Computer Science Honours Project Report

The Afriweb Project: The Zulu Search Engine

Nkosana Malumba
mlmnko002@myuct.ac.za

Supervised by Dr. Hussein Suleman
hussein@cs.uct.ac.za

	Category	**Min**	**Max**	**Chosen**
1	Requirements Analysis and Design	0	20	5
2	Theoretical Analysis	0	25	0
3	Experiment Design and Execution	0	20	15
4	Systems Development and Implementation	0	15	10
5	Results, Findings and Conclusion	10	20	15
6	Aim Formulation and Background Work	15	20	15
7	Quality of Report Writing and Presentation	10		10
8	Adherence to Project Proposal and Quality of Deliverables	10		10
9	Overall General Project Evaluation	0	10	0
	Total Marks	80		80

Abstract

Search engines provide a very useful tool for finding information that may be stored on the World Wide Web or any particular database. However, for African languages, it is a challenge to find information due to the lack of technologies and algorithms that support the retrieval of this information. This report looks at the development of an IsiZulu search engine, which belongs to a group of African languages termed the Bantu languages. The system was built using existing technologies that were customized to suit the morphology of the language.

Keywords:

Information retrieval, morphological analysis, stemming, search engine, IsiZulu

Acknowledgements

Firstly, I would like to thank my family for their continual support throughout my university studies – their support has helped me times of strength and weaknesses, to continue to work towards my goals and aspirations

I would like to thank the Allan Gray Orbis Foundation for investing in my education and personal development through the course of my studies at the university. Being a fellow of this organization has been one of the most rewarding experiences that I have encountered during my stay at the university.

I would like to thank my supervisor Dr. Hussein Suleman, for his assistance throughout the project and pushing my project partner and I, to produce satisfactory results throughout the course of this project.

Lastly, I would like to thank everyone that has supported me during this stage of my life. May you all live in interesting times...

TABLE OF CONTENTS

List of Figures

This page has been intentionally left blank

Chapter 1

1 INTRODUCTION

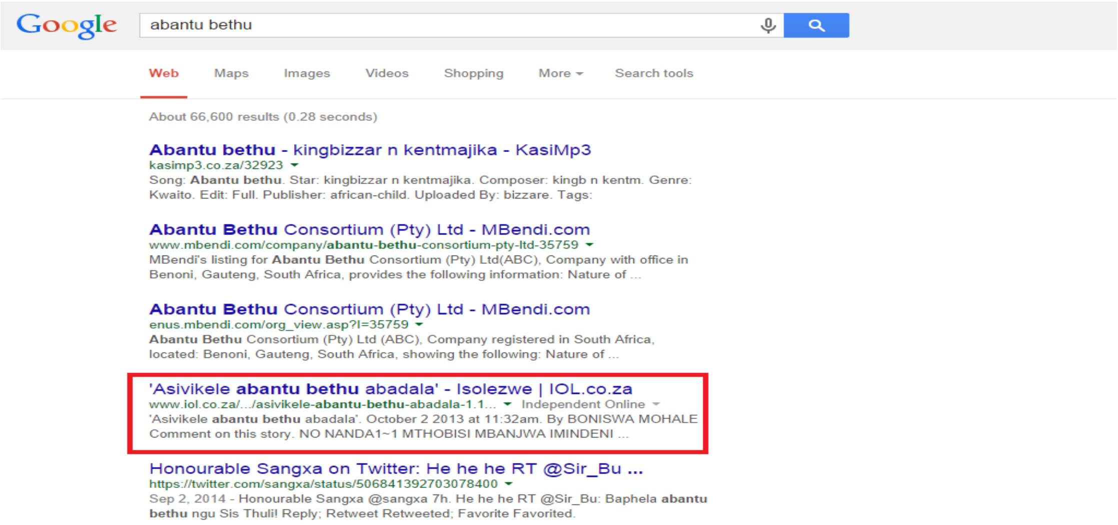
The purpose of this report is to document the design and implementation of an IsiZulu search engine. The search engine was built using existing open source technologies that were customized to suit the language. The project comprised of two main aspects, the harvesting of information from the World Wide Web and the indexing and retrieval of information.

Harvesting of information from the World Wide Web required the use of a focused crawler to detect pages that had content that was written in IsiZulu. These pages were then harvested and indexed in an open source search engine. The indexing and retrieval of information required that pre-processing algorithms be built in order to leverage the morphology of the language in the optimization of retrieval.

The project forms part of a broader research project into the building of information retrieval technologies for African languages, which a specific focus on South African languages. The growth of the World Wide Web has resulted in an increase of search engine usage in finding information. Thus the development of these technologies for African language will potentially make it easier to find language specific information.

1.1 PROBLEM OUTLINE

IsiZulu is one of South Africa's 11 official languages. According to ethnologies, it is the most widely spoken home language in South Africa and is understood by more than 50% of South Africa's population of about 53 million people (SA, 2012). Given the prevalence of this language in South Africa, it is almost impossible to find information on the World Wide Web, written in IsiZulu i.e. submitting a query in IsiZulu to a popular search engine such as Google, and getting results only in IsiZulu.



About 66,600 results (0.28 seconds)

Abantu bethu - kingbizzar n kentmajika - KasiMp3
kasimp3.co.za/32923
Song: **Abantu bethu**. Star: kingbizzar n kentmajika. Composer: kingb n kentm. Genre: Kwaito. Edit: Full. Publisher: african-child. Uploaded By: bizzare. Tags:

Abantu Bethu Consortium (Pty) Ltd - MBendi.com
www.mbendi.com/company/**abantu-bethu**-consortium-pty-ltd-35759
MBendi's listing for **Abantu Bethu** Consortium (Pty) Ltd(ABC), Company with office in Benoni, Gauteng, South Africa, provides the following information: Nature of ...

Abantu Bethu Consortium (Pty) Ltd - MBendi.com
enus.mbendi.com/org_view.asp?I=35759
Abantu Bethu Consortium (Pty) Ltd (ABC), Company registered in South Africa, located: Benoni, Gauteng, South Africa, showing the following: Nature of ...

'Asivikele **abantu bethu** abadala' - Isolezwe | IOL.co.za
www.iol.co.za/.../asivikele-**abantu-bethu**-abadala-1.1... Independent Online
'Asivikele **abantu bethu** abadala'. October 2 2013 at 11:32am. By BONISWA MOHALE Comment on this story. NO NANDA1~1 MTHOBISI MBANJWA IMINDENI ...

Honourable Sangxa on Twitter: He he he RT @Sir_Bu ...
https://twitter.com/sangxa/status/506841392703078400
Sep 2, 2014 - Honourable Sangxa @sangxa 7h. He he he RT @Sir_Bu: Baphela **abantu bethu** ngu Sis Thuli! Reply; Retweet Retweeted; Favorite Favorited.

Figure 1: Screen shot of Google search results for "abantu bethu"

As seen in the Fig. 1, a Google search for the word “abantu bethu”, which means our people, results in the first page in with content written in IsiZulu being ranked fourth. This is due to fact that Google uses several ranking algorithms, one of which is PageRank, which ranks the importance of a page given the number of incoming links it has (Langville & Meyer, 2011). Given that IsiZulu is only spoken by roughly 50% of the South Africa population, there is a high probability that the amount of IsiZulu content on the World Wide Web is not sufficient to use such a ranking algorithm as an advantage in providing relevant results.

Therefore, to have better access to IsiZulu content, tools that are built specifically for this language have to be available. These tools need to ensure that, not only results that are specific to the IsiZulu language are available, but that there is an increase relevancy in the results that are returned by the search engine.

1.2 Proposed Solution

It is proposed that a search system be developed for the IsiZulu language, through which users can search for IsiZulu specific information and get relevent and contextual results. To develop this system, two important aspects will have to be investigated. Firstly, the development of a search platfrom that has been customized for the IsiZulu language. This is because most open source platforms have been built and optimized for the more popular langauges such as English. Secondly, a focused crawler will have to be either developed or customized to harvest information from the World Wide Web that is either of relevance to or written in the IsiZulu language.

The project will also serve as a feasibility study into the practicality of building IR technologies for African languages with a specific focus on the South African languages. The feasibility study will outline aspects such as African language corpora and the extent to which certain algorithms effect the retrieval of language specific information.

The project will be split into two components, one that focuses on the harvesting of information from the World Wide Web and the other, the indexing and retrieval of information. This report will be focusing on the indexing and retrieval of information. Additionally, the are two research questions that the development of the IsiZulu search engine will aid in answering. The research questions are:

1) Can a morphological parser result in a more accurate derivation of root words as opposed to a stemming algorithm?

2) Will the use of a morphological parser as opposed to a stemming algorithm result in an improved precision of results given a user's query?

1.3 REPORT OUTLINE

The following chapter will look at the existing works in information retrieval in relative to smaller languages. This includes the various technologies and techniques that were implemented in these studies. The last section of the chapter will look at the harvesting of data, which forms the other half of the Afriweb project.

Chapter 3, the design chapter, will focus on the requirements analysis, design, and implementation of the system. This includes the methodology used and the iterations that resulted in the complete system. The evaluation of the project is discussed in Chapter 4, where the research questions are discussed relative to the experiments that were carried out. The last chapter, chapter 5, will include future works and a conclusion to the project.

Chapter 2

2 BACKGROUND

2.1 INTRODUCTION

Africa has 54 countries that have an estimated over 2000 languages in total. Some languages are endangered due to the assimilation of other dominant groups and the adoption of Western cultures (Mukami, 2013). In Africa, the language is an element of its culture as it presents the philosophy, history, stories, and medicinal practices of that particular culture. Therefore, an extinction of language will inevitably result in the loss of the diversity of the community that is Africa (Mukami, 2013).

Although there are a large amount of spoken languages in African, many of these, especially the Bantu languages of South Africa, still form part of the less researched languages in the world (Pretorius & Bosch, 2003). As a result, technologies that are crucial in the advancement of information retrieval research, such as corpora and dictionaries are still either undeveloped or incomplete. In the case of IsiZulu, which is the focal language of the Afriweb project, many researchers in linguistics have provided different perspectives which has resulted in a distributed and non-cohesive body of knowledge (Madondo & Muziwenhlanhla, 2000).

The first section will cover existing works in information retrieval that focus on uncommon languages such as Arabic, IsiZulu, and Swahili. This will include some of the strategies that were used in the development of these IR[1] systems and any challenges that were outlined in the development of these systems. The second section will provide an overview into the technologies that aid in information retrieval with relevance to the Afriweb project.

2.2 LANGUAGE BASED INFORMATION RETRIEVAL

Information retrieval is a field in computer science that is concerned with organization, storage, and displaying of information. The retrieved information is usually of a ranked nature that is sensitive to the user submitted query. (Manning, et al., 2008).

[1] Information retrieval

An interest in language-based information retrieval has been on the rise due to failure of the more popular search engines failing to provide language specific results for the smaller languages

Although websites such as Wikipedia do contain some pages that are in South African languages such as IsiXhosa and IsiZulu, it is close to impossible to find this information. According to the census conducted in South Africa in the year 2011, there are about 8.1 million Xhosa and about 10.6 million Zulu speaking people. Therefore, a significant fraction of the population has difficulty in finding information that is written in their language. One area that is influenced by this is research, because research is largely based on finding information that is of relevance and usefulness to the topic at hand.

As the digital age advances, a vital necessity for tools that are language specific is increasing as this is aiding many areas such as teaching, learning, and research. The following sections will discuss topics in information retrieval of small languages. These topics include Arabic IR[2], Swahili IR, Afaan Oromo IR, and IsiZulu CLIR[3]

2.2.1 Arabic Information Retrieval

Arabic is a Semitic language that belongs to a family of languages dominant in parts of North Africa and Western Asia. Semitic languages are known for their nonconcatenative morphology were the words are created by adding in vowels to a set of root consonants. Arabic has distinctive morphological features and its writing orientation is from right-to-left (Abdelila, et al., 2004).

The effectiveness of an information retrieval system depends on the system’s capacity to conform to the language in use. Therefore, understanding of characteristics of the language is very critical in building high quality information retrieval systems (El-Khair, 2007). As described by Nwesri et, al. (2007), Arabic is an inflectional language that requires the use of morphological analysis in the retrieval of Arabic text. This is to ensure that the various inflected forms of a word can be mapped to their root equivalent. Once converted to the root form, the search engine has a better chance of matching these words in a search through their relation to the root form. In the case of cross language informational retrieval, translations were achieved through morphological analysis by reducing Arabic text to its stem form. This made it easier to then translate the Arabic text to English. An additional post-processing step was required to correct an grammatical errors that may have occurred during the conversion to root form (Nwesri, Tahaghoghi & Scholer, 2007).

The major limitation in the development of the Arabic IR system has been the fact that there is a lack of adequate resources available to evaluate the system’s performance in real world settings. These include tools such as complete morphological analyzers, a language corpus,

[2] Information retrieval

[3] Cross language information retrieval

and a machine-readable lexicon. For research purposes, a language corpus was derived from a newspaper that is printed in the language. Although this did not provide an exhaustive list of the words that are part of the language, it provided a good estimation of the words that are commonly used in the Arabic language. On evaluation of the corpus, inconsistencies in the transliteration of proper names were discovered, which could affected the accuracy of the information retrieval system. This inconsistency was pointed out in the TREC[4] evaluation of the LDC[5] corpus, which has been central to most research conducted on Arabic IR systems (Abdelila, et al., 2004).

Arabic information retrieval systems have also attempted to make use of some of the strategies available to improve the performance of the retrieval system, such as stemming, stop lists, and term weighting. These techniques are all aimed at improving the relevancy of the results.

This section has outlined the benefits of using the language's morphology in developing an IR system. Additionally, lack of resources such as a language corpus has been shown to have a limiting effect on the effectiveness of the IR system. The next section will discuss Swahili IR system. Swahili is a Bantu language that is closely related to IsiZulu in its origins.

2.2.2 Swahili Information Retrieval

Swahili is a language of Bantu origin, which is part of the language group that South African languages such as IsiZulu and IsiXhosa belong. The language also has borrowed words from Arabic, which is because of the prevalence of Muslim groups in the areas that Swahili is dominant.

In 1992, Hurskainen described the first morphological analyzer for the Swahili language. The motivation for the development of this parser was that, in agglutinating languages such as Swahili, it is not convenient to use a direct string matching search method. This is due to the various inflected forms that a word can have because of its use in a piece of text (Hurskainen, 1995). As discussed in section 2.2.1, morphological analysis makes it possible to map the various inflected forms to a root word, which allows the search engine to pick up all these related terms given a query. The information retrieval system, SWATWOL[6] was designed to analyze the Standard Swahili language based on a two level formalism, where each character has a lexical and surface representation. A few morphological inconsistencies were found in this process. These were resolved using rules that are parsed in to the system as input (Hurskainen, 1995).

SWATWOL was implemented in the Swahili Language Manager (SALAMA), which is a computational system that facilitates many kinds of applications that are based on written

[4] Text Retrieval Conference

[5] Linguistic data consortium

[6] Swahili Two Level

Swahili text. Some of the abilities of SALAMA include producing the full vocabulary of a given text, translation of a particular text and syntactic analysis (Hurskainen, 1999). In the information retrieval space, SWATWOL was used as the morphological analyzer for unrestricted and non-encoded Swahili text. According to Hurskainen, there was a notable improvement in the accuracy of the search because of the integration. This is because the morphological characteristics of a language can be far more reliable as a key for information retrieval as opposed the direct search method (Hurskainen, 1995).

The application of morphological analysis in the development of information retrieval systems for languages that have two-directional word formation has proved to be a powerful method (Hurskainen, 1992). These formations are affected by the affixes that are applied to a word and change in morphemes that occur as a result of these applications. As Swahili and IsiZulu belong to the same language group, an assumption can be made that the use of the language's morphology will aid in the development of an effective IR system.

The next section will discuss cross language cross language information language in a Semitic language, which has many similarities with IsiZulu, in terms of its morphology.

2.2.3 Cross Language Information Retrieval for the Afaan Oromo Language

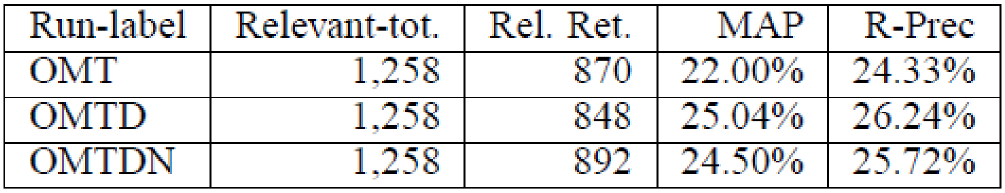
Afaan Oromo is a language that is widely spoken in Ethiopia. It belongs to the group of Semitic languages similar to Arabic. Similar to IsiZulu, the grammatical information of the language is conveyed by prefixes and suffixes. Afaan Oromo is the instructional medium in junior and secondary schools. Additionally, a number of works such as newspapers, magazines, education resources, official documents, and religious languages have been published in the language (Tune, et al., 2007).

In Afaan Oromo, the lack of a rich language source did not present a barrier as in the IsiZulu case. An Oromo-English dictionary was available that had been developed from hard copies of a bilingual dictionary. It was enhanced by incorporating additional entries, other language reference sources (Tune, et al., 2007).

Two popular information retrieval techniques, stopword lists and stemming, were adopted because of a rich language source. This significantly reduced the number of words in Afaan Oromo topics as the search engine could omit the words given as stopwords (Tune, et al., 2007). A light stemmer was used to automatically remove frequent inflectional suffixes that were attached to base form words. This was to ensure that the translation process could be simplified by translation of root words. The stemmed words were then translated using the bilingual dictionary by taking into account all possible keywords. Unmatched words were manually added to the dictionary (Tune, et al., 2007).

In the evaluation of the system, Afaan Oromo queries were translated into the English language and tested on the LA Times and GH Herald newspaper sources, which totaled about 169 477 documents (Tune, et al., 2007). Table 1 denotes the results of the experiment.

Table 1: Afaan Oromo experiment results



Run-label	Relevant-tot.	Rel. Ret.	MAP	R-Prec
OMT	1,258	870	22.00%	24.33%
OMTD	1,258	848	25.04%	26.24%
OMTDN	1,258	892	24.50%	25.72%

OMT refers to the Oromo title, OMTD referees to Oromo title and description and OMTDN refers to Oromo title, description, and narration. This was for assessing the overall performance of the Oromo-English cross language information retrieval system .The Relevant-tot is the total number of relevant documents, the Rel. Ret refers to the relevant retrieved documents, the MAP is the mean average precision, and R-Prec is the non-interpolated average precision.

OMTD was shown to have a better performance, which was attributed to the fact that most title fields in the CLEF[7] topics where very short. Therefore, the description of the document provided a better sense of what the content of the document was, which resulted in increased relevancy. Including the narrative lead to the increase of data to be considered by the matching, which negatively affected precision (Tune, Varma & Pingali, 2007).

