/>

¹⁷<https://github.com/pjox/goclassy>

Benchmarks in table 2.1 of our goclassy pipeline show a drastic reduction in processing time compared to the original fastText preprocessing pipeline. We show that in our particular infrastructure, we are capable of reducing the *real* time as measured by the `time` UNIX tool almost always by half. The *user* time which represents the amount of CPU time spent in user-mode code (outside the kernel) within the process is almost three times lower for our goclassy pipeline, this particular benchmark strongly suggest a substantial reduction in energy consumption of goclassy with respect to the fastText pipeline.

As we understand that even an infrastructure with more than 20TB of free space in traditional electromechanical storage is not available to everyone and we propose a simple parametrization in our pipeline that actively deletes already processed data and that only downloads and decompresses files when needed, thus ensuring that no more than 10TB of storage are used at a given time. We nevertheless note that delaying decompression increases the amount of computation time, which is a trade-off that some users might make as it might be more suitable for their available infrastructure.

2.8 OSCAR

Finally, we are aware that some users might not even have access to a big enough infrastructure to run our pipelines or just to store all the Common Crawl data. Moreover, even if previously used and cited in NLP and Machine Learning research, we note that there is currently no public distribution of Common Crawl that is filtered, classified by language and ready to use for Machine Learning or NLP applications. Thus we decide to publish a pre-processed version of the November 2018 copy of Common Crawl which is comprised of usable data in 166 different languages, we publish¹⁸ our version under the name OSCAR which is short for *Open Super-large Crawled ALMAAnaCH¹⁹ coRpus*.

After processing all the data with goclassy, the size of the whole Common Crawl corpus is reduced to 6.3TB, but in spite of this considerable reduction, OSCAR still dwarfs all previous mentioned corpora having more 800 billion “words” or spaced separated tokens and noting that this in fact is an understatement of how big OSCAR is, as some of the largest languages within OSCAR such as Chinese and Japanese do not use spaces. The sizes in bytes for both the original and the deduplicated versions of OSCAR can be found in table 2.2. OSCAR is published under the *Creative Commons CC0 license* (“no rights reserved”)²⁰, so it is free to use for all applications.

¹⁸<https://team.inria.fr/almanach/oscar/>

¹⁹<https://team.inria.fr/almanach/>

²⁰<http://creativecommons.org/publicdomain/zero/1.0/>

2.9 CONCLUSIONS

We are sure that our work will greatly benefit researchers working on an either constrain infrastructure or a low budget setting. We are also confident, that by publishing a classified version of Common Crawl, we will substantially increase the amount of available public data for medium to low resource languages, thus improving and facilitating NLP research for them. Furthermore, as our pipeline speeds-up and simplifies the treatment of Common Crawl, we believe that our contribution can be further parallelised and adapted to treat multiple snapshots of Common Crawl opening the door to what would be otherwise costly diachronic studies of the use of a given language throughout the internet.

Finally, we note that both our proposed pipeline is data independent, which means that they can be reused to process, clean and classify any sort of big multilingual corpus that is available in plain text form and that is UTF-8 encoded; meaning that the impact of our work goes way beyond a single corpus.

2.9 Conclusions

Language	Size		Words		Language	Size		Words	
	Orig	Dedup	Orig	Dedup		Orig	Dedup	Orig	Dedup
Afrikaans	241M	163M	43,482,801	29,533,437	Lower Sorbian	13K	7.1K	1,787	966
Albanian	2.3G	1.2G	374,196,110	186,856,699	Luxembourgish	29M	21M	4,403,577	3,087,650
Amharic	360M	206M	28,301,601	16,086,628	Macedonian	2.1G	1.2G	189,289,873	102,849,595
Arabic	82G	32G	8,117,162,828	3,171,221,354	Maithili	317K	11K	69,161	874
Aragonese	1.3M	801K	52,896	45,669	Malagasy	21M	13M	3,068,360	1,872,044
Armenian	3.7G	1.5G	273,919,388	110,196,043	Malay	111M	42M	16,696,882	6,045,753
Assamese	113M	71M	6,956,663	4,366,570	Malayalam	4.9G	2.5G	189,534,472	95,892,551
Asturian	2.4M	2.0M	381,005	325,237	Maltese	24M	17M	2,995,654	2,163,358
Avaric	409K	324K	24,720	19,478	Marathi	2.7G	1.4G	162,609,404	82,130,803
Azerbaijani	2.8G	1.5G	322,641,710	167,742,296	Mazanderani	691K	602K	73,870	64,481
Bashkir	128M	90M	9,796,764	6,922,589	Minangkabau	608K	310K	5,682	4,825
Basque	848M	342M	120,456,652	45,359,710	Mingrelian	5.8M	4.4M	299,098	228,629
Bavarian	503	503	399	399	Mirandese	1.2K	1.1K	171	152
Belarusian	1.8G	1.1G	144,579,630	83,499,037	Modern Greek	62G	27G	5,479,180,137	2,412,419,435
Bengali	11G	5.8G	623,575,733	363,766,143	Mongolian	2.2G	838M	181,307,167	68,362,013
Bihari	110K	34K	8,848	2,875	Nahuatl languages	12K	11K	1,234	1,193
Bishnupriya	4.1M	1.7M	198,286	96,940	Neapolitan	17K	13K	5,282	4,147
Bosnian	447K	116K	106,448	20,485	Nepali	1.8G	1.2G	107,448,208	71,628,317
Breton	29M	16M	5,013,241	2,890,384	Newari	5.5M	4.1M	564,697	288,995
Bulgarian	32G	14G	2,947,648,106	1,268,114,977	Northern Frisian	4.4K	4.4K	1,516	1,516
Burmese	1.9G	1.1G	56,111,184	30,102,173	Northern Luri	76K	63K	8,022	6,740
Catalan	8.0G	4.3G	1,360,212,450	729,333,440	Norwegian	8.0G	4.7G	1,344,326,388	804,894,377
Cebuano	39M	24M	6,603,567	3,675,024	Norwegian Nynorsk	85M	54M	14,764,980	9,435,139
Central Bikol	885	885	312	312	Occitan	5.8M	3.7M	750,301	512,678
Central Khmer	1.1G	581M	20,690,610	10,082,245	Oriya	248M	188M	14,938,567	11,321,740
Central Kurdish	487M	226M	48,478,334	18,726,721	Ossetian	13M	11M	1,031,268	878,765
Chavacano	520	520	130	130	Pampanga	760	304	130	52
Chechen	8.3M	6.7M	711,051	568,146	Panjabi	763M	460M	61,847,806	37,555,835
Chinese	508G	249G	14,986,424,850	6,350,215,113	Persian	79G	38G	9,096,554,121	4,363,505,319
Chuvash	39M	26M	3,041,614	2,054,810	Piemontese	2.1M	1.9M	362,013	337,246
Cornish	44K	14K	8,329	2,704	Polish	109G	47G	15,277,255,137	6,708,709,674
Croatian	226M	110M	34,232,765	16,727,640	Portuguese	124G	64G	20,641,903,898	10,751,156,918
Czech	53G	24G	7,715,977,441	3,540,997,509	Pushto	361M	242M	46,559,441	31,347,348
Danish	16G	9.5G	2,637,463,889	1,620,091,317	Quechua	78K	67K	10,186	8,691
Dhivehi	126M	79M	7,559,472	4,726,660	Romanian	25G	11G	3,984,317,058	1,741,794,069
Dimli	146	146	19	19	Romansh	7.4K	6.5K	1,093	960
Dutch	78G	39G	13,020,136,373	6,598,786,137	Russia Buriat	13K	11K	963	809
Eastern Mari	7.2M	6.0M	565,992	469,297	Russian	1.2T	568G	92,522,407,837	46,692,691,520
Egyptian Arabic	66M	33M	7,305,151	3,659,419	Sanskrit	93M	37M	4,331,569	1,713,930
Emilian-Romagnol	25K	24K	6,376	6,121	Scottish Gaelic	1.9M	1.3M	310,689	207,110
English	2.3T	1.2T	418,187,793,408	215,841,256,971	Serbian	3.9G	2.2G	364,395,411	207,561,168
Erzya	1.4K	1.2K	90	78	Serbo-Croatian	25M	5.8M	5,292,184	1,040,573
Esperanto	299M	228M	48,486,161	37,324,446	Sicilian	3.3K	2.8K	554	468
Estonian	4.8G	2.3G	643,163,730	309,931,463	Sindhi	347M	263M	43,530,158	33,028,015
Finnish	27G	13G	3,196,666,419	1,597,855,468	Sinhala	1.4G	802M	93,053,465	50,864,857
French	282G	138G	46,896,036,417	23,206,776,649	Slovak	9.1G	4.5G	1,322,247,763	656,346,179
Galician	620M	384M	102,011,291	63,600,602	Slovenian	2.5G	1.3G	387,399,700	193,926,684
Georgian	3.6G	1.9G	171,950,621	91,569,739	Somali	61K	16K	1,202	472
German	308G	145G	44,878,908,446	21,529,164,172	South Azerbaijani	27M	19M	2,175,054	1,528,709
Goan Konkani	2.2M	1.8M	124,277	102,306	Spanish	278G	149G	47,545,122,279	25,928,290,727
Guarani	36K	24K	7,382	4,680	Sundanese	211K	141K	30,321	20,728
Gujarati	1.1G	722M	72,045,701	50,023,432	Swahili	13M	8.1M	2,211,927	1,376,963
Haitian	3.9K	3.3K	1,014	832	Swedish	44G	25G	7,155,994,312	4,106,120,608
Hebrew	20G	9.8G	2,067,753,528	1,032,018,056	Tagalog	573M	407M	98,949,299	70,121,601
Hindi	17G	8.9G	1,372,234,782	745,774,934	Tajik	379M	249M	31,758,142	21,029,893
Hungarian	40G	18G	5,163,936,345	2,339,127,555	Tamil	9.3G	5.1G	420,537,132	226,013,330
Icelandic	1.5G	846M	219,900,094	129,818,331	Tatar	670M	305M	51,034,893	23,825,695
Ido	147K	130K	25,702	22,773	Telugu	2.5G	1.6G	123,711,517	79,094,167
Iloko	874K	636K	142,942	105,564	Thai	36G	16G	951,743,087	368,965,207
Indonesian	30G	16G	4,574,692,265	2,394,957,629	Tibetan	187M	138M	1,483,589	936,556
Interlingua	662K	360K	180,231	100,019	Tosk Albanian	5.0M	2.8M	841,750	459,001
Interlingue	24K	1.6K	5,352	602	Turkish	60G	27G	7,577,388,700	3,365,734,289
Irish	88M	60M	14,483,593	10,017,303	Turkmen	11M	6.8M	1,113,869	752,326
Italian	137G	69G	22,248,707,341	11,250,012,896	Tuvinian	12K	7.9K	759	540
Japanese	216G	106G	4,962,979,182	1,123,067,063	Uighur	122M	83M	8,657,141	5,852,225
Javanese	659K	583K	104,896	86,654	Ukrainian	53G	28G	4,204,381,276	2,252,380,351
Kalmyk	113K	112K	10,277	10,155	Upper Sorbian	4.2M	1.8M	545,351	236,867
Kannada	1.7G	1.1G	81,186,863	49,343,462	Urdu	2.7G	1.7G	331,817,982	218,030,228
Karachay-Balkar	2.6M	2.3M	185,436	166,496	Uzbek	21M	12M	2,450,256	1,381,644
Kazakh	2.7G	1.5G	191,126,469	108,388,743	Venetian	18K	17K	3,492	3,199
Kirghiz	600M	388M	44,194,823	28,982,620	Vietnamese	68G	32G	12,036,845,359	5,577,159,843
Komi	2.3M	1.2M	201,404	95,243	Volapük	2.0M	2.0M	321,121	318,568
Korean	24G	12G	2,368,765,142	1,120,375,149	Walloon	273K	203K	50,720	37,543
Kurdish	94M	60M	15,561,003	9,946,440	Waray	2.5M	2.2M	397,315	336,311
Lao	174M	114M	4,133,311	2,583,342	Welsh	213M	133M	37,422,441	23,574,673
Latin	26M	8.3M	4,122,201	1,328,038	Western Frisian	35M	26M	5,691,077	4,223,816
Latvian	4.0G	1.8G	520,761,977	236,428,905	Western Mari	1.2M	1.1M	93,338	87,780
Lezghian	3.3M	3.0M	247,646	224,871	Western Panjabi	12M	9.0M	1,426,986	1,111,112
Limbungan	29K	27K	4,730	4,283	Wu Chinese	109K	32K	11,189	4,333
Lithuanian	8.8G	3.9G	1,159,661,742	516,183,525	Yakut	42M	26M	2,547,623	1,789,174
Lojban	736K	678K	154,330	141,973	Yiddish	141M	84M	13,834,320	8,212,970
Lombard	443K	433K	75,229	73,665	Yoruba	55K	27K	8,906	3,518
Low German	18M	13M	2,906,347	2,146,417	Yue Chinese	3.7K	2.2K	186	128
Total	6.3T	3.2T	844,315,434,723	425,651,344,234					

Table 2.2: Size of the OSCAR corpus by language measured in bytes and number of words. Standard UNIX human-readable notation is used for the size in byte. We define “words” as spaced separated tokens, which gives a good estimate of the size of each corpus for languages using Latin or Cyrillic alphabets, but might give a misleading size for other languages such as Chinese or Japanese.

2.10 INTRODUCTION

Access to multilingual datasets for NLP research has vastly improved over the past years. A variety of web-derived collections for hundreds of languages is available for anyone to download, such as ParaCrawl (Esplà et al., 2019; Bañón et al., 2020), WikiMatrix (Schwenk et al., 2021) CCAIined (El-Kishky et al., 2020), OSCAR (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020), and several others. These have in turn enabled a variety of highly multilingual models, like mT5 (Xue et al., 2021), M2M-100 (Fan et al., 2020), M4 (Arivazhagan et al., 2019).

Curating such datasets relies on the websites giving clues about the language of their contents (e.g. a language identifier in the URL) and on automatic language classification (LangID). It is commonly known that these automatically crawled and filtered datasets tend to have overall lower quality than hand-curated collections (Koehn et al., 2020), but their quality is rarely measured directly, and is rather judged through the improvements they bring to downstream applications (Schwenk et al., 2021).

Building NLP technologies with automatically crawled datasets is promising. This is especially true for low-resource languages, because data scarcity is one of the major bottlenecks for deep learning approaches. However, there is a problem: There exists very little research on evaluating both data collections and automatic crawling and filtering tools for low-resource languages. As a result, although many low-resource languages are covered by the latest multilingual crawl data releases, their quality and thus usability is unknown.

To shed light on the quality of data crawls for the lowest resource languages, we perform a manual data audit for 230 per-language subsets of five major crawled multilingual datasets: CCAIined (El-Kishky et al., 2020), ParaCrawl (Esplà et al., 2019; Bañón et al., 2020), WikiMatrix (Schwenk et al., 2021), OSCAR (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020) and mC4 (Xue et al., 2021). We propose solutions for effective, low-effort data auditing (section 2.13), including an error taxonomy. Our quantitative analysis reveals surprisingly low amounts of valid in-language data, and identifies systematic issues across datasets and languages. In addition, we find that a large number of datasets is labeled with nontransparent or incorrect language codes (section 2.14). This leads us to reflect on the potential harm of low-quality data releases for low-resource languages (section 2.15), and provide a set of recommendations for future multilingual data releases (section 2.17).

