CorpusCollie
A Web Corpus Mining Tool for Resource-Scarce Languages

Doris Hoogeveen[1,2]
[1]_textkernel
Amsterdam, The Netherlands
d1_doris@hotmail.com

Guy De Pauw[2]
[2]CLiPS - Computational Linguistics Group
University of Antwerp
Antwerp, Belgium
guy.depauw@ua.ac.be

Abstract

This paper describes CORPUSCOLLIE, an open-source software package that is geared towards the collection of clean web corpora of resource-scarce languages. CORPUSCOLLIE uses a wide range of information sources to find, classify and clean documents for a given target language. One of the most powerful components in CORPUSCOLLIE is a maximum-entropy based language identification module that is able to classify documents for over five hundred different languages with state-of-the-art accuracy. As a proof-of-concept, we describe and evaluate the fully automatic compilation of a web corpus for the Nilotic language of Luo (Dholuo) using CORPUSCOLLIE.

1 Introduction

In the field of human language technology, corpora are the cornerstone to the development of robust language technology tools and applications, such as part-of-speech taggers, syntactic parsers, machine translation and text-mining systems. Many of these systems use some kind of statistical processing and the adage "*There is no data like more data*" has never been more relevant: many studies suggest that a system's accuracy is often a function of corpus size. This unfortunately also means that for *resource-scarce* languages, the full language technological potential can often not be released.

Before the Internet era, corpora for resource-scarce languages were almost impossible to get hold of, unless one went through the slow and tedious process of collecting and digitizing (often severely outdated) printed works. Not until recently have researchers started looking at the Internet as a valuable source for language data and corpus material (Baroni and Bernardini, 2006). *Web mining* corpora has proved to be a relatively fast and cheap way to harvest digital language data. A lot of web mining approaches have been described over the years (Resnik, 1999; Ghani and Jones, 2000; Scannell, 2007; Kilgarriff et al., 2010) and quite a few of those tools are publicly available (Baroni and Bernardini, 2004; CorpusCatcher, 2011).

The CORPUSCOLLIE web mining tool described in this paper, differs from the current state-of-the-art in that it attempts to incorporate a wide range of text processing and classification modules that ensure the resulting corpus is as clean and expansive as possible. It furthermore adheres and makes use of the authoritative Ethnologue classification system. In this paper we will outline the different components of CORPUSCOLLIE and will provide a quantitative evaluation of the tool as a web miner for the resource-scarce, Nilotic language of Luo (Dholuo).

2 Harvesting data

As a very first step, the user is asked to specify the ISO 639-3 language code, as defined by Paul (2009), as well as a number of user-definable parameters (see below). The user will also need to obtain a Bing API AppID, which is free of charge[1]. Unfortunately other major search engines such as Google and Yahoo do no longer provide such APIs.

[1]http://www.bing.com/developers

Conference on Human Language Technology for Development, Alexandria, Egypt, 2-5 May 2011.