The Afaan Oromo CLIR system has outlined the various effects that search optimization methods such as stemming and the use of stopwords have on the efficiency of retrieval systems. The next section will discuss the retrieval of IsiZulu indigenous knowledge. This will include the challenges that were encountered when attempting to use the CLIR approach to the retrieval of IsiZulu text

2.2.4 Indigenous Knowledge Retrieval in IsiZulu

Indigenous knowledge is the local knowledge that is unique to cultures and societies. It is a body of knowledge that enables communities to survive, and it commonly held by the people (Cosjin, et al., 2002). Indigenous knowledge is based on the ideas, experiences, practices, and information that has been generated either locally or elsewhere and has gone through a certain amount of transformation in other to be incorporated in the way of life of a particular culture. Indigenous knowledge is not confined to rural areas; it is present throughout all types of communities (Njiraine, et al., 2010). Indigenous knowledge has been originally collected and stored in paper archives, however, the digitization of these achieves has necessitated the development of an information retrieval system.

Due to government legislature, efforts have been increased to collect indigenous knowledge from various language groups, to prevent loss. A study that was carried out in Kenya shows that there has been a significant increase the publications in South Africa (Njiraine, et al., 2010). In

[7] Conference and Labs of the Evaluation Forum

the case of Kenya, the slow growth has been attributed to the possible absence of legislature regulations in the publications of indigenous knowledge.

Cross Language Information Retrieval Systems were the domain of research concerning indigenous knowledge. This is due to the South Africa situation, where there are 11 official languages. As seen in Fig. 2, the publications in South Africa increased rapidly from the year 2000 and reached its peak in 2005. The rate of publication then decreased sharply in the next few years.

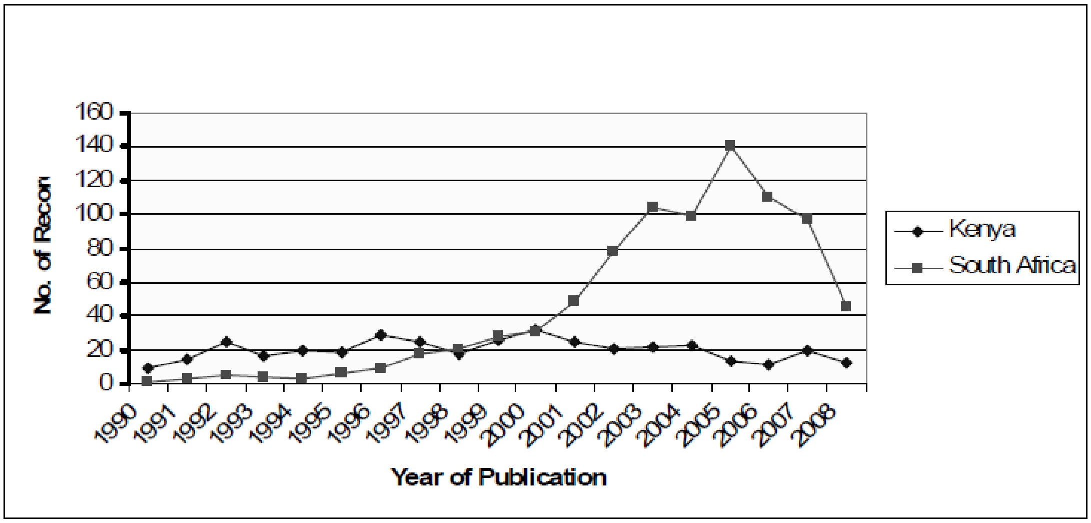


Figure 2: Trend of publications of IK Literature

In IsiZulu, grammatical information is conveyed by the morphology of the language. This presented several challenges when attempting to submit user queries in English to retrieve indigenous knowledge in IsiZulu. For example, the word “esesabekayo” comes from the root “esabeka” which can either mean capable, fearful, feared, awe inspiring, or wonderful. However, in English, the words capable, fearful and wonderful all have different semantics. This resulted in a loss of semantics in the translated meanings, which affected the precision of the returned results (Cosijn, et al., 2007). Similar problems were found when attempting to translate from an IsiZulu term to the English language, when the results had been retrieved from a query.

The next section will discuss web crawling, which is the process of harvesting web pages from the web page. The relevance of this top stems from the fact that the IsiZulu search engine will need to index web pages that will be aiding in providing search results given a user’s query.

2.3 WEB CRAWLING

A web crawler is an Internet software application that systematically browses the World Wide Web to index the contents of websites or the Internet as a whole. Given a set of URLs as input, the crawler visits these URLs and, based on a set of rules, indexes the page and also scans for other URLs within the same page, which it can next visit (Spiegler, Van Der Spuy & Flach, 2010).

As the focus of this project is to create an African Language based search engine, there is need for a focused crawler that is able to harvest documents that contain some IsiZulu text. This section will discuss two types of focused crawlers (topic and language based) and language corpora.

2.3.1 Topic Based Crawler

A topic-based crawler generally seeks, acquires, indexes, and maintains pages on a set of specific topics that represent a relatively narrow segment of the web. A topic-based crawler has two main components, which is a classifier and a distiller. The classifier makes relevance judgments on the crawled pages based on the selected topics that are given by the user as input. The distiller ranks the importance of the crawled pages and aims to identify hubs for a particular topic to improve the rate of harvesting (Chakrabarti, et al., 1999).

Ideally, a focused crawler should be able to download webpages that are only relevant to the topic being that is required. The probability that a link to a particular page is relevant can be predicted before downloading the page. This can be done with the use of anchor text analysis, which checks to see if the anchor text has a correlation to the topic being investigated. Another approach would be to determine the relevance of a page after its content has been downloaded (Yamana & Chan, 2010). Once found, relevant pages would be harvested by the crawler and any links found on the harvested page can be added to the crawl frontier. The frontier is responsible for determining the next URI to be visited by the crawler.

Relevance is enforced on to the crawler with the use of a hypertext classifier that uses a categorized taxonomy to judge which categories a certain document belongs to. There are two strategies used, the soft focused and hard focused strategy. In the soft crawler strategy, the crawler uses the relevancy score of each crawled page as a priority value for all the unvisited pages that are added to the crawler's frontier (Chakrabarti, et al., 1999). In hard focused crawling, the classifier is invoked on a newly crawled page and checks the taxonomy for the best matching category. The URLs found on the page as also checked against this category, if they are nodes on the best matching path, they are added to the crawler's frontier.

Focused crawlers have been found to provide highly specialized fields with highly relevant information using a topic based classifier. However, there is need for human input in training the crawler to become highly specialized through examples and manual classification of the results that are received (Chakrabarti, et al., 1999).

The next section will discuss language-based crawlers. The language based crawler differs with the topic based crawler in that its classifier requires a language model to determine if a web page should be harvested or not.

2.3.2 Language Based Crawler

As the World Wide Web has expanded over the past few years, the CJK (Chinese, Japanese, and Korean) web has seen an increase in its websites. Focused crawlers have been developed to capture these unique languages as the English crawlers have given unsatisfactory results in the harvesting of non-English web pages (Yamana & Chan, 2010). The interconnectivity of the Web has led also to several issues, as it is possible to find a CJK webpage that links to an English one or vice versa.

Two strategies were used in ensuring that the crawler was able to identify the language of the Web page. The first method was to extract the domain name from the hyperlink's URL and then determine the top-level domain, for example, ".js" for Japanese Web pages. Once the URL has been determined to be of the targeted domain, the next step was to enqueue the URL for crawling. Alternatively, if the anchor text is also in the target language, then it is also added to the queue of URLs to be visited (Yamana & Chan, 2010).

The second method that was used is similar to the topic-based crawling concept, but required replacing the topical classifier with a language identifier that would indicate if the downloaded page is written in the target language (Yamana & Chan, 2010). This required a language model to be built, considering the language and character encoding schemes from the occurrence frequency of each n-gram in each language's text corpus (Yamana & Chan, 2010). A language model, usually known as a statistical language model, assigns a probability to a set of words using a probability distribution, to estimate the probability that a particular text is in a particular language.

The next section will discuss language corpora and their effects on language based focused crawlers.

2.4 LANGUAGE CORPUS

A corpus is a large set of structured texts that are used to do statistical analysis, hypothesis testing and rule validation. In the case of information retrieval, corpora provide information about a language that is used to create faster and efficient retrieval systems (Spiegler, et al., 2010).

Corpora have been found to affect the performance of focused crawlers and cross language information retrieval systems. In the English-Zulu CLIR[8] system, a parallel text corpus was created to translate from one language to the other (Cosjin, et al., 2002). Several ambiguities resulted in some translations being incorrect and thus affecting the quality of the queries that are submitted. In language based crawling, a language corpus is used to create a language model which is a statistic model which approximates if a given stream of text is of a particular language. This allows the crawler to classify if the page should be harvested or not.

In the Afaan Oromo case, it was found that the quality of the corpus affects the stopword lists that are generated by the stopword algorithm, which has several effects on the words that are omitted. Additionally, some terms were not found in the dictionary source, which reflects on the quality of the language corpus being an issue (Tune, et al., 2007).

2.5 SUMMARY

The purpose of this chapter has been to provide an overview into the field of information retrieval and an investigation into existing works in the field. This included several strategies that were used in improving the efficiency of the retrieval systems. The problems that were encountered in the development of the existing works will also be considered when establishing the scope of the project.

[8] Cross language information retrieval

Chapter 3

3 DESIGN

The first section of this chapter will outline the requirements analysis and a high level design of the system. Following on from this, the implementation of the search system will be discussed. This includes the implementation of a prototype and the design and development of the algorithms that will aid in the answering of research questions

SECTION 1: HIGH LEVEL DESIGN AND USER INTERFACE IMPLEMENTATION

This section outlines the analysis and design process that was undertaken in the designing of the IsiZulu search engine. The focus of the requirements analysis and design process is mainly centred around the user interface. The interface provides the point of interaction between the users and the retrieval of information that has been indexed in the search engine.

Additionally, the components that are required to help in the development of the system have been analysed to ensure that their implementation will be feasible and contribute to the overall success of the project.

3.1 AN OVERVIEW OF THE PROPOSED SOLUTION

The proposed solution requires various existing technologies to be customized and be used as complementary parts to construct the system.

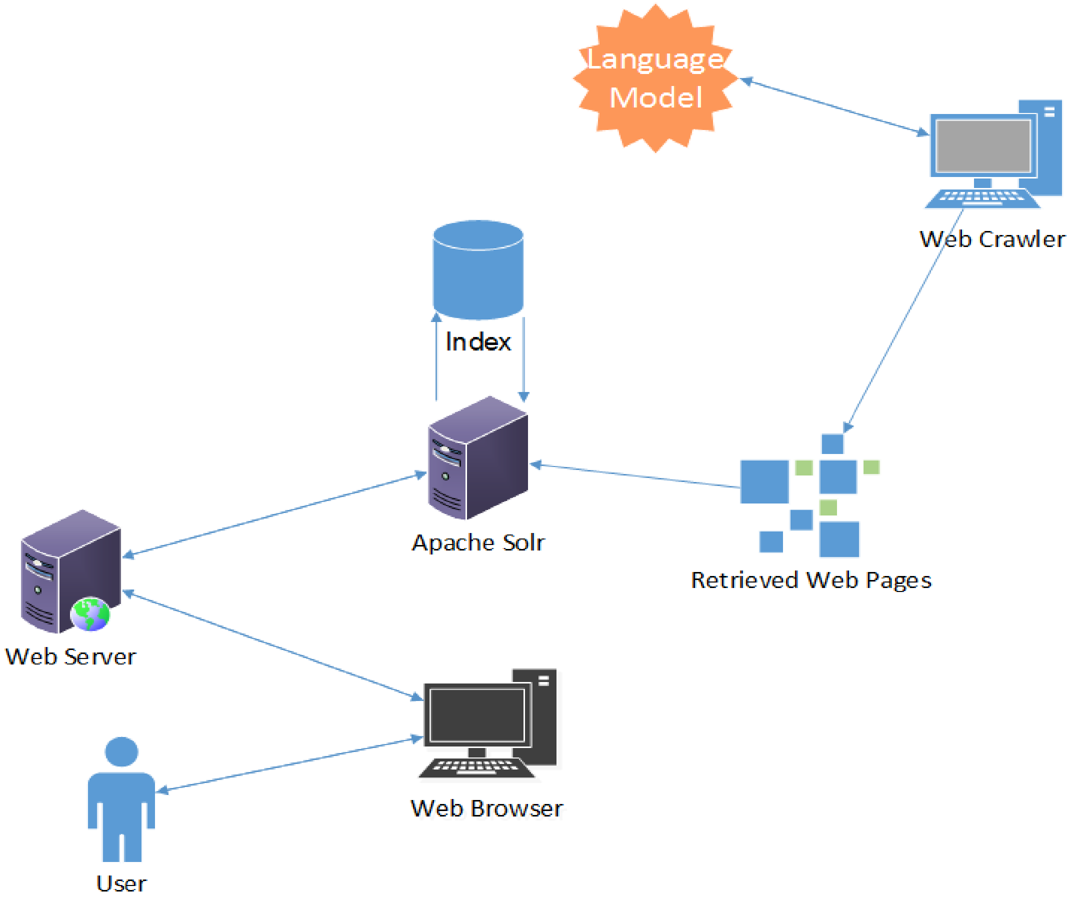


Figure 3: An overview of the Afriweb IR system

Fig. 3 outlines the main components of the system. There are two main parts to the system which, are the indexing and retrieval of data, and the harvesting of web pages and documents from the World Wide Web. The split of the system into two parts allowed for concurrent development with the only dependency being the Web pages and documents that were to be indexed in the system.

An outline of the existing technologies that are to be employed in the development of the system are as follows:

3.1.1 Indexing and Retrieval of Information

a) Web Server – a Web server is computer or software system that processes requests via HTTP(hypertext transfer protocol), which is a protocol used to distribute information through the World Wide Web (Webopedia, 2014). For this project, a web server is required to host the IsiZulu Search Engine web pages and publish the results of the requests that are sent through to the search engine.

b) Search Engine Interface – the search engine interface is the set of web pages that the user is going to interact with which will be hosted by the web server. Through this interface, a user is going to be able to submit a query that is based on a particular information need and be able to view the results from the search engine.The search engine interface will be accessbile via the web browser.

c) Apache Solr - Apache Solr is a search platform, which will primarily be used to index and retrieve documents based on a query supplied by the user. Additionally, the search platform will be customized through plugins and alteration of the schema properties to ensure efficient indexing and retrieval of documents.

3.1.2 Harvesting web pages

a) Web Crawler - A web crawler is an Internet software application that systematically browses the World Wide Web to index the contents of websites or the Internet as a whole. Given a set of URLs as inputs, the crawler visits these URLs and based on its set of rules, indexes the page and scans for other URLs within the same page, which it can next visit. The crawler os used for harvesting IsiZulu text from the World Wide Web so that these pages can be indexed in the search engine.

b) Language model – a language model assigns a probability to a certain sequence of words to estimate the likehood of the sequence of being a particular language or a particular part of speech(Medelyan et al., 2006). Language modelling is used innatural language processing techniques that use a computational approach to a language. In this case, a language model will be used to classify a web page as either being written or having relations to the IsiZulu language.

3.2 Search Interface Analysis

The main focus of this project was the development of the search engine with a specific focus to the algorithms that analyze the documents that were to be indexed in the search engine. However, a search interface was required to ensure that users could interact with the search engine. The design and implementation of the search engine was primarily based on existing interfaces such as Google and Bing.

3.2.1 Analysis of existing search interfaces

The images below show screen shots of the most popular search engines, Yahoo, Bing, Google and Yandex. An analysis of the interfaces show that each user interface contains a logo, input text box and a search button. Various other widgets have also been included, such as language options, links to privacy policies and terms of service. As seen in the following figures, there has been a standardization of the search interface across the popular search engines.

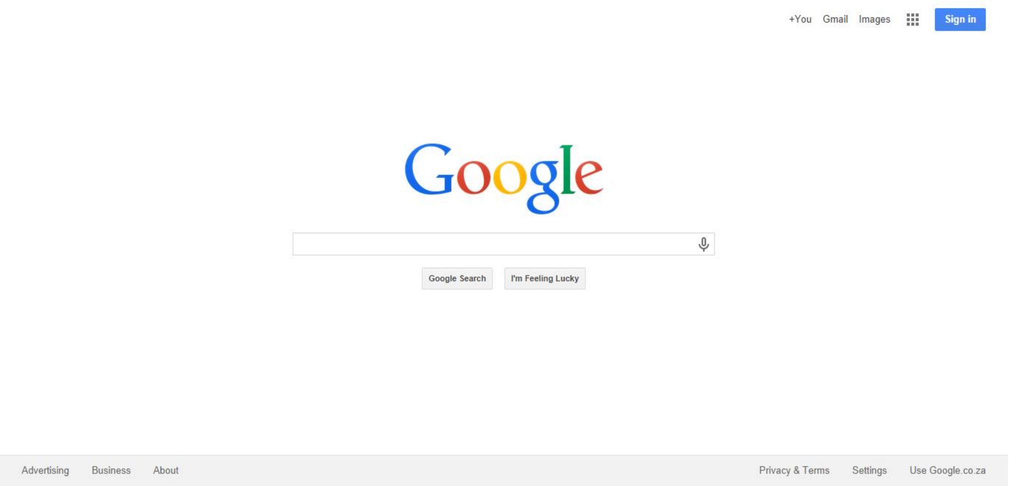


Figure 4: A screen shot of the Google Search interface

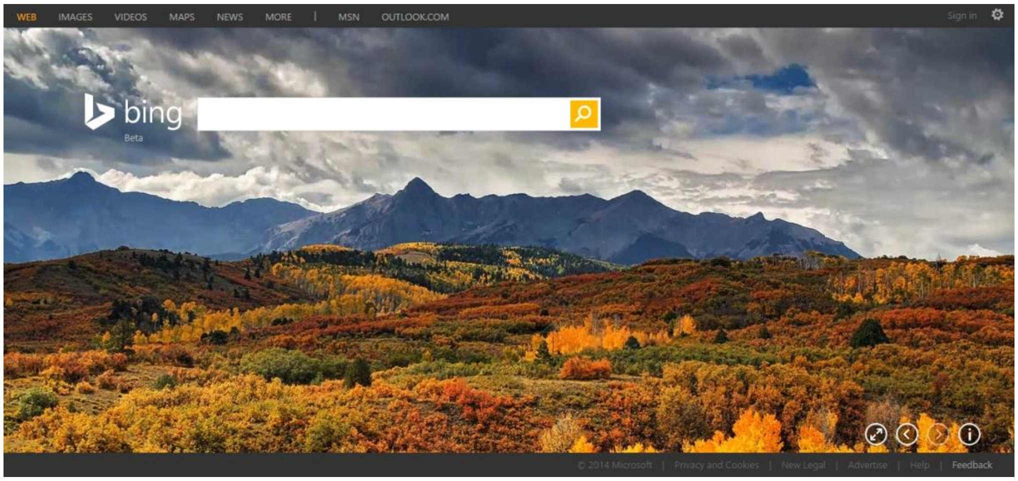


Figure 5: Screenshot of the Bing Search Interface

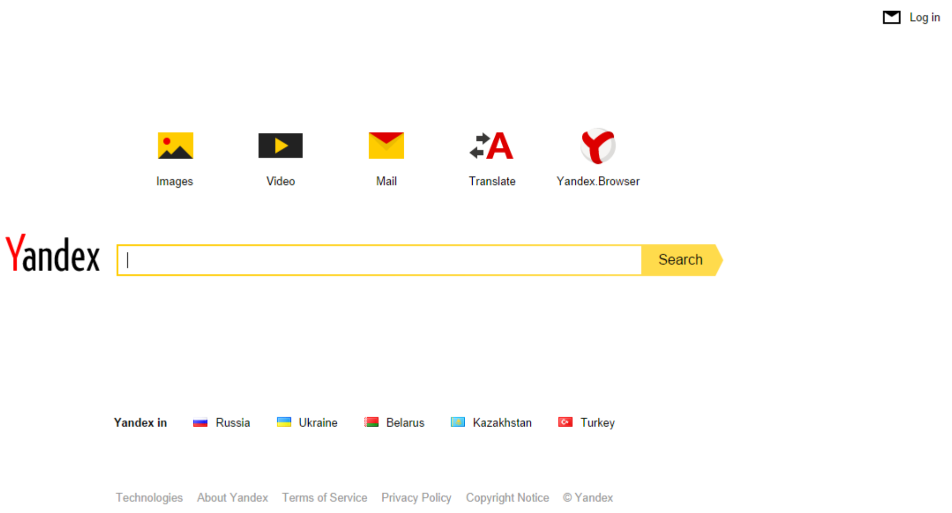


Figure 6: Screenshot of the Yandex search interface

There are several similarities that can be noticed in these the search interface screen shots. First, the user interfaces provide a relatively “clean” interface with a very specific purpose, to provide search results to a user. Intrusion is kept to a minimum to ensure that the least amount of options are required searching for information (Hearst, 2009). Additionally, as seen in the screenshots in Appendix A, each of the search interfaces use a list format in the presentation of results.