2.11 RELATED WORK

Corpora collected by web crawlers are known to be noisy (Junczys-Dowmunt, 2019). In highly multilingual settings, past work found that web-crawls of lower-resource languages have serious issues, especially with segment-level LangID (Caswell et al.,

	Parallel			Monolingual	
	CCAligned	ParaCrawl v7.1	WikiMatrix	OSCAR	mC4
#languages	137	41	85	166	101
Source	CC 2013–2020	selected websites	Wikipedia	CC 11/2018	CC all
Filtering level	document	sentence	sentence	document	document
Langid	FastText	CLD2	FastText	FastText	CLD3
Alignment	LASER	Vec/Hun/BLEU-Align	LASER	-	-
Evaluation	TED-6	WMT-5	TED-45	POS/DEP-5	XTREME

Table 2.3: Comparison of parallel and monolingual corpora extracted from web documents, including their downstream evaluation tasks. All parallel corpora are evaluated through machine translation evaluation with BLEU. TED-6: da, cr, sl, sk, lt, et; TED-45: 45-language subset of (Qi et al., 2018); WMT-5: cs, de, fi, lv, ro. POS/DEP-5: part-of-speech labeling and dependency parsing for bg, ca, da, fi, id.

2020). Cleaning and filtering web-crawls can boost general language modeling (Gao et al., 2020; Brown et al., 2020; Raffel et al., 2020) and downstream task performance (Moore and Lewis, 2010; Xu and Koehn, 2017; Khayrallah and Koehn, 2018; Brown et al., 2020).

As the scale of ML research grows, it becomes increasingly difficult to validate automatically collected and curated datasets (Biderman and Scheirer, 2020; ?; ?). Several works have focused on advancing methodologies and best practices to address these challenges. ? introduced data statements, a documentary framework for NLP datasets that seeks to provide a universal minimum bar for dataset description. Similar work has focused on online news (Kevin et al., 2018), data ethics (?), and data exploration (?), as well as generalist work such as (?). There is a large literature on filtering text data for various NLP tasks, e.g. (Axelrod et al., 2011; Moore and Lewis, 2010; ?; Kamholz et al., 2014; Junczys-Dowmunt, 2018; Caswell et al., 2020).

Closest to our work is the analysis of a highly multilingual (non-publicly available) web-crawl and LangID related quality issues by Caswell et al. (2020). They perform a brief analysis of the quality of OSCAR with the focus only on the presence of in-language content.

2.12 MULTILINGUAL CORPORA

Table 2.3 provides an overview of the corpora of interest in this work. We selected the corpora for their multilinguality and the inclusion of understudied languages in NLP. With the exception of WikiMatrix and Paracrawl, all corpora are derived from CommonCrawl (CC).²¹

²¹<http://commoncrawl.org/>

CCALIGNED (EL-KISHKY ET AL., 2020) is a parallel dataset built off 68 CC snapshots. Documents are aligned if they are in the same language according to FastText LangID (Joulin et al., 2016; Joulin et al., 2017), and have the same URL but for a differing language code. These alignments are refined with cross-lingual LASER embeddings (Artetxe and Schwenk, 2019). For sentence-level data, they split on newlines and align with LASER, but perform no further filtering. Human annotators evaluated the quality of document alignments for six languages (de, zh, ar, ro, et, my) selected for their different scripts and amount of retrieved documents, reporting precision of over 90%. The quality of the extracted parallel sentences was evaluated in a machine translation (MT) task on six European (da, cr, sl, sk, lt, et) languages of the TED corpus (Qi et al., 2018), where it compared favorably to systems built on crawled sentences from WikiMatrix and ParaCrawl v6.

MULTILINGUAL C4 (mC4) (XUE ET AL., 2021) is a document-level dataset used for training the mT5 language model. It consists of monolingual text in 101 languages and is generated from 71 CC snapshots. It filters out pages that contain less than three lines of at least 200 characters and pages that contain bad words (?). Since this is a document-level dataset, we split it by sentence and deduplicate it before rating. For language identification, it uses CLD3 (Botha et al., 2017), a small feed-forward neural network that was trained to detect 107 languages. The mT5 language model pre-trained on mC4 is evaluated on 6 tasks of the XTREME benchmark (?) covering a variety of languages and outperforms other multilingual pre-trained language models such as mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020).

OSCAR (ORTIZ SUÁREZ ET AL., 2019; ORTIZ SUÁREZ ET AL., 2020) is a set of monolingual corpora extracted from CC snapshots, specifically from the plain text WET format distributed by CC which removes all the HTML tags and converts the text formatting to UTF-8. It is deduplicated and follows the same approach as Grave et al. (2018) by using FastText LangID on a line-level. For five languages (bg, ca, da, fi, id) OSCAR corpora were used to train ELMo (Peters et al., 2018) embeddings for POS tagging and dependency parsing, outperforming those trained on Wikipedia (Ortiz Suárez et al., 2020).

PARACRAWL v7.1 is a parallel dataset with 41 language pairs primarily aligned with English (39 out of 41) and mined using the parallel-data-crawling tool Bitextor (Esplà et al., 2019; Bañón et al., 2020) which includes downloading documents, preprocessing and normalization, aligning documents and segments, and filtering noisy data via Bicleaner. ParaCrawl focuses on European languages, but also includes 9 lower-resource, non-European language pairs in v7.1. Sentence alignment and sentence pair filtering choices were optimized for five languages (mt, et, hu, cs, de) by training and evaluating MT models on the resulting parallel sentences. An

earlier version, ParaCrawl v5, was shown to improve translation quality on WMT benchmarks for cs, de, fi, lv, ro.

WIKIMATRIX (SCHWENK ET AL., 2021) is a public dataset containing 135M parallel sentences in 1620 language pairs (85 languages) mined from Wikipedia. Out of the 135M parallel sentences, 34M are aligned with English. The text is extracted from Wikipedia pages, split into sentences, and duplicate sentences are removed. FastText LangID is used before identifying bitext with LASER’s distance-based mining approach. The margin threshold is optimized by training and evaluating downstream MT models on four WMT benchmarks (de-en, de-fr, cs-de, cs-fr). The final dataset is evaluated through TED translation models between 45 languages, with highest quality for translations between English and e.g. pt, es, da, and lowest for sr, ja, mr, zh_TW. We focus on language pairs with English on one side.

2.13 AUDITING DATA QUALITY

None of the above datasets has been evaluated for quality on the sentence level (exception: several languages in ParaCrawl v3), and downstream evaluations are centered around a small fraction of higher-resource languages. This is insufficient for drawing conclusions about the quality of individual or aligned sentences, and about the entirety of languages. To close this gap, we conduct a data quality audit that focuses on the lowest-resource and most under-evaluated languages, but also covers mid- and high-resource languages for comparison.

2.13.1 AUDITING PROCESS

PARTICIPANTS We recruited 51 volunteers from the NLP community, covering about 70 languages with proficient language skills. To verify our hypothesis that those annotations can largely be done by non-native speakers, we repeat a set of language expert annotations by a non-expert, and measure the accuracy of the non-expert.

SAMPLE SELECTION For each language in each dataset, we took a random sample of 100 lines, which may be anywhere from single words to short paragraphs depending on segmentation. We manually annotated them according to the error taxonomy described below. For WikiMatrix and CCAligned, we selected those languages that are paired with English, and for ParaCrawl, we also included those paired with Spanish (“total” counts in table 2.5). We did not annotate all languages, but focused on the ones with the least number of sentences in each dataset (at least the smallest 10) and languages for which we found proficient speakers.

Correct Codes	
CC: <i>Correct translation, natural sentence</i>	
en The Constitution of South Africa	nso Molaotheo wa Rephabliki ya Afrika Borwa
en Transforming your swimming pool into a pond	de Umbau Ihres Swimmingpools zum Teich
CB: <i>Correct translation, Boilerplate or low quality</i>	
en Reference number: 13634	ln Motango ya référence: 13634
en Latest Smell Stop Articles	fi l Pinakabagong mga Artikulo Smell Stop
CS: <i>Correct translation, Short</i>	
en movies, dad	it cinema, papà
en Halloween - without me	ay Halloween – janiw nayampejj
Error Codes	
X: <i>Incorrect translation, but both correct languages</i>	
en A map of the arrondissements of Paris	kg Paris kele mbanza ya kimfumu ya Fwalansa.
en Ask a question	tr Soru sor Kullanıma göre seçim
WL: <i>Source OR target wrong language, but both still linguistic content</i>	
en The ISO3 language code is zho	zza Tāim eadra brachach mar bhionns na frogannaidhe.
en Der Werwolf — sprach der gute Mann,	de des Weswolfs, Genitiv sodann,
NL: <i>Not a language: at least one of source and target are not linguistic content</i>	
en EntryScan 4 _	tn TSA PM704 _
en organic peanut butter	ckb 🚫🚫🚫🚫🚫🚫🚫

Table 2.4: Annotation codes for parallel data with sentence pair examples. The language code before each sentence indicates the language it is supposed to be in.

NON-EXPERT LABELING STRATEGIES Although many of the volunteers were familiar with the languages in question or spoke related languages, in cases where no speaker of a relevant language could be found, volunteers used dictionaries and internet search to form educated guesses. We discuss this deeper in appendix 2.23 to highlight how much of this low-resource focused evaluation can actually be done by non-proficient speakers with relatively low effort. In general, we aim to find an upper bound on quality, so we encouraged annotators to be forgiving of translation mistakes when the overall meaning of the sentence or large parts thereof are conveyed, or when most of the sentence is in the correct language.

TAXONOMY In order to quantify errors, we developed a simple error taxonomy. Sentences and sentence pairs were annotated according to a simple rubric with error classes of Incorrect Translation (X, excluded for monolingual data), Wrong Language (WL), and Non-Linguistic Content (NL). Of correct sentences (C), we further mark single words or phrases (CS) and boilerplate contents (CB). The appendix contains the detailed instructions, and table 2.4 provides examples for parallel data. In addition, we asked annotators to flag offensive or pornographic content.

2.13.2 HUMAN AUDIT RESULTS

INTERPRETATION OF RESULTS For each language, we compute the percentage of each label within the 100 audited sentences. Then, we either aggregate the labels across languages with equal weights (macro-average), or weight them according to their presence in the overall dataset (micro-average). Note that the number of languages, the numbers of sentences per language and the choice of languages differ across datasets, both in the original release and in the selection for our audit, so the comparison of numbers across datasets has to be taken with a grain of salt. Our audit captures a decent ratio of languages (25–55%, 2nd row in table 2.5), but only a tiny fraction of the overall number of sentences (0.00004–0.002%). Appendix 2.25 contains the detailed audit results for each language and dataset. When we speak of “low-” and “high”-resource languages, we mean languages with smaller or larger representation in the datasets at hand. When reporting language-specific results we use the original language identifiers of the datasets.

WHICH DATASETS HAVE QUALITY ISSUES? The macro-averaged results show that the ratio of correct samples (“C”) ranges from 24% to 87%, with a large variance across the five audited datasets. Particularly severe problems were found in CCAIined and WikiMatrix, with 44 of the 65 languages that we audited for CCAIined containing under 50% correct sentences, and 19 of the 20 in WikiMatrix. In total, 15 of the 205 language specific samples (7.3%) contained not a single correct sentence. For the parallel datasets we are also interested in the quantity of misaligned/mistranslated sentences (X). For WikiMatrix, two-thirds of the audited samples were on average misaligned. We noticed that sentences were often similar in structure, but described different facts (see table 2.12).

While Table 2.5 gives means and numbers of corpora passing certain thresholds, Figure 2.3 illustrates per-corpus correctness more completely, showing for each dataset what percent of audited corpora are under each possible threshold of correctness.

WHY HAVEN’T THESE PROBLEMS BEEN REPORTED BEFORE? The findings above are averaged on a per-language basis (i.e. macro-average), and therefore give low and high-resource languages equal weight. If we instead estimate the quality on a per-sentence basis, i.e. down-weight the the lower-resource languages in the computation of the average, the numbers paint a more optimistic picture (“micro” block in table 2.5). This is especially relevant for the monolingual datasets because they contain audits for English, which makes up for 43% of all sentences in OSCAR and 36% in mC4. To illustrate the effect of this imbalance: A random sample from the entire mC4 dataset will with over 63% chance be from one of the 8 largest languages (en, ru, es, de, fr, it, pt, pl, >100M sentences each), of which all have near perfect quality. Analogously, evaluation and tuning of web mining pipelines and resulting corpora in downstream

applications focused largely on higher-resource languages (section 2.12), so the low quality of underrepresented languages might go unnoticed if there is no dedicated evaluation, or no proficient speakers are involved in the curation (?).

HOW MUCH CONTENT IS NONLINGUISTIC OR IN THE WRONG LANGUAGE? In general, non-linguistic content was a larger problem than wrong-language content. Among the parallel datasets, CCAIined contains the highest percentage of nonlinguistic content, at 31.42% on average across all rated corpora, and also the highest percent of wrong-language content, at 9.44% on average. Among the monolingual datasets, mC4 contains the highest ratio both of sentences in incorrect languages (15.98% average) and nonlinguistic content (11.40% average), with 4 of the 48 audited languages having more than 50% contents in other languages. The low amount of wrong language in ParaCrawl shows the benefits of selecting domains by the amount in-language text, but the dataset also covers the smallest amount of languages. The relatively low ratio of wrong language samples in OSCAR may reflect the success of line-level LangID filtering. These numbers provide evidence that more research in improved LangID could improve the overall quality, especially with respect to nonlinguistic content.

WHICH LANGUAGES GOT CONFUSED? The languages that were confused were frequently related higher-resource languages. However, there were also a significant number of “out-of-model cousin” cases, where languages not supported by the LangID model ended up in a similar-seeming language. For instance in mC4, much of the Shona (sn) corpus is actually Kinyarwanda (rw) – and, peculiarly, much of the Hawaiian (haw) is actually Twi (tw/ak).

DO LOW-RESOURCE LANGUAGES HAVE LOWER QUALITY? Low-resource datasets tend to have lower human-judged quality. The Spearman rank correlation between quality (%C) and size is positive in all cases. The trend is strongest for mC4 ($r = 0.66$), and gradually declines for CCAIined ($r = 0.53$), WikiMatrix ($r = 0.49$), ParaCrawl ($r = 0.43$), and OSCAR ($r = 0.37$). Figure 2.2 compares the number of sentences for each language against the proportion of correct sentences that we found during the audit: Not all high-resource languages have high quality, in particular for CCAIined (e.g. en-jv_ID with 5%C, or en-tl_XX with 13%C). For mid-resource languages (10^4 – 10^6 sentences) the picture is rather inconclusive.

WHICH LANGUAGES HAVE THE LOWEST QUALITY? Across datasets we observe that the quality is particularly poor for languages that are included in the datasets in romanized script, but are more commonly written in other scripts, e.g., Urdu (ur), Hindi (hi), Arabic (ar). In terms of geography, the poorest quality is found for African languages (bm, ff, kg, lg, ln, nso, om, sn, so, tn, wo), minority languages in Europe

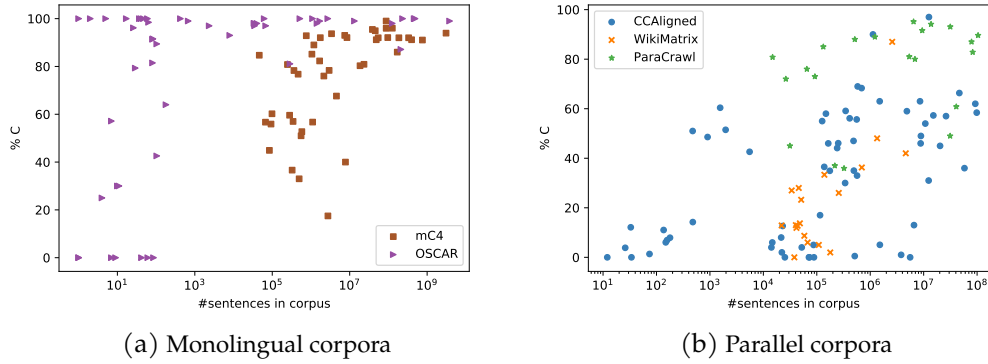


Figure 2.2: Percentage of sentences labeled as correct vs. $\log N$ sentences for all audited languages.

and the Middle East that are closely related to higher-resource languages (az-IR, frr, nap, szl, zza), lesser spoken Chinese languages sharing a script with Mandarin (yue, wuu), and four major Austronesian languages (bcl, cbk, jv, su).

