Looking into Developing a Corpus for South African English

M.Umar Khan
khnmuh038@myuct.ac.za
University of Cape Town
Rondebosch, South Africa

ABSTRACT

A corpus is made for the study of a language and play a key part in linguistic investigations. A corpus contains the vocabulary of a language, its structure and the evolution of a language as time goes by. This literature review aims to look at why there is a need for a South African English corpus (SAE), previous attempts or work done for the development of a corpus for South African English and also attempts to explore the various ways in which a corpus for South African English may be developed using various techniques available for corpus development. Various techniques were used around the world which included: gathering data for the corpus, cleaning and filtering the data and annotation of the corpus etc, after analysis, we believe that these techniques can provide an insight on how one may go about developing a corpus for South African English.

KEYWORDS

Corpus, SAE

1 INTRODUCTION

As the popularity of the Internet has skyrocketed in the past few years, The web has become a valuable data bank for language resources of many kinds, this is seen especially with mono- and bilingual text corpora [17]. Making the Web an ideal place to build a corpus from.

Corpora are becoming quite popular and now play a central role in multiple branches of linguistics and similar disciplines [1]. [1] stated that corpora have been used to a great extent to provide usage evidence required for questions posed by theoretical and applied linguistics such as with multiple tasks in natural language processing and lexicography [1].

This literature review aims to look at existing work done on corpora for South African and the methods used to develop such. The aim of this is to create a corpus for South African English using previous methods used as well as incorporating new methods currently being used in corpus creation throughout the world.

According to [2] a corpus of SAE is needed by the Dictionary Unit for South African English (DSAE), the DSAE are an acknowledged authority on South African English and are the publishers of the Oxford South African Concise Dictionary.A corpus is also required by The Linguistics and Language department for Rhodes University [2]. Currently the DSAEs methods scale poorly, and limit the amount of data that can be gathered and pose a hindering factor for the depth of the analysis, this is because they are manually collecting data used to formulate dictionaries for South African English and the data collected is stored in spreadsheets.The need for a South African English corpus is further emphasized by it being a component of the international Corpus of English (ICE) [6].

After looking at existing work done on developing a South African English corpus we looked at some existing large web corpora and the techniques that are used to develop such corpora such as the process of gathering data, storing retrieved data, the cleaning process and the annotation of the corpus. We aim highlight these sections to shed some light on how a South African English corpus may be developed.

2 EXISTING CORPORA

Minimal work has been done for developing corpora for South African English as currently there exists no electronic corpus for South African English [2], the work that has been done for South African English was to do with the sub varieties of South African English such as Indian South African English or sub varieties of Black South African English.[4] mentions that a big challenge faced with developing a South African English corpus is that there are many sub-varieties such as Indian English, Black English and Afrikaans English, each of these sub varieties have some words or phrases that are not present in the other sub-varieties making it challenging to develop a South African English corpus.

A look at general corpora and other language specific corpora will also be looked at in this section to extract various techniques how corpora are developed around the world.

2.1 South African Corpus

Currently there exists no complete corpus for south African English [2, 4], work has been done on the sub-varieties of South African English this includes work done by Peinaar and de Klerk [4] with respect to Indian South African English, de Klerk [5] with respect to Xhosa English in which De Klerrk came up with a spoken corpus of Xhosa English consisting of about 500000 words [5] and Jeffery [6] with an attempt to create a South African English corpus for the International Corpus of English but this corpus has not yet been completed.The issue with all these corpora was that they only focus on sub-varieties of South African English and there means of collecting data for the corpus was limited to audio recordings as seen with [4] and [6] this resulted in a very small sample size [2].

The decision to use audio recording was explained by [4], was due to the fact that to transform from spoken data to written text required listening to the recordings multiple times and manual transcription of the words.It was acknowledged by [4] that automatic transcription methods do exist but the issue with those is that they tend to perform well with clear speech such as dialogues or broadcast monologues, whilst with unpredictable and spontaneous speech auto transcribers are still unreliable [4].