Secondly, search engine interfaces are designed to optimally attain the goals of usability. These goals are learnability, efficiency, memorability, and satisfaction, which aim to ensure that an interface provides a high amount satisfaction without compromising the functional goals of the interface(Hearst, 2009). This is achieved through following a user centred design approach which allows users to be incorporated into the design and implementation of the interface. Additionally, the users are able to assess the interface throughout the process and provide useful feedback on how the interface can be better designed.

Practitioners in Human-Computer Interaction have put forward a set of guidelines that are directed at the design of search interfaces. According to Shneiderman et al (1997), the 8 design guidelines for user interfaces are offering informative feedback, supporting user control, short-term memory load, provision of short cuts for skilled users, offering simple error handling, consistency, easy reversal of actions, and designing for closure(Hearst, 2009).

Therefore, it can be assumed that the popular search engines have been standardized due to the use of search engine design guidelines and following the user centred design process. By following the design of the popular search interfaces, it can be assumed that the IsiZulu search interface will accomplish the functional goals of a search engine interface as well as attain the usability goals.

3.3 SEARCH INTERFACE PROTOTYPING ITERATION

This section will outline the iterative design process that was followed in the development of the search interface. This includes the development of a rapid prototype, the evaluation process and the improvements made to the initial design based on user recommendations.

As outlined in the research proposal, an iterative development process was adopted in the project to allow for the incremental development of the various artifacts ensuring that a working system will be produced at the end of each development cycle. Additionally, the incremental development approach will ensure that the project was sensitive to change and could be easily altered based on new requirements.

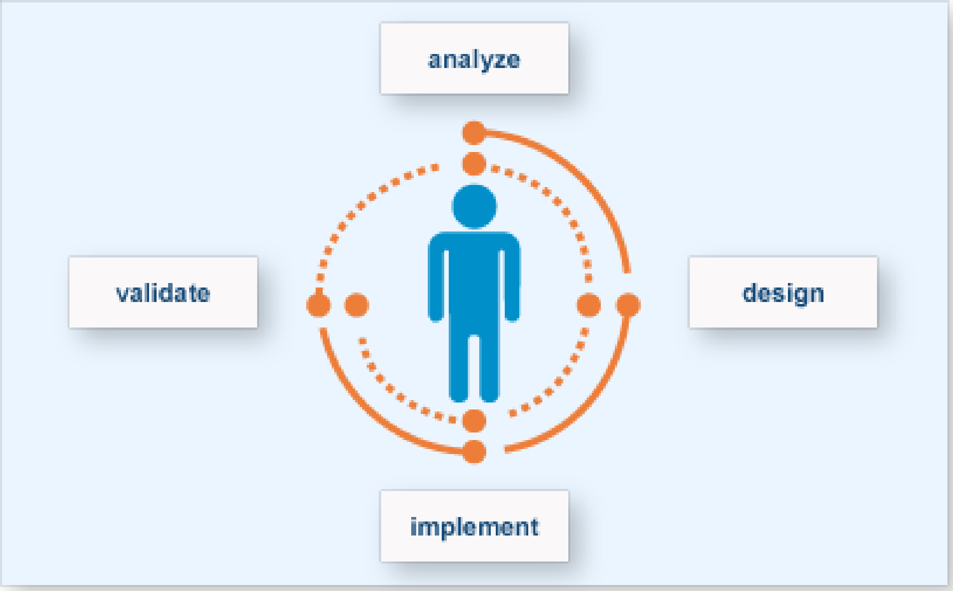


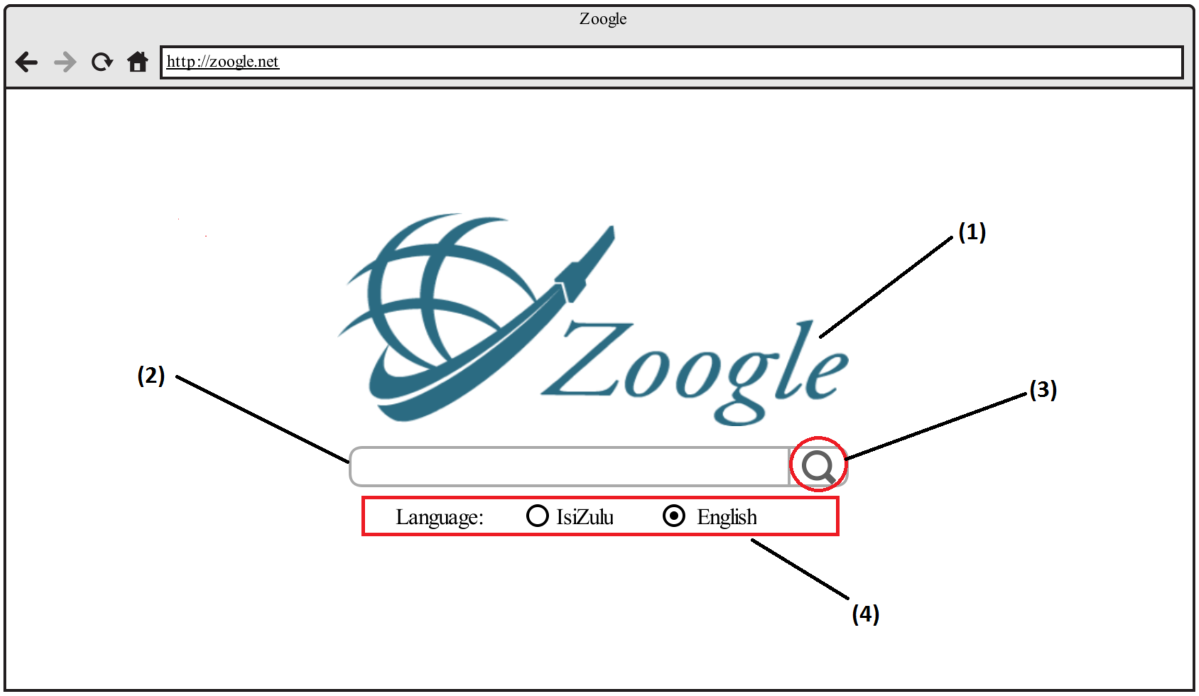
Figure 7: A visual representation of the iterative methodology

Fig. 7 above shows the four main stages that were incorporated in the development of the search interface. Each stage began with an analysis, which informed the design of the feature or artifact. The next step was to implement the feature and validate it before integrating into the system.

3.3.1 User Interface Design

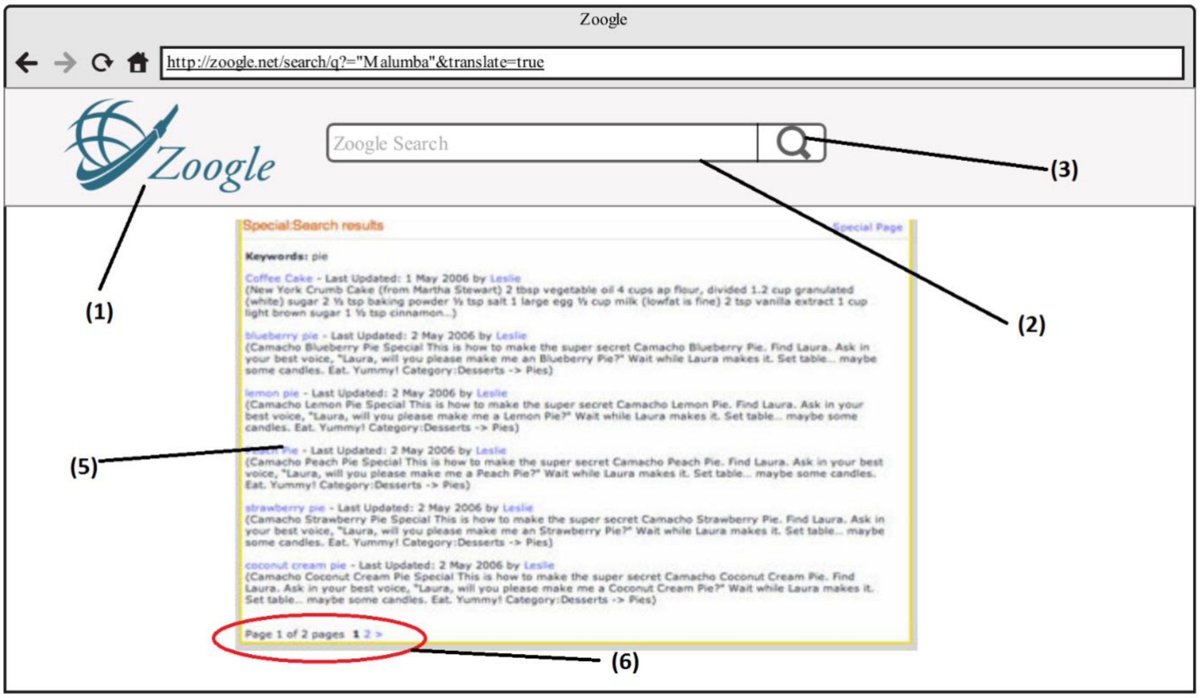
On completion of the analysis phase, the next step was to create a prototype of the user interface. A key feature that had to be catered for was allowing the user input both IsiZulu and English language queries using the same interface. Initially, the idea was to make use of language recognition APIs or plugins that are produced commercially. However, this option proved to be immensely difficult because the technologies available were either unable to complete the task in the required fashion or were not available to the public.

A wireframing tool was used in the development of the search interface. This provided various benefits such as the ability to incorporate the flow of screens into the process and automation of the design process. Additionally, the tool has the option of creating design blueprints that have the measurements and various layout options that can be used in the development process.



Language: ○ IsiZulu ● English

Figure 8: Homepage of the proposed search engine



Special:Search results

Special Page

Keywords: pie

Coffee Cake - Last Updated: 1 May 2006 by Leslie
(New York Crumb Cake (from Martha Stewart) 2 tbsp vegetable oil 4 cups ap flour, divided 1.2 cup granulated (white) sugar 2 ½ tsp baking powder ½ tsp salt 1 large egg ½ cup milk (lowfat is fine) 2 tsp vanilla extract 1 cup light brown sugar 1 ½ tsp cinnamon...)

blueberry pie - Last Updated: 2 May 2006 by Leslie
(Camacho Blueberry Pie Special This is how to make the super secret Camacho Blueberry Pie. Find Laura. Ask in your best voice, "Laura, will you please make me an Blueberry Pie?" Wait while Laura makes it. Set table... maybe some candles. Eat. Yummy! Category:Desserts -> Pies)

lemon pie - Last Updated: 2 May 2006 by Leslie
(Camacho Lemon Pie Special This is how to make the super secret Camacho Lemon Pie. Find Laura. Ask in your best voice, "Laura, will you please make me a Lemon Pie?" Wait while Laura makes it. Set table... maybe some candles. Eat. Yummy! Category:Desserts -> Pies)

Peach Pie - Last Updated: 2 May 2006 by Leslie
(Camacho Peach Pie Special This is how to make the super secret Camacho Peach Pie. Find Laura. Ask in your best voice, "Laura, will you please make me a Peach Pie?" Wait while Laura makes it. Set table... maybe some candles. Eat. Yummy! Category:Desserts -> Pies)

strawberry pie - Last Updated: 2 May 2006 by Leslie
(Camacho Strawberry Pie Special This is how to make the super secret Camacho Strawberry Pie. Find Laura. Ask in your best voice, "Laura, will you please make me an Strawberry Pie?" Wait while Laura makes it. Set table... maybe some candles. Eat. Yummy! Category:Desserts -> Pies)

coconut cream pie - Last Updated: 2 May 2006 by Leslie
(Camacho Coconut Cream Pie Special This is how to make the super secret Camacho Coconut Cream Pie. Find Laura. Ask in your best voice, "Laura, will you please make me a Coconut Cream Pie?" Wait while Laura makes it. Set table... maybe some candles. Eat. Yummy! Category:Desserts -> Pies)

Page 1 of 2 pages 1 2 >

Figure 9: Results page of the proposed search engine

The images above shows the initial design of the IsiZulu search engine interface. Fig. 8 shows the initial interface(home page) that the user encounters when accessing the search platform. Fig. 9 depicts the layout of the results that are given once a user has provided a query.

The description of each of the numbered features is as follows:

1) Logo – the logo does not necessarily have a particular functional uses, however, it has been included to give the look and feel of the existing search engine user interfaces.

2) Text box – the text box is to be used for accept the user input as a query.

3) Search Icon – the search icon is a submit button that sends a user query through to the search engine for processing.

4) Language Selector – the language selector has been included for the user to select the input language of the query. This feature has been included to tackle to problem of not having an open source plugin that can be easily integrated into the interface. Additionally, those that are available either do not cater for the IsiZulu language or require licensing fees.

5) Search Results – the query results will be displayed in a list format with a preview of the text contained in the results truncated.

6) Pagination controls– the user interface will display about 10 results per page. The pagination buttons will help the user to page through the results.

3.3.2 Informal Evaluation of the User Interface

An informal evaluation of the initial search interface was carried out to validate if a user could easily search for information using the search interface as an intermediary tool for completing this task. the evaluation was based on the goals of usability which are learnability, efficiency, and satisfaction.

The wireframing tool that was used (moqups[9]) had the option of making the wireframes clickable and providing some feedback on the action that the user had taken. The evaluation included an introduction to the task that required completion and a scenario, which provided the context to the process. The users were observed to record any errors that during the task and to probe about any potential areas of improvement.

3.3.2.1 Overview of the evaluation process

A sample of 5 users were asked to participate in the evaluation process to determine whether the search interface was an effective tool to interact with the search engine. The scenario that set the context of the evaluation process was a follows:

“You have just been told by a friend that there is a new search engine that is able to search for information in a specific language. The search engine currently caters for information in the IsiZulu language. The queries can be submitted in either English or IsiZulu. Please complete the task of searching for information using the search engine”

After completing the process, the results were gathered and a list of suggesions and problems were compiled. The list of the suggestions and problems are as follows:

1. The search icon – a few users assumed that this option represented an option to zoom in as, in other search interfaces, this tool is for either zooming in and out. The suggestion was to replace this with the word “Search”.

2. Issues with the language selection option – there was a general assumption that the language selection option [number (4) in Fig. 9], was for setting the language of the search interface. The suggestion was that this be made clear to users what the option achieves.

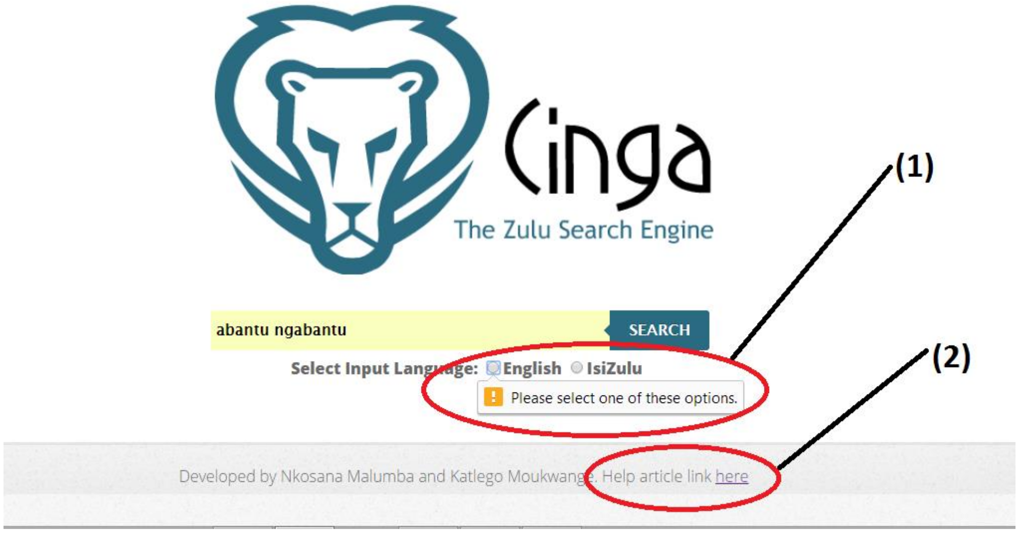
3. Lack of a help or “how to” article – it was also mentioned that a help article that would outline the process would be helpful to first time users or those that were not technologically adept.

[9] https://moqups.com/

These recommendations were considered and implemented in the next iteration of designing the interface.