WHAT IS THE INCIDENCE OF OFFENSIVE AND PORNOGRAPHIC CONTENT? Overall, the sampled sentences did not contain a large amount of offensive contents. However, there were notable amounts of pornographic content ($> 10\%$) found in CCAIined for 11 languages.

ANNOTATION QUALITY For six audited languages from OSCAR and ten from CCAIined we measure the accuracy of the labels assigned by non-proficient speakers against the labels assigned by proficient speakers for all audited sentences. With the full 6-class taxonomy we find a mean accuracy of 0.66 for CCAIined audits, and 0.98 for OSCAR audits (see appendix 2.20 for language-specific results). With a binary taxonomy distinguishing C from the rest, the accuracy further increases to 0.79 for CCAIined. This provides strong evidence that good quality annotations are not limited to those proficient in a language.

2.13.3 AUTOMATIC FILTERING

Given the frequency of WL and NL annotations, it might be tempting to use open-source LangID models to post-filter data on a per-sentence(-pair) level, as OSCAR does. Unfortunately, this turns out to have its own issues.

SENTENCE-LEVEL N-GRAM FILTERING We classify all sentence pairs of CCAIined with CLD3. By comparing its predictions to the audit labels, we evaluate its quality on the subset of annotated samples: the classifier should detect both correct languages

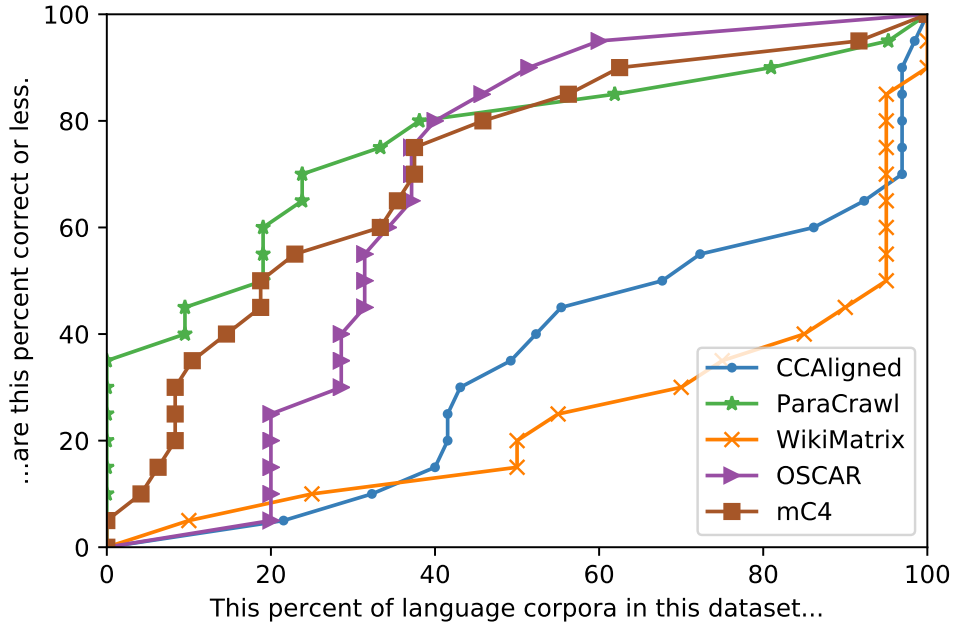


Figure 2.3: Fraction of languages in each dataset below a given quality threshold (percent correct).

when the pair is annotated as C and X, and should detect incorrect languages in the pair when WL and NL. On this task, the CLD3 classifier achieves an average precision of only 40.6%.

TRANSFORMER-BASED LANGID FILTERING N-gram LangID models like CLD3 have known problems. However, [Caswell et al. \(2020\)](#) demonstrate that semi-supervised transformer-based LangID models strongly out-perform them. We train a comparable transformer-based LangID model and apply it to our annotated CCAIined data. We find that filtering noisy corpora (< 50% correct) on LangID for both source and target leads to gains in median precision, rising from 13.8% pre-filter to 43.9% post-filter. However, this comes at a steep cost of 77.5% loss in recall. The biggest winners were Lingala, whose precision climbs from 8% to 80%, and Oromo, which soars from 2% to 33% in-language. Both of these, however, come at the cost of losing 50% of the correct in-language sentences. The moral is that, at least at the current stage, there is no one-size-fits-all approach for sentence-level LangID.

2.14 DATASET MIS-LABELING

Standardized and unambiguous representations of language codes are important for practical data use and exchange. The standard for unambiguous language codes used by most academic and industry applications is BCP-47 (?), which would enhance transparency and interoperability if adopted consistently.

We find a variety of errors in language code usage, ranging from serious mislabelings to small transgressions against standard conventions. . For this analysis, we also include the JW300 (Agić and Vulić, 2019) dataset, a multilingual dataset crawled from jw.org. In summary, we find 8 nonstandard codes in CCAligned, 3 in OSCAR, 1 in mC4, 1 in WikiMatrix, and 70 in JW300, for 83 in total. This does not include the 59 codes affected by superset issues. Full details are given in appendix 2.18.

INCONSISTENT LANGUAGE CODES One common issue is simply using nonstandard or invented codes. For example, CCAligned uses only two-letter codes, so when the BCP-47 code for a language is three letters it is either shortened (e.g. `zza` → `zz`) or invented (`shn` → `qa`). Similarly, OSCAR contains data labeled as `als` (BCP-47 for Tosk Albanian) that is actually in `gsw` (Allemanic). 22 additional language codes in JW300 have similar issues, including 12 codes that start with `jw_` but are not Javanese.

FALSE SIGN LANGUAGES 12% (48/417) of JW300 carry language codes for sign languages. Instead of sign language transcripts they are texts in another high resource language, mostly English or Spanish—for example, the `en-zs1` data is actually English-English parallel data (i.e. copies).

MYSTERIOUS SUPERSETS When datasets contain language codes that are supersets of other language codes, it is difficult to determine which particular language the data are in. WikiMatrix has Serbian (`sr`), Croatian (`hr`), Bosnian (`bs`), and Serbo-Croatian (`sh`)—their superset. In some cases this may not be an issue, as with Arabic, where `ar` conventionally refers to Modern Standard Arabic, even though the code technically encompasses all dialects. But in many cases, the nature of the data in the superset code remains a mystery.

DEPRECATED CODES Finally, there are several deprecated codes that are used: `sh` in Wikimatrix, `iw` in mC4, `sh` and `eml` in Oscar, and `daf` in JW300.

2.15 RISKS OF LOW-QUALITY DATA

LOW QUALITY IN DOWNSTREAM APPLICATIONS Text corpora today are building blocks for many downstream NLP applications like question answering and text summarization—for instance, a common approach is to first train translation models on such data and

then automatically translate training data for downstream models (Conneau et al., 2018). If the data used for the original systems is flawed, derived technology may fail for those languages far down the line without knowing the causes.

REPRESENTATION WASHING Since there are datasets which contain many low-resource languages, the community may feel a sense of progress and growing equity, despite the actual quality of the resources for these languages. Similarly, if low-quality datasets are used as benchmarks they may exaggerate model performance, making low-resource NLP appear more solved than it is—or conversely, if models perform poorly when trained with such data, it may be wrongly assumed that the task of learning models for these languages is harder than it actually is or infeasible given current resources. These effects could result in productive effort being redirected away from these tasks and languages.

TRUST IN INCORRECT “FACTS” We found many instances of parallel-looking sentences that are actually not semantically similar (appendix table 2.12). They can cause models to produce plausible “translations” that are factually wrong, but users may still trust them (*algorithmic trust*) without verifying the information. Similarly, *automation bias* (?), referring to humans favoring decisions made by automated systems over decisions made by humans, might amplify the issues of inaccurate translations caused by misaligned sentences.

2.16 FUTURE WORK

There are a variety of ways to improve both the ease and accuracy of human evaluation, as well a few classes of issues we ignored in this paper, like close dialects. We present a slightly improved suggested rubric in appendix 2.24.

Ideally there can be a standard suite of automatic metrics for datasets, but more study is necessary to determine what the appropriate metrics would be. One important area missing from our analyses however is the estimated portion of a dataset which has been generated by MT, LM systems, or bots/templates. The information captured in machine-generated content might still be useful for modeling, but might falsely overrepresent typical generation patterns and introduce linguistic errors or unnatural artifacts.

2.17 CONCLUSION & RECOMMENDATIONS

Of the five multilingual corpora evaluated, we consistently found severe issues with quality, especially in the lower-resource languages. We rated samples of 205 languages, and found that 87 of them had under 50% usable data, with a full 15

languages at 0% in-language. We furthermore found consistent issues with mislabeled data and nonstandard language codes, particularly in the JW300 dataset, and identified 83 affected corpora, at least 48 of which were entirely spurious (Section 2.14). While there might have been anecdotal evidence of insufficient quality for some of the datasets, the majority of these quality issues had not been reported, nor been investigated in depth. These issues might go unnoticed for languages that are not represented in the evaluation of the crawling methods, and cause harm in downstream applications.

We therefore strongly recommend looking at samples of any dataset before using it or releasing it to the public. As we have shown, one does not need to be proficient in a language to see when there are serious quality issues, and a quick scan of 100 (or fewer!) sentences can be sufficient to detect major problems. Moreover, going through and annotating a small sample of data can bring useful insights about new ways to filter or use it.

If data quality issues are found, a wide variety of techniques can be explored, like filtering on length-ratio, LangID, TF-IDF wordlists (Caswell et al., 2020) or dictionaries (Kamholz et al., 2014); to neural approaches like LM scoring (Axelrod et al., 2011; Moore and Lewis, 2010; ?). Unfortunately, none of these provides a quick and easy fix, especially for low-resource languages – data cleaning is no trivial task!

Noisy datasets are however by no means useless, at least if they contain some usable content. Therefore an alternative to filtering can be documentation (?). This can take the form of a per-language quality score and notes about known issues, a datasheet (?) or nutrition label (?). However, we suggest researchers not release corpora with near-zero in-language content, as this may give the mistaken impression of usable resources.

Finally, we encourage the community to continue conducting evaluations and audits of public datasets – similar to system comparison papers.

2.18 DETAILS ON LANGUAGE CODE ISSUES

Section 2.14 describes a variety of issues surrounding language codes that are unclear or incorrect. This section provides more details, focusing on the JW300 dataset.

In table 2.6 we provide a complete table of the datasets where one code is defined as a superset of the other by the ISO standard, and in table 2.7 we provide a complete list of the language codes in JW300 which purport to be sign language but are actually unrelated high-resource languages.

Special attention needs to be given to the JW300 dataset, which, in addition to the sign languages and superset code issues, has a variety of other peculiarities. These problems seem to originate in the codes used by jw.org²², which were apparently

²²The jw.org website seems to use correct BCP-47 extensions now, however, and entering a code such as “`jw_dmr`” redirects to “`naq_x_dmr`”

not checked in the creation of the JW300 dataset. An overview is provided in Table 2.8, and the following paragraphs give specifics.

Twelve languages in JW300 have codes starting in `jw_`, suggesting they are varieties of Javanese (ISO639-1 `jw`), but are instead attempts to represent language dialects for which there are not BCP-47 codes. These codes seem to have been updated in `jw.org` to appropriate BCP-47 private-use extensions in the form `<supercode>_x_<tag>`, which are provided in Table 2.8.

In addition to the `jw_` tags, there are two other mis-used private subtags: `hy_arevmda`, which in addition to lacking the mandatory `_x_` appears to represent standard Western Armenian (`hyw`); and `rmy_AR`, which, rather than being Romany from Argentina, is Kalderash Romany.

There are also a few anomalies where private use extensions should have been used but other methods were found to convey the distinctions. Three codes appear in addition to equivalent ISO codes, making it unclear which languages they are. Two of these are equivalencies between ISO639-2 and ISO639-3 (`nya` and `ny` are both Chichewa, `qu` and `que` are both Quechua). and one is a script equivalency (`kmr` and `kmr_latn` are both in Latin script). In these three cases the two codes do represent different languages — so a private use extension would have been appropriate.

Finally, there is the more minor issue that three languages use the ISO639-3 code instead of the ISO639-2 code, and therefore are not BCP-47.

In addition to the JW300-specific tables, Table 2.11 summarizes misc errors in CCAligned and OSCAR that were detailed in Section 2.14.

2.19 COMPLETE ERROR TAXONOMY AND INSTRUCTIONS

In addition to the table given in table 2.4, raters were provided with the following verbal notes on the error codes

- **CC: Correct translation, natural sentence:** It's OK if it's a sentence fragment instead of a whole sentence, as long as it is not too short (about 5 words or greater). The translation does not have to be perfect.
- **CS: Correct Translation, but single word or short phrase:** Also includes highly repeated short phrases, like "the cat the cat the cat the cat the cat ..."
- **CB: Correct translation, but boilerplate:** This can be auto-generated or formulaic content, or content that one deems "technically correct but generally not very useful to NLP models". Unfortunately, it's often not clear what should be counted as boilerplate...do your best.
- **X: Incorrect translation** [for parallel sentences] both source and target are in the correct language, but they are not adequate translations.

- **WL: Wrong language** For short sentences, especially with proper nouns, there is often a fine line between “Wrong language” and “Not language”. Do your best.
- **NL: Not language** At least one of source and target are not linguistic content. Any sentence consisting only of a proper noun (e.g. “Tyrone Ping”) should be marked as NL.
- **U: Unknown** for sentences that need verification by a native speaker. This is an auxiliary label that is resolved in most cases.

Finally, for future work please consider using the aspirational error taxonomy in appendix [2.24](#), rather than the one presented above.

		Parallel			Monolingual	
		CCAligned	ParaCrawl v7.1	WikiMatrix	OSCAR	mC4
#langs audited / total		65 / 119	21 / 38	20 / 78	51 / 166	48 / 108
%langs audited		54.62%	55.26%	25.64%	30.72%	44.44%
#sents audited / total		8037 / 907M	2214 / 521M	1997 / 95M	3517 / 8.4B	5314 / 8.5B
%sents audited		0.00089%	0.00043%	0.00211%	0.00004%	0.00006%
macro	C	29.25%	76.14%	23.74%	87.21%	72.40%
	X	29.46%	19.17%	68.18%	-	-
	WL	9.44%	3.43%	6.08%	6.26%	15.98%
	NL	31.42%	1.13%	1.60%	6.54%	11.40%
	offensive	0.01%	0.00%	0.00%	0.14%	0.06%
	porn	5.30%	0.63%	0.00%	0.48%	0.36%
micro	C	53.52%	83.00%	50.58%	98.72%	92.66%
	X	32.25%	15.27%	47.10%	-	-
	WL	3.60%	1.04%	1.35%	0.52%	2.33%
	NL	10.53%	0.69%	0.94%	0.75%	5.01%
	offensive	0.00%	0.00%	0.00%	0.18%	0.03%
	porn	2.86%	0.33%	0.00%	1.63%	0.08%
#langs =0% C		7	0	1	7	0
#langs <50% C		44	4	19	11	9
#langs >50% NL		13	0	0	7	1
#langs >50% WL		1	0	0	3	4

Table 2.5: Averages of sentence-level annotations across datasets and selected languages. Macro-avg: Each language is weighted equally in the aggregation, regardless of its size. Micro-avg: Each label is weighted by the fraction of sentences for that language in the overall annotated corpus, i.e., the annotations for higher-represented languages are upweighted, and annotations for lower-represented languages are downweighted. The bottom rows contain the number of languages that have 0% sentences labeled C etc.