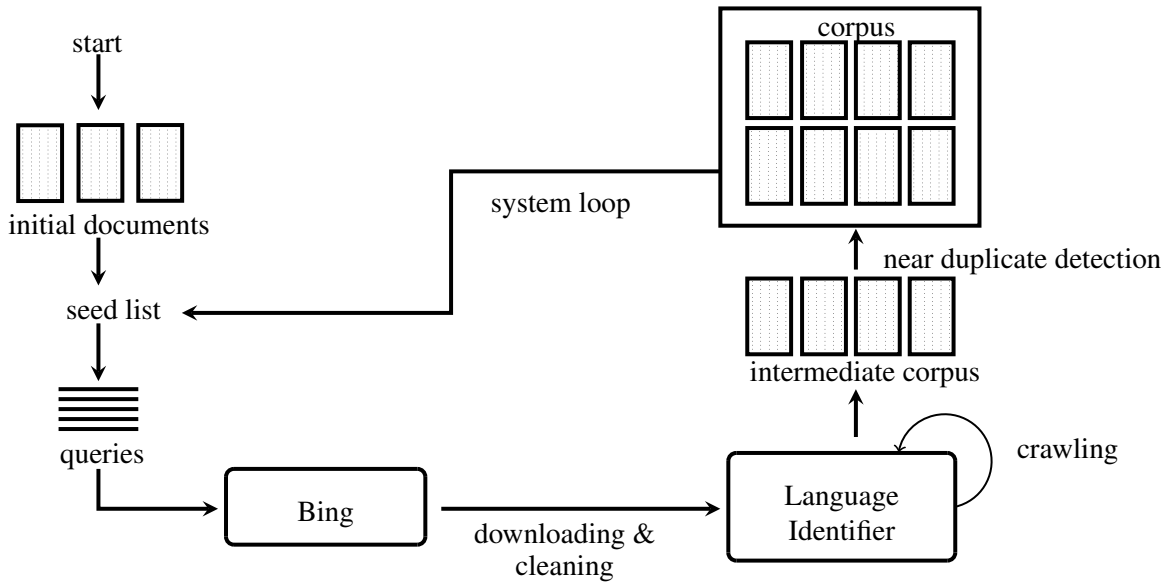

Figure 1: General Architecture of CORPUSCOLLIE.

The general architecture of CORPUSCOLLIE is outlined in Figure 1. The tool starts out with a number of documents in the target language from which a *seed list* of words is extracted. Words from the seed lists are randomly combined into search queries, which are sent to the search engine. Candidate documents in the target language are then classified by a language identification module and added to the final web mined corpus, provided they pass the (near) duplicate detection filter. From the newly found documents, a new seed list is generated and the whole process starts anew. In this section, we will go into more detail on the respective components of the CORPUSCOLLIE tool.

2.1 Query generation

The default settings of CORPUSCOLLIE assume that the user has a number of documents available in the target language. These need to be in text-only format and should be encoded in UTF-8. From these documents, a seed list is generated, containing the n most frequent words in the initial documents[2].

Highly frequent words are often short function words however, and many frequently occurring homographs are shared among languages[3]. Obviously, such words do not constitute appropriate search terms to mine one particular target language. We therefore downloaded all of the available Wikipedia data. For those languages for which a critical amount of Wikipedia data was available ($\pm$240), we constructed a *safetynet*: if a word in the seed list occurs in one of the languages of the safetynet file, it is not retained for possible search query generation. This safetynet helps ensure that the seed list contains the most frequent words of a language, that are not common in other languages.

From this seed list, a number of search queries are generated by randomly combining three words from the seed list. These queries are then sent to the Bing Search API module[4], which returns a list of URLs that match the search query in question.

We tried to reduce the risk of getting pages in the wrong language even further by specifying the domain extensions the search engine needed to take into account. For this purpose we used Ethnologue data to construct a database where each language code is listed with the countries where it is spo-

[2]It is also possible to skip automatic seed list generation and start the web mining process with a manually defined seed list.

[3]For example, "*san*" is among the 100 most frequent words in more than 30 languages.

[4]http://uswaretech.com/blog/2009/06/bing-python-api

ken, and the associated Internet domain extensions[5]. These extensions were appended to the search engine queries, together with more general domain names (e.g. .com, .biz, .org, ...). It seems however, that Bing does not yet handle extensive lists of domain specifications well and this functionality is currently disabled by default.

2.2 Processing the documents

CORPUSCOLLIE only harvests HTML pages from the Internet, as automatic conversion of PDF files and other legacy formats is unreliable, often leading to messy results and noise in the web mined corpus. Once it has been established that the URL returned by Bing, points to a true HTML page, we download it. Most web miners use external tools with crawling functionality for this task. We have opted for a standard python module (URLLIB) to download the pages. This allows us to save bandwidth by downloading a page from a web site first and then decide whether or not to crawl the rest of the site, based on the decision of the language identification module.

Encoding - A very important, but all too often ignored issue is that of encoding. We want the web mined corpus to be uniformly encoded in UTF-8. CORPUSCOLLIE uses the Python version of the Universal Encoding Detector, CHARDET[6], which displays the encoding of a page with a reliability score. If the encoding of a web page is determined with a reliability of over 70% (manually defined threshold), we keep the page, after converting it to UTF-8 when necessary.

HTML to (clean) text - After downloading the pages, they need to be *cleaned*. HTML needs to be converted to plain text, JavaScript, css and other forms of code need to be removed, as well as comments. We opted to use the Python module HTML2TEXT[7] to convert HTML pages to *Markdown* format, a plain text format that attempts to preserve the structure of the text, by using simple, non-obtrusive markings such as square brackets around links, and hash signs to indicate headers.