3.4 Second User Interface Design Iteration

After completing the user evaluation of the initial prototype, the next stage was to develop the interface and implement the recommendations that were extracted from the evaluation process. The implementation of the interface formed part of the vertical prototype that was to be developed at the end of the search interface design.The new interface, which can be seen in the images below, has introduced some new features and a slight redesign of the previous prototype.



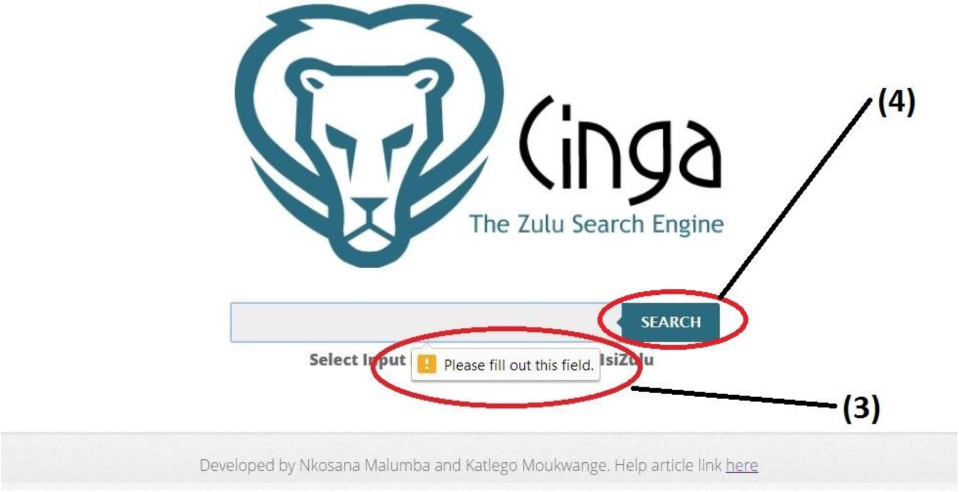
abantu ngabantu

Select Input Language: ○ English ○ IsiZulu

! Please select one of these options.

Developed by Nkosana Malumba and Katlego Moukwange. Help article link here

Figure 10: Redesigned homepage



Please fill out this field.

Developed by Nkosana Malumba and Katlego Moukwange. Help article link here

Figure 11: Redesigned homepage

An overview of the features of the user interface is as follows:

a) Feedback – Two new feedback features were implemented to guide the user from selecting the wrong option. The first, depicted in Fig. 10 above as (1), ensures that user selects the language option before submitting the query. The message is displayed if the user omits this action before clicking on the submit button. The second, depicted in the diagram as (3), ensures that no blank queries are submitted to the search engine.

b) Help article – As noted in the user evaluation process, there was a request to add a help article so that users can be able to read the steps required to submit a query. The help article is available by clicking the link, depicted as (2) in Fig. 10, which will redirect the user to a page which has all the necessary details.

c) Changing the search icon – the search icon was changed to conform to the standards of the more popular search engines. The icon has been changed to a button with the words “Search” as shown in the Fig. 11 as (4).

Additionally, a clean interface was chosen to reduce the cognitive load that is required when a user is performing a task using the search interface.

After the search interface was developed, the next step was to ensure that the pages and the links were working in the required manner. To host the website, a Web server software package was required. In this case, XAMPP, an open source software package that comes with equipped with PHP support was chosen as it offered a simple integration of both the web server and the translation API.

This section describes the system development iterations that followed on from the development of the user interface. Due to the split of the project, this section of the report will only focus on the indexing and retrieval aspects of the system. This stage was meant to provide light on the search engine system as a whole, ensuring that the aspects that had been chosen could be implemented, and provide a good workflow that will accomplish the necessary tasks. The scope of the vertical prototype included the user interface, translation API and the open source search platform.

3.5 THIRD ITERATION: SYSTEM INTEGRATION

The third iteration focused on the integration of the user interface with the translation API and the open source search platform

3.5.1 Translation API Integration

The language translation API was integrated into the search interface to enable the feature that allows users to input queries in English. Initially the plugins that were selected are the Google translate API and the Bing Translation API. However, these were not implemented because a limited number of requests could be processed per day. Additionally, users would be required to pay a licensing fee to make use of the service on a larger scale.

The next alternative was to make use of an open source API that was just as effective. The chosen was the MyMemory[10] translation API that offered a free statistical machine translation service. The service translates words based on a translation memory that is both crowd-sourced and uses some of the more popular dictionaries. The service is available via a RESTful API, which makes it relatively simple to integrate into the web server. Additionally the service requires a key, which can be request after registering onto the platform.

For the translation API to make requests through PHP, the cURL[11] library has to be installed. The XAMPP server comes bundled with this library but it can also be installed separately. Once the integration was completed, the API was tested by sending translation requests using the user interface. As mentioned in the previous section, the translation API was also tested using the Google Chrome and the Mozilla Firefox browser and it produced the expected results.

[10] http://mymemory.translated.net/doc/spec.php

[11] http://curl.haxx.se/

3.5.2 Open Source Search Platform Integration

The final phase of developing the iteration required integration with the search platform. The chosen search engine is Apache Solr, which an open source search toolkit. Apache Solr provides features such as full-text search, near-real time indexing, database integration and full customization via plugins and XML configuration (Foundation, 2011). The most important feature of this application is the fact that new functionality can be introduced through custom-built plugins. This will be a key feature that will ensure that the indexing and searching can be optimized in later stages.

In order to integrate the search engine platform with the user interface, a Solr client had to be developed to process requests. Apache Solr has PHP support, which ensured simple integration with translation API that has been discussed in the previous section. An existing library, Solarium[12] was used to provide the PHP support for the Solr client. Solarium is an open source project that provides a PHP library that provides a simple HTTP communication protocol between the platform and the user interface. Additionally, the library offers good object orientated programming practices, which allows the user to customize and extend the features that have been included in the library.

To use the Solarium library, one has to download it via a software tool called Composer[13]; a package manager for PHP projects makes it simpler to declare dependencies amongst the packages that have been included in the project. Once installed, the next step was to create a JSON file that stated the version of the Solarium library. By running the composer install command through command prompt, the library dependency was built. A few scripts had to be written to ensure that the ports for communication between the user interface and Apache Solr were configured. A few ping queries were executed to ensure that the integration had been successful.

The last step in this exercise was to ensure that the correct data flow would be established. For example, if the user selected the English query option, the query has to be sent through to the translation API and then to Apache Solr to query the data stored in the index. This required some simple server side scripting to cater for the two language selection options.

Once this process was completed, the components were tested using the Google Chrome and Mozilla Firefox browsers. Sample data was indexed in the search engine to ensure that the data was being received and that some results would be received as a query.

[12] http://www.solarium-project.org/

[13] https://getcomposer.org/

SECTION 2: PRE PROCESSING ALGORITHMS

This section describes the development of two pre-processing algorithms that were used in the indexing and querying of information stored in the system.

3.6 FIRST ITERATION: STEMMER DEVELOPMENT

3.6.1 Introduction

This section describes the development of the stemming algorithm that is used for manipulating the indexing and querying of data that is indexed on the search platform. According to Lovins (1986), a stemming algorithm is a computational procedure, which reduces all words with the same root to a common form by stripping each word of its derivational and inflectional suffixes. Stemming is widely used in computational linguistics in the development of applications such as thesauruses and some search engines such as Google.

3.6.2 Stemming algorithm analysis

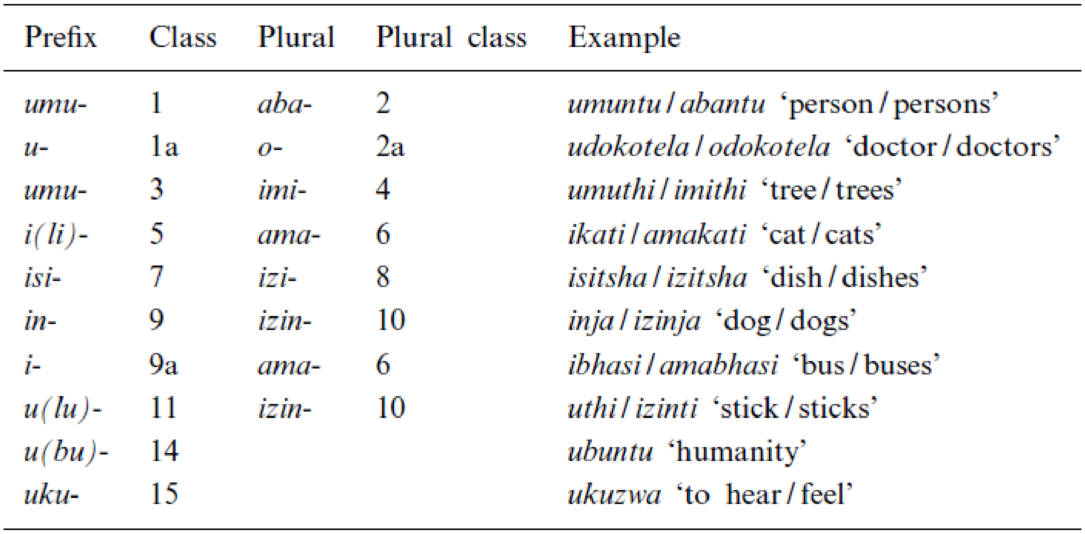
The relationship between a query and a document is determined by the frequency of the terms contained in the query that the document has (Lovins, 1968). However, as documents have to adhere to language constraints, a single word may have morphological variants and the term matching algorithms that are used to retrieve stored documents may not recognize the variants (Hull, 1996). Thus, the implementation of an effective stemming algorithm will ensure that the various inflected forms of the word can be mapped to a single stem.

There are two main principles that are used in the development of stemming algorithms, the iteration and longest-match principle. According to Lejnieks (1967), the iteration principle is based on the fact affixes are attached to stems in a certain order using a predefined class of affixes. The algorithm simply removes the affixes from either start to end or end to start; based on which class the detected affix matches. The second principle, the longest match; states that within any given class of endings, if more than a single ending provides a match, the longest one should be removed from the word.

IsiZulu is an agglutinative language in which complex words are derived by a combination of morphemes that have a grammatical or semantic meaning. The meaning of a particular word is determined by the affixes that are applied to a particular root (Spiegler, Van Der Spuy & Flach, 2010). Depending on the category of the affix that is applied to the word, certain language morphology phenomenon can occur, altering a particular morpheme or segment of the word. IsiZulu has a noun classification system that categorizes nouns into several noun classes that are determined by noun prefixal morphemes (Pretorius & Bosch, 2003).

These categories determine the types of nouns that can be applied a particular now, for example, classes 1 and 3 both have the prefix “umu-“. However, class 1 can only be applied to living things and class 3 to abstract objects. Therefore, the possible noun classes determine a word’s class firstly that it belongs to and whether it is a noun or verb.

Table 2: The IsiZulu noun classification system



Prefix	Class	Plural	Plural class	Example
umu-	1	*aba-*	2	*umuntu / abantu* ‘person / persons’
u-	1a	*o-*	2a	*udokotela / odokotela* ‘doctor / doctors’
umu-	3	*imi-*	4	*umuthi / imithi* ‘tree / trees’
i(li)-	5	*ama-*	6	*ikati / amakati* ‘cat / cats’
isi-	7	*izi-*	8	*isitsha / izitsha* ‘dish / dishes’
in-	9	*izin-*	10	*inja / izinja* ‘dog / dogs’
i-	9a	*ama-*	6	*ibhasi / amabhasi* ‘bus / buses’
u(lu)-	11	*izin-*	10	*uthi / izinti* ‘stick / sticks’
u(bu)-	14			*ubuntu* ‘humanity’
uku-	15			*ukuzwa* ‘to hear / feel’

As seen in the Table 2, each class has a singular and a plural form, for example, classes 1a and 2a. Noun classes 12 and 13 do not exist in IsiZulu but they form an integral part of the noun classification system of Bantu languages (Pretorius & Bosch, 2003). Hence, in the table above, these classes were omitted. Thus, the existence of the classification system allows for the development of the stemming algorithm iteration principle, which removes an affix based on the longest match that found within a finite sent of prefixal morphemes.

There are a given set of suffixes that can be applied to a root word, that convey aspects of the language known as nominal suffixes. The nominal suffixes that the stemming algorithm removes include diminutive, feminine, augmentative, deverbative, and locative suffixes. Table 3 denotes the different types of suffixes.

Table 3: A table showing the suffixes in the IsiZulu language

Suffix Type	Suffix Ending
Diminutive	-ana
Feminine	-azi or –kazi
Locative	-eni
Augmentative	–kazi
Deverbative	-i or -o
Evaluative	-se

Depending on which suffix that has been used in the formation of a word, certain phenomena such as palatalization and vowel elision can occur.

Palatalization is a change in the phonetics of a word when the diminutive suffix is applied to it:

For example:

Ikhanda + -ana = ikha**nda**ana > ikha**nj**ana
root *diminutive* *result*

In the example above, when applying the diminutive suffix "-ana" to the root word "ikhanda", the ending "nda" changes to "nja".

Vowel elision is a process when a subject concord consisting of a consonant and a vowel are prefixed to a vowel verb stem. The vowel of the subject concord is then remove when the word is formed. An example is

Ngi- + -eba > ngeba (I steal)
(*I)* *(steal)*

(Pretorius & Bosch, 2003)

The suffixes will require the longest matching principle, which results in the longest matching suffix being removed from the word. Thus, the word will be analyzed against the suffixes that have been listed above to check which applies to the word. Once this is determine, the suffix will be removed from the word and the same process will apply to the prefix. However, for a prefix, the noun classification system will be used to match the prefix and remove it from the root word. The algorithm for the stemming process is as follows:

```
#Suffix Matching Algorithm
    IF (word.length >= 4):
    //using longest match principle
        ITERATE over order list of suffixes:
            IF word endsWith (suffixList[i]):
                MARK suffix as longest
                Matching endings ++
    ELSE:

        Matching endings = 0;



FOR EACH word:
    CHECK the suffix match:
        IF (matching endings >= 1):
            REMOVE longest ending
            Check if the suffix is a special case:
                IF special case suffix:
                    LOOKUP suffix category
                        IF category == diminutive suffix:
                            REVERSE palatalization
                        ELSE IF category == locative suffix:
                            REMOVE locative prefix
                        ELSE IF category == feminine suffix:
                            REVERSE vowel elision
                        RETURN word
        ELSE:
            RETURN word
    Check the prefix ending of the word:
        IF prefix in Noun Classification List:
            REMOVE prefix
    RETURN WORD
```

Figure 12: Stemming algorithm description

In Fig 12, a “special case suffix” refers to the suffixes (feminine, diminutive and locative) that result in a change in the morphemes of the word when they are applied to the root form. The suffix-matching algorithm only checks the suffix if the word is greater than or equal to four characters. This is because the smallest possible word that contains a suffix is about four characters in length (Pretorius & Bosch, 2003).

3.6.3 Stemming Algorithm Design and Implementation

In the design stage of the IsiZulu stemmer, two existing algorithms were investigated, the Porter stemming and the Spanish stemming algorithm. The Porter Stemming algorithm simplified the rules for suffix stripping in the English language (Willett, 2006). The increase in the interest of the development of conflation techniques in the searching of text, thus the Porter algorithm is the standard stemming algorithm used on English text.

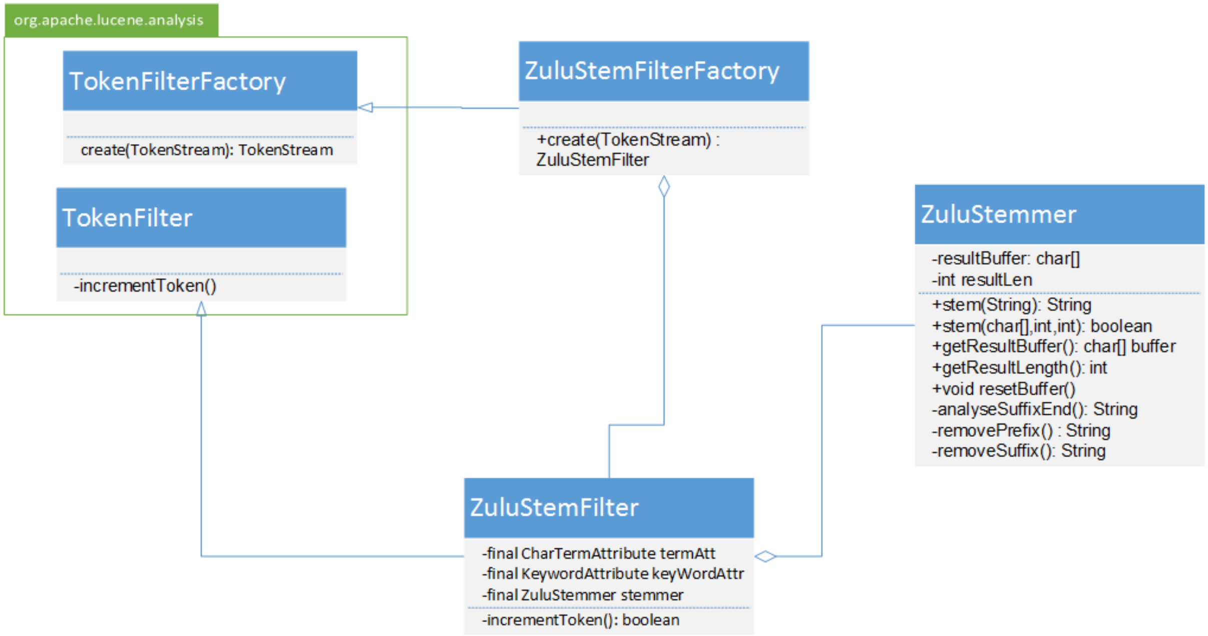

org.apache.lucene.analysis
TokenFilterFactory
create(TokenStream): TokenStream
TokenFilter
-incrementToken()
ZuluStemFilterFactory
+create(TokenStream) : ZuluStemFilter
ZuluStemmer
-resultBuffer: char[]
-int resultLen
+stem(String): String
+stem(char[],int,int): boolean
+getResultBuffer(): char[] buffer
+getResultLength(): int
+void resetBuffer()
-analyseSuffixEnd(): String
-removePrefix() : String
-removeSuffix(): String
ZuluStemFilter
-final CharTermAttribute termAtt
-final KeywordAttribute keyWordAttr
-final ZuluStemmer stemmer
-incrementToken(): boolean

Figure 13: IsiZulu Stemmer class diagram

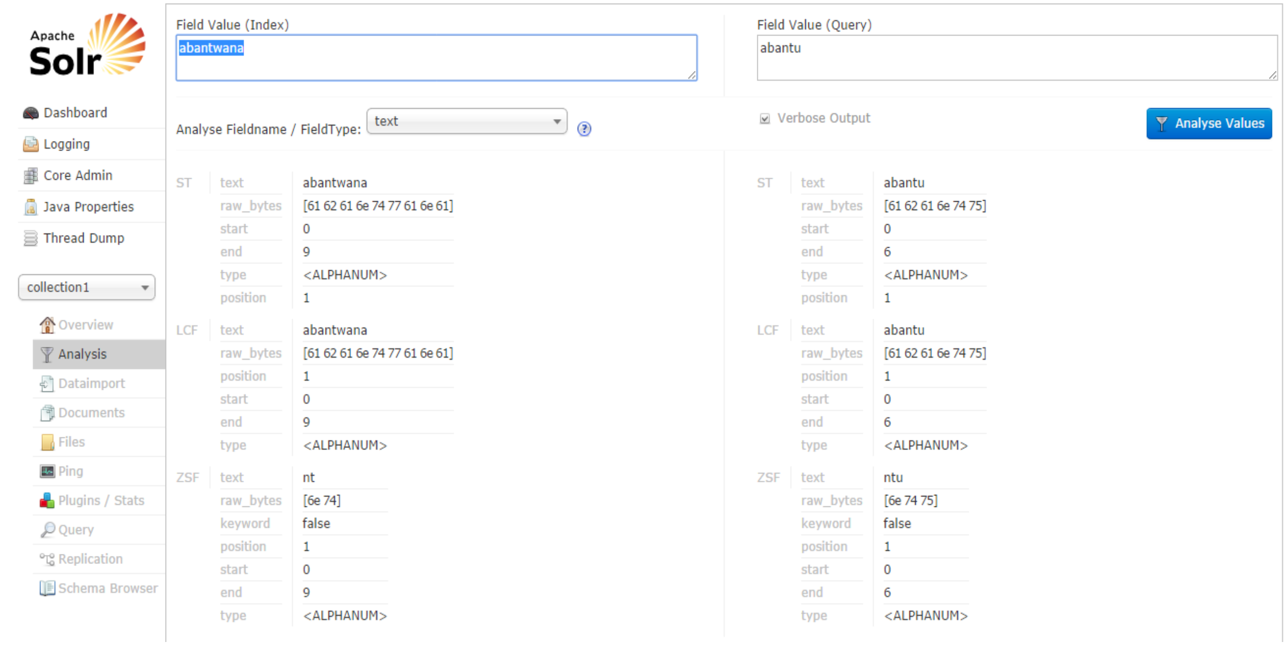
Fig. 13 shows the class diagram for the IsiZulu stemmer that was designed based on the Porter stemming algorithm that has been implemented in the Apache Solr search platform. The package "org.apache.lucene.analysis" is not available in the 4.7.2 release of the platform, but it can be downloaded from the Apache Lucene project. The package provided the token filtering functionality, which allows the stemmer to process a token (word). The TokenFilterFactory allows the bundling of several filters and tokenizers in the development process. The IsiZulu stemmer has been written in the Java programming language.