2.2 Other relevant corpora

We will now look at some corpora that have been developed around the world to gain an understanding of techniques used in the the development of corpora.

2.2.1 **Wacky corpus.** Are massive corpora of English , German and Italian and consist of over a billion words. The corpora were developed using web crawlers, and they aim to provide general-purpose resources for the languages specified[8].

2.2.2 **Czech Web corpus.** Is a corpus made up of about 2.65 billion words and the data is divided into three categories encompassing speech from different types of media such as magazines,blogs and discussions [17].The author describes in-depth the techniques used to develop this corpus some of which will be discussed in the coming sections.

3 STEPS REQUIRED FOR CORPUS DEVELOPMENT

A description of common techniques currently being used worldwide for corpus development.

The following steps will be covered in depth and follow the general procedure for corpus development:

(1) Gathering data for the corpus
(2) Storage of acquired corpus data
(3) Filtering and cleaning acquired data
(4) Annotation of the corpus

4 GATHERING DATA FOR THE CORPUS

We now look at various methods used by researchers to gather data for corpora.We will be looking at three methods being used which are search engine queries, web crawls and search engine hit counts.

4.1 Search engine queries

An approach to gather corpus data could be, that one can issue automated queries to the search engine such as Google which have Web service APIs that allow users to perform a number of automated queries per day, the user can then retrieve the pages returned by the search engine, and process them for building a corpus [1].

This approach has been explored by various scholars according to [1], and to a great extent by [23] and [24].

This approach relies less on search engines such as Google when compared to some methods such as the search engine "hit" counts method which will be discussed later on. The search engine is only used to obtain a list of documents but these documents sill have to be retrieved and post-processed by the user, so this means that the stability of the data will no longer depend on the search engine, the researcher then has full access to the corpus and the user can then integrate the corpus for complex linguistic queries [1].

BootCaT toolkit BootCat uses search engine queries and contains perl programs which perform an iterative approach to bootstrap specialized corpora and web terms, which require a list of seeds (which are items of interest) as input [7].

[7] performed the BootCaT procedure by building a corpus by automatically searching Google with only a small set of "seed" terms and then extracted new terms from the current corpus, these terms were then again used to build a new corpus through a new set of "seed" terms from automated Google queries , and then extract new terms which were from this corpus and and repeat the process. Unigram term list and the the corpus achieved at the end were then used to build a list of multi-word terms [7].

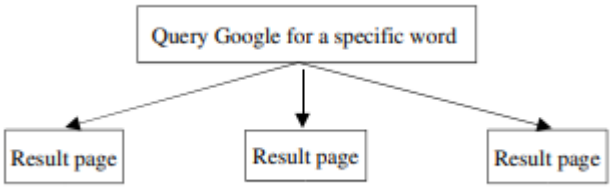

Figure 1: The search engine query approach [9]**.**

Though having its advantages this approach still has its fair share of problems [1].

The data that is recovered still depended on the search engines matching and ranking criteria. Furthermore, amounts of data that can be obtained by automated querying is often limited by search engines such as Google which only allows a the retrieval of max 10000 result URLs in a day, and in these results not all the data retrieved is relevant to the user [1].Whilst the vastness of data available on the Web is one of the attractive factors for linguists to use the Web for building massive corpora, using search engine queries for this is extremely impractical because of the reasons stated above [1].

4.2 Web crawls

A Web crawler is a program that recursively visits web pages to gather information for a specific need [9].

Considered one of the best long term and viable approaches for linguists to construct corpora, is the use of crawlers to perform crawls of the internet[1].This approach is considered more feasible than search engine queries, as it makes linguists independent of commercial search engines and provides full control over the procedure of corpus construction [1].

Implementing web crawls is generally the most difficult approach to implement, especially when constructing sizable corpora [1].Large crawls tend to require a substantial amount of computational resources, thereafter the crawled data requires post processing to remove unnecessary data such as html codes or data duplicates [1].