Boilerplate is a term denoting all the linguistically uninteresting parts of web pages, such as navigation bars and disclaimers. These need to be removed. Unfortunately, there are no unambiguous cues that indicate if something is part of the boilerplate of a page or not. Several researchers (Baroni and Bernardini, 2006; Sharoff, 2006; Ferraresi et al., 2008) have used a reimplementation of the BTE tool (Finn et al., 2001), which uses tag density as a heuristic. Other suggested techniques for boilerplate removal work on the basis of document size (Fletcher, 2004; Kilgarriff et al., 2010), word count (Ide et al., 2002) or the presence of function words (Ferraresi et al., 2008). Unfortunately, these approaches are too slow to be used as a module in our tool or are not publicly available.

The user can ask CORPUSCOLLIE not to remove Boilerplate at all (leaving open the option of using the aforementioned techniques post-hoc). Alternatively, a module can be enabled that uses regular expressions to remove patterns that are likely to be boilerplate. For instance, a sentence with a ©sign in it, is likely to be a disclaimer and can be removed. A pattern of words with pipe-signs (|) is likely to be a list of links at the end of a page. At this time, we have no experimental results on the effectiveness of our boilerplate removal algorithm.

2.3 Filtering the results

The cleaned pages are then classified two times. An extensive language identification module (see Section 3) checks whether the document is indeed written in the target language. If that is the case, CORPUSCOLLIE can be configured to crawl the rest of the site up to a user-defined depth, as it is likely that the same web site has more pages in the target language.

Finally, each page is classified by a (near) duplicate detection module. Near-duplicates are documents that share a significant portion of their content with a page already present in the corpus. These need to be removed because having identical documents in the corpus will negatively affect statistical processing.

For an extensive overview of the existing work on near duplicate detection, we would like to refer to Kumar and Govindarajulu (2009). CORPUSCOLLIE makes use of the *simhash* method, described in Charikar (2002), because of its speed and its ability to accurately perform near duplicate detection on smaller documents.

[5] For example Luo is associated with .ke, .tz and .ug.

[6] http://chardet.feedparser.org

[7] http://www.aaronsw.com/2002/html2text.py

2.4 Looping

Figure 1 shows the *system loop* link. Once we have harvested more data, we can construct a new seed list and repeat the entire mining operation again. This iteration can in principle be performed ad infinitum, but CORPUSCOLLIE is configured to loop a user-defined number of times.

The way the seed list is constructed in the second (and subsequent) iteration(s), differs from the very first iteration. Ghani et al. (2001) identified the *odds ratio* of a word as the best metric to select candidate seed words. The odds ratio of a word is determined, by calculating the probability of a word in the target language and the probability of that word in a non-target language and applying Equation 1.

$$log_2 \frac{P(w|rightlang) * (1 - P(w|wronglang))}{(1 - P(w|rightlang)) * P(w|wronglang)} \quad (1)$$

But while it is fairly easy to calculate P(w|rightlang) for a language, given a sizable amount of data, P(w|wronglang) is harder to calculate, unless we supply CORPUSCOLLIE with a prohibitively large lexicon of the world's languages. This is why in the first loop, we restrict CORPUSCOLLIE to selecting seed words on the basis of P(w|rightlang) and the *safetynet* only.

In subsequent loops however, we can use the output of the language identification module, to calculate P(w|wronglang), i.e. the probability that a word occurs in a document that was classified as a non-target language document. Alternatively, if a user knows that his target language**A** has a lot of lexical overlap with another language**B**, (s)he can add documents in language**B** as *wronglang* data to the system, so that appropriate odds ratios can be calculated before the first loop as well.

3 Language Identification

One of the core modules in CORPUSCOLLIE performs language identification. Language identification is a well-studied problem and most techniques use some sort of variant of the TextCat approach[8], which is based on the text categorization algorithm coined in Cavnar and Trenkle (1994).

[8]http://odur.let.rug.nl/~vannoord/TextCat

We are very much obliged to Kevin Scannell, who supplied us with a trigram frequency list for 450 languages. We also used the aforementioned Wikipedia data to construct trigram lists for languages not yet covered by Scannell's data, resulting in language models for over 500 languages.

Unfortunately, the TextCat approach becomes very slow when so many languages are to be considered. We therefore developed a different, faster classification technique, based on maximum-entropy learning. We discard all of the frequency information and construct a training set where each line lists the class (i.e. the ISO 639-3 code), followed by the top-400 trigrams in that language. This is exemplified for Swahili and Zulu in Figure 2.