During the process of indexing and querying, the TokenFilter class in the Lucene Analysis package allows the each individual word to be filtered using a predefined filter class that is specified in the configuration. Therefore, the ZuluStemFilter is able to apply the stemming process to each token (word), which will result in the derivation of the root or stem.

3.6.4 Testing and Evaluation

Once development was completed, the stemmer was integrated into the search platform using a jar built from the project. In order for the IsiZulu stemmer to be used during indexing and querying, the jar file has to be specified in the Apache Solr schema that loads all the instructions during the search platform start-up. The ZuluStemFilter was compounded with the WhiteSpaceTokenzier class and the LowerCaseFilter class to provide the necessary filters to complete the stemming process.

Once the plugins had been loaded, they were tested using the built in analysis feature of the search platform. By navigating to the administration panel, one can access the analysis panel and load the particular field that has been customized.



Field Value (Index): abantwana

Field Value (Query): abantu

Analyse Fieldname / FieldType: text

☑ Verbose Output

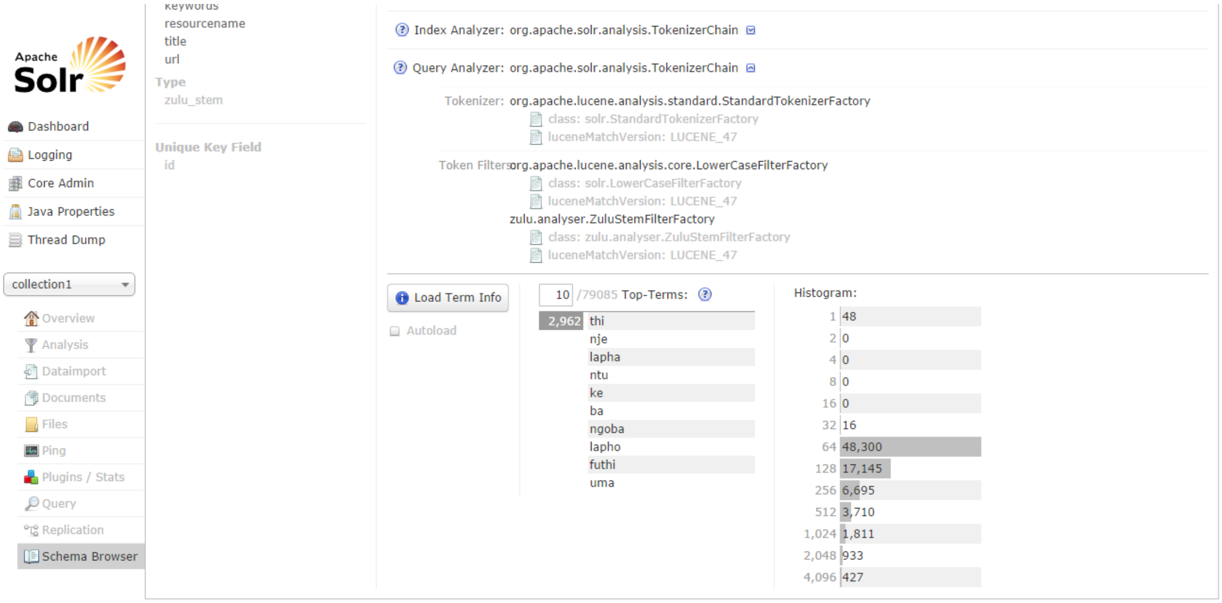
ST	text	abantwana
	raw_bytes	[61 62 61 6e 74 77 61 6e 61]
	start	0
	end	9
	type	<ALPHANUM>
	position	1
LCF	text	abantwana
	raw_bytes	[61 62 61 6e 74 77 61 6e 61]
	position	1
	start	0
	end	9
	type	<ALPHANUM>
ZSF	text	nt
	raw_bytes	[6e 74]
	keyword	false
	position	1
	start	0
	end	9
	type	<ALPHANUM>

ST	text	abantu
	raw_bytes	[61 62 61 6e 74 75]
	start	0
	end	6
	type	<ALPHANUM>
	position	1
LCF	text	abantu
	raw_bytes	[61 62 61 6e 74 75]
	position	1
	start	0
	end	6
	type	<ALPHANUM>
ZSF	text	ntu
	raw_bytes	[6e 74 75]
	keyword	false
	position	1
	start	0
	end	6
	type	<ALPHANUM>

Figure 14: Screenshot of the Apache Solr analysis interface with the IsiZulu Stemmer loaded (ZSF)

Fig. 15 shows a screenshot from the testing process of the stemming algorithm. The Field Index (Value) and Field Index (Query) text boxes allow text to be analyzed to ensure that the text is being transformed in the correct way using the appropriate filters.

In Fig.16, the filters can also be viewed to ensure that they have been loaded to the appropriate query and index analyzers.



keywords
resourcename
title
url

Type
zulu_stem

Unique Key Field
id

Index Analyzer: org.apache.solr.analysis.TokenizerChain

Query Analyzer: org.apache.solr.analysis.TokenizerChain

```
Tokenizer: org.apache.lucene.analysis.standard.StandardTokenizerFactory
               class: solr.StandardTokenizerFactory
               luceneMatchVersion: LUCENE_47

Token Filtersorg.apache.lucene.analysis.core.LowerCaseFilterFactory
               class: solr.LowerCaseFilterFactory
               luceneMatchVersion: LUCENE_47
           zulu.analyser.ZuluStemFilterFactory
               class: zulu.analyser.ZuluStemFilterFactory
               luceneMatchVersion: LUCENE_47
```

☐ Autoload

10 /79085 Top-Terms:

2,962	thi
	nje
	lapha
	ntu
	ke
	ba
	ngoba
	lapho
	futhi
	uma

1	48
2	0
4	0
8	0
16	0
32	16
64	48,300
128	17,145
256	6,695
512	3,710
1,024	1,811
2,048	933
4,096	427

Figure 15: Screenshot of Apache Solr using the IsiZulu stemmer to analyze indexed documents

3.7 SECOND ITERATION: MORPHOLOGICAL PARSER DEVELOPMENT

3.7.1 Introduction

This section describes the development and implementation of a morphological parser for the IsiZulu language. The morphological parser has been developed to investigate how a language's morphology can aid in the extraction of a root from a particular word. At a later stage, the morphological parser will aid in answering research question that will assess the extent to which the morphology of a language affects the precision of the results during the information retrieval process.

3.7.2 Analysis of IsiZulu morphology

In the field of linguistics, the morphology of a language is the study of the word formation process in a language based on the parts of the language structure such as morphemes, affixes and other language phenomena that occur because of the word formation process. Morphological analysis allows the breakdown of a word into various components that would have been overlooked by light stemming algorithms (McEnery, 2001). Once the semantic structure is obtained from morphological analysis, the parser is then able to apply predefined computations to the word to extract the root word.

Currently, a complete computational morphological analyzer for IsiZulu does not exist and thus one had to be constructed from scratch. Therefore, a firm understanding of the word formation process was crucial to the development of an effective parser. Additionally, IsiZulu is one of the lesser-studied languages of the world, which presents several challenges to the development of a complete morphological analyzer (Pretorius & Bosch, 2003).

The scope of the morphological parser was primarily based on the affixes and word formation rules. In terms of the prefixal analysis, the scope focused on the noun classification and concordial systems. The noun classification system forms the basis of all prefixes and determines the types of concords that can be applied to a stem. These concords have different categories, which have different semantics that must be considered when analyzing a particular word. On analyzing the concordial system that is attached in Appendix B, it was noted that there are overlaps in morphemes that belong to several concord classes such as the subject and possessive concord systems.

For suffix analysis, the parser catered for nominal suffixes that include diminutive, feminine, augmentative, deverbative, and locative suffixes. Certain suffixes such as the locative suffix, have some specialized rules that must be followed to ensure that the correct word formation procedure is followed (Madondo & Muziwenhlanhla, 2000). Other language phenomena such as vowel elision, consonantalization, and palatalization were also included to reverse the morphological changes that occur when applying certain affixes to a word.

Consonantalization is the process when subject concords that consist of only the vowels u- and i- respectively change to semi-vowels w- and y-. An example of this is:

i- + -ala > yala (it refuses)
(the) *(refuse)*

Palatalization and vowel elision have been explained in section 3.6.2.

The morphological parser was developed using approximations that were derived from the concordial system and noun classification rules. Although some overlaps are present between concordial types, it was presumed that these would not significantly affect the parsing of a word. For example, the concord “ba” can be either a plural prefix, a subject concord or an object concord.

The morphological analysis algorithm is as follows:

```
#Suffix Matching Algorithm
      IF (word.length >= 4):
      #using longest match principle
            ITERATE over order list of suffixes:
                  IF word endsWith (suffixList[i]):
                        MARK suffix as longest
                        Matching endings++
      ELSE:

            Matching endings = 0;



FOR EACH word:
      Check the suffix match:
            IF (matching endings >= 1):
                  MARK longest suffix
                  Check if the suffix is a special case:
                        IF special suffix:
                              LOOKUP suffix category
                                    IF category == diminutive suffix:
                                          REVERSE palatalization
                                    ELSE IF category == locative suffix:
                                          REMOVE locative prefix
                                    ELSE IF category == feminine suffix:
                                          REVERSE vowel elation
                                    ELSE IF category == evaluative suffix:
                                          REMOVE ‘se’ suffix
                                          REMOVE ‘e’ preprefix
                        ELSE:
                              MARK simple-suffix category
            ELSE:
                  MARK word as non-suffix category
            REMOVE suffix from word
            RETURN word and suffix category
      Check the suffix category:
              If (word has a preprefix):
                        IF (consonant preprefix AND non-suffix category):
                              CHECK concord class:
                                    IF possessive concord:
                                          MARK as root
                                    ELSE:
                                          CHECK concord pattern
                                          MARK and REMOVE concord
                                          MARK word as root

                        ELSE IF (consonant preprefix AND simple-suffix
category):
                              CHECK concord class:
                                    MARK and REMOVE concord
                                    MARK word as root

                        ELSE IF (consonant preprefix AND special suffix
category):
                              CHECK formation rule for suffix:
                                    REVERSE word formation
                                    REMOVE prefix
                                    MARK word as root

                        ELSE IF (vowel prefix and special suffix category):
                              CHECK preprefix class:
                                    REVERSE word formation
                                    MATCH prefix pattern
                                    REMOVE prefix
                                    MARK word as root

                        ELSE:
                              #word has no preprefix
                              MARK word as root


      RETURN root;
```

Figure 16: Morphological analysis algorithm

Fig 16 describes the algorithm used in the morphological analysis of the word. The morphology of the word begins with the analysis of the suffix pattern. This is because certain suffix patterns have particular prefixes that are part of the formation rule. For example, the evaluative suffix "-

se” has the preprefix pattern “-e” and a prefix pattern that is of the subject concord class. Once the suffix category has been defined, it is used an argument to check which rules may have been applied in the formation of the word. The type of preprefix defines class of prefixes that can be matched. A prefix is usually a vowel and a concord class, for example, “umu-“ has the preprefix “u-“ and an object concord “-mu”.

The overlaps in the concord classes were resolved by using approximate pattern matching to detect different types of concords. For example, the possessive concord “zika” has the pattern CVCV where C is the consonant and V is the vowel. However, a future negative subject concord “zuku” also has CVCV pattern. Thus, the resolution of this clash would have to be based on the suffix and the preprefix of the word, which would determine the formation rule to be applied to the word.

3.7.3 Morphological Parser Design and Implementation

The development of the morphological parser focused on the word formation rules that were described by Pretorius and Laurette (2003). This presented a few challenges that needed to be considering when defining the scope of the morphological parser. The software that provided most of the analysis functionality could not be integrated with the Apache Solr platform. Therefore, a considerable amount of time had to be allocated to developing the process flow of the parser. Additionally, there is not an exhaustive list of all possible word formation rules. Thus, some of the words were deconstructed based on approximations of what possible classes the morphemes belong.

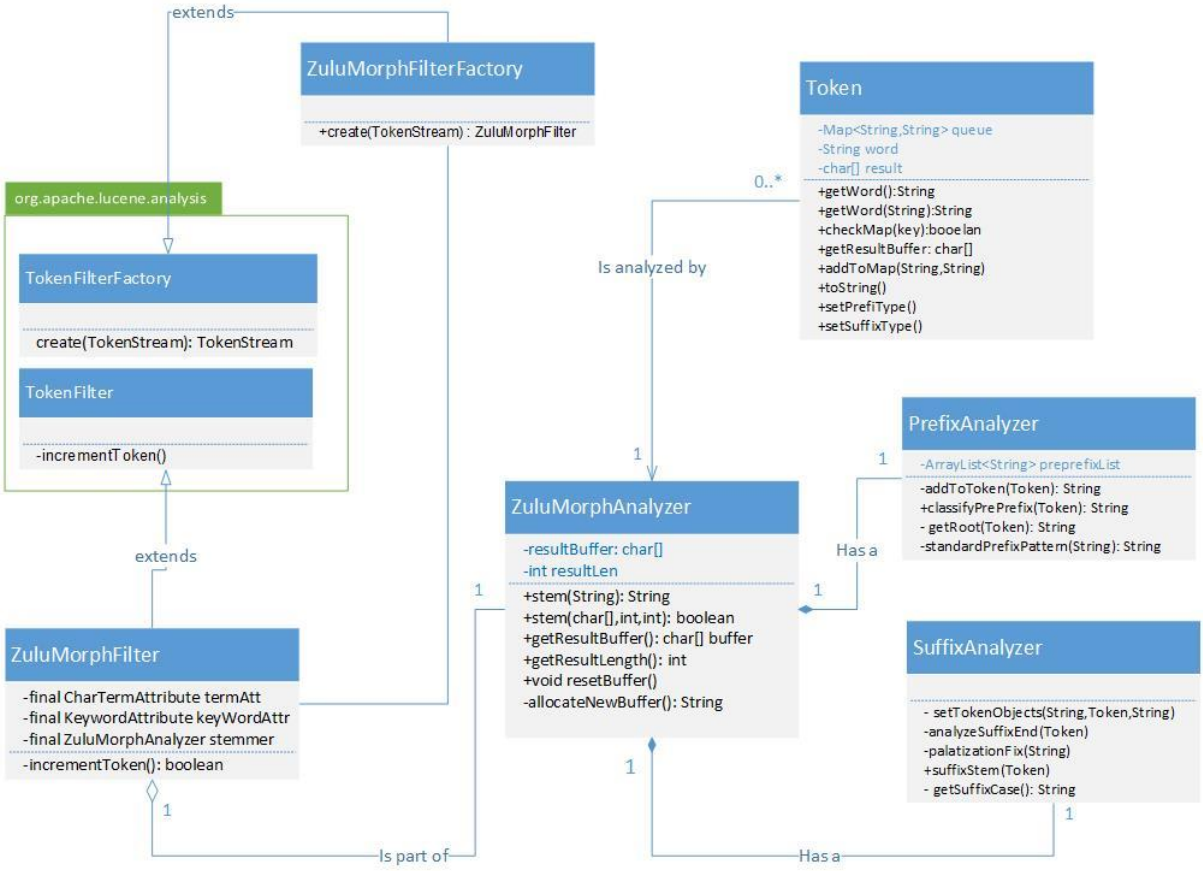


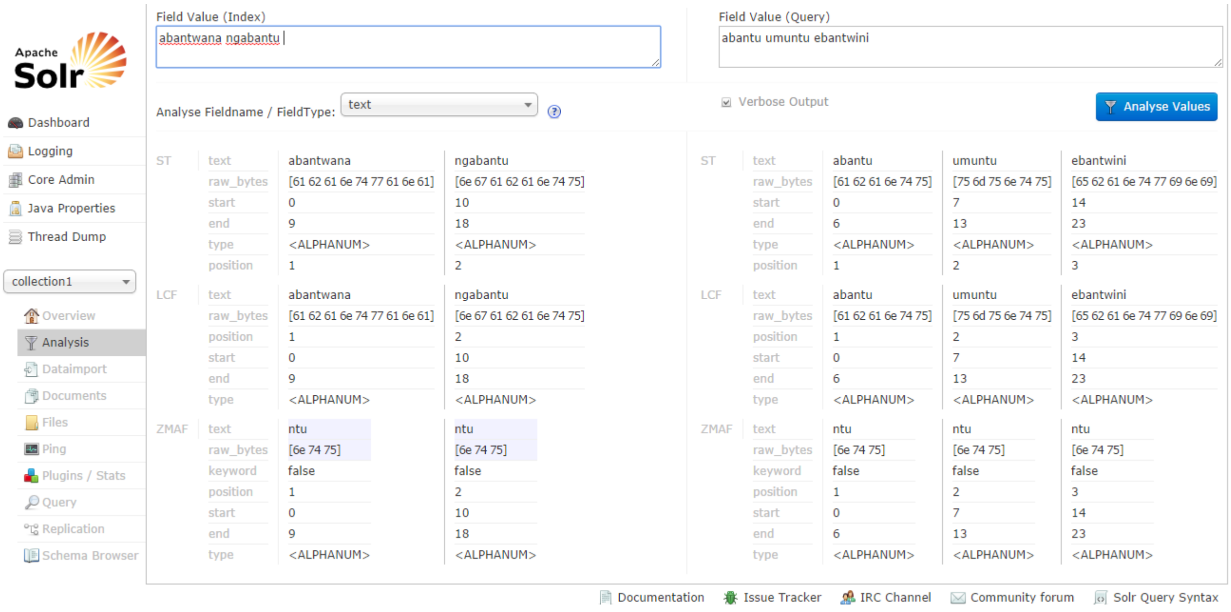
Figure 17: Morphological parser class diagram

Fig 17 shows the class diagram for the morphological parser that was designed using the Porter stemming algorithm. The package “org.apache.lucene.analysis” contained the token filtering packages that allow access to the indexed data using the TokenStream object.

