Originally published in: Lüngen, Harald/Kupietz, Marc/Bański, Piotr/Barbaresi, Adrien/Clematide, Simon/Pisetta, Ines (Eds.): Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-9) 2021. Limerick, 12 July 2021 (Online-Event). – Mannheim: Leibniz-Institut für Deutsche Sprache, 2021, Pp. 1-9.
DOI: *https://doi.org/10.14618/ids-pub-10467*

Ungoliant: An Optimized Pipeline for the Generation of a Very Large-Scale Multilingual Web Corpus

Julien Abadji[1] **Pedro Javier Ortiz Suárez**[1,2] **Laurent Romary**[1] **Benoît Sagot**[1]
[1]Inria, Paris, France
[2]Sorbonne Université, Paris, France
{julien.abadji, pedro.ortiz
benoit.sagot, laurent.romary}@inria.fr

Abstract

Since the introduction of large language models in Natural Language Processing, large raw corpora have played a crucial role in Computational Linguistics. However, most of these large raw corpora are either available only for English or not available to the general public due to copyright issues. Nevertheless, there are some examples of freely available multilingual corpora for training Deep Learning NLP models, such as the OSCAR and Paracrawl corpora. However, they have quality issues, especially for low-resource languages. Moreover, recreating or updating these corpora is very complex. In this work, we try to reproduce and improve the goclassy pipeline used to create the OSCAR corpus. We propose a new pipeline that is faster, modular, parameterizable, and well documented. We use it to create a corpus similar to OSCAR but larger and based on recent data.Also, unlike OSCAR, the metadata information is at the document level. We release our pipeline under an open source license and publish the corpus under a research-only license.

1 Introduction

With the increasing interest in language modeling in recent years in Natural Language Processing (NLP) (Rogers et al., 2020), particularly concerning contextualized word representations[1] (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., 2020) and OSCAR (Ortiz Suárez et al., 2019, 2020) for pre-training language models, as well as, Paracrawl (Esplà et al., 2019; Bañón et al., 2020), CCAligned (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, CCaligned 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[2] (Ortiz Suárez et al., 2019, 2020), but we propose a new pipeline *Ungoliant*[3] 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, 2020) and Paracrawl (Esplà et al., 2019; Bañón et al., 2020).

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

[2]https://github.com/oscar-corpus/goclassy

[3]https://github.com/oscar-corpus/ungoliant

DOI: *https://doi.org/10.14618/ids-pub-10468*

2 Limitations of the OSCAR Corpus and its Generation Pipeline

2.1 OSCAR

OSCAR is a multilingual corpus derived from CommonCrawl[4], 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. 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, 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 data[5] 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, 2021), several problems have been described, and the state of the publicly available code raises questions about maintenance and maintenability 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, 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.

2.2 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, 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.

[4] https://commoncrawl.org

[5] OSCAR-orig: 6.3TB, OSCAR-dedup: 3.2TB

3 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.[6]

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.

3.1 Ungoliant

3.1.1 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.

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.

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

Through benchmarking we found that the best parallelization strategy is to use rayon[7], 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 1.

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.

3.2 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 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.

[6] https://benchmarksgame-team.pages.debian.net/benchmarksgame/fastest/rust-go.html

[7] https://github.com/rayon-rs/rayon

3.2.1 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 1, with a more precise view on record processing and metadata extraction in Figure 2.

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.

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.

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: 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 3: 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.

We compare running times for the reimplementation of the goclassy pipeline, and our new pipeline adding metadata extraction, using both desktop and HPC contexts. The results are reported in Table 2.

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.

3.3 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.

3.3.1 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.