Web Crawlers start by crawling the pages of websites. Then it indexes the words and contents found on that website. Followed by visits of the links available on that website.So the process begins with a start URL and the crawler visits the page and extracts all the URLs from the given page, these are then added to queue, and the crawler then recursively crawls each URL in this queue and saves each page as the process moves forward.

A simple implementation of the crawler as stated by [9] is given below : Crawling begins at the root page and then advances to the links on that page as given by Fig 1 [9]. Processing of the contents of the returned pages takes place, and the links on each of these pages are followed. The content is collected in this way as the process advances forward [9].

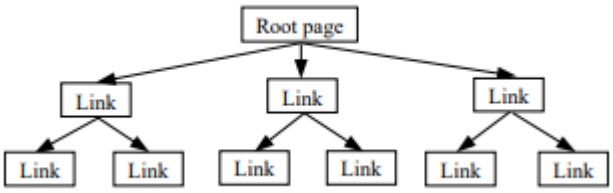

Figure 2: Basic architecture of a web crawler, starting with an initial document, the system recursively follows all the links in all subsequent documents [9]**.**

A common web crawler used by [8], [1] and [10] was the Heritrix crawler which is a free web crawler written in Java making it an attractive choice for many researchers [20].

4.3 Search engine hit counts as frequency estimates

One of the older but still most commonly used techniques for gathering data of the web is using the "hit counts" reported by popular search engines such as Google or Bing to estimate how frequently a searched string occurs with respect to a target language [1].

However, there are some issues when opting for this approach for gathering data. To start off, the types of queries a user can make are limited, [1] states the example that the user cannot make the use regular expressions or some parts-of-speech to restrict the type of search query.Many of the results become inconsistent or unreliable because companies often do not reveal on how query results are gathered,indexed or even returned, this creates a big issue with the "hit" counts approach and makes it quite unsuitable[1].

5 DATA STORAGE AND ANALYSIS

[22] saw that it was not appropriate to store corpus data in relational databases such as MySQL or sqlserver [22] . Firstly, the pages were barely related,the relationships were either of URL or text descriptions of corresponding pages. It was also seen that a relational database system with support of proper queries and relations would have had a lot of additional costs and essentially not be worth it [22]. Because web crawlers are efficient in crawling web pages, if a relational database

was used to store pages, a large number of data would be inserted into I/O in a short span of time. This insertion was also seen to have been a potential hassle, which was also a big concern for the physical disk and database maintenance network . That is why [22] found it suitable to store data text directly.

[22] believed that processes such as frequent file creation, writing, flush, shutdown, and system overhead were quite substantial. This led to the creation of a physical file that stored multiple pages. An index file was created to correspond with the data file so that proper search, merge and segmentation operations could be performed [22].Due Querying or traversing through data was very fast due to the index file being much smaller than the data file. The specific format is shown in Figure 3.

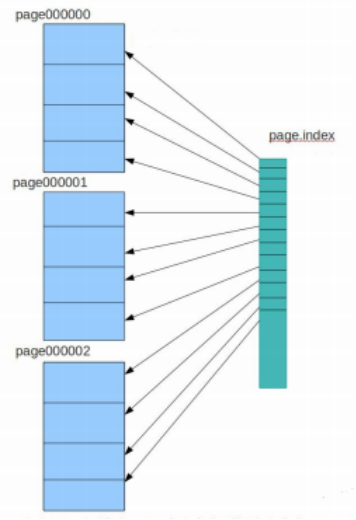

Figure 3: Storage format of [22]

6 FILTERING AND CLEANING ACQUIRED DATA

Data taken from the Internet can often be messy and not ideal for the format of a corpus [2].Many contents of web pages can be hidden in headers,footers, HTML tags e.t.c.The fact that many links can all point towards a single document or source t can lead to data duplication, which is another matter that needs to be resolved during corpus development.

6.1 Initial filtering

A common approach before stripping out any HTML tags or removing near duplicates was that scholars tended to discard documents less than 5KB and documents above 200KB [1], [8], [10]. [15] gives a reason for this and that is that very small documents barely contain any useful text (due to the HTML overhead) and large documents tend to lists of some sort such as store catalogs.