Dataset	supercode	subcode(s)
JW300	kg	kwy
JW300	mg	tdx
JW300	qu	que,qug,qus,quw,quy,quz,qvi,qvz
JW300	sw	swc
OSCAR	ar	arz
OSCAR	az	azb
OSCAR	sh	bs,hr,sr
OSCAR	ku	ckb
OSCAR	ms	id,min
OSCAR	no	nn
OSCAR	sq	als*
OSCAR	zh	yue,wuu
Wikimatrix	ar	arz
Wikimatrix	sh	bs,hr,sr
Wikimatrix	zh	wuu

Table 2.6: Situations where two language codes are represented, but one is a superset of another by the ISO standard, leading to unclarity about the data in the supercode dataset. *The als dataset is actually in gsw.

Actual language	Code in JW300
cs	cse
de	gsg
el	gss
en	ase,asf,bfi,ins,psp,sfs,zib,zsl
es	aed,bvl,csf,csg,csn,csr,ecs,esn, gsm,hds,lsp,mfs,ncs,prl,pys,ssp,vsl
fi	fse
fr	fcs,fsl
hu	hsh
id	inl
it	ise
ja	jsl
ko	kvk
pl	pso
pt	bzs,mzy,psr,sgn_AO
ro	rms
ru	rsl
sk	svk
sq	sql
st	jw_ssa
zh	csl,tss

Table 2.7: There are 48 languages in the JW300 corpus with language codes that correspond to sign languages, but in reality are unrelated high-resource languages (usually the most spoken language in the country of origin of the sign language). This table shows the actual language of the data corresponding to each sign language code.

Code in JW300	BCP-47 code	Actual Language Name
Incorrect private-use extensions		
hy_arevmnda	hyw	Western Armenian
jw_dgr	os_x_dgr	Digor Ossetian
jw_dmr	naq_x_dmr	Damara Khoekhoe
jw_ibi	yom_x_ibi	Ibinda Kongo
jw_paa	pap_x_paa	Papiamentu (Aruba)
jw_qcs	qxl	Salasaca Highland Kichwa
jw_rmg	rmn_x_rmg	Greek Romani (South)
jw_rmv	rmy_x_rmv	Vlax Romani, Russia
jw_spl	nso_x_spl	Sepulana
jw_ssa	st_ZA	Sesotho (South Africa)
jw_tpo	pt_PT	Portuguese (Portugal)
jw_vlc	ca_x_vlc	Catalan (Valencia)
jw_vz	skg_x_vz	Vezo Malagasy
rmy_AR	rmy_x_?	Kalderash
Equivalent codes used in place of extensions		
kmr_latn	kmr_x_rdu	Kurmanji (Caucasus)
nya	ny_x_?	Chinyanja (Zambia)
que	qu_x_?	Quechua (Ancash)
Deprecated codes		
daf	dnj/lda	Dan
ISO-693-3 used in place of ISO-693-2		
cat	ca	Catalan
gug	gn	Guarani
run	rn	Kirundi
tso_MZ	ts_MZ	Changana (Mozambique)

Table 2.8: Language code issues in the JW300 datasets for 22 language varieties not covered by Tables 2.6 and 2.7. Twelve languages have codes starting in `jw_`, suggesting they are varieties of Javanese, but are instead mis-parsed private-use extensions. Three codes appear in addition to equivalent ISO codes, making it unclear which languages they are. One language uses a deprecated ISO code. Four languages use the ISO639-3 code instead of the ISO639-2 code, and therefore are not BCP-47. (Note: in this table, private use extensions are given as they appear in `jw.org`, and specified as ‘?’ if they are absent from `jw.org`.)

	es_XX	bm_ML	yo_NG	tr_TR	ku_TR	zh_CN	af_ZA	jv_ID	zh_TW	it_IT	mean
Acc-6	0.58	0.73	0.41	0.45	0.43	0.55	0.65	0.55	0.46	0.55	0.66
Acc-4	0.77	0.73	0.60	0.55	0.56	0.72	0.72	0.57	0.58	0.66	0.72
Acc-2	0.91	0.96	0.72	0.64	0.71	0.79	0.77	0.92	0.81	0.69	0.79

Table 2.9: Rater evaluation for a subset of audits from **CCAligned** (translated from English) measured by the accuracy (Acc- n) of labels assigned by non-proficient speaker against those assigned by proficient speakers. n indicates the granularity of the classes. For $n = 6$ all classes of the taxonomy were distinguished, for $n = 4$ the C subclasses were combined, and for $n = 2$ it is binary decision between C and the rest of the error classes.

	tyv	rm	bar	eml	zh	la	mean
Acc-6	1.0	0.98	1.0	1.0	0.86	1.0	0.98
Acc-4	1.0	1.0	1.0	1.0	0.87	1.0	0.98
Acc-2	1.0	1.0	1.0	1.0	0.87	1.0	0.98

Table 2.10: Rater evaluation for a subset of audits from **OSCAR** measured by the accuracy (Acc- n) of labels assigned by non-proficient speaker against those assigned by proficient speakers. n indicates the granularity of the classes. For $n = 6$ all classes of the taxonomy were distinguished, for $n = 4$ the C subclasses were combined, and for $n = 2$ it is binary decision between C and the rest of the error classes.

corpus	code in corpus	correct code
CCAligned	zz	zza
CCAligned	sz	szl
CCAligned	ns	nso
CCAligned	cb	ckb
CCAligned	tz	ber
CCAligned	qa	shn
CCAligned	qd	kac
CCAligned	cx	ceb
mC4	iw	he
OSCAR	eml	egl
OSCAR	als	gsw
OSCAR	sh	hbs
Wikimatrix	sh	hbs

Table 2.11: Miscellaneous errors in language codes not in other tables (mentioned in the text in Section 2.14).

2.20 NON-PROFICIENT RATER EVALUATION

Tables 2.9 and 2.10 show the detailed rating accuracy scores for all selected languages for several levels of annotation granularity. We can see that for the CCAligned data, reducing the labels to a binary scale naturally increases the accuracy (except for tr_TR), so a binary interpretation (“correct” sentence vs. error) is the most reliable. For monolingual data, the accuracy appears exceptionally high since the bar and tyv corpora contain < 100 sentences each (4 and 25, respectively).

en	The prime minister of the UK is Boris Johnson .
nl	De minister-president van Nederland is Mark Rutte .
en	24 March 2018
pt	14 Novembro 2018
en	The current local time in Sarasota is 89 minutes.
nn	Den lokale tiden i Miami er 86 minutt.
en	In 1932 the highway was extended north to LA .
bar	1938 is de Autobahn bei Inglstod fertig gstellt.

Table 2.12: Examples of “parallel” data where the translation has a different meaning than the source, but the form looks the same. Such data may encourage hallucinations of fake “facts”.

2.21 NOT-SO-PARALLEL DATA

Table 2.12 contains a list of examples from the audited datasets that were misaligned (X). These examples in particular illustrate that structurally similar sentences can easily describe very different facts. Translation models trained on such examples might hallucinate such fact-altering translations.

2.22 QUALITY VS SIZE

To understand the relation between the amount of data available for each language in each corpus and the quality as estimated by our audit, we plot the ratio of X, NL and WL labels against the number of sentences in figures 2.4, 2.5, 2.6, 2.7.

2.23 METHODOLOGICAL NOTES

A surprising amount of work can be done without being an expert in the languages involved. The easiest approach is simply to search the internet for the sentence, which usually results in finding the exact page the sentence came from, which in turn frequently contains clues like language codes in the URL, or a headline like *News in X language*, sometimes with references to a translated version of the same page. However, for the cases where this is insufficient, here are a few tips, tricks, and observations.

NO SKILLS REQUIRED

Things that do not require knowledge of the language(s) in question.

1. “Not language” can usually be identified by anyone who can read the script, though there are tricky cases with proper nouns.

2. Frequently, “parallel” sentences contain different numbers in the source and target (especially autogenerated content), and are easy to disqualify
3. Errors tend to repeat. If a word is mistranslated once, it will often be mistranslated many more times throughout a corpus, making it easy to spot

BASIC RESEARCH REQUIRED

Things that do not require knowledge of the language(s) in question but can be done with basic research.

1. If it’s written in the wrong script it’s considered wrong language. (Sometimes the writing system is indicated in the published corpus, e.g. `bg-Latn`, but usually the language has a “default” script defined by ISO.)
2. Some types of texts come with inherent labels or markers, such as enumerators or verse numbers.
3. When all else fails, search the internet for the whole sentence or n-grams thereof! If the whole sentence can be found, frequently the language is betrayed by the webpage (the language’s autonym is useful in this case).

2.24 ASPIRATIONAL ERROR TAXONOMY

Although the error taxonomy used in this paper did the job, there are a variety of ways to improve both the ease and accuracy of human evaluation, as well as the ease of automatically detecting issues and fixing them. With respect to improved annotations, the error taxonomy presented in this paper lacks at least one significant category of error, namely “correct/in-language but unnatural”. Similarly, the definition of “correct-short” and “correct-boilerplate” were not understood equally by all annotators, leading us to collapse the categories into one for most analyses. Similarly, a concept like “correct-short” has potential issues for agglutinative languages like Turkish. Finally, it was unclear what to do with related dialects, e.g. when a sentence is “almost correct but wrong dialect” or when it is unclear which dialect a sentence belongs to.

Therefore, we present here a slightly modified version which we hope is both more explicit and finer-grained. The main changes are 1) replacing “CB” and “CS” with a catch-all for lower-quality sentences “CL”, and 2) incorporating two codes for languages with related dialects. This is also by no means a perfect rubric, and would benefit from some fine-tuning and workshopping based on the particular dataset or application in question.

2 OSCAR

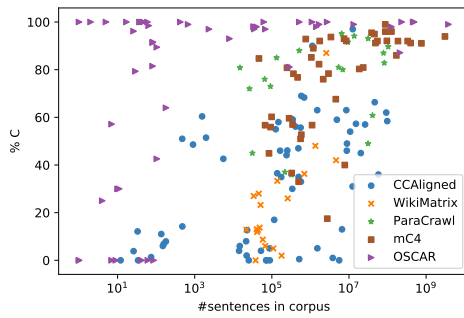


Figure 2.4: Ratio of C ratings vs size.

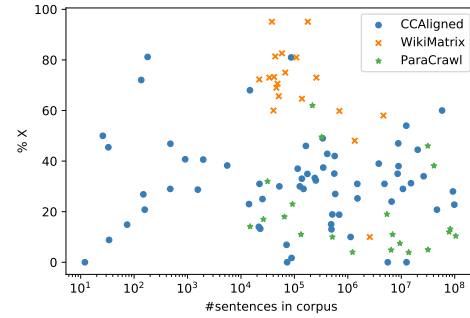


Figure 2.5: Ratio of X ratings vs size.

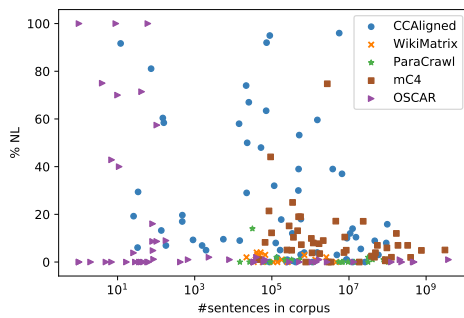


Figure 2.6: Ratio of NL ratings vs size.

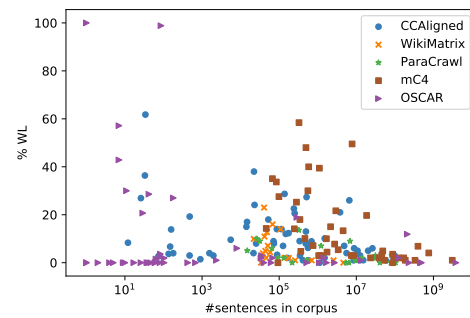


Figure 2.7: Ratio of WL ratings vs size.

- **CC: Correct:** Natural in-language sentence. It's ok if it has a few small issues, like spelling errors or a few words from another language, or if it's a sentence fragment of reasonable length (about 5 words or more). For translations, there may be minor mistakes in the translation.
- **CL: Correct Low-quality:** In-language sentence, but low-quality. This could be ungrammatical text, boilerplate, or very short fragments. For translations, this is the appropriate code for a low-quality translation.
- **X: Incorrect translation** [for parallel sentences] both source and target are in the correct language, but they are not adequate translations.
- **DW: Wrong Dialect** *This code is only applicable for dialects that are closely related to other languages/dialects.* This sentence is in a related but different dialect to the language it's supposed to be in. For instance, it's supposed to be in Sa'idi Arabic but it's in Egyptian Arabic.

- **DA: Ambiguous Dialect** *This code is only applicable for dialects that are closely related to other languages/dialects.* Correct but ambiguous whether it's in the correct language. For instance, many short sentences in Gulf Arabic may also be valid in MSA, and many written Cantonese sentences might also be valid in Mandarin.
- **WL: Wrong language** This sentence is not in the language it's supposed to be. For short sentences, especially with proper nouns, there is often a fine line between "Wrong language" and "Not language". Do your best.
- **NL: Not language** At least one of source and target are not linguistic content. Any sentence consisting only of a proper noun (e.g. "Ibuprofen", "Calvin Klein", or "Washington DC") should be marked as NL
- **U: Unknown** for sentences that need verification by a native speaker. This is an auxiliary label that is resolved in most cases.

Special note on Boilerplate: "Boilerplate" generally refers to autogenerated text found on websites. It's not always clear when a sentence is boilerplate or not. If you see a lot of similar formulaic sentences in the sample, however, that's a good sign that they are boilerplate, and you can mark them all as "CL" ! Common types of boilerplate include sentences like "Convert Euro to Pound", "Online gambling games", "Download Game of Thrones Free Torrent" and so on.

Special note on Mixed Language: Some samples are mixed between the right language and some other language. Some out-of-language content is fine, but a majority out-of-language content is not. We can mark the sentence "CC" if: 1) it is majority in-language, and 2) the in-language portion is more than a short phrase. For unclear border cases you can use "CL".

2.25 COMPLETE TABLES

Tables 2.13, 2.16, 2.17, 2.15, and 2.14 give the complete annotation percentages for CCAIghed, MC4, OSCAR, Paracrawl, and Wikimatrix, respectively.