To evaluate the language identification module, we constructed a test set of (up to) twenty documents for each of the 242 languages for which a sizable Wikipedia data set exists[9]. Table 1 compares our results to that of the TextCat approach, using the trigram frequency lists, and Google's language identification module.

Algorithm	Accuracy	CPU time
Maximum Entropy	89%	0.3ms
TextCat	54%	23s
Google	39%	1s

Table 1: Experimental Results for language identification task.

It is clear that the maximum-entropy approach outperforms the alternatives, not only in terms of accuracy, but also in terms of execution time. The low accuracy score for Google is surprising, but it is important to point out that Google only supports 104 languages out of the 240 in the test set. When we compare the maximum-entropy approach to Google's on languages only supported by the latter, Google scores marginally better (92.8% accuracy vs 92.6%). Google's approach has the added disadvantage that it uses bandwidth for classification, while the maximum entropy approach performs its classification offline.

TextCat's low accuracy score is rather surprising,

[9]In the interest of methodologically sound held-out validation, the training data for the classifier was constructed without using documents from the test set.

swa wa_ a_k _wa ya_ _ya na_ a_m _na ka_ _ku ni_ _ka a_n ika ali ati a_w ili kat _ma a_y i_y aka ia_ _kw ana kwa za_ tik la_ i_w i_k ani _ki ish wak ina ha_ a_u li_ _la mba uwa ma_ ini kuw _ni a_h ...
zul aba _ng ulu ing la_ thi hi_ a_n uth wa_ ama nga zin _ba uku ezi ngo izi ban ni_ _ab a_u a_i _uk lo_ a_k ase a_e isi eni we_ oku _iz and esi _ne _ku nge hul ala ant hla na_ khu eth lan imi ela nda ntu olo ...

Figure 2: Training Data for Language Identification. Underscores represent whitespace.

particularly compared to those reported in the literature. Error analysis indicates that TextCat does not seem to scale well to a large array of languages, particularly when many of them are closely related and share distinct orthographic features. The Maximum Entropy approach seems less vulnerable to this effect, as the features seem to function as constraints, rather than as probabilistic cues towards disambiguation.

To the best of our knowledge, CORPUSCOLLIE includes the largest language identification module currently available. Despite the large coverage of the module, it may occur that a user is mining a language that is not yet covered. In such a case CORPUSCOLLIE can construct a new language model on the basis of the initial documents used to bootstrap the web mining process.

4 Case-Study: web mining a Luo corpus

When evaluating web miners, it is inherently impossible to calculate the recall of the system, because there is no way to find out how many target-language documents exist on the world wide web. However, we can evaluate precision, by looking at how many pages in a web mined corpus are written in the target language. In this section we describe a short case study, an empirical, quantitative evaluation of CORPUSCOLLIE as a web corpus mining tool for the resource-scarce language of Luo.

Luo (Dholuo) is a Western Nilotic language spoken by around 5 million people in Kenya, Tanzania and Uganda. It has no official status and can easily be called a resource-scarce language and therefore serves as an ideal case-study to evaluate CORPUSCOLLIE. We decided to work with fairly restrictive user-defined parameters (400 seed words, crawl-depth 5 and 2 loops) to limit the number of documents the Luo native speaker needs to evaluate.

We start off with a collection of Luo documents ($\pm$200,000 words), once again kindly supplied by Kevin Scannell. In the first loop we simply select the most frequent words in this set, not present in the *safetynet*. 400 queries were constructed with these seeds, for which Bing returned 14 Luo pages. Further crawling resulted in another 262 Luo pages. In the second loop (with a new and updated seed list), 77 results were returned by Bing, while 1054 were found through crawling. After near-duplicate detection, 830 documents remained.

	Total	**Mainly Luo**	**Some Luo**	**No Luo**
Documents	830	292	535	3
%	100	35	64	0.5
Words	410k	212k	197k	46

Table 2: Experimental Results for Luo CORPUSCOLLIE Case-Study.

We then built a simple evaluation interface that displays each of the 830 documents and offers three evaluation options: (1) pages containing mainly Luo, (2) pages containing mainly another language, but some Luo as well, and (3) pages not containing any Luo.

The results can be found in Table 2. Less than 1% of the documents did not contain any Luo whatsoever. This more than encouraging precision score for CORPUSCOLLIE can be explained by a combination of having a good language identifier, selecting appropriate seed words, crawling and accurate text-processing modules.