At this stage some scholars also removed perfect duplicates. [10] used sha-1 fingerprints present in the crawler logs to identify and then remove duplicates. [10] chose to remove any documents that had atleast one duplicate, although this could be noted as drastic measure by some, [10] noted that usually such documents came from the same sites and were warning messages, copyright statements and were not much of linguistic interests.

6.2 Boilerplate removal

One common practice found in sources [1],[8], [10] and [17] post retrieval of data was known as "Boilerplate stripping". “Boilerplate” means all those components of Web pages which are the same across many pages [9].This encompassed the removal of contents such as HTML markup, javascript and other non-linguistic material.Boilerplate removal was seen as an absolute necessity when it came to corpus creation by [9] as it distorted the statistics collected from the corpus [9].

The BTE tool[11] was adopted by [10] in which it was stated that sections rich with content in a page would often have a low html tag density, where as boilerplate contains a substantial amount of html due to its special formatting,newlines e.t.c .This approach was independent of the crawling procedure used because it is based on the general properties of web documents [9] .

[17] decided to go with manually writing scripts for each website to deal with HTML markup and boilerplate removal, the reason for this was because they believed this method would result in desired content and would save them from the problem of fundamental duplicates such as samples from articles and blogs e.t.c [17].

Automatic cleaning tools were also developed by some, such as "Victor the Cleaner" by [18] which was aimed at cleaning HTML pages by removing all text except headers and main page content.Continuous text sections (sections not including any HTML tags) are considered a single block that should be marked by a label as a whole [18].

6.3 Duplicates and near Duplicate removal

Duplication or near duplicates was noted by many sources which included: [10], [1], [8] and [14]. Duplication is quite prevelant on the internet, for multiple reasons, from caching to quotation and plagiarism [14]. In some instances perfect duplicates may exist and at other times near duplicates might be present [14].

1. order texts, from longest to shortest.
2. set sentence-db to empty
3. for each text
   a. set sentence-count and duplicate-sentence-count to 0 and empty the buffer
   b. break into sentences
   c. for each sentence over 25 characters long
      i. normalize:
         1. delete all non-alphanumeric characters and characters above ASCII 127
         2. convert all characters to lower case
      ii. if normalized sentence is in sentence-db (using an exact match), increment duplicate-sentence-count; else add normalized sentence to buffer
      iii. increment sentence-count
   d. if duplicate-sentence-count > x% of sentence-count reject text; else accept text, add sentences in buffer to sentence-db.

Figure 4: The algorithm used by [14] to get rid of duplicates.

This was applied to the whole corpus as one large process.

Approaches used by others such as [1] to remove near duplicates was the application of the "shingling" algorithm which was implemented in perl/mysql this involved the use of n-grams and fingerprints the process is given below:

The procedure began with taking the fingerprints of a fixed number of randomly selected n-grams, in which only distinct n-grams were taken and repeated n-grams were discarded , then for each pair of documents the number of shared n-grams were counted, as to provide an unbiased estimate of the overlap between the two documents [1]. Pairs of documents sharing more than t n-grams were identified, and one of the two was discarded. The pairs were ordered by document ID. To avoid inconsistencies, [1] always removed the second document of each pair.[1] mentions explains this with an example which states that "the pairs A-B, B-C and C-D were present in the list, then only the document A is kept and if the list contained the pairs such as A-C and B-C, only C was removed."

7 ANNOTATION

Corpus annotation is a procedure that aims to provide interpretive linguistic information to a corpus [12]. A common implementation of annotation is the adding of tags which indicates the word class of the words contained in a specific text [12].

Corpus Annotation is a valuable resource when development of NLP systems is concerned, and has use in linguistic analysis of languages [25]. The vast amounts of unstructured user-generated data tend to be a challenge for NLP technologies, specially with the boom of the Internet in recent years, this makes the need annotated corpora more[25].