	C	CC	CS	CB	X	WL	NL	porn	#sentences	avg target length
en-sz_PL	0.00%	0.00%	0.00%	0.00%	0.00%	8.33%	91.67%	0.00%	12	71.42
en-mt_MT	3.85%	0.00%	3.85%	0.00%	50.00%	26.92%	19.23%	0.00%	26	12.58
en-tz_MA	12.12%	6.06%	6.06%	0.00%	45.45%	36.36%	6.06%	0.00%	33	57.33
en-zz_TR	0.00%	0.00%	0.00%	0.00%	8.82%	61.76%	29.41%	0.00%	34	46.53
en-kg_AO	1.35%	0.00%	1.35%	0.00%	14.86%	2.70%	81.08%	0.00%	74	29.20
en-qa_MM	11.03%	5.88%	3.68%	1.47%	72.06%	3.68%	13.24%	0.00%	136	55.28
en-bm_ML	6.04%	4.03%	2.01%	0.00%	26.85%	6.71%	60.40%	0.00%	149	32.19
en-az_IR	6.93%	6.93%	0.00%	0.00%	20.79%	13.86%	58.42%	0.00%	158	115.85
en-qd_MM	7.92%	4.95%	1.98%	0.99%	81.19%	3.96%	6.93%	0.00%	179	60.34
en-ay_BO	51.00%	33.00%	18.00%	0.00%	29.00%	3.00%	17.00%	0.00%	475	92.19
en-ak_GH	14.23%	13.60%	0.63%	0.00%	46.86%	19.25%	19.67%	0.00%	478	45.85
en-st_ZA	48.57%	42.14%	0.00%	6.43%	40.71%	1.43%	9.29%	0.00%	904	111.83
en-ve_ZA	60.40%	29.70%	21.78%	8.91%	28.71%	3.96%	6.93%	0.00%	1555	82.99
en-ts_ZA	51.49%	34.65%	11.88%	4.95%	40.59%	2.97%	4.95%	0.00%	1967	73.93
en-or_IN	42.61%	6.09%	24.35%	12.17%	38.26%	9.57%	9.57%	0.00%	5526	71.39
en-ns_ZA	4.00%	2.00%	0.00%	2.00%	23.00%	15.00%	58.00%	4.00%	14138	33.52
en-lg_UG	6.00%	0.00%	6.00%	0.00%	68.00%	17.00%	9.00%	2.00%	14701	15.83
en-ln_CD	8.00%	4.00%	3.00%	1.00%	14.00%	4.00%	74.00%	4.00%	21562	28.80
en-om_KE	2.00%	2.00%	0.00%	0.00%	31.00%	38.00%	29.00%	24.00%	22206	23.83
en-ss_SZ	12.65%	9.04%	3.61%	0.00%	13.25%	24.10%	50.00%	13.86%	22960	25.30
en-te_IN_rom	0.00%	0.00%	0.00%	0.00%	25.00%	8.00%	67.00%	5.00%	25272	24.21
en-cb_IQ	4.00%	1.00%	3.00%	0.00%	30.00%	18.00%	48.00%	11.00%	52297	30.04
en-tn_BW	0.00%	0.00%	0.00%	0.00%	6.90%	8.97%	63.45%	10.34%	71253	16.80
en-ff_NG	0.00%	0.00%	0.00%	0.00%	0.00%	8.00%	92.00%	2.00%	73022	33.59
en-sn_ZW	5.00%	1.00%	3.00%	1.00%	81.00%	14.00%	0.00%	0.00%	86868	102.59
en-wo_SN	0.00%	0.00%	0.00%	0.00%	1.71%	3.31%	94.98%	18.46%	88441	27.25
en-br_FR	17.00%	3.00%	1.00%	13.00%	37.00%	14.00%	32.00%	1.00%	115128	41.68
en-zu_ZA	55.00%	39.00%	3.00%	13.00%	30.00%	7.00%	8.00%	3.00%	126101	79.32
en-ku_TR	36.52%	12.17%	13.04%	11.30%	33.04%	28.70%	1.74%	1.74%	137874	90.51
en-ig_NG	58.00%	49.00%	3.00%	6.00%	29.00%	12.00%	1.00%	0.00%	148146	83.42
en-kn_IN	46.00%	9.00%	6.00%	31.00%	46.00%	2.00%	5.00%	4.00%	163921	70.20
en-yo_NG	34.93%	6.16%	10.96%	17.81%	34.93%	12.33%	17.81%	0.00%	175192	75.01
en-ky_KG	44.12%	24.51%	17.65%	1.96%	33.33%	22.55%	0.00%	0.98%	240657	69.56
en-ig_TJ	46.08%	18.63%	24.51%	2.94%	32.35%	20.59%	0.98%	4.90%	251865	75.31
en-ha_NG	30.00%	25.00%	3.00%	2.00%	49.00%	9.00%	12.00%	1.00%	339176	60.78
en-am_ET	59.11%	35.47%	2.46%	21.18%	37.44%	2.96%	0.49%	0.00%	346517	58.29
en-km_KH	56.12%	12.24%	33.67%	10.20%	42.86%	1.02%	0.00%	0.00%	412381	71.35
en-ne_NP	47.00%	10.00%	13.00%	24.00%	15.00%	8.00%	30.00%	14.00%	487155	79.14
en-su_ID	35.00%	15.00%	15.00%	5.00%	13.00%	13.00%	39.00%	0.00%	494142	57.08
en-ur_PK_rom	0.50%	0.00%	0.50%	0.00%	18.91%	27.36%	53.23%	5.47%	513123	18.41
en-ht_HT	55.67%	8.25%	10.31%	37.11%	35.05%	6.19%	3.09%	1.03%	558167	101.95
en-mn_MN	33.00%	8.00%	14.00%	11.00%	42.00%	7.00%	18.00%	12.00%	566885	44.43
en-te_IN	69.00%	42.00%	11.00%	16.00%	27.00%	1.00%	3.00%	1.00%	581651	97.95
en-kk_KZ	68.32%	40.59%	18.81%	8.91%	18.81%	8.91%	3.96%	1.98%	689651	72.36
en-be_BY	90.00%	57.00%	13.00%	20.00%	10.00%	0.00%	0.00%	2.00%	1125772	118.45
en-af_ZA	63.00%	40.00%	23.00%	0.00%	31.00%	2.00%	4.00%	12.00%	1504061	105.45
en-jv_ID	5.05%	1.01%	1.01%	3.03%	25.25%	10.10%	59.60%	8.08%	1513974	18.34
en-nl_NL	46.00%	27.00%	19.00%	0.00%	49.00%	2.00%	3.00%	0.00%	36324231	85.95
en-hi_IN_rom	1.00%	0.00%	0.00%	1.00%	39.00%	21.00%	39.00%	8.00%	3789571	18.13
en-lv_LV	59.00%	37.00%	9.00%	13.00%	31.00%	7.00%	3.00%	14.00%	4850957	83.67
en-ar_AR_rom	0.00%	0.00%	0.00%	0.00%	0.00%	4.00%	96.00%	4.00%	5584724	16.69
en-tl_XX	13.00%	6.00%	3.00%	4.00%	24.00%	26.00%	37.00%	5.00%	6593250	37.03
en-uk_UA	63.00%	42.00%	8.00%	13.00%	35.00%	1.00%	1.00%	5.00%	8547348	67.88
en-zh_TW	46.00%	11.00%	31.00%	4.00%	47.00%	6.00%	1.00%	1.00%	8778971	24.89
en-el_GR	49.00%	15.00%	5.00%	29.00%	38.00%	3.00%	10.00%	8.00%	8878492	54.90
en-da_DK	54.00%	31.00%	18.00%	5.00%	29.00%	5.00%	12.00%	7.00%	10738582	73.99
en-vi_VN	31.00%	18.00%	0.00%	13.00%	54.00%	1.00%	14.00%	6.00%	12394379	74.19
en-sv_SE	97.00%	91.00%	3.00%	3.00%	0.00%	3.00%	0.00%	0.00%	12544075	103.91
en-zh_CN	57.29%	22.92%	12.50%	21.88%	31.25%	1.04%	10.42%	1.04%	15181410	33.55
en-tr_TR	45.00%	14.50%	14.00%	16.50%	44.50%	5.00%	5.50%	4.00%	20282339	83.80
en-ja_XX	57.00%	35.00%	21.00%	1.00%	34.00%	6.00%	0.00%	0.00%	26201214	34.44
en-pt_XX	66.34%	36.63%	10.89%	18.81%	20.79%	3.96%	8.91%	0.00%	46525410	87.20
en-it_IT	36.00%	14.00%	18.00%	4.00%	60.00%	1.00%	3.00%	0.00%	58022366	97.44
en-de_DE	62.00%	29.00%	14.00%	19.00%	28.00%	2.00%	8.00%	2.00%	92597196	78.08
en-es_XX	58.42%	16.83%	25.74%	15.84%	22.77%	2.97%	15.84%	4.95%	98351611	72.18
mean	27.01%	29.35%	8.62%	28.97%	14.48%	6.49%	5.89%	0.00%	5.26%	

Table 2.13: Audit results for a sample of 100 sentences from **CCAligned** for each language pair, compared to the number of sentences available in the dataset. If fewer than 100 sentences were available, all sentences were audited. Language codes are as originally published. The length is measured in number of characters and averaged across the audited portion of each corpus. Languages with less than 20% correct sentences are boldfaced.

	C	CC	CS	CB	X	WL	NL	porn	# sentences	avg target length
en-ug	12.87%	8.91%	1.98%	1.98%	72.28%	9.90%	1.98%	0.00%	22012	95.55
en-mwl	27.00%	26.00%	0.00%	1.00%	73.00%	0.00%	0.00%	0.00%	33899	135.26
en-tg	0.00%	0.00%	0.00%	0.00%	95.10%	3.92%	0.98%	0.00%	37975	88.87
en-ne	13.00%	7.00%	6.00%	0.00%	60.00%	23.00%	4.00%	0.00%	40549	69.26
en-ka	11.88%	2.97%	2.97%	5.94%	73.27%	10.89%	2.97%	0.00%	41638	144.74
en-lmo	12.75%	11.76%	0.00%	0.98%	81.37%	4.90%	0.98%	0.00%	43790	89.38
en-io	28.00%	27.00%	0.00%	1.00%	69.00%	2.00%	1.00%	0.00%	45999	83.26
en-jv	13.73%	9.80%	0.00%	3.92%	70.59%	12.75%	2.94%	0.00%	48301	91.87
en-wuu	23.23%	14.14%	7.07%	2.02%	65.66%	7.07%	4.04%	0.00%	51024	34.77
br-en	8.70%	7.61%	1.09%	0.00%	82.61%	4.35%	0.00%	0.00%	58400	90.68
bar-en	6.00%	6.00%	0.00%	0.00%	75.00%	16.00%	3.00%	0.00%	67394	103.51
en-kk	5.00%	2.00%	2.00%	1.00%	81.00%	14.00%	0.00%	0.00%	109074	56.03
en-sw	33.33%	27.27%	4.04%	2.02%	64.65%	2.02%	0.00%	0.00%	138590	111.61
en-nds	1.96%	1.96%	0.00%	0.00%	95.10%	1.96%	0.98%	0.00%	178533	91.95
be-en	26.00%	24.00%	2.00%	0.00%	73.00%	1.00%	0.00%	0.00%	257946	121.22
en-hi	36.27%	32.35%	0.98%	2.94%	59.80%	0.98%	2.94%	0.00%	696125	96.77
en-ko	48.04%	33.33%	2.94%	11.76%	48.04%	2.94%	0.98%	0.00%	1345630	55.18
en-uk	87.00%	84.00%	2.00%	1.00%	10.00%	1.00%	2.00%	0.00%	2576425	104.39
en-it	42.00%	42.00%	0.00%	0.00%	58.00%	0.00%	0.00%	0.00%	4626048	140.27
en-simple	37.62%	24.75%	0.00%	12.87%	56.44%	2.97%	2.97%	0.00%	nan	77.53

Table 2.14: Audit results for a sample of 100 sentences from **WikiMatrix** for each language pair, compared to the number of sentences available in the dataset. Language codes are as originally published. The length is measured in number of characters and averaged across the audited portion of each corpus. Languages with less than 20% correct sentences are boldfaced.

	C	CC	CS	CB	X	WL	NL	porn	# sentences	avg target length
en-so	80.81%	61.62%	1.01%	18.18%	14.14%	5.05%	0.00%	0.00%	14879	189.83
en-ps	72.00%	53.00%	9.00%	10.00%	17.00%	10.00%	0.00%	0.00%	26321	141.01
en-my	45.00%	9.00%	16.00%	20.00%	32.00%	9.00%	14.00%	0.00%	31374	147.07
en-km	76.00%	51.00%	13.00%	12.00%	18.00%	6.00%	0.00%	0.00%	65113	121.20
en-ne	73.00%	48.00%	1.00%	24.00%	23.00%	2.00%	0.00%	0.00%	92084	153.42
en-sw	85.00%	60.00%	15.00%	10.00%	11.00%	2.00%	2.00%	0.00%	132517	167.34
en-si	37.00%	31.00%	6.00%	0.00%	62.00%	0.00%	1.00%	0.00%	217407	123.06
en-nn	35.92%	24.27%	8.74%	2.91%	49.51%	13.59%	0.97%	0.00%	323519	56.24
es-eu	88.00%	66.00%	15.00%	7.00%	10.00%	1.00%	1.00%	0.00%	514610	121.31
es-gl	89.00%	46.00%	6.00%	37.00%	4.00%	7.00%	0.00%	0.00%	1222837	107.88
en-ru	81.00%	73.00%	6.00%	2.00%	19.00%	0.00%	0.00%	6.00%	5377911	101.28
en-bg	95.15%	85.44%	0.97%	8.74%	4.85%	0.00%	0.00%	0.97%	6470710	112.29
es-ca	80.00%	54.00%	19.00%	7.00%	11.00%	9.00%	0.00%	5.00%	6870183	107.21
en-el	91.59%	68.22%	0.93%	22.43%	7.48%	0.93%	0.00%	0.00%	9402646	135.66
en-pl	94.12%	76.47%	0.98%	16.67%	3.92%	1.96%	0.00%	0.98%	13744860	95.95
en-nl	49.00%	32.00%	17.00%	0.00%	46.00%	3.00%	2.00%	0.00%	31295016	95.05
en-pt	93.07%	92.08%	0.00%	0.99%	4.95%	1.98%	0.00%	0.00%	31486963	108.68
en-it	60.82%	36.08%	16.49%	8.25%	38.14%	0.00%	1.03%	0.00%	40798278	127.55
en-es	87.00%	54.00%	20.00%	13.00%	12.00%	0.00%	1.00%	0.50%	78662122	119.72
en-de	82.83%	64.65%	13.13%	5.05%	13.13%	3.03%	1.01%	0.00%	82638202	111.43
en-fr	89.62%	82.08%	4.72%	2.83%	10.38%	0.00%	0.00%	0.00%	104351522	144.20

Table 2.15: Audit results for a sample of 100 sentences from **ParaCrawl** for each language pair, compared to the number of sentences available in the dataset. Language codes are as originally published. The length is measured in number of characters and averaged across the audited portion of each corpus.