The size augmentation is not the same for each language, and while the whole corpus is bigger

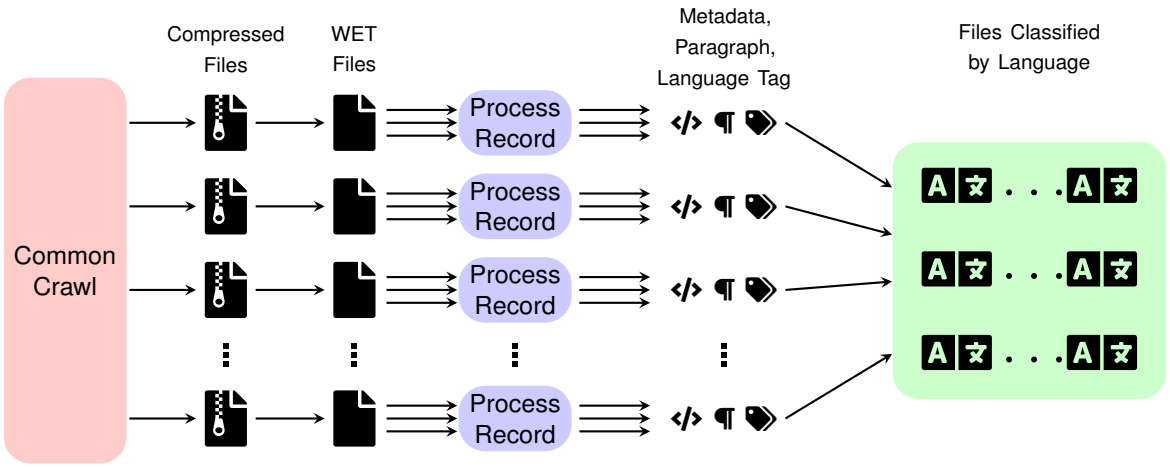

Figure 1: Scheme of the Ungoliant pipeline. The red square represents CommonCrawl content hosting, where the compressed shards are fetched. The *Process Shard* steps hold shard processing, paragraph creation and merging (see Figure 2), and are internally parallelized.

- : CommonCrawl compressed shard.
- : Uncompressed shard, containing records.
- </>: Record Metadata
- : Language identification
- ¶: Paragraph, composed of sentences identified as