7.1 Genre separation

The texts in corpora such as the (British National corpus) are classified according to a number of parameters, as knowing the contents of the corpus carries great importance [26].Generally this classification is manually done and at least covers the domain and genre of target texts [26]. The Web has being an increasingly popular source of linguistic data, However, the texts collected from the Internet have the problem of the metadata not having adequate descriptions of their domain and genre [26].

The linguistic features analyzed by [27] included the average word length, relative frequency of nouns and pronouns ,type/token ratio , average sentence length and the relative frequency of lexical elements that were only present one time, known as the hapax legomenon ratio. It was found that these features captured some original aspects of the text and which could then be used to separate the text into different genres [27].

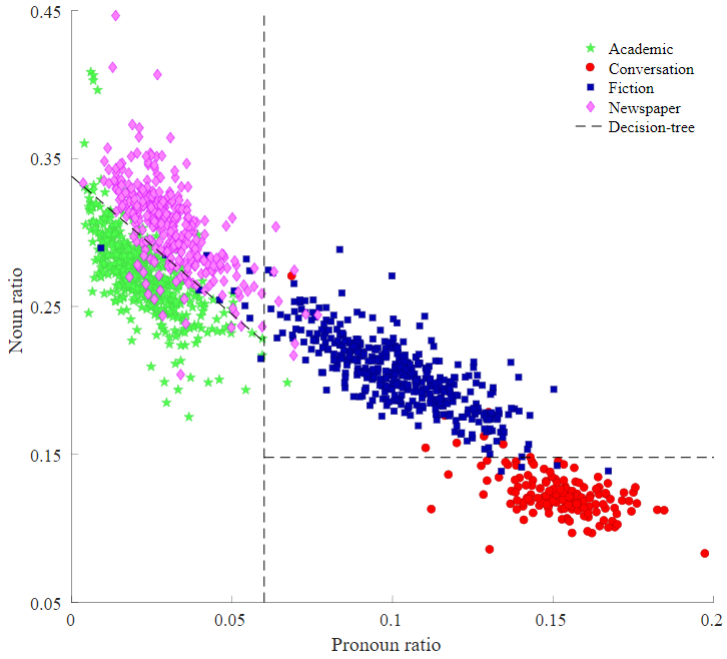

Figure 5: One of the results found by [27] when looking to categorize data.

7.2 Tagging

An annotation technique used by [8], [1] , [10], [14] and [12] was the Parts Of Speech tagging or POS tagging.

POS Tagger A part-of-speech(POS) tagger, processes a number of words and attaches a tag to each of these words [19]

POS tagging uses can be found e.g for differentiating words which have the same spelling, but different meanings or pronunciation [12].

Below are some examples of various types of POS taggers:

Tree Tagger: Is used for annotating text with part-of-speech and lemma information [21]. The TreeTagger has already been used to tag multiple languages and is can be tailored to be used for other languages if a lexicon and a manually tagged training corpus are available [21].This tagger was used by [10] and [8].

Penn Treebank tagger: Another type of POS tagger that was used by [13] and [25] was the Penn Treebank tagger the tagger was part of the Penn Treebank project [16] and was modified to correctly tokenise URLs, emails, and other web-specific text.

NLTK: Natural Language Toolkit or NLTK is a free Python platform which provides advanced POS tagging which can be customized by a user to be used with other taggers making it an attractive option for researchers [19].

8 CONCLUSIONS

In this review , we looked what existing work has been done for the development of a corpus for South African English, We then looked at how we can go about developing a corpus for South African English by inspecting previous work done on developing a South African corpus and techniques being currently being used around the globe.

To look for methods on how various corpora were being constructed around the world we looked at examples of some large web corpora such as [8], [10], [17] and [1] etc.We then looked at in-depth how the development of these corpora took place all the way from gathering the data for the corpus till annotation of the acquired data.Multiple techniques for each process were looked at as seen by various scholars and the pros and cons were also discussed.We feel like the next step from this would be integrating the methods discussed above for the purpose of developing an electronic South African English corpus.