	C	CC	CS	CB	WL	NL	porn	# sentences	avg length
yo	84.69%	71.43%	2.04%	11.22%	14.29%	1.02%	0.00%	46214	117.71
st	56.70%	42.27%	14.43%	0.00%	35.05%	8.25%	0.00%	66837	132.13
haw	44.90%	34.69%	1.02%	9.18%	33.67%	21.43%	1.02%	84312	129.99
ig	55.91%	41.73%	10.24%	3.94%	0.00%	44.09%	0.79%	92909	98.03
sm	60.20%	58.16%	2.04%	0.00%	27.55%	12.24%	0.00%	98467	126.42
ha	80.81%	79.80%	1.01%	0.00%	14.14%	5.05%	2.02%	247479	155.76
su	59.60%	58.59%	1.01%	0.00%	25.25%	15.15%	2.02%	280719	107.10
sn	36.63%	32.67%	2.97%	0.99%	58.42%	4.95%	0.00%	326392	145.59
mg	57.00%	57.00%	0.00%	0.00%	18.00%	25.00%	0.00%	345040	116.23
pa	78.30%	68.87%	3.77%	5.66%	4.72%	10.38%	0.00%	363399	134.43
ga	76.77%	58.59%	6.06%	12.12%	10.10%	13.13%	0.00%	465670	147.35
co	33.00%	29.00%	2.00%	2.00%	48.00%	19.00%	0.00%	494913	195.30
zu	51.00%	48.00%	2.00%	1.00%	30.00%	19.00%	0.00%	555458	137.81
jv	52.73%	19.09%	19.09%	14.55%	40.00%	7.27%	1.82%	581528	97.96
km	92.86%	92.86%	0.00%	0.00%	7.14%	0.00%	0.00%	756612	162.57
kn	85.15%	73.27%	3.96%	7.92%	2.97%	9.90%	0.00%	1056849	105.39
fy	56.73%	50.00%	3.85%	2.88%	39.42%	3.85%	0.00%	1104359	234.25
te	89.00%	76.00%	9.00%	4.00%	3.00%	8.00%	0.00%	1188243	108.49
la	82.31%	65.38%	6.15%	10.77%	10.00%	7.69%	0.00%	1674463	67.25
be	92.04%	86.73%	2.65%	2.65%	4.42%	3.54%	0.00%	1742030	110.86
af	76.00%	76.00%	0.00%	0.00%	15.00%	9.00%	0.00%	2152243	99.52
lb	17.48%	17.48%	0.00%	0.00%	7.77%	74.76%	0.00%	2740336	481.68
ne	78.35%	77.32%	1.03%	0.00%	21.65%	0.00%	0.00%	2942785	102.88
sr	93.69%	85.59%	7.21%	0.90%	5.41%	0.00%	0.00%	3398483	131.72
gl	67.62%	57.14%	10.48%	0.00%	13.33%	17.14%	0.00%	4549465	151.45
bn	93.00%	86.00%	1.00%	6.00%	3.00%	4.00%	0.00%	7444098	92.60
mr	40.00%	35.24%	2.86%	1.90%	49.52%	10.48%	0.00%	7774331	281.94
sl	92.08%	82.18%	4.95%	4.95%	2.97%	4.95%	0.00%	8499456	149.45
hi	80.30%	76.77%	1.01%	2.53%	19.70%	0.00%	2.53%	18507273	105.54
bg	80.90%	75.88%	2.51%	2.51%	2.01%	17.09%	0.00%	23409799	93.86
uk	95.48%	81.41%	7.54%	6.53%	2.01%	2.51%	0.00%	38556465	116.79
ro	94.95%	78.79%	12.12%	4.04%	3.03%	2.02%	0.00%	45738857	130.08
sv	91.18%	84.31%	2.94%	3.92%	4.90%	3.92%	1.96%	48570979	114.45
zh	92.00%	87.00%	1.00%	4.00%	1.00%	7.00%	0.00%	54542308	94.77
ja	99.00%	89.00%	6.00%	4.00%	0.00%	1.00%	1.00%	87337884	59.94
tr	95.96%	88.89%	0.00%	7.07%	3.54%	0.51%	0.00%	87595290	152.75
nl	92.08%	85.15%	6.93%	0.00%	1.98%	5.94%	0.00%	96210458	103.67
pl	96.00%	82.00%	7.00%	7.00%	2.00%	2.00%	0.00%	126164277	170.70
pt	86.00%	79.00%	4.00%	3.00%	2.00%	12.00%	1.00%	169239084	133.51
it	92.00%	79.00%	9.00%	4.00%	1.00%	7.00%	0.00%	186404508	180.26
fr	92.00%	82.00%	7.00%	3.00%	1.00%	7.00%	0.00%	332674575	143.69
de	91.18%	77.45%	7.84%	5.88%	6.86%	1.96%	0.00%	397006993	107.71
ru	91.06%	69.11%	11.38%	10.57%	4.07%	4.88%	0.00%	755585265	109.28
en	93.94%	83.84%	8.08%	2.02%	1.01%	5.05%	0.00%	3079081989	130.97
bg_latn	9.09%	9.09%	0.00%	0.00%	51.52%	39.39%	1.01%	N/A	139.92
ja_latn	13.00%	7.00%	4.00%	2.00%	60.00%	27.00%	0.00%	N/A	218.92
ru_latn	36.45%	25.23%	10.28%	0.93%	34.58%	28.97%	0.93%	N/A	123.14
zh_latn	5.00%	4.00%	1.00%	0.00%	64.00%	31.00%	0.00%	N/A	186.84

Table 2.16: Audit results for a sample of 100 sentences from **mC4** for each language, compared to the number of sentences available in the dataset. Language codes are as originally published. The length is measured in number of characters and averaged across the audited portion of each corpus. Languages with less than 20% correct sentences are boldfaced.

	C	CC	CS	CB	WL	NL	porn	# sentences	avg length
diq	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	1	131.00
bcl	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	1	623.00
cbk	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	1	519.00
pam	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	2	139.00
bar	25.00%	25.00%	0.00%	0.00%	0.00%	75.00%	0.00%	4	53.50
myv	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	5	127.00
yue	0.00%	0.00%	0.00%	0.00%	57.14%	42.86%	0.00%	7	177.00
mw1	57.14%	57.14%	0.00%	0.00%	42.86%	0.00%	0.00%	7	141.00
frr	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	9	231.56
ht	30.00%	30.00%	0.00%	0.00%	0.00%	70.00%	0.00%	10	329.10
ie	30.00%	30.00%	0.00%	0.00%	30.00%	40.00%	0.00%	11	121.70
scn	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	17	155.59
tyv	96.15%	96.15%	0.00%	0.00%	0.00%	3.85%	0.00%	26	167.96
mai	79.31%	75.86%	0.00%	3.45%	20.69%	0.00%	0.00%	29	141.17
bxr	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	37	160.76
dsb	100.00%	97.56%	0.00%	2.44%	0.00%	0.00%	0.00%	41	155.15
so	0.00%	0.00%	0.00%	0.00%	28.57%	71.43%	0.00%	42	208.24
rm	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	47	137.66
nah	100.00%	96.67%	0.00%	3.33%	0.00%	0.00%	0.00%	60	164.53
nap	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	61	152.11
yo	98.46%	96.92%	0.00%	1.54%	1.54%	0.00%	0.00%	64	281.57
gn	81.48%	81.48%	0.00%	0.00%	2.47%	16.05%	0.00%	81	234.95
vec	91.36%	91.36%	0.00%	0.00%	0.00%	8.64%	0.00%	81	184.90
kw	91.57%	90.36%	0.00%	1.20%	3.61%	4.82%	0.00%	83	162.75
wuu	0.00%	0.00%	0.00%	0.00%	98.84%	1.16%	0.00%	86	157.15
eml	42.57%	42.57%	0.00%	0.00%	0.00%	57.43%	0.00%	104	177.88
bh	89.42%	21.15%	0.00%	68.27%	1.92%	8.65%	0.00%	104	137.17
min	64.00%	6.00%	0.00%	58.00%	27.00%	9.00%	0.00%	180	649.85
qu	100.00%	98.97%	0.00%	1.03%	0.00%	0.00%	0.00%	425	167.27
su	99.00%	99.00%	0.00%	0.00%	0.00%	1.00%	0.00%	676	221.00
jv	97.00%	86.00%	0.00%	11.00%	1.00%	2.00%	0.00%	2350	203.08
als	93.00%	93.00%	0.00%	0.00%	6.00%	1.00%	0.00%	7997	375.44
la	98.00%	98.00%	0.00%	0.00%	2.00%	0.00%	0.00%	33838	224.11
uz	98.00%	98.00%	0.00%	0.00%	2.00%	0.00%	0.00%	34244	369.99
nds	97.03%	95.05%	0.00%	1.98%	2.97%	0.00%	0.00%	35032	344.74
sw	98.00%	98.00%	0.00%	0.00%	0.00%	2.00%	0.00%	40066	196.70
br	100.00%	96.00%	0.00%	4.00%	0.00%	0.00%	0.00%	61941	239.56
fy	97.00%	97.00%	0.00%	0.00%	2.00%	1.00%	0.00%	67762	340.23
am	81.09%	79.10%	0.00%	1.99%	18.91%	0.00%	0.00%	287142	267.43
af	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	517353	339.18
eu	100.00%	98.00%	0.00%	2.00%	0.00%	0.00%	0.00%	1099498	330.93
mn	98.00%	94.00%	0.00%	4.00%	2.00%	0.00%	0.00%	1430527	309.94
te	98.99%	93.94%	1.01%	4.04%	0.00%	1.01%	1.01%	1685185	412.31
kk	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	2719851	318.93
ca	99.00%	91.00%	0.00%	8.00%	1.00%	0.00%	0.00%	13292843	333.38
nl	98.00%	94.00%	2.00%	2.00%	2.00%	0.00%	4.00%	126067610	305.01
it	87.13%	71.29%	1.98%	13.86%	11.88%	0.99%	1.98%	210348435	393.66
zh	100.00%	97.00%	0.00%	3.00%	0.00%	0.00%	1.00%	232673578	195.60
fr	100.00%	93.00%	0.00%	7.00%	0.00%	0.00%	5.00%	461349575	306.62
es	100.00%	94.00%	0.00%	6.00%	0.00%	0.00%	3.00%	488616724	268.07
en	99.00%	96.00%	0.00%	3.00%	0.00%	1.00%	1.00%	3809525119	364.65

Table 2.17: Audit results for a sample of 100 sentences from **OSCAR** for each language, compared to the number of sentences available in the dataset. If fewer than 100 sentences were available, all sentences were audited. Language codes are as originally published. Length is measured in number of characters. Languages with less than 20% correct sentences are boldfaced.

2 OSCAR

corpus	language	C	CC	CS	CB	X	WL	NL	porn	#sentences	avg target length
CCAligned	en-tz_MA	12.12%	6.06%	6.06%	0.00%	45.45%	36.36%	6.06%	0.00%	33	57.33
CCAligned	en-kg_AO	1.35%	0.00%	1.35%	0.00%	14.86%	2.70%	81.08%	0.00%	74	29.20
CCAligned	en-bm_ML	6.04%	4.03%	2.01%	0.00%	26.85%	6.71%	60.40%	0.00%	149	32.19
CCAligned	en-ak_GH	14.23%	13.60%	0.63%	0.00%	46.86%	19.25%	19.67%	0.00%	478	45.85
CCAligned	en-st_ZA	48.57%	42.14%	0.00%	6.43%	40.71%	1.43%	9.29%	0.00%	904	111.83
CCAligned	en-ve_ZA	60.40%	29.70%	21.78%	8.91%	28.71%	3.96%	6.93%	0.00%	1555	82.99
CCAligned	en-ts_ZA	51.49%	34.65%	11.88%	4.95%	40.59%	2.97%	4.95%	0.00%	1967	73.93
CCAligned	en-ns_ZA	4.00%	2.00%	0.00%	2.00%	23.00%	15.00%	58.00%	4.00%	14138	33.52
CCAligned	en-ig_UG	6.00%	0.00%	6.00%	0.00%	68.00%	17.00%	9.00%	2.00%	14701	15.83
CCAligned	en-ln_CD	8.00%	4.00%	3.00%	1.00%	14.00%	4.00%	74.00%	4.00%	21562	28.80
CCAligned	en-om_KE	2.00%	2.00%	0.00%	0.00%	31.00%	38.00%	29.00%	24.00%	22206	23.83
CCAligned	en-ss_SZ	12.65%	9.04%	3.61%	0.00%	13.25%	24.10%	50.00%	13.86%	22960	25.30
CCAligned	en-tn_BW	0.00%	0.00%	0.00%	0.00%	6.90%	8.97%	63.45%	10.34%	71253	16.80
CCAligned	en-ff_NG	0.00%	0.00%	0.00%	0.00%	0.00%	8.00%	92.00%	2.00%	73022	33.59
CCAligned	en-sn_ZW	5.00%	1.00%	3.00%	1.00%	81.00%	14.00%	0.00%	0.00%	86868	102.59
CCAligned	en-wo_SN	0.00%	0.00%	0.00%	0.00%	1.71%	3.31%	94.98%	18.46%	88441	27.25
CCAligned	en-zu_ZA	55.00%	39.00%	3.00%	13.00%	30.00%	7.00%	8.00%	3.00%	126101	79.32
CCAligned	en-ig_NG	58.00%	49.00%	3.00%	6.00%	29.00%	12.00%	1.00%	0.00%	148146	83.42
CCAligned	en-yo_NG	34.93%	6.16%	10.96%	17.81%	34.93%	12.33%	17.81%	0.00%	175192	75.01
CCAligned	en-ha_NG	30.00%	25.00%	3.00%	2.00%	49.00%	9.00%	12.00%	1.00%	339176	60.78
CCAligned	en-am_ET	59.11%	35.47%	2.46%	21.18%	37.44%	2.96%	0.49%	0.00%	346517	58.29
CCAligned	en-af_ZA	63.00%	40.00%	23.00%	0.00%	31.00%	2.00%	4.00%	12.00%	1504061	105.45
CCAligned	en-ar_AR_rom	0.00%	0.00%	0.00%	0.00%	0.00%	4.00%	96.00%	4.00%	5584724	16.69
Wikimatrix	en-sw	33.33%	27.27%	4.04%	2.02%	64.65%	2.02%	0.00%	0.00%	138590	111.61
ParaCrawl	en-so	80.81%	61.62%	1.01%	18.18%	14.14%	5.05%	0.00%	0.00%	14879	189.83
ParaCrawl	en-sw	85.00%	60.00%	15.00%	10.00%	11.00%	2.00%	2.00%	0.00%	132517	167.34
MC4	yo	84.69%	71.43%	2.04%	11.22%	N/A	14.29%	1.02%	0.00%	46214	117.71
MC4	st	56.70%	42.27%	14.43%	0.00%	N/A	35.05%	8.25%	0.00%	66837	132.13
MC4	ig	55.91%	41.73%	10.24%	3.94%	N/A	0.00%	44.09%	0.79%	92909	98.03
MC4	ha	80.81%	79.80%	1.01%	0.00%	N/A	14.14%	5.05%	2.02%	247479	155.76
MC4	sn	36.63%	32.67%	2.97%	0.99%	N/A	58.42%	4.95%	0.00%	326392	145.59
MC4	mg	57.00%	57.00%	0.00%	0.00%	N/A	18.00%	25.00%	0.00%	345040	116.23
MC4	zu	51.00%	48.00%	2.00%	1.00%	N/A	30.00%	19.00%	0.00%	555458	137.81
MC4	af	76.00%	76.00%	0.00%	0.00%	N/A	15.00%	9.00%	0.00%	2152243	99.52
OSCAR	so	0.00%	0.00%	0.00%	0.00%	N/A	28.57%	71.43%	0.00%	42	208.24
OSCAR	yo	98.46%	96.92%	0.00%	1.54%	N/A	1.54%	0.00%	0.00%	64	281.57
OSCAR	sw	98.00%	98.00%	0.00%	0.00%	N/A	0.00%	2.00%	0.00%	40066	196.70
OSCAR	am	81.09%	79.10%	0.00%	1.99%	N/A	18.91%	0.00%	0.00%	287142	267.43
OSCAR	af	100.00%	100.00%	0.00%	0.00%	N/A	0.00%	0.00%	0.00%	517353	339.18

Table 2.18: Results on African languages.

With the increasing interest in language modeling in recent years in Natural Language Processing (NLP) (Rogers et al., 2020), particularly concerning contextualized word representations²³ (Peters et al., 2018; Devlin et al., 2019), there has also been an explosion in interest for large raw corpora, as some of these latest models require almost 1TiB of raw text for pre-training (Raffel et al., 2020; Brown et al., 2020).