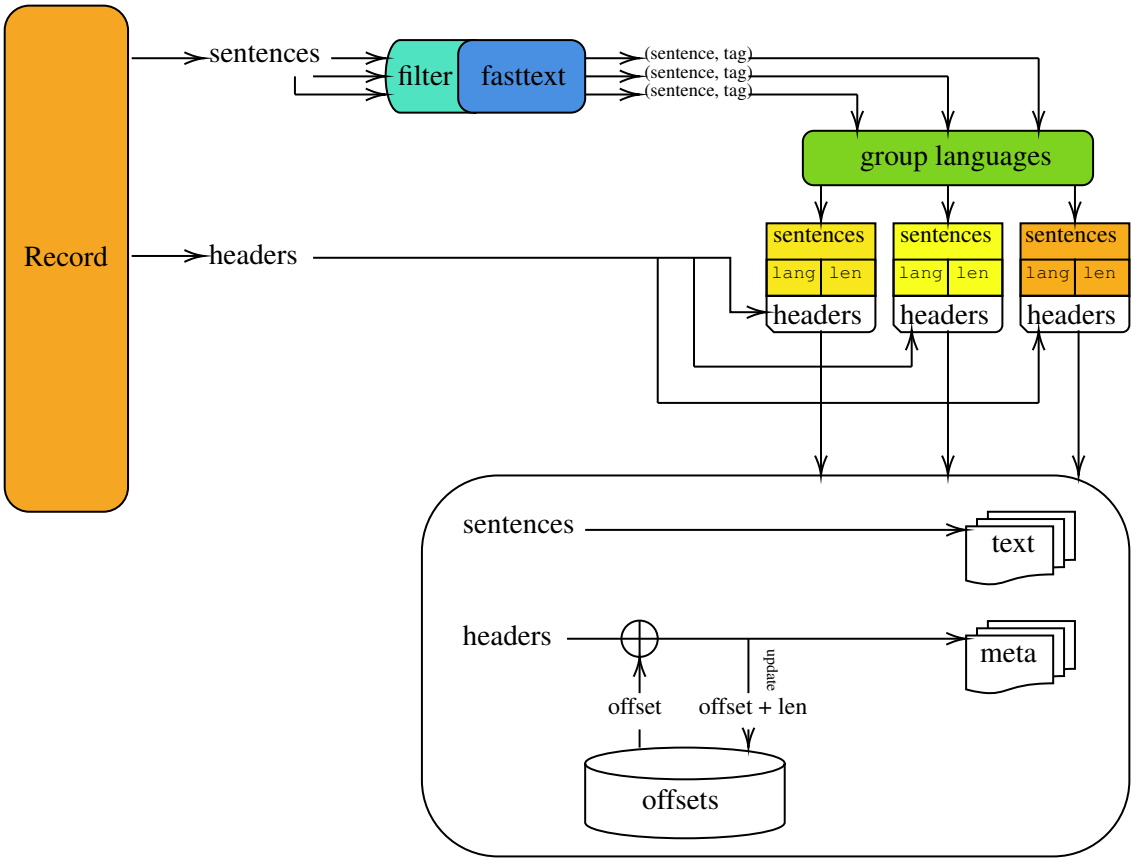

Figure 2: Record processing with metadata extraction. Headers are kept aside while sentences are identified and grouped into same-language bins. Headers are then cloned for each bin, and are sequentially stamped with an offset that is recorded for the whole operation, and written to disk into text and metadata files by language.

now, some languages are smaller than they were before.

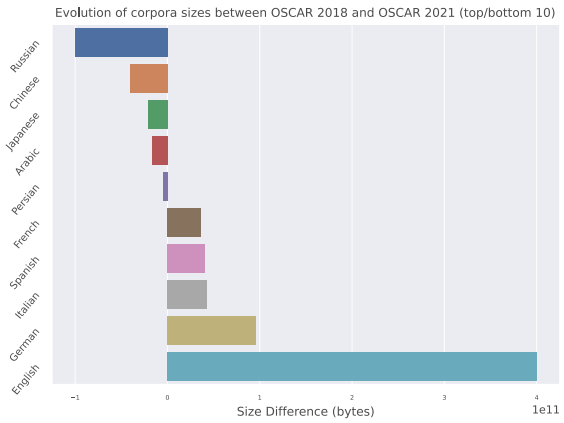

Figure 3: Comparison of language size (in bytes) between OSCAR 2018 and OSCAR 2021 (top/bottom 5 only).

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 3.

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 4.

Numerous languages have been omitted from Figure 4, 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.

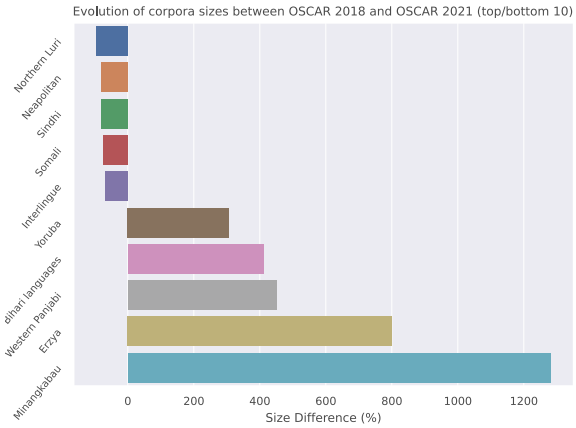

Figure 4: Comparison of language percentage between OSCAR 2018 and OSCAR 2021 (top/bottom 5 only).

3.3.2 Metadata

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

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.

3.3.3 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 4.

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).

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 4: 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.

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.

3.4 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.

4 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.

References

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 Sarrías, 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.

Robert D. Blumofe and Charles E. Leiserson. 1999. Scheduling multithreaded computations by work stealing. *J. ACM*, 46(5):720–748.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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, Iroro Orife, Kelechi Ogueji, Rubungo Andre Niyongabo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhalov, 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:2103.12028 [cs]*. Presented at the AfricaNLP 2021 workshop.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

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 Volume 2: Translator, Project and User Tracks*, pages 118–119, Dublin, Ireland. European Association for Machine Translation.

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, Hervé Jégou, and Tomás Mikolov. 2016. Fasttext.zip: Compressing text classification models. *CoRR*, abs/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.

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 7th Workshop on Challenges in the Management of Large Corpora*, 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.

Matthew 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.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

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.

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. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *CoRR*, abs/2010.11934.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, 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 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.