Filtering Pipeline of OSCAR

Thursday 16th December, 2021

The OSCAR filtering pipeline consists of two steps.

First, the documents will be modified to essentially remove excessively long or incorrect words (links, for example).

Second, on these newly modified documents, a filtering is performed to say whether or not the document is kept.

For each method presented in these two parts, one can, depending on the language, choose to use it or not.

The order of the filters is also indifferent and the methods are commutative.

All filters work with cutoffs or parameters, which are user-defined depending on the language in this file.

The code is available here and most methods are defined in this file.

Contents

1 Modification of documents

1.1 Whitespace standardization

There are many characters for whitespace. This method converts all whitespace in the sentence to the same whitespace, which is the classic space.

This method will later facilitate the segmentation of the document into words, which we will need for many methods.

1.2 Remove long words

To be able to extract the words of a document, and to rebuild it thereafter in a reversible way, one starts by splitting the document according to the new line character `"\n"`, then for each element split according to the tabulation `"\t"`, then for each sub-element split according to the whitespace `" "`. We can then perform the filtering we want on the words obtained, then reconstruct the document by joining the list in the opposite direction.

To know if a word is kept or not, we start by striping this word according to the special characters (by removing on the left and on the right of the word all possible special characters). Thus, for the word `situation,` (with a comma), we obtain the word `situation` (without a comma) after strip.

If the length of the word obtained after stripping it is greater than the specified cutoff manually defined by the user, we count the number of special characters in the word. If this number is exactly equal to 1, we keep the word. If not, it is deleted. The purpose of this heuristic is not to remove words that are long only because someone forgot a space between two words separated by punctuation, for example for the word `situation.Regardless`.

Of course, the stripping of words on special characters is only useful to decide whether to keep a word or not, but when reconstructing the sentence after the word filtering, it is the original words and not the stripped words that are considered.

1.3 Remove words with incorrect substrings

For this method, we use exactly the same strategy as for the previous method to extract the words and reconstruct the sentence. Only the filtering strategy changes.

Here, we decide to remove a word if it contains any substring that we consider incorrect. These substrings are defined manually by the user, and are by default `["http", "www", ".com", "href", "//"]`. The goal is to remove links and words related to the source code of the page.

2 Filtering of documents

2.1 Filtering on the number of words

We count the number of words in the sentence (separated by a line break, a tab or a whitespace). If this number is less than or greater than two cutoffs, the document is removed.

Documents that are too short are very often incorrect sentences, or contain no context for a model to learn correctly.

2.2 Filtering on the special characters ratio

Special characters are defined in this file. If a document has a special characters ratio greater than a certain cutoff, it is removed.

Some documents do not contain any correct sentences and mostly special characters. This filter allows you to remove them.

2.3 Filtering on the stop words ratio

We made a list of stop words for each language, available in this file. If the stop words ratio for a document is higher than a certain cutoff, it is removed.

This filter is the best method to remove automatically generated sentences by computers that do not make sense as a whole.

2.4 Filtering on the flagged words ratio

We have defined a list of flagged words for the different languages in this file.

For English, several existing long lists were concatenated and manually reviewed to keep only relevant words. In particular, we made an effort to keep only words centered on porn or racial slurs (the latter being present in much smaller proportions than the former). For example, `motherfucker` or `retard` are not present in the list. We also made an effort to not include non-offensive words referring to minorities. Thus, `gay`, `lesbian` and `transsexual` are not in the list either.

For the other languages, the lists have not yet been checked manually, and we will need native speakers to filter them in the same way as we did for English. Since English flagged words are regularly present in documents of other languages, we have also added them to those of all other languages.

The purpose of this filter is not to remove erotic texts, but to remove the numerous Oscar documents consisting of an accumulation of buzzwords centered on porn, most often without having any cohesion within the same sentence, which would harmful for the learning of the model.

We see empirically by looking at the data that the texts with a flagged word ratio above a certain cutoff are not at all texts concerning minorities. On the contrary, some of them are very sexist, glorify rape or pedophilia, etc...

2.5 Filtering on the language identification prediction score

We use the fastText model to quickly obtain an estimate of the language of a document, along with a prediction confidence score. If this score is below a certain cutoff, the document is removed.

This filter removes documents in which the spoken language changes several times, as well as documents of another language, which cannot be correctly analyzed by filters that have been specified with parameters related to a particular language.

2.6 Filtering on the perplexity score

We use the KenLM models trained by Facebook on each language on Wikipedia to obtain the perplexity scores of the documents. If the perplexity score is above a certain cutoff, the document is removed.

The purpose of this filter is not necessarily to remove a large number of documents. Indeed, documents with a different structure from Wikipedia, or with a more familiar language or

on the contrary very technical on specialized subjects, would be unfairly penalized. The goal here is to remove outliers which are mainly documents with unrelated words, for example a list of tags, information related to the extraction of the text of the page (date and time) or multiple repetitions of the same sentences.

3 Playing with the filtering parameters

The Spaces text-data-filtering is a demo that allows you to play with the filtering parameters to filter 5000 English documents and get familiar with the pipeline.

It is possible to increase the number of documents (up to 15000) and to use it for other languages. To do this, please run this code on your computer.

4 What's next?

Some methods are still under development and not yet officially integrated to the filtering pipeline. We quickly present these methods in this section.

First of all, it is necessary to adapt the methods to Chinese which does not have spaces between words. For this, we can perform a tokenization to obtain sub-words. The methods must also be adapted to Vietnamese, which has spaces between syllables, and therefore whose whitespace does not necessarily indicate a separation between two words.

It is interesting to code a filter allowing to remove documents with too many repetitions.

KenLM models have also been trained on OSCAR (@Eduardo). We are interested in a way to combine the two scores obtained by the KenLM trained on OSCAR and on Wikipedia.

The data team members are interested in anonymizing the documents. They are also interested in doing experiments to see if this anonymization can have negative consequences on the performance of the model.