REFERENCES

[1] M.Baroni and M.Ueyama. 2006 ."Building general- and special-purpose corpora by Web crawling.

[2] Gareth Terence Bryant Dwyer. 2014. "Towards Automated Creation and Management of a South African English Web Corpus

[3] Jan Pomikálek, Miloš Jakubícek, Pavel Rychlý, "Building a 70 billion word corpus of English from ClueWeb

[4] Leela Pienaar and Vivian de Klerk, "Towards a Corpus of South African English: Corralling the Subvarieties. Thirteenth International Conference of the African Association for Lexicography, 2018.

[5] De Klerk, V. 2002a. Starting with Xhosa English ... Towards a Spoken Corpus. International Journal of Corpus Linguistics 7(1): 21-42.

[6] Chris Jeffery (2003) On compiling a corpus of South African English, Southern African Linguistics and Applied Language Studies, 21:4, 341-344, DOI: 10.2989/16073610309486353

[7] Marco Baroni and Silvia Bernardini , "BootCaT: Bootstrapping Corpora and Terms from the Web April 2004

[8] Baroni, Marco, Bernardini, Silvia,Ferraresi, Adriano, Zanchetta, Eros. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. Language resources and evaluation, 43(3), 209–226.

[9] Gerrit Botha, Etienne Barnard "Two approaches to gathering text corpora from the World Wide Web, January 2005

[10] Adam Kilgarriff and Marco Baroni "Large linguistically-processed Web corpora for multiple languages, January 2006

[11] A. Kilgarriff. 2003. Linguistic search engine. In K. Simov, editor, Proc. SPROLAC Workshop, Lancaster.

[12] Developing Linguistic Corpora: a Guide to Good Practice Chapter 2: Adding Linguistic Annotation (Geoffrey Leech, Lancaster University © Geoffrey Leech 2004)

[13] Vinci Liu and James R. Curran.2006." Web Text Corpus for Natural Language Processing.

[14] Adam Kilgarriff, Michael Rundell and Elaine Uí Dhonnchadha.2006. "Efficient corpus development for lexicography: building the New Corpus for Ireland

[15] W. Fletcher. 2004. Making the web more useful as a source for linguistic corpora. In U. Connor and T. Upton, editors, Corpus Linguistics in North America 2002.

[16] Robert MacIntyre. 1995. Sed script to produce Penn Treebank tokenization on arbitrary raw text. From http://www.cis.upenn.edu/ treebank/tokenizer.sed.

[17] Johanka Spoustova, Miroslav Spousta.2012 "A High-Quality Web Corpus of Czech

[18] Miroslav Spousta, Michal Marek, and Pavel Pecina.2008.Victor: the web-page cleaning tool. In Proceedings of the Web as Corpus Workshop (WAC-4), Marrakech, Morocco

[19] NLTK. 2020. NLTK: the Natural Language Toolkit. Retrieved from http://www.nltk.org/book/ch05.html

[20] Wikipedia.2020. Heritrix Crawler. Retrieved from https://en.wikipedia.org/wiki/Heritrix

[21] TreeTagger.2020. TreeTagger - a part-of-speech tagger for many languages. Retrieved from https://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/

[22] Liujun Zhao, Weizheng Kong1, Qiuling Wang and Lihua Song.2019. "Construction of power industry corpus based on data mining and machine learning intelligent algorithm

[23] Sharoff, S. (2006) 'Creating general-purpose corpora using automated search engine queries'
[24] Igor LETURIA "Evaluating different methods for automatically collecting large general corpora for Basque from the web
[25] Xuansong Li, Martha Palmer, Nianwen Xue, Lance Ramshaw, Mohamed Maamouri, Ann Bies, Kathryn Summerville Conger, Stephen Grimes, Stephanie Strassel. "Large Multilingual, Multi-level and Multi-genre Annotation Corpus
[26] Serge Sharoff. 2007 ."Classifying Web corpora into domain and genre using automatic feature identification
[27] Jefrey Lijffijt."A simple model for recognizing core genres in the BNC