While most of these language models were initially trained in English (Devlin et al., 2019; Yang et al., 2019; Clark et al., 2020; Zaheer et al., 2020; Xiong et al., 2021) and consequently most of the large corpora used to pre-train them were in English, there has been a recent push to produce larger high quality corpora for other languages, namely those of Grave et al. (2018), CCNet (Wenzek et al., 2020), Multilingual C4 (mC4) (Xue et al., 2021) and OSCAR (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020) for pre-training language models, as well as, Paracrawl (Esplà et al., 2019; Bañón et al., 2020), CCAIghed (El-Kishky et al., 2020) and WikiMatrix (Schwenk et al., 2021) which are parallel corpora for training Machine Translation (MT) models. Of these, only OSCAR, Paracrawl, CCAIghed and WikiMatrix are freely available and easily downloadable.

In this paper we propose a new multilingual corpus for language modeling, and for that we take inspiration in the OSCAR corpus and its pipeline *goclassy*²⁴ (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020), but we propose a new pipeline *Ungoliant*²⁵ that is faster, modular, parametrizable and well-documented. We then use it to produce a new corpus similar to OSCAR, yet larger, based on recent data containing mentions of last years' events such as the COVID-19 pandemic, the 2020–2021 United States racial unrest, the Australian wildfires, the Beirut explosion and Brexit among others. Moreover, contrarily to OSCAR, our corpus retains metadata information at the document level. We release our pipeline under an Apache 2.0 open source license and we publish the corpus under a research-only use license following the licensing schemes proposed by OSCAR (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020) and Paracrawl (Esplà et al., 2019; Bañón et al., 2020).

2.26 LIMITATIONS OF THE OSCAR CORPUS AND ITS GENERATION PIPELINE

2.26.1 OSCAR

OSCAR is a multilingual corpus derived from CommonCrawl²⁶, a project that provides web crawl data for everyone on a periodic manner, usually each month. CommonCrawl provides data in several formats, from raw HTML source code to pure text.

²³In which one takes a unannotated large textual corpus in a particular language and tries to predict a missing word in order to learn a vector space representation for it.

²⁴<https://github.com/oscar-corpus/goclassy>

²⁵<https://github.com/oscar-corpus/ungoliant>

²⁶<https://commoncrawl.org>

OSCAR was generated from the pure text data version (WET files) of the November 2018 crawl, distributed in the form of 56,000 *shards*, that were then filtered and classified by language (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020). OSCAR is available through several means, and has been used in numerous projects (Ortiz Suárez et al., 2019). OSCAR’s generation pipeline also suffers from numerous issues, which we plan to address simultaneously with the release of a new, more powerful, stable, and higher quality pipeline

Simply put, OSCAR is composed of single language files that contain textual data (ta.txt for the Tamil language, for example). However, due to the often huge sizes of these files, and subsequently the impracticality of storage and distribution, OSCAR files are split and compressed in equally sized parts.

OSCAR comes in four different versions, each suited differently for different tasks, and allows less limited ways of sharing the corpus more widely. These versions are either *unshuffled* or *shuffled* (that is, for each language, lines have been shuffled, destroying records integrity), and *non-deduplicated* or *deduplicated* (since duplicate lines account for more than half of the total data²⁷ generated by the pipeline). For the unshuffled versions, each language file contains paragraphs that come from the same record, and each paragraph is separated by a newline.

OSCAR is inherently linked to its generation pipeline, and as such its quality partly depends on the pipeline’s quality. While OSCAR is considered to be one of the cleanest multilingual corpora available (Caswell et al., 2020; Caswell et al., 2021), several problems have been described, and the state of the publicly available code raises questions about maintenance and maintainability of the pipeline itself.

Apart from the fact that its content dates back to 2018, the current OSCAR corpus suffers from quality issues discussed in (Caswell et al., 2020; Caswell et al., 2021), including:

- **Language label mismatches and inconsistencies**, which occurs earlier in the pipeline and would be fixable downstream,
- **Representation washing** as defined by Caswell et al. (2021), whereby low resource languages, while present in the corpus, are of a significantly lower quality than higher resource languages without any quality metric available publicly.

The most recent Common Crawl dump contains 64,000 shards. Each shard is composed of numerous records, and each record holds textual content along with metadata. While CommonCrawl shards hold document-level metadata that could be useful downstream, they were discarded and do not appear in OSCAR, whereas other corpora generated from the same source include them, e.g. CCNet (Wenzek et al., 2020). This limits OSCAR users to the textual content only, whereas metadata could have been distributed along with the corpus itself.

²⁷OSCAR-orig: 6.3TB, OSCAR-dedup: 3.2TB

2.26.2 GOCLASSY

OSCAR was built using *goclassy*, a high-performance asynchronous pipeline written in Go (Ortiz Suárez et al., 2019). However, it suffers from several caveats that makes the re-generation and update of the corpus relatively complex in practice.

While *goclassy*'s source code is easily readable thanks to the choice of an uncluttered language and a pragmatic approach, the lack of structure in both the source and the project itself makes *goclassy* difficult to extend and maintain.

The pipeline is not functional out-of-the-box, as the user has to provide the compressed shards from CommonCrawl, manually install *fasttext* (Joulin et al., 2016; Joulin et al., 2017) and create specific directories by themselves, since only partial instructions are given in the supplied README file.

goclassy also makes heavy use of I/O, as data is saved and loaded repeatedly between steps; as an example, the identification step stores language identification data and individual sentences in two files, before generating the final files (one per language). Despite these limitations, *goclassy*'s performance is good due to Go's emphasis on easy and efficient parallelization and inherent speed. The pipeline uses clever handling of file descriptors, limiting I/O calls cost in some parts.

2.27 BUILDING A NEW OSCAR-LIKE CORPUS

We introduce *Ungoliant*, a new corpus generation pipeline that, like *goclassy*, creates a large-scale multilingual text corpus from a CommonCrawl dump. Contrarily to *goclassy*, *Ungoliant* is fully modular, better structured, and highly parametrizable; thereby allowing comparisons between several parallelization strategies. A specific effort was put in testing and documentation. Parts of *Ungoliant* are heavily inspired by *goclassy*, although it is implemented in Rust rather than in Go, which is sometimes faster.²⁸

Additionally, we use *Ungoliant* to generate a new corpus from a recent Common Crawl dump. The new corpus includes metadata information while retaining backward compatibility with the OSCAR corpus.

2.27.1 UNGOLIANT

RATIONALE AND SCOPE

While *Ungoliant* is heavily inspired by *goclassy*, it provides a better set of tools to download, process, filter and aggregate textual and contextual data from CommonCrawl. These operations can be sequential, parallel or both, depending on contexts and performance requirements.

²⁸<https://benchmarksgame-team.pages.debian.net/benchmarksgame/fastest/rust-go.html>

Platform	#shards	goclassy	Ungoliant	Approx. speedup
Desktop	1	30s	13s	×2.3
	10	3m6s	2m12s	×1.3
	25	9m10s	5m47s	×1.5
HPC	1	40s	6s	×6.6
	25	2m40s	1m6s	×2.4
	100	7m59s	4m14s	×1.8

Table 2.19: Comparison of approximate generation times depending on platform and number of shards.

We provide both batch and streaming processing, so that the whole pipeline could be run either online, with every step running on streams of data, or offline, with every step running on tangible files, or a mix of both, using already downloaded CommonCrawl dumps but streaming the rest of the process. Moreover, we embed numerous filtering and deduplication utilities directly inside Ungoliant, making these features available for pipeline composition and post-processing.

Ungoliant features a loosely defined pipeline interface, on which we re-implement goclassy’s one, while improving performance by threading more aggressively and avoiding I/O where it is not necessary: While goclassy uses intermediate files for tags and sentences, we try to keep everything in memory in order to avoid losing time loading or writing files. The Rust language provides constructs that helps us build complex abstractions and pipelines while limiting proactive file I/O or computing, since nearly all the reimplemented pipeline is built around lazy evaluation. File I/O is only used when loading shards, and when writing sentences in language files.

Through benchmarking we found that the best parallelization strategy is to use rayon²⁹, a work-stealing (Blumofe and Leiserson, 1999) parallel and concurrent library enabling massive parallelization. We parallelize on shard-, record- and sentence-level processing.

To evaluate Ungoliant performance, we run both goclassy and Ungoliant’s implementation on 1, 10, 25 and 100 Common Crawl shards both on a middle-range laptop computer (i5-7200u, 8GB RAM, NVMe SSD) and a HPC node (Xeon 5218 (64 Threads), 180GB RAM). Results are shown in Table 2.19.

Ungoliant performs better than goclassy on all tasks, independently of the platform or number of shards processed. However, we can note that Ungoliant’s speedup is higher on short tasks, which is explained by its aggressive multithreading strategy, while goclassy uses a record-scope multithreading at its finest granularity.

2.27.2 ITERATING ON THE GOCLASSY PIPELINE

CommonCrawl dumps contain metadata that hold useful information such as related records, recognized language(s), or origin URLs. Since OSCAR pipeline discards

²⁹<https://github.com/rayon-rs/rayon>

metadata and sentences can be shuffled, we lose the ability to investigate those metadata themselves, as well as working on potentially multilingual documents, since we separate text from metadata.

The new pipeline (and the resulting new corpus schema) aims to establish a first link between textual data and metadata from CommonCrawl, while staying backward compatible with the existing OSCAR schema.

In other words, switching from the original OSCAR corpus and the newly generated one should be a drop-in operation.

METADATA EXTRACTION AND LINKING

Our choice of keeping the corpus backward compatible with the original OSCAR introduces changes in the way the corpus is generated, namely regarding metadata: a record's body is composed of sentences that aren't guaranteed to be of the same language. Since OSCAR merges sentences from multiple records into a single file, special attention has to be paid to the metadata dispatch too.

Approaches to tackle this problem range from (1) storing all metadata in a single location to (2) having language-specific metadata files that contain the metadata for each line in the language file.

Both (1) and (2) have their strengths and weaknesses, namely:

1. Having all metadata at the same place may facilitate wide queries about whole metadata, but at a cost of a very large size (which harms both accessibility and performance).
2. Getting the metadata for a given line is fast since line numbers are synchronized, but there is repeated information and a potentially important increase in size.

We choose a hybrid approach which keeps metadata local to each language, while trying to limit the information repetition by keeping an entry by group of chunks rather than by line, where a chunk is a series of contiguous sentences that share the same language from the same document.

An overview of the pipeline can be seen in Figure 2.8, with a more precise view on record processing and metadata extraction in Figure 2.8.

Metadata are distributed via JSON-encoded files holding an ordered list of metadata entries, along with offsets (o) and paragraph lengths (l), enabling any user to get the content of a said metadata by querying for lines $(o, o + l]$ in the content file.

This approach still has drawbacks, in particular when looking for the corresponding metadata of a given sentence/paragraph, where one has to perform a search on the metadata file, or when working with multilingual documents. Another drawback is the resulting cost of potentially merging back numerous language parts: Since metadata query is offset-based, merging back metadata files implies updating those offsets.

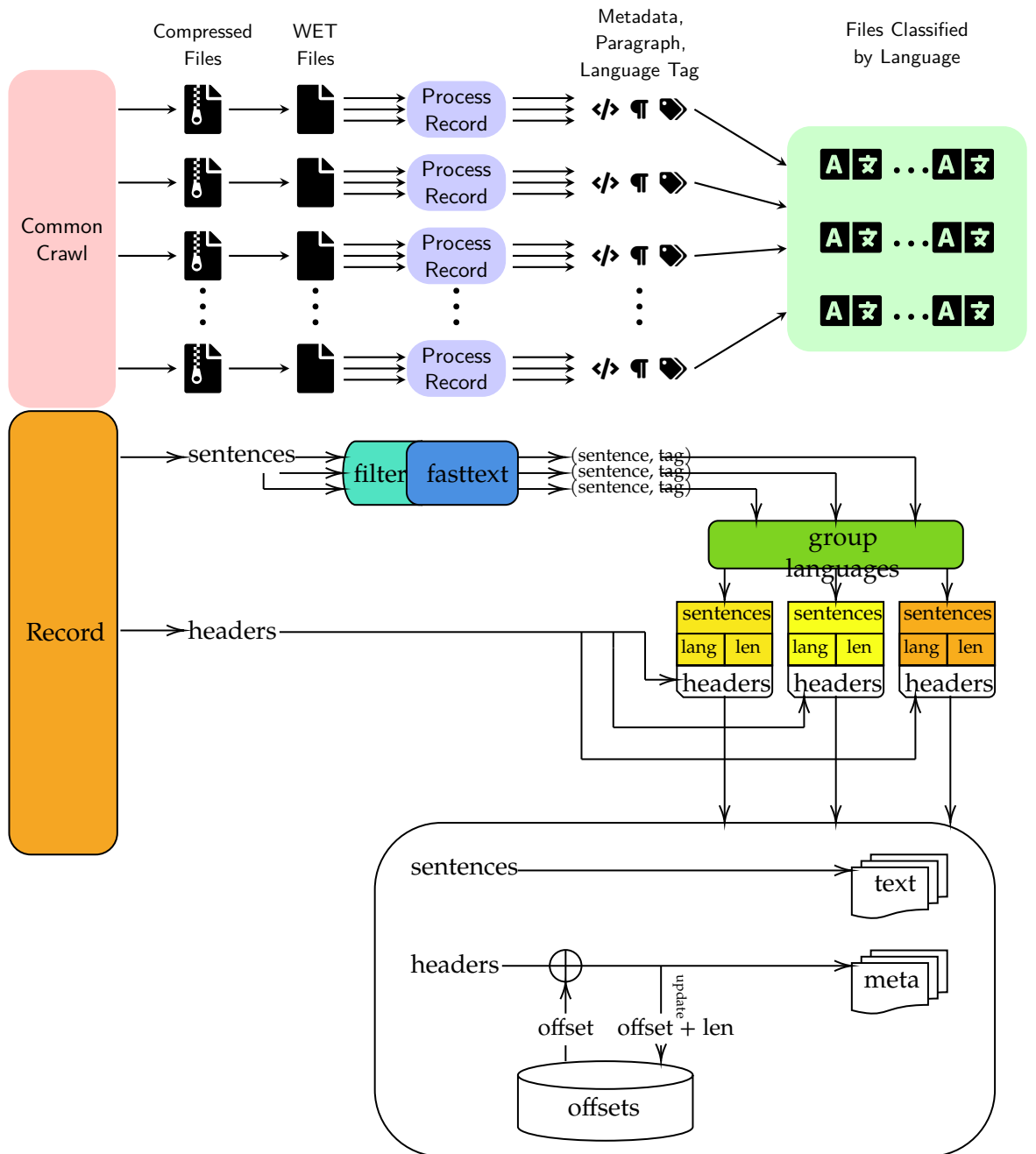


Figure 2.8: Record processing with metadata extraction. Headers are kept aside while sentences are identified and grouped into same-language bins. Headers are then cloned for each bin, and are sequentially stamped with an offset that is recorded for the whole operation, and written to disk into text and metadata files by language.

Platform	#shards	OSCAR	With Metadata	Speedup
Desktop	1	13s	12s	×1.1
	10	2m12s	1m55s	×1.1
	25	5m47s	4m50s	×1.2
HPC	1	6s	7s	×0.9
	25	1m6s	1m12s	×0.9
	100	4m14s	4m36s	×0.9

Table 2.20: Comparison of approximate generation times with and without metadata generation.