The SuffixAnalyzer object contains a predefined set of suffixes that are evaluated using the longest match principle that has been discussed in section 4.3.2, which selects a suffix based on the longest matching suffix. The PrefixAnalyzer object uses regex patterns to match the possible pre-prefix values and the category of the concordial system to determine what appropriate action to apply to a particular word.

3.7.4 Testing and Evaluation

As described in the previous section 3.3.4, the morphological parser had to be packaged into a jar file with the appropriate dependent resources. Once the plugins had been loaded, they were tested using the built in analysis feature of Apache Solr. By navigating to the administration panel, one can access the analysis panel and load the particular field that has been customized.



Field Value (Index): abantwana ngabantu

Field Value (Query): abantu umuntu ebantwini

Analyse Fieldname / FieldType: text

☑ Verbose Output

ST	text	abantwana	ngabantu
	raw_bytes	[61 62 61 6e 74 77 61 6e 61]	[6e 67 61 62 61 6e 74 75]
	start	0	10
	end	9	18
	type	<ALPHANUM>	<ALPHANUM>
	position	1	2
LCF	text	abantwana	ngabantu
	raw_bytes	[61 62 61 6e 74 77 61 6e 61]	[6e 67 61 62 61 6e 74 75]
	position	1	2
	start	0	10
	end	9	18
	type	<ALPHANUM>	<ALPHANUM>
ZMAF	text	ntu	ntu
	raw_bytes	[6e 74 75]	[6e 74 75]
	keyword	false	false
	position	1	2
	start	0	10
	end	9	18
	type	<ALPHANUM>	<ALPHANUM>

ST	text	abantu	umuntu	ebantwini
	raw_bytes	[61 62 61 6e 74 75]	[75 6d 75 6e 74 75]	[65 62 61 6e 74 77 69 6e 69]
	start	0	7	14
	end	6	13	23
	type	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>
	position	1	2	3
LCF	text	abantu	umuntu	ebantwini
	raw_bytes	[61 62 61 6e 74 75]	[75 6d 75 6e 74 75]	[65 62 61 6e 74 77 69 6e 69]
	position	1	2	3
	start	0	7	14
	end	6	13	23
	type	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>
ZMAF	text	ntu	ntu	ntu
	raw_bytes	[6e 74 75]	[6e 74 75]	[6e 74 75]
	keyword	false	false	false
	position	1	2	3
	start	0	7	14
	end	6	13	23
	type	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>

Figure 18: Screenshot of Apache Solr with the morphological analyzer loaded

Fig 18 shows the testing process for the morphological parser. Various forms of the root “-ntu” which means person, were used as test cases. As seen in the image, the various inflected forms (abantu, ngabantu, ebantwini, umunt, abantwana) of the root where reduced to their base form. These inflected forms also provided sufficient coverage of the rules that were built into the parser.

3.8 SUMMARY

The initial iterations of development were focused on the development of a search interface. Once the search interface was completed, the system implementation process followed suit. Once the system was tested and the necessary adjustments made, experiments were designed to test some of the assumptions and investigate the research questions that had been established before the implementation of the search engine. The experimental process will be discussed in the next chapter

Chapter 4

4 EVALUATION

4.1 INTRODUCTION

Throughout the project, evaluation was done to offer improvements to the usability of the search engine and the performance of the developed algorithms. At the end of the final iteration, the developed system was tested in various scenarios and use cases to ensure its overall reliability.

A set of experiments were designed to evaluate the extent of how the project had tackled the research questions. Due to the split in the project, the report will focus on the following research questions.

1) Can a morphological parser result in a more accurate derivation of root words as opposed to a stemming algorithm?

2) Will the use of a morphological parser as opposed to a stemming algorithm result in an improved precision of results given a user's query?

In total, three experiments were designed to evaluate the project, two of which were focused on the research questions established at the onset of the project. The third was an evaluation of the search interface.

The experiments will be presented in this chapter. Each experiment will provide background, a hypothesis, the purpose, results and conclude with a discussion.

4.2 EXPERIMENT 1

Evaluating the accuracy of the morphological parser as compared to the stemming algorithm given a language corpus

4.2.1 Background

A morphological parser and a stemming algorithm have been developed to derive the root from of each word that has been indexed in the search engine. Thus improving the recall of the search engine due to the various canonical forms of a word that can be mapped to the root.

4.2.2 Purpose

A morphological parser and stemming algorithm to derive the root or stem of a word given some input. The experiment is aimed at comparing the accuracy of each of the algorithm's ability to reduce words to their root form.

4.2.3 Hypothesis

The use of a morphological parser that is sensitive to the morphemes such as suffixes, prefixes, and word formation rules of the IsiZulu language will result in the accurate extraction of the root word.

4.2.4 Experiment Requirements

- A morphological parser
- IsiZulu stemming algorithm
- Data set – Ukwabelana corpus[14]
- Data set – WordNet stemmed list

[14] Source: http://www.cs.bris.ac.uk/Research/MachineLearning/Morphology/resources.jsp

4.2.5 Methodology

The experiment was conducted using the Ukwabelana open source corpus. The corpus has a set of 10040 words that have been deconstructed into morphemes and roots. A set of words from the lexical database for the English language (WordNet[15]) were also included in the experiment as controls. Essentially, the set of English words were expected to either provide false positives or remain unaltered through the stemming process.

The tests were run using the Netbeans IDE and the JUnit testing framework. Two sets of tests were run to measure 100% accuracy and Off-by-one morpheme accuracy.

The methodology used in the experiment is as follows:

1) The list of input words was parsed using the morphological parser and stemming algorithm.
2) For each derived root:
    a. A comparison was made to the existing stem that is stored in the corpus.
    b. The stem was then determined to be either 100% accuracy or off by one morpheme class.
3) The process was also repeated for the control set of words.
4) The incorrectly stemmed words were sorted into categories based on observations of how the resultant word differed with the expected stem.

4.2.6 Results

Table 4: The results from the morphological parser experiment

Data Set	100% Accuracy Tests	Off-by-one morpheme
Ukwabelana Corpus	4824	7731
WordNet	0	0

Table 5: The results from the stemming experiment

Data Set	100% Accuracy Tests	Off-by-one morpheme
Ukwabelana Corpus	4217	5892
WordNet	0	0

[15] Source: http://wordnet.princeton.edu/

4.2.7 Observations

4.2.7.1 Morphological Parser Results

In terms of the 100% accuracy tests, the morphological parser managed to stem 4824 words, which produced a 48% accuracy. In the off-by-at most one-morpheme tests, the morphological parser yielded a 77% accuracy, stemming 7731 words. The results are denoted in Table 4.

The incorrectly stemmed words, which form 23% of the Ukwabelana corpus, had very similar characteristics. In the first instance, it was found that the language corpus had several words that could not be reduced to a stem or verb. About half of these words, 11.5% of the corpus, included mostly pronouns such as absolute and personal pronouns and a class of words known as demonstratives. These words are usually formed by concatenating a particular concord and the roots "-dwa", "-nye", "-nke" and "-aya".

In about 3% of the corpus, the words did not have enough morphological information to provide clues for the formation rules could have been used in the formulation of the word. For example:

> The word "abafana" which means boys, has the root "umufo"
>
> When tagged by the morphological parser, it produces the following:
>
> > a- [NPrePref2] -ba [OC2] f [NStem] -ana [DimSuf]
>
> Therefore, the stem is "f" tagged as "[NStem]".

The combination of the diminutive suffix ([DimSuf]) and the root "f", is a legal formation rule but it does not give clues of the vowel elision that would have occurred in the joining of "o" from the root "umufo" and the "-a" from the suffix "-ana. The prefix of the word "abafana" is "aba-", which is further broken down into a noun pre-prefix "a-" tagged as "[NPrePref2]" and a prefix "ba-" which is an object concord, tagged in the 2nd class of the concordial system as "[OC2]". The prefix "aba" converts a word to its plural form and is usually remove. However, in the case of "abafana", the prefix is meant to be converted to its singular form "umu-". Thus, the stemming of the word "abafana" results in a complex process that has not been explained fully in the existing works done on the parsing of the IsiZulu language.

The remaining 8.5% of cases comprised of a set of words that had gone through several morphological processes based on their use in a sentence. For example:

> The word "abazukuthengisa" which means, "they will not sell it" has the root word "theng".
>
> The correct form of the tagged word is:
>
> > a[NegPre]ba[SC2]zuku[FutNeg]theng[VRoot]is[CauExt]a[VerbTerm]

(Pretorius & Bosch, 2003)

The word provides two new morpheme classes that are used in the future tense. As the root is a verb, it requires a future negative ([FutNeg]) and a causative extension ([CauExt]). In this case, a dictionary would be used to detect the word “theng” in the pre-processing step. Thus, using a different set of rules to decompose the word in to the particular morphemes (Pretorius & Bosch, 2003).

4.2.7.2 Stemming algorithm results

The results from the stemming algorithm are denoted in Table 5. The stemming algorithm correctly stemmed 4217 words, which formed about 42% of the Ukwabelana corpus. In the control group, no words were correctly stemmed. In the off-by-at-most one morpheme group, the stemming algorithm managed to correctly stem 5892 words, which is 59% of the corpus.

The stemmer had an advantage in that the only words that are stemmed have either a nominal suffix or a prefix in the noun classification system. Therefore, words that fall into the category of pronouns that form 11.5% (see section 4.2.7.1 for examples) of the corpus were not stemmed as a result. In most cases, the stemmer lost its accuracy due to under stemming words, which occurs when only the morphemes that can be detected as accurate are stemmed.

4.3 EXPERIMENT 2

Evaluating the effect on precision of using a morphological parser vs a stemming algorithm when a user submits a query to the search engine

4.3.1 Purpose

The second experiment was conducted to measure if the relevance of the results given a user’s query would increase by using a morphological parser in comparison to a stemming algorithm when indexing and querying results.

4.3.2 Hypothesis

A morphological parser, which is analyzes words based on its morphemes and word formation rules, will result in a higher precision of results as opposed to a stemming algorithm that is sensitive to noun classes and suffixes.

4.3.3 Experiment Requirements

- Two search platforms – one implementing the stemming algorithm and the other implementing the morphological parser.
- User population – a user population of 12 people were used in the experiment. They were selected on the basis or either having learned IsiZulu in high school or spoke it as a home language.
- Data set – a collection of IsiZulu documents that have been harvested from the World Wide Web.
- Feedback form –categorization of results as relevant or irrelevant based on the user's query.

4.3.4 Methodology

The methodology used in the experiment is as follows:

1) A participant was invited to the experiment based on being literate in the IsiZulu language. This included participants who did not speak the IsiZulu language but were able to demonstrate a sound grasp of the language.

2) The participants were required to sign a form that acknowledged their participation in the experiment and give consent for their results to be used in compiling the results of the experiment. The participants were also notified that their personal information would not be used in any part of the experiment.

3) The participant was given a scenario that would require the user to find n information that read as: "You have been trying to research about African cultures in terms of their way of life and religious practices. Please formulate 3 queries that you think will provide you with the information that you need."

4) The participant was then asked to select the best query from the 3 and submit it to the search engine.

5) For each query, the participant was asked to input the query into both search engine. This was a result of each search engine implementing a different preprocessing algorithm

6) The participant was then asked to rate the results are either relevant or not relevant for the first 50 results that were returned by the search engine.

7) The participants were then asked to reformulate the query and complete the process described in (5).

8) The results were collected and tabulated so that the appropriate observations and conclusions could be drawn from them.

4.3.5 Results

Table 6: Results of the relevancy experiments using the morphological parser and stemming algorithm

	Morphological Parser			Stemming Algorithm		
	Q1	Q2	MPAv	Q1	Q2	SAAv
Part001	0,2	0,3	0,25	0,14	0,1	0,12
Part002	0,08	0,12	0,1	0,06	0,02	0,04
Part003	0,04	0,06	0,05	0,08	0,04	0,06
Part004	0,2	0,16	0,18	0,14	0,12	0,13
Part005	0,14	0,04	0,09	0,08	0,1	0,09
Part006	0,26	0,1	0,18	0,18	0,2	0,19
Part007	0,14	0,2	0,17	0,1	0,16	0,13
Part008	0,22	0,18	0,2	0,08	0,14	0,11
Part009	0,12	0,1	0,11	0,08	0,12	0,1
Part010	0,06	0,01	0,035	0,1	0,06	0,08
Part011	0,08	0,12	0,1	0,02	0,04	0,03
Part012	0,22	0,16	0,19	0,18	0,1	0,14
	Mean Average		0,13792	Mean Average		0,10167

Table 6 refers to the Precision @ 50 results.

4.3.6 Observations

Initially, the experiment required users to rate to categorize the first 100 results. However, the users quickly became restless and issued complaints about the process taking too long to complete. This was because they had to run over the process twice, for each query that was submitted to the search engine. After consultation with the supervisor, the number was changed to 50. The precision of the search engine was then calculated based on the results that were categorized.

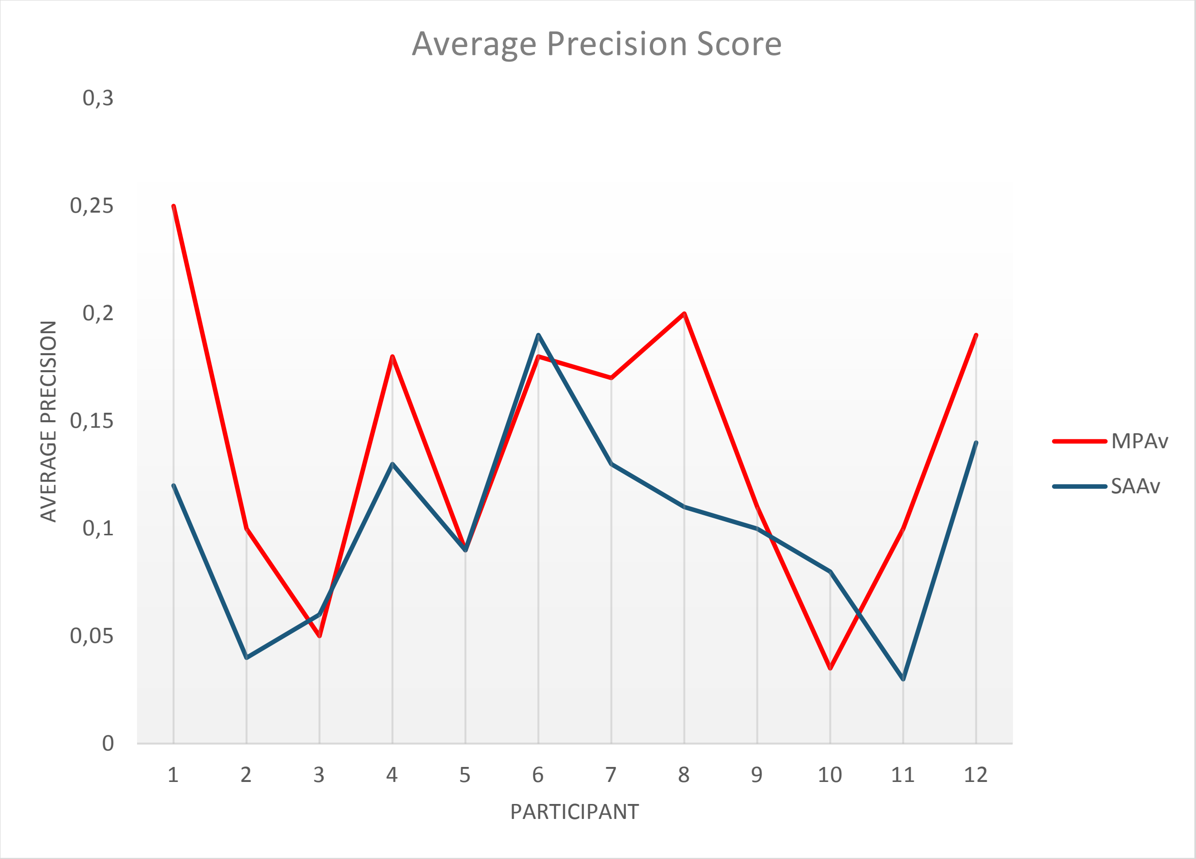


Figure 19: A graph of the MPAv and the SAAv

Fig 19 shows the average precision score determined by each participant during the experiment. The average precision of each participant was an average of two queries that were submitted into the search engine. Each query was aimed at trying to satisfy the information need that the participant had identified at the beginning of the experiment.

The legend labels MPAv and SAAv refer to Morphological Parser Average and Stemming Algorithm Average respectively. The mean precision of the morphological parser was found to be 0.138 as compared to the stemming algorithm with a precision score of 0.102. During the study, the queries that were submitted to the search engine were also recorded to determine if there was a correlation between the relevance of the results and the length of a query.

