How to create a (text) corpus for NLP applications

Shekhar Pradhan

8 Mar 2021 • 4 min read

You may be considering building a corpus for natural language processing tasks. But before you proceed you should be clear why you want to build one.

What is your primary motivation for creating a corpus? Is it to build a model for a specific NLP application? Or do you want to create a corpus primarily to create a resource for others so they can build their applications using your corpus?

If your primary motivation in building a corpus is to create a task-specific model, you should stop and ask yourself, "Do I really need to create a corpus? Maybe there is a ready made corpus that I can use." Some of the many sites where you can find ready-made corpora (plural of corpus):

Going a step further, you should also ask whether you need to create your own model for whatever NLP task you have in mind. Today there are many models you can use (check out the models hub on huggingface.co), which you may use as they are, or by fine-tuning a model using your training set (for example, for text classification). So, you may not even need to create a model or a corpus.

Okay, let us suppose after consideration of the points raised so far, you have decided to create a corpus, whether for you own needs or to provide a resource for others (not that these are mutually exclusive goals).

Now you need to think about what types of documents you need to collect. That depends on the downstream tasks you want to use the corpus for. For example, if your primary purpose is to find which

tropes, topics, and issues dominated the Victorian period in England, you will want to assemble a corpus consisting primarily of documents from the Victorian period (literary, news, legal, etc.). On the other hand, if you want to build a model for detecting shifting sentiments about immigration in the USA over a certain time period, you will want to assemble a corpus consisting of news stories, blog postings, tweets, political speeches, legal rulings, etc., pertaining to immigration, where these documents span the time period of interest. Or, if you want to build a Word2Vec model that is applicable to the medical domain, the Word2Vec model created by Google based on news articles will not work well for your needs and you will want to assemble a corpus consisting of medical articles.

Now that you have settled these questions, you need to decide how many documents you need. For some applications, such as building a word embedding model, you may need need many more than for building, say, a Naïve Bayes classifier. You may think, the more, the better. But that is not always the case. There are diminishing returns and increasing costs for building and maintaining a large corpus. There is also the danger of over-fitting if your model is too closely tuned to your corpus. The larger the corpus, the more the risk of over-fitting unless one takes special care to make your corpus representative of the cases in which your model will be deployed.

So, now you have to figure out how to get the documents to build your corpus. Are they available in some repository such as a Wikipedia dump? (Don't confuse a document repository with a corpus---the documents in a repository may be in different formats, with all sorts of mark-ups, etc. The text from the documents in a repository will need to be extracted, cleaned up, etc., before it can go into a corpus. More on that below.)

If the documents you need for your corpus are not available in a repository, you will need to find these documents, say on the Internet or the intranet of an organization. To access these documents, you will need their URL. How do you get those? Here you have two choices: You could use a service (like webhose.io) that finds these documents. (These services also extract the content). Or, you'll have to use a search engine like Bing or Google. Bing provides an API for

programmatically finding documents (their URLs), but then you have to retrieve the documents and extract (scrape) their content.

Services like webhose.io (Octoparse, diffbot, etc.) are great for retrieving documents from the Internet and scraping their content and categorizing them. However, they tend to focus on news feeds and blogs. So, if you want to create a corpus of scholarly articles, these services are not for you. Also, they are not free.

If you decide not to use these paid services, you may want to do your own crawling using a framework like Scrapy, which does both web crawling and scraping. Or, you may use an API provided by Bing for crawling and do scraping using the tools mentioned in the next step.

You may need to extract the content of the documents you retrieved so they are all in the same format, text files most likely. You may use Python libraries like Beautiful Soup to extract the content of html documents or PDFMiner to extract the content of PDF documents.

If you want your corpus to consist of documents in a particular language, such as English, you'll need to do language detection to remove non-English documents. Some English documents may have non-English parts and you may need to identify those and remove them, depending on what you are trying to do.

Further, depending on your application, you may need to remove duplicates and nearly duplicates from your corpus. This is called de-duping. For that, you may want to use locality-sensitive hashing techniques for de-duping.

Some portions of some of your documents may have gotten garbled in the process of applying the above steps. You need to have a way of detecting them and correcting or removing them. One way to detect garbled characters would be to tokenize each document and check whether every token is in the vocabulary of your language. (Presumably, you have already compiled or procured such a vocabulary.) Then for any token that is out of vocabulary, you will want to check whether any of the characters in the token fall outside the range of the character encoding (such as unicode) for that language.

But, if you decide to use this strategy, be sure to include the encoding values for mathematical and logical symbols.

Depending on your application you may need to normalize your documents. Do some of the documents use American spelling and some use British spelling? You may need to make them the same spelling.

There are probably more steps, depending on why you are creating a corpus. A rule of thumb is that 80% of the effort in a data mining project is in procuring, cleaning, normalizing data. This may well hold for text-mining projects as well.

Any questions? Ask the expert below.