Version	Source	Textual (dedup)	Metadata	Total (increase)
2018	7.42TB	6.3TB (3.2TB)	N/A	6.3TB
2021	8.06TB	7.2TB (3.3TB)	1.2TB	8.4TB (+33%)

Table 2.21: Comparison of CommonCrawl and OSCAR sizes between 2018 and 2021 versions. Compressed (CommonCrawl) sources are from November 2018 and February 2021. Total is Textual + Metadata without deduplication.

Having paragraphs and metadata linked by offsets in a highly parallelized pipeline implies to take special care at the offset level. The solution is to use shard-scoped offsets (starting from 0 for each language), and to keep global offsets protected by a mutex guard. This way, when a given shard is done processing and is ready to be written on disk, we convert shard-scoped offsets to global-scoped ones, update the global-scoped ones and then write text and metadata on disk.

We compare running times for the reimplementations of the goclassy pipeline, and our new pipeline adding metadata extraction, using both desktop and HPC contexts. The results are reported in Table 2.20.

Metadata generation does not seem to influence generation time dramatically. However, we can notice a slight performance difference between HPC and Desktop contexts. These differences may lie in the storage medium differences, I/O layout, or algorithmic peculiarities benefiting desktop contexts because of other bottlenecks.

2.27.3 CHARACTERISTICS OF OUR NEW BACKWARD COMPATIBLE OSCAR-LIKE CORPUS

We evaluate the newly generated corpus, assessing its ability to reflect events that occurred after the publication of OSCAR 2018 and detail the metadata format and potential use.

COMPARISON WITH OSCAR

While it is expected that our new corpus has a larger file size than OSCAR since CommonCrawl itself grew from 7.42TB to 8.06TB, metadata quickly adds up and take for nearly 15% of the whole uncompressed data.

2 OSCAR

The size augmentation is not the same for each language, and while the whole corpus is bigger now, some languages are smaller than they were before.

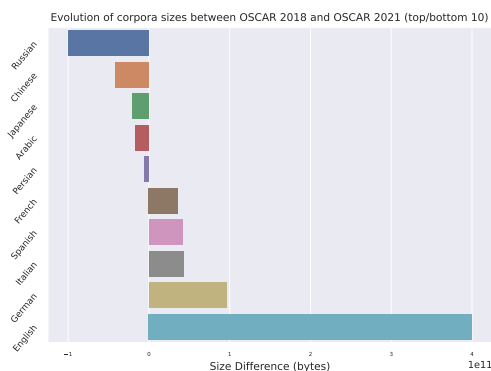


Figure 2.9: Comparison of language size (in bytes) between OSCAR 2018 and OSCAR 2021 (top/bottom 5 only).

Results show that already largely represented languages gain more and more data (like the English language, which constitutes more than a third of the original OSCAR), except for the Russian language which loses approximately 100Gb of textual content. These results are summarized in Figure 2.9.

However, in a context where the number of languages is very high (higher than 150) and of varying sizes, evolution can't be analyzed via a mere size evaluation. By computing, for each language, the relative size difference between the 2018 and 2021 releases of OSCAR, less resourced languages do appear, hinting at a better representation of some of them. These results can be found in Figure 2.10.

Numerous languages have been omitted from Figure 2.10, either:

- because they were present in the original OSCAR and are now absent (*Central Bikol* and *Cantonese*)
- because they were absent in the original OSCAR and are now present (*Manx*, *Rusyn*, *Scots* and *West Flemish*)

Precautions have to be taken when using these corpora and further work has to be done to correctly assess the quality of low-to-mid resource languages in order to better reflect the quality of each corpus to the OSCAR users. Some languages exhibited either a particularly low number of sentences or a very low quality, and as such couldn't be usable, while still accounting for a language in the total language count of the original OSCAR.

METADATA

Metadata provides new contextual data that is useful to evaluate the corpus and draw metrics.

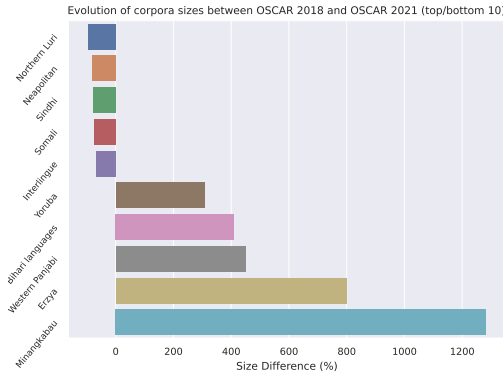


Figure 2.10: Comparison of language percentage between OSCAR 2018 and OSCAR 2021 (top/bottom 5 only).

The total size of metadata is 1.2TB, ranging from 4Kb to 500Gb, depending on the number of lines. Relative size varies from 100% to 20%, diminishing with the textual data size, which is expected.

Metadata are provided in single files for now, but split versions of both textual and contextual data will be released soon after the release of the corpus, enabling easy access.

Our choice of keeping metadata aside from the main content adds some complexity when working with both textual and contextual data:

- When trying to get the metadata of given sentence, one has to get the line number k , then sequentially (or use a search algorithm since offsets are sorted) look for the record (with offset o and length l), where $k \in [o, o + l]$.
- Looking for lines corresponding to a particular metadata entry is easier: one has to read the textual file, skipping until the o -th line, then read l lines.

PRESENCE OF EVENTS

Using a sample of an English part of our corpus, we perform a simple search of terms in order to assess and compare the presence of pre- and post- 2018 events and persons in both corpora. Terms and frequency are grouped in Table 2.22.

Our corpus keeps around the same number of occurrences for pre-2018 events or public figures such as Barack Obama, while increasing the occurrence of people linked to more recent events (Joe Biden).

We include search terms linked to post-2018 events in French and Arabic which are smaller corpora (resp. 200 and 80 GB), and in Burmese, a mid-resource language (approximately 2GB). We observe a term occurrences evolution that reflects the linked events' timing and importance.

Language	Term	2018	2021
Arabic	Beirut port explosion	0	31
Burmese*	Min Aung Hlaing	387	3439
English	Obama	30039	27639
English	Biden	990	19299
French	Yellow Vests	2	96

Table 2.22: Comparison of occurrences of news-related terms between OSCAR and our corpus in a sample of 100 CommonCrawl shards. For the Burmese language, we use the whole 2018 and 2021 corpus since it is a low resource language. Terms are translated in the corpus language.

2.27.4 LICENSE

This new corpus will be released under a research-only license that is compliant with the EU’s exceptions for research in text and data mining. Contrarily to the original OSCAR, no shuffled version of the corpus will be distributed, instead we will put in place an authentication system that will allow us to verify that requests for the corpus come from research institutions. A contact form will be also provided for independent researchers so that we can study their particular cases and determine if the utilization of the corpus corresponds to a legitimate research use.

Moreover, the introduction of metadata makes our corpus far more queryable, thus simplifying and speeding up the handling of take-down GDPR requests. For this reason, we will be releasing the complete set of metadata under a CC0 public domain license, so that any individual can check if their personal or even copyrighted data is in our new corpus and make a request accordingly.

2.28 CONCLUSION

We show that our solution is able to generate an OSCAR-like corpus that is augmented with metadata without breaking compatibility, while being faster, better tested and thoroughly documented. We believe our new pipeline and corpus will be useful for applications in computational linguistics as well as in corpus linguistics in general.

The generated corpus is of a larger size when including metadata and without deduplication. However, deduplicated textual content is of the same magnitude between OSCAR 2018 and OSCAR 2021, while reflecting topic changes from all over the world. This fact suggests that old data may be lost with the time passing, and could be resolved by using CommonCrawl releases to build an incremental corpus, with every version augmenting the corpus size.

Metadata enables queries and statistics on the generated data, and we believe that it can be used to filter OSCAR to generate corpora that respond to certain criteria.

We plan to make this new version of OSCAR available under research constraints, with split versions of both textual content and metadata along with tools to operate on the corpus, enabling fast and easy operation on the corpus for researchers.

3 MODERN FRENCH DATA

3.1 LEM17

3.2 PRESTO MAX

3.3 PRESTO GOLD

4 ANCIENT/MEDIEVAL FRENCH DATA

4.1 BERTRADE CORPUS

5 OTHER DATA

PART II

MODELS

6 CAMeMBERT

7 FrELMo

8 D'ALEMBERT

9 BERT_{TRADE}

PART III

DOWNSTREAM TASKS

10 PARSING

11 POS TAGGING

12 NAMED-ENTITY RECOGNITION

13

TEXT NORMALIZATION

14 DOCUMENT STRUCTURATION

PART IV

REAL WORLD APPLICATION

15 BASNUM

16 NAMED-ENTITY RECOGNITION CORPORA

BIBLIOGRAPHY

- Željko Agić and Ivan Vulić. 2019. [JW300: A wide-coverage parallel corpus for low-resource languages](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3204–3210, Florence, Italy. Association for Computational Linguistics.
- Rami Al-Rfou', Bryan Perozzi, and Steven Skiena. 2013. [Polyglot: Distributed word representations for multilingual NLP](#). In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192, Sofia, Bulgaria. Association for Computational Linguistics.
- Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. [Massively Multilingual Neural Machine Translation in the Wild: Findings and Challenges](#). *arXiv e-prints*, page arXiv:1907.05019.
- Mikel Artetxe and Holger Schwenk. 2019. [Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond](#). *Transactions of the Association for Computational Linguistics*, 7:597–610.
- Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. [Domain adaptation via pseudo in-domain data selection](#). In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK. Association for Computational Linguistics.
- Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. 2019. [Cloze-driven pretraining of self-attention networks](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5360–5369, Hong Kong, China. Association for Computational Linguistics.
- Marta Bañón, 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 Sarriás, Marek Strelec, Brian Thompson, William Waites, Dion Wiggins, and Jaume Zaragoza. 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.

- Stella Biderman and Walter J. Scheirer. 2020. [Pitfalls in Machine Learning Research: Reexamining the Development Cycle](#). *arXiv e-prints*, page arXiv:2011.02832.
- Robert D. Blumofe and Charles E. Leiserson. 1999. [Scheduling multithreaded computations by work stealing](#). *J. ACM*, 46(5):720–748.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. [Enriching word vectors with subword information](#). *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. [Findings of the 2018 conference on machine translation \(WMT18\)](#). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.
- Jan A. Botha, Emily Pitler, Ji Ma, Anton Bakalov, Alex Salcianu, David Weiss, Ryan McDonald, and Slav Petrov. 2017. [Natural language processing with small feed-forward networks](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2879–2885, Copenhagen, Denmark. Association for Computational Linguistics.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.
- Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. 2009. Clueweb09 data set.
- Isaac Caswell, Theresa Breiner, Daan van Esch, and Ankur Bapna. 2020. [Language ID in the wild: Unexpected challenges on the path to a thousand-language web text corpus](#). In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6588–6608, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Isaac Caswell, Julia Kreutzer, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Javier Ortiz Suárez, Iroko Orife, Kelechi Ogueji, Rubungo Andre Niyongabo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhlov, Tapiwanashe

- Matangira, Colin Leong, Nze Lawson, Sneha Kudugunta, Yacine Jernite, Mathias Jenny, Orhan Firat, Bonaventure F. P. Dossou, Sakhile Dlamini, Nisansa de Silva, Sakine Çabuk Ballı, Stella Biderman, Alessia Battisti, Ahmed Baruwa, Ankur Bapna, Pallavi Baljekar, Israel Abebe Azime, Ayodele Awokoya, Duygu Ataman, Orevaoghene Ahia, Oghenefego Ahia, Sweta Agrawal, and Mofetoluwa Adeyemi. 2021. [Quality at a Glance: An Audit of Web-Crawled Multilingual Datasets](#). *arXiv e-prints*, page arXiv:2103.12028.
- Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. [Electra: Pre-training text encoders as discriminators rather than generators](#). In *International Conference on Learning Representations*.
- Alexis Conneau, 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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. [XNLI: Evaluating cross-lingual sentence representations](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. [Transformer-XL: Attentive language models beyond a fixed-length context](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *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. Association for Computational Linguistics.
- Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzmán, and Philipp Koehn. 2020. [CCAligned: A massive collection of cross-lingual web-document pairs](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5960–5969, Online. Association for Computational Linguistics.
- Miquel Esplà, Mikel Forcada, Gema Ramírez-Sánchez, and Hieu Hoang. 2019. [ParaCrawl: Web-scale parallel corpora for the languages of the EU](#). In *Proceedings of Machine Translation Summit XVII: Translator, Project and User Tracks*, pages 118–119, Dublin, Ireland. European Association for Machine Translation.

- Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. [Beyond English-Centric Multilingual Machine Translation](#). *arXiv e-prints*, page arXiv:2010.11125.
- Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. [The Pile: An 800GB Dataset of Diverse Text for Language Modeling](#). *arXiv e-prints*, page arXiv:2101.00027.
- Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. [Learning word vectors for 157 languages](#). In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).
- Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, H erve J egou, and Tomas Mikolov. 2016. [FastText.zip: Compressing text classification models](#). *arXiv e-prints*, page arXiv:1612.03651.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. [Bag of tricks for efficient text classification](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.
- Marcin Junczys-Dowmunt. 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.
- Marcin Junczys-Dowmunt. 2019. [Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation](#). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 225–233, Florence, Italy. Association for Computational Linguistics.
- David Kamholz, Jonathan Pool, and Susan Colowick. 2014. [PanLex: Building a resource for panlingual lexical translation](#). In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC’14)*, pages 3145–3150, Reykjavik, Iceland. European Language Resources Association (ELRA).
- Vincentius Kevin, Birte H ogden, Claudia Schwenger, Ali  ahan, Neelu Madan, Piush Aggarwal, Anusha Bangaru, Farid Muradov, and Ahmet Aker. 2018. [Information nutrition labels: A plugin for online news evaluation](#). In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 28–33, Brussels, Belgium. Association for Computational Linguistics.

- Huda Khayrallah and Philipp Koehn. 2018. [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.
- Philipp Koehn, 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 *Proceedings of the Fifth Conference on Machine Translation*, pages 726–742, Online. Association for Computational Linguistics.
- Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhersch, and Armand Joulin. 2018. [Advances in pre-training distributed word representations](#). In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. [Distributed representations of words and phrases and their compositionality](#). In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.
- Robert C. Moore and William Lewis. 2010. [Intelligent selection of language model training data](#). In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.
- Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. [A monolingual approach to contextualized word embeddings for mid-resource languages](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.
- Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. [Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures](#). In *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019*, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.
- Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition, linguistic data consortium. *Technical report, Technical Report. Linguistic Data Consortium*.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. [GloVe: Global vectors for word representation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. [Deep contextualized word representations](#). In *Proceedings of the 2018 Conference of the North American Chapter of the*

- Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.
- Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. [When and why are pre-trained word embeddings useful for neural machine translation?](#) In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana. Association for Computational Linguistics.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1:8.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. [Exploring the limits of transfer learning with a unified text-to-text transformer](#). *Journal of Machine Learning Research*, 21(140):1–67.
- Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. [A primer in BERTology: What we know about how BERT works](#). *Transactions of the Association for Computational Linguistics*, 8:842–866.
- Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. [WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia](#). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. [CCNet: Extracting high quality monolingual datasets from web crawl data](#). In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.
- Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. 2021. [Nyströmformer: A nyström-based algorithm for approximating self-attention](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14138–14148.

- Hainan Xu and Philipp Koehn. 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.
- Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. [mT5: A massively multilingual pre-trained text-to-text transformer](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. [Xlnet: Generalized autoregressive pretraining for language understanding](#). In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.
- Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. [Big bird: Transformers for longer sequences](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc.
- Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. [Aligning books and movies: Towards story-like visual explanations by watching movies and reading books](#). In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.