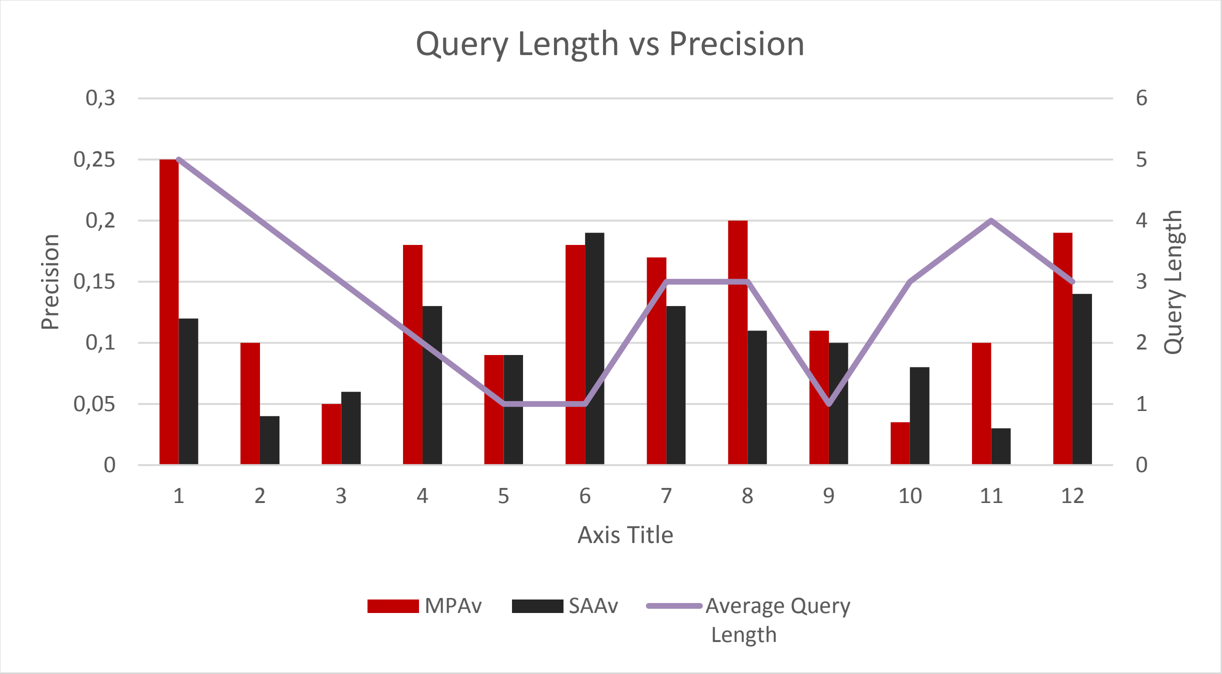


Figure 20: A graph that shows the effect of the query length on MPAv and SAAv

Fig 20 shows the differences between average precision of results for the morphological parser and the stemming algorithm based on the average length of the query. The length of the query has an effect on the differences in precision as seen in participants 1, 2, 11, and 8. As the length of the query decreases, the differences between the precision of the algorithms start to decrease and almost evens out in cases 5, 6, and 9. Thus, a certain trend can be derived from these results, i.e., as the word count per query decreases, the stemming algorithm and the morphological parser produce an almost equal precision score. Inversely, as the query length increases, average precision rate increases and there is a greater difference between the scores for the morphological parser and the stemming algorithm. The table of queries can be found in the Appendix C as Table C-2.

This can be attributed to the fact that, when searching for one word, a user is likely to search for a verb or noun. The word's prefix may fall in the noun classification system, which will result in a stemming procedure that is identical to both the stemming algorithm and the morphological parser. However, as the query's length increases, there is a higher possibility that the query can be a phrase or question. This will result in the increase of inflected forms in the query, which the morphological parser is most likely to deconstruct into the root form as it uses the language's morphology to extract it.

An increase in the query length will increase the number of terms that must be matched in a document to deem it relevant to the search. Thus, the search engine will be able to filter out larger number of given the increase in terms that must be matched to a given document (Belkin et al., 2003).

Therefore, the experiment proved the hypothesis true. Making use of a morphological parser resulted in a higher precision of results as compared to the stemming algorithm

4.4 EXPERIMENT 3

Evaluation of the search interface

4.4.1 Purpose

The experiment was carried out to assess whether the search interface provided an effective intermediary tool for the user to complete the task of searching for information.

4.4.2 Experiment Requirements

- Search interface
- User population (population used in experiment 2)
- Feedback form – the form was used in the rating of certain interface elements

4.4.3 Methodology

1) The experiment was run concurrently with Experiment 2
2) At the completion of Experiment 2, the users were asked to use an online form to rate their experience and the search interface
3) The results were then used to draw conclusions on the quality of the search interface.

4.4.4 Results

See Appendix D

4.4.5 Observations

In the evaluation of the user interface, the users were asked to rate their experience of the search engine to assess whether the search interface had managed to achieved the goals of usability; learnability, efficiency and satisfaction. Additionally, the participants were asked to list three positives and negatives that they encountered during the experiments.

The text was rated overall as visible and clear which allowed the results to be well read. Most participants said the system was easy to use and had no trouble entering a query. The user interface scored an average of 7.6 out of 10 for ease of use, which is a positive indication of the low level of learnability that is required to operate the search interface. Additionally, the user population noted a few positives of the visual aspects such as the use of color, clarity of text and minimalist design of the interface.

A few negatives were also given in the feedback such as the results that either did not make sense or were irrelevant given some queries. A participant said, “I searched for something in IsiZulu, but there is English here.”

Additionally, there was an expectation for the search engine to provide the functionality that is found on the more popular search engine, such as news, current information, and definition of words.

Chapter 5

5 Conclusion and Future Work

5.1 Conclusion

The two research questions that were investigated in the experiments were:

a) Can a morphological parser result in a more accurate derivation of root words as opposed to a stemming algorithm?

b) Will the use of a morphological parser as opposed to a stemming algorithm result in an improved precision of results given a user's query?
.

In the first evaluation, research question in (a) was positively answered, as the morphological parser produced about 48% accuracy in which the result contained the stem or root word as compared to the 42% produced by the stemming algorithm. Currently, the morphological parser detects the affixes based on patterns and predefined affixes whereas the stemming algorithm only uses a predefined list of affixes.

It is noteworthy that both algorithms would improve significantly by using a pre-processing step that detects the word given a language source such as dictionary. According to Willet (2006), the use of a dictionary in computational linguistics processing algorithms allow the algorithm to detect the word before the processing steps, which can increase the efficiency of these algorithms. The morphological parser is a more desirable algorithm in this case in that it sensitive to the word formation rules which can result in a more accurate derivation of a root word from an inflected form. Additionally, the morphological parser's accuracy can be increased by a further 11% by including a step to check whether a word is a pronoun before the reduction process begins. This can be done by either checking the length of the word or separating the pronouns into prefixes and suffixes and detecting the morphemes.

In the second evaluation, the research question denoted as (b) was positively answered. The use of a morphological parser in the indexing and querying of data resulted in a higher precision score as opposed to using a stemming algorithm. One key factor that contributed to this is that the morphological parser was able to reduce words to their root forms using their morphology as a guide. In the case of the stemming, a brute force stripping of suffixes and prefixes may have resulted in a phenomenon called understemming or overstemming. These processes usually result in the word being incorrectly stemmed due to the some of the morphemes either being incorrectly detected, or being totally omitted by the algorithm (Lovins, 1968).

Additionally, the quality of the results of the search engine has been found to affect the precision of the search engine. A few participants that noted that the results did not make sense, thus, resulting in the irrelevance of the returned results. An analysis of the results shows that some of the indexed documents are multilingual with a few sentences in IsiZulu. In these cases, the title usually is based on what the page is mainly about and thus provides no correlation with the IsiZulu text. Therefore, when these results are returned, the anchor text does not reflect any relevance with the query.

In terms of feasibility, the project has shown that it is possible to develop technologies for Bantu languages through the development of algorithms and the customization of existing technologies. Supporting technologies such as language corpora, dictionaries, and the development of a machine-readable lexicon for IsiZulu may provide interesting areas of research.

5.2 Future Work

In the development of the IsiZulu search engine, several issues that were encountered could form potential areas for future work.

5.2.1 IsiZulu Corpus

The corpus formed on of the most important aspects as it aided in the development of the morphological parser and the language model that was used in the harvesting of the IsiZulu text. An increase in the quality of the corpus would ensure that a richer language model could be developed. Additionally, the morphological parser will have a larger corpus to provide test cases and expose potential areas of weakness.

5.2.2 Morphological Parser

The morphological parser has a few area for improvement. The results from Experiment 1 (section 5.2) showed that the parser achieved a 48% accuracy. Thus, the parser can benefit from the development of an IsiZulu lexicon and a defined set of rules. Additionally, given a dictionary, the parser would also improve its ability to detect words in the preprocessing step.

BIBLIOGRAPHY

El-Khair, l. . A., 2007. Arabic Information Retreival. In: *Annual Review of Informatiomn Science and Technology.* Egypt: John Wiley and Sons , pp. 505 - 533.

Abdelila, A., Cowie, J. & Soliman, H. S., 2004. Arabic Information Retrieval Perspectives. *Arabic Language Processing.*

Chakrabarti, S., van der Berg, M. & Dom, B., 1999. Focused crawling: A new approach to topic-specific Web resource discovery. *Computer Networks,* Volume 31, pp. 1623-1640.

Cosijn, E. et al., 2007. RETRIEVING INFORMATION IN ONE LANGUAGE VIA ANOTHER.

Cosjin, E., Pirkola, A., Bothma, T. & Jarvelin, K., 2002. Information access in indigenous languages: a case study in Zulu. *South African Journal of Libraries and Information Science,* 68(2), p. 94.

Foundation, T. A. S., 2011. *Apache Solr.* [Online]
Available at: http://lucene.apache.org/solr/
[Accessed 26 October 2014].

Hurskainen, A., 1995. Information Retreival and Two-directional Word Formation. *Nordic Journal of African Studies,* 4(2), pp. 81-92.

Hurskainen, A., 1995. Information Retrieval and Two Directional Word Formation. *Nordic Journal of African Studies,* 4(2), pp. 81-92.

Hurskainen, A., 1999. Swahili Language Manager. *Nordic Journal of African Studies,* 8(2), pp. 139-157.

Manning, D. C., Prabhakar, R. & Schutze, H., 2008. *Boolean Retrieval.* 1 ed. s.l.:Cambridge University Press.

Manning, D. C., Prabhakar, R. & Schutze, H., 2008. *Boolean Retrieval.* Volume 1 ed. Cambridge: Cambridge University Press.

Mukami, L., 2013. *African Review.* [Online]
Available at: http://www.africareview.com/Special-Reports/Africas-endangered-languages/-/979182/2008252/-/12yos0s/-/index.html
[Accessed 14 May 2014].

Njiraine, D., Ocholla, D. & Onyancha, O. B., 2010. Indigenous knowledge research in Kenya and South Africa: an informetric study.. *Indilinga African Journal of Indigenous Knowledge Systems: Indigenous Knowledge and Poverty Eradication,* 9(2), pp. 194-210.

SA, S., 2012. *Census 2011.* [Online]
Available at: http://www.statssa.gov.za/census2011/default.asp
[Accessed 29 October 2014].

Spiegler, S., van der Spuy, A. & Flach, P., 2010. s.l., s.n.

Tune, K. T., Varma, V. & Pingali, P., 2007. *Evalutation of Oromo-English Cross Language Information Retreival.* Hyderabad, India, Cross Language Evaluation Forum.

Webopedia, 2014. *Webopedia.* [Online]
Available at: http://www.webopedia.com/TERM/W/Web_server.html
[Accessed 26 October 2014].

Yamana, H. & Chan, S.-B., 2010. *The method of improving the specific language focused crawler. In Proceedings of the CIPS-SIGHAN Joint Conference on Chinese Language Processing.* s.l., s.n.

Belkin, N.J., Kelly, D., Kim, G., Kim, J., Lee, H., Muresan, G., Tang, M., Yuan, X. et al. 2003. Query length in interactive information retrieval. *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM. 205.

Hearst, M. 2009. *Search user interfaces.* Cambridge University Press.

Hull, D.A. 1996. Stemming algorithms: A case study for detailed evaluation. *Jasis.* 47(1): 70-84.

Langville, A.N. & Meyer, C.D. 2011. *Google's PageRank and beyond: The science of search engine rankings.* Princeton University Press.

Lovins, J.B. 1968. *Development of a stemming algorithm.* MIT Information Processing Group, Electronic Systems Laboratory.

Madondo, L.M. & Muziwenhlanhla, S. 2000. Some aspects of evaluative morphology in Zulu.

McEnery, T. 2001. *Corpus linguistics: An introduction.* Edinburgh University Press.

Medelyan, O., Schulz, S., Paetzold, J., Poprat, M. & Markó, K. 2006. Language specific and topic focused web crawling. *Proceedings of the Language Resources Conference LREC.*

Nwesri, A.F., Tahaghoghi, S.M. & Scholer, F. 2007. Answering English Queries in Automatically Transcribed Arabic Speech. *Computer and Information Science, 2007. ICIS 2007. 6th IEEE/ACIS International Conference on.* IEEE. 11.

Pretorius, L. & Bosch, S.E. 2003. Finite-state computational morphology: An analyzer prototype for Zulu. *Machine Translation.* 18(3): 195-216.

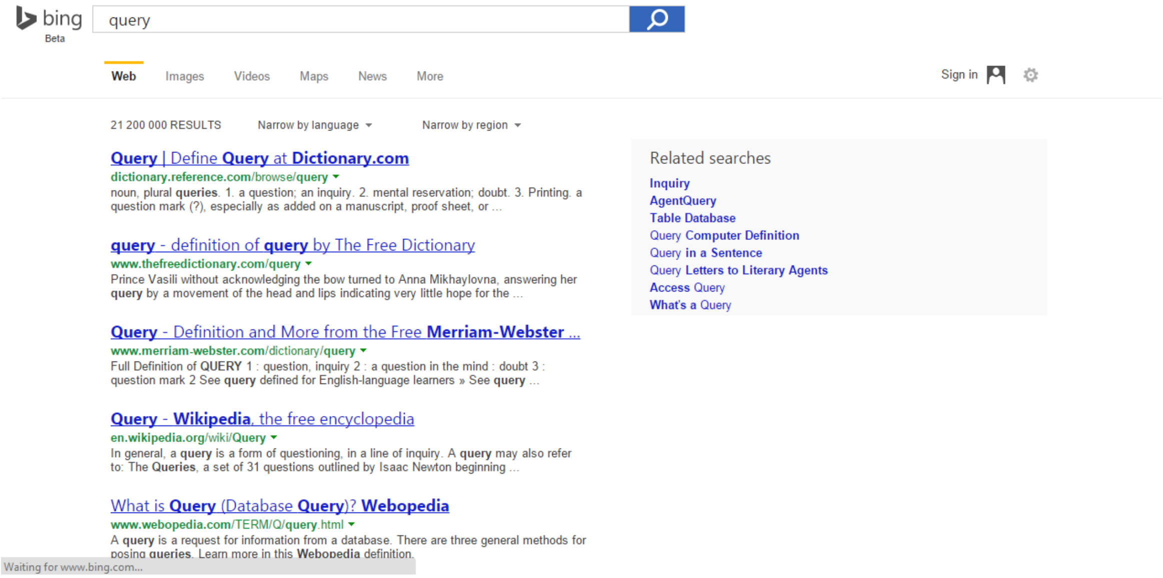
Spiegler, S., Van Der Spuy, A. & Flach, P.A. 2010. Ukwabelana: an open-source morphological Zulu corpus. *Proceedings of the 23rd International Conference on Computational Linguistics.* Association for Computational Linguistics. 1020.

Tune, K.K., Varma, V. & Pingali, P. 2007. Evaluation of Oromo-English Cross-Language Information Retrieval. *Language Technologies Research Centre IIIT, Hyderabad India.*

Willett, P. 2006. The Porter stemming algorithm: then and now. *Program: Electronic Library and Information Systems.* 40(3): 219-223.

Appendices

APPENDIX A: SEARCH RESULTS SCREENSHOTS



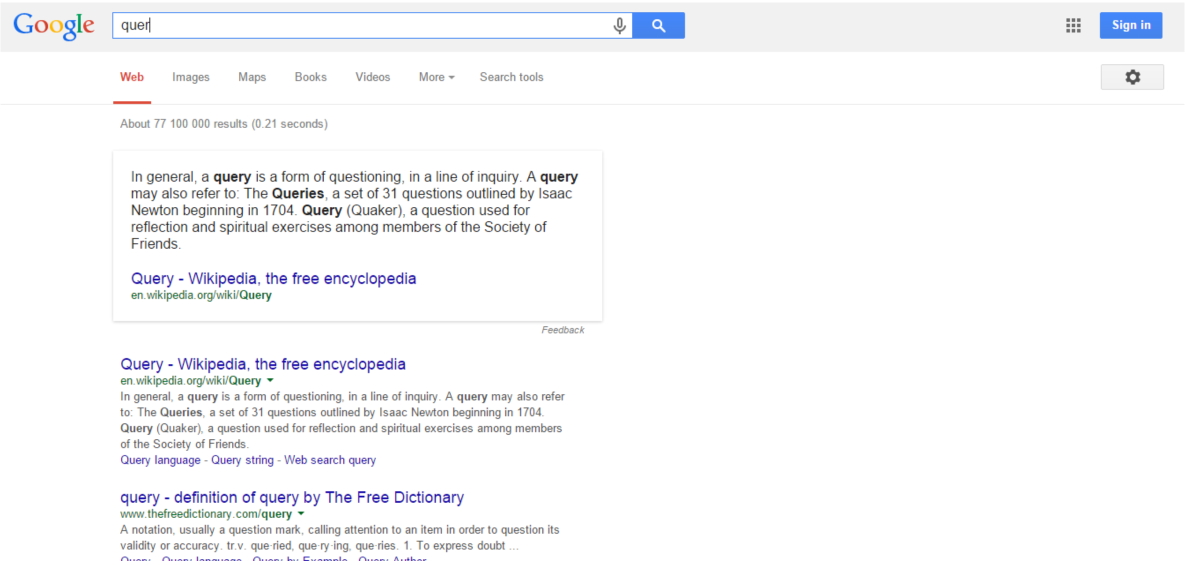
Query | Define Query at Dictionary.com
dictionary.reference.com/browse/query
noun, plural **queries**. 1. a question; an inquiry. 2. mental reservation; doubt. 3. Printing. a question mark (?), especially as added on a manuscript, proof sheet, or ...

query - definition of **query** by The Free Dictionary
www.thefreedictionary.com/query
Prince Vasili without acknowledging the bow turned to Anna Mikhaylovna, answering her **query** by a movement of the head and lips indicating very little hope for the ...

Query - Definition and More from the Free **Merriam-Webster** ...
www.merriam-webster.com/dictionary/query
Full Definition of QUERY 1 : question, inquiry 2 : a question in the mind : doubt 3 : question mark 2 See **query** defined for English-language learners » See **query** ...