5 Conclusion and Future Work

We presented CORPUSCOLLIE, a new web mining tool that incorporates a wide range of text processing and classification modules, in particular a wide-coverage and accurate language identification system. We presented experimental results that underline the capability of CORPUSCOLLIE in web mining a corpus for a resource-scarce language.

We are currently constructing a new evaluation framework, MININET, a closed-world data set of interlinked documents in different languages. This will allow us to evaluate CORPUSCOLLIE and other web corpus miners, not only in terms of precision, but now also in terms of recall, i.e. how many of the documents in the target language that exist in the "closed world" were actually retrieved by the tool.

Furthermore, we will extend our language identification module to not only work on the document level, but also on the sentence level. This will allow us to retrieve target-language data within a multilingual document. This is particularly useful when sourcing data from web forums, where typically a mixture of official and regional languages are used interchangeably.

We aim to publicly release CORPUSCOLLIE and the language identification module through http://AfLaT.org as a free and open-source software package in Q2 of 2011. This will hopefully lead to further refinements to the tool through the user community, a well as more extensive experimental results for a larger variety of languages.

Acknowledgments

The second author is funded as a Postdoctoral Fellow of the Research Foundation - Flanders (FWO). The authors wish to thank Naomi Maajabu for her annotation efforts and Kevin Scannell for making his data available to us.

References

M. Baroni and S. Bernardini. 2004. Bootcat: Bootstrapping corpora and terms from the web. In *Proceedings of LREC-2004*, pages 1313–1316, Lisbon, Portugal.

M. Baroni and S. Bernardini. 2006. *Wacky! Working papers on the Web as Corpus*. Gedit Edizioni, Bologna, Italy.

W.B. Cavnar and J.M. Trenkle. 1994. N-gram based text categorization. In *Proceedings of SDAIR-94, the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, USA.

M. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual Symposium on Theory of Computing (STOC)*, pages 380–388, Montreal, Canada.

CorpusCatcher. 2011. *by translate.org.za*. Available from: http://translate.sourceforge.net (Accessed: 22 January 2011).

A. Ferraresi, E. Zanchetta, M. Baroni, and S. Bernardini. 2008. Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4): Can we beat Google?*, pages 47–54, Marrakech, Marocco.

A. Finn, N. Kushmerick, and B. Smyth. 2001. Fact or fiction: Content classification for digital libraries. In *Proceedings of the Joint DELOS-NSF Workshop on Personalisation and Recommender Systems in Digital Libraries*, pages 1–6, Dublin.

W.H. Fletcher. 2004. Making the web more useful as a source for linguistic corpora. In U. Connor and T.Upton, editors, *Applied Corpus Linguistics: A Multidimensional Perspective*, pages 191–205. Rodopi, Amsterdam.

R. Ghani and R. Jones. 2000. Automatically building a corpus for a minority language from the web. In *Proceedings of the Student Workshop at the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, pages 29–36, Hong Kong.

R. Ghani, R. Jones, and D. Mladenic. 2001. Mining the web to create minority language corpora. In *Proceedings of the ACM CIKM International Conference 2001*, pages 279–286, New York, USA.

N. Ide, R. Reppen, and K. Suderman. 2002. The American national corpus: More than the web can provide. In *Proceedings of the 3rd Language Resources and Evaluation Conference (LREC)*, pages 839–844, Canary Islands.

A. Kilgarriff, S. Reddy, J. Pomik'alek, and A. PVS. 2010. A corpus factory for many languages. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC-2011)*, Valletta, Malta.

J. Prasanna Kumar and P. Govindarajulu. 2009. Duplicate and near duplicate document detection: A review. *European Journal of Scientific Research*, 32(4):514–527.

L.M. Paul, editor. 2009. *Ethnologue: Languages of the World*. SIL International, Dallas, USA. Online version: http://www.ethnologue.com.

Ph. Resnik. 1999. Mining the web for bilingual text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*, pages 527–534, College Park, USA.

K.P. Scannell. 2007. The Crúbadán project: Corpus building for under-resourced languages. In *Proceedings of the Third Web as Corpus Workshop: Building and Exploring Web Corpora*, pages 5–15, Louvain-la-Neuve, Belgium.

S. Sharoff. 2006. Creating general-purpose corpora using automated search engine queries. In *Wacky! Working papers on the Web as Corpus*, pages 63–98. GEDIT, Bologna, Italy.