Query - **Wikipedia**, the free encyclopedia
en.wikipedia.org/wiki/Query
In general, a **query** is a form of questioning, in a line of inquiry. A **query** may also refer to: The **Queries**, a set of 31 questions outlined by Isaac Newton beginning ...

What is **Query** (Database **Query**)? **Webopedia**
www.webopedia.com/TERM/Q/query.html
A **query** is a request for information from a database. There are three general methods for posing **queries**. Learn more in this **Webopedia** definition.



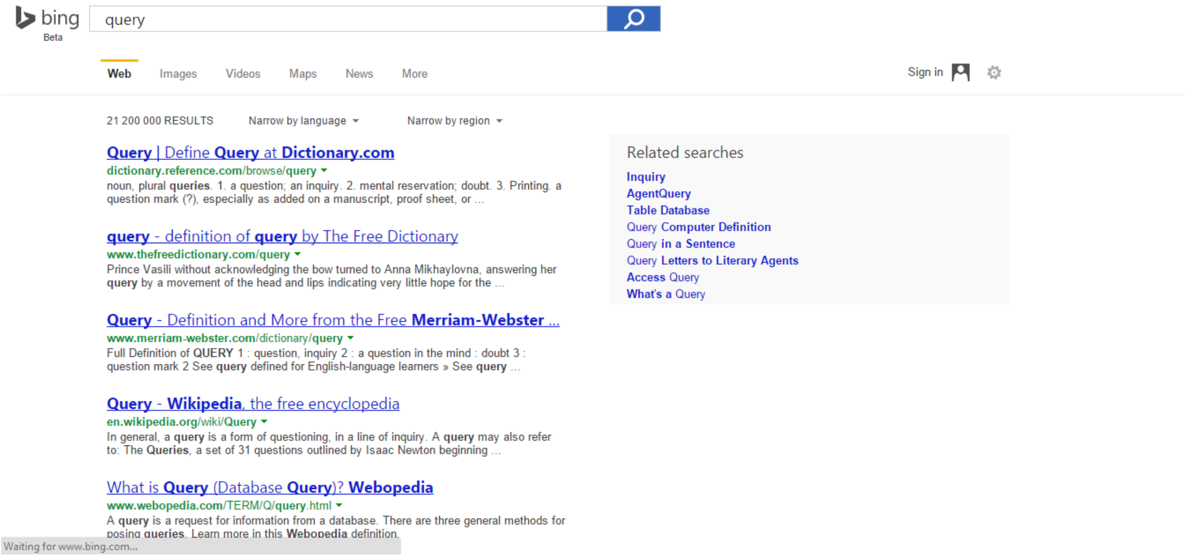
About 77 100 000 results (0.21 seconds)

In general, a **query** is a form of questioning, in a line of inquiry. A **query** may also refer to: The **Queries**, a set of 31 questions outlined by Isaac Newton beginning in 1704. **Query** (Quaker), a question used for reflection and spiritual exercises among members of the Society of Friends.

Query - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Query

Query - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Query
In general, a **query** is a form of questioning, in a line of inquiry. A **query** may also refer to: The **Queries**, a set of 31 questions outlined by Isaac Newton beginning in 1704. **Query** (Quaker), a question used for reflection and spiritual exercises among members of the Society of Friends.
Query language - Query string - Web search query

query - definition of query by The Free Dictionary
www.thefreedictionary.com/query
A notation, usually a question mark, calling attention to an item in order to question its validity or accuracy. tr.v. que·ried, que·ry·ing, que·ries. 1. To express doubt ...



21 200 000 RESULTS   Narrow by language   Narrow by region

Query | Define Query at Dictionary.com
dictionary.reference.com/browse/query
noun, plural **queries**. 1. a question; an inquiry. 2. mental reservation; doubt. 3. Printing. a question mark (?), especially as added on a manuscript, proof sheet, or ...

query - definition of query by The Free Dictionary
www.thefreedictionary.com/query
Prince Vasili without acknowledging the bow turned to Anna Mikhaylovna, answering her **query** by a movement of the head and lips indicating very little hope for the ...

Query - Definition and More from the Free Merriam-Webster ...
www.merriam-webster.com/dictionary/query
Full Definition of **QUERY** 1 : question, inquiry 2 : a question in the mind : doubt 3 : question mark 2 See **query** defined for English-language learners » See **query** ...

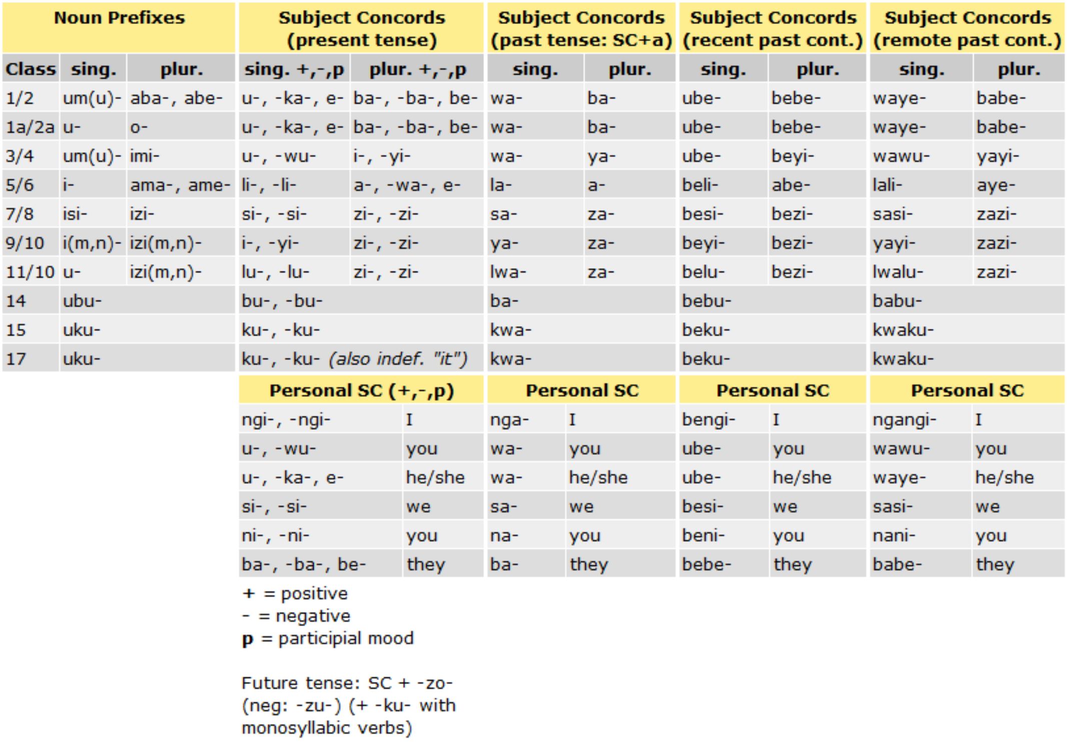
Query - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Query
In general, a **query** is a form of questioning, in a line of inquiry. A **query** may also refer to: The **Queries**, a set of 31 questions outlined by Isaac Newton beginning ...

What is Query (Database Query)? Webopedia
www.webopedia.com/TERM/Q/query.html
A **query** is a request for information from a database. There are three general methods for posing **queries**. Learn more in this **Webopedia** definition.

Related searches
Inquiry
AgentQuery
Table Database
Query Computer Definition
Query in a Sentence
Query Letters to Literary Agents
Access Query
What's a Query

Waiting for www.bing.com...

APPENDIX B: NOUN CLASSIFICATION AND CONCORDIAL SYSTEM

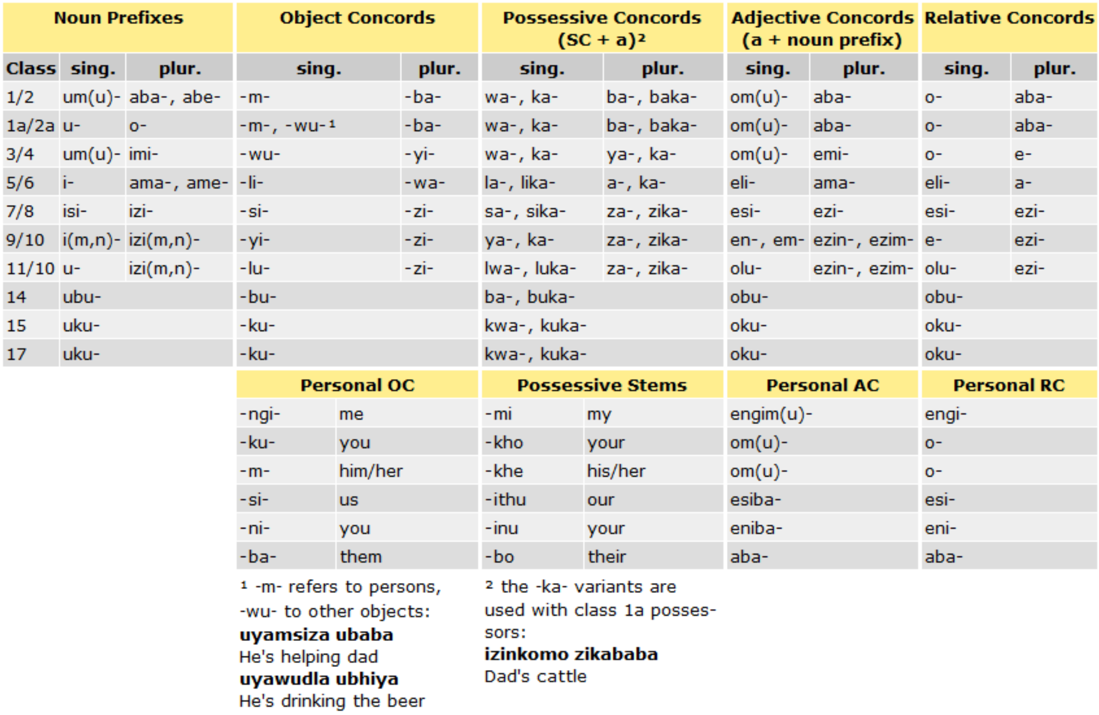


Noun Prefixes			Subject Concords (present tense)		Subject Concords (past tense: SC+a)		Subject Concords (recent past cont.)		Subject Concords (remote past cont.)	
Class	**sing.**	**plur.**	**sing. +,-,p**	**plur. +,-,p**	**sing.**	**plur.**	**sing.**	**plur.**	**sing.**	**plur.**
1/2	um(u)-	aba-, abe-	u-, -ka-, e-	ba-, -ba-, be-	wa-	ba-	ube-	bebe-	waye-	babe-
1a/2a	u-	o-	u-, -ka-, e-	ba-, -ba-, be-	wa-	ba-	ube-	bebe-	waye-	babe-
3/4	um(u)-	imi-	u-, -wu-	i-, -yi-	wa-	ya-	ube-	beyi-	wawu-	yayi-
5/6	i-	ama-, ame-	li-, -li-	a-, -wa-, e-	la-	a-	beli-	abe-	lali-	aye-
7/8	isi-	izi-	si-, -si-	zi-, -zi-	sa-	za-	besi-	bezi-	sasi-	zazi-
9/10	i(m,n)-	izi(m,n)-	i-, -yi-	zi-, -zi-	ya-	za-	beyi-	bezi-	yayi-	zazi-
11/10	u-	izi(m,n)-	lu-, -lu-	zi-, -zi-	lwa-	za-	belu-	bezi-	lwalu-	zazi-
14	ubu-		bu-, -bu-		ba-		bebu-		babu-	
15	uku-		ku-, -ku-		kwa-		beku-		kwaku-	
17	uku-		ku-, -ku- *(also indef. "it")*		kwa-		beku-		kwaku-	

Personal SC (+,-,p)		Personal SC		Personal SC		Personal SC	
ngi-, -ngi-	I	nga-	I	bengi-	I	ngangi-	I
u-, -wu-	you	wa-	you	ube-	you	wawu-	you
u-, -ka-, e-	he/she	wa-	he/she	ube-	he/she	waye-	he/she
si-, -si-	we	sa-	we	besi-	we	sasi-	we
ni-, -ni-	you	na-	you	beni-	you	nani-	you
ba-, -ba-, be-	they	ba-	they	bebe-	they	babe-	they

+ = positive
- = negative
p = participial mood

Future tense: SC + -zo- (neg: -zu-) (+ -ku- with monosyllabic verbs)

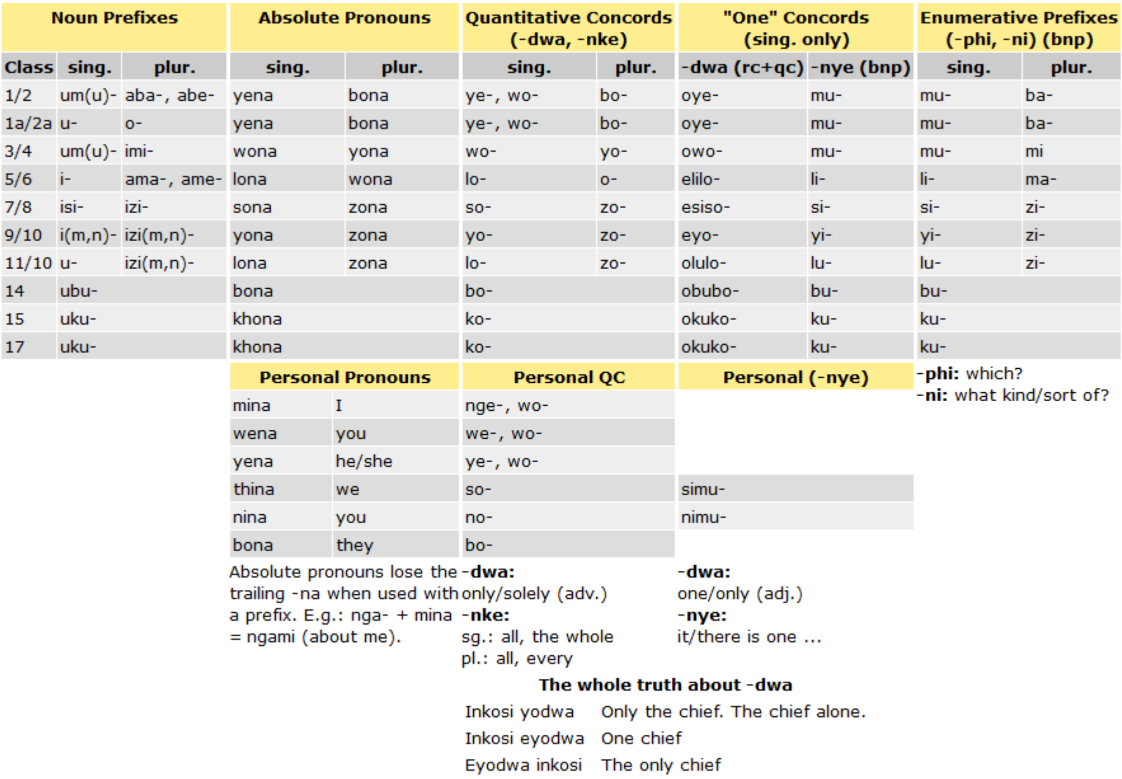


Noun Prefixes			Object Concords		Possessive Concords (SC + a)[2]		Adjective Concords (a + noun prefix)		Relative Concords	
Class	**sing.**	**plur.**	**sing.**	**plur.**	**sing.**	**plur.**	**sing.**	**plur.**	**sing.**	**plur.**
1/2	um(u)-	aba-, abe-	-m-	-ba-	wa-, ka-	ba-, baka-	om(u)-	aba-	o-	aba-
1a/2a	u-	o-	-m-, -wu-[1]	-ba-	wa-, ka-	ba-, baka-	om(u)-	aba-	o-	aba-
3/4	um(u)-	imi-	-wu-	-yi-	wa-, ka-	ya-, ka-	om(u)-	emi-	o-	e-
5/6	i-	ama-, ame-	-li-	-wa-	la-, lika-	a-, ka-	eli-	ama-	eli-	a-
7/8	isi-	izi-	-si-	-zi-	sa-, sika-	za-, zika-	esi-	ezi-	esi-	ezi-
9/10	i(m,n)-	izi(m,n)-	-yi-	-zi-	ya-, ka-	za-, zika-	en-, em-	ezin-, ezim-	e-	ezi-
11/10	u-	izi(m,n)-	-lu-	-zi-	lwa-, luka-	za-, zika-	olu-	ezin-, ezim-	olu-	ezi-
14	ubu-		-bu-		ba-, buka-		obu-		obu-	
15	uku-		-ku-		kwa-, kuka-		oku-		oku-	
17	uku-		-ku-		kwa-, kuka-		oku-		oku-	

Personal OC		Possessive Stems		Personal AC	Personal RC
-ngi-	me	-mi	my	engim(u)-	engi-
-ku-	you	-kho	your	om(u)-	o-
-m-	him/her	-khe	his/her	om(u)-	o-
-si-	us	-ithu	our	esiba-	esi-
-ni-	you	-inu	your	eniba-	eni-
-ba-	them	-bo	their	aba-	aba-

[1] -m- refers to persons, -wu- to other objects:
uyamsiza ubaba
He's helping dad
uyawudla ubhiya
He's drinking the beer

[2] the -ka- variants are used with class 1a possessors:
izinkomo zikababa
Dad's cattle



Noun Prefixes			Absolute Pronouns		Quantitative Concords (-dwa, -nke)		"One" Concords (sing. only)		Enumerative Prefixes (-phi, -ni) (bnp)	
Class	sing.	plur.	sing.	plur.	sing.	plur.	-dwa (rc+qc)	-nye (bnp)	sing.	plur.
1/2	um(u)-	aba-, abe-	yena	bona	ye-, wo-	bo-	oye-	mu-	mu-	ba-
1a/2a	u-	o-	yena	bona	ye-, wo-	bo-	oye-	mu-	mu-	ba-
3/4	um(u)-	imi-	wona	yona	wo-	yo-	owo-	mu-	mu-	mi
5/6	i-	ama-, ame-	lona	wona	lo-	o-	elilo-	li-	li-	ma-
7/8	isi-	izi-	sona	zona	so-	zo-	esiso-	si-	si-	zi-
9/10	i(m,n)-	izi(m,n)-	yona	zona	yo-	zo-	eyo-	yi-	yi-	zi-
11/10	u-	izi(m,n)-	lona	zona	lo-	zo-	olulo-	lu-	lu-	zi-
14	ubu-		bona		bo-		obubo-	bu-	bu-	
15	uku-		khona		ko-		okuko-	ku-	ku-	
17	uku-		khona		ko-		okuko-	ku-	ku-	

Personal Pronouns		Personal QC	Personal (-nye)
mina	I	nge-, wo-	
wena	you	we-, wo-	
yena	he/she	ye-, wo-	
thina	we	so-	simu-
nina	you	no-	nimu-
bona	they	bo-	

-**phi:** which?
-**ni:** what kind/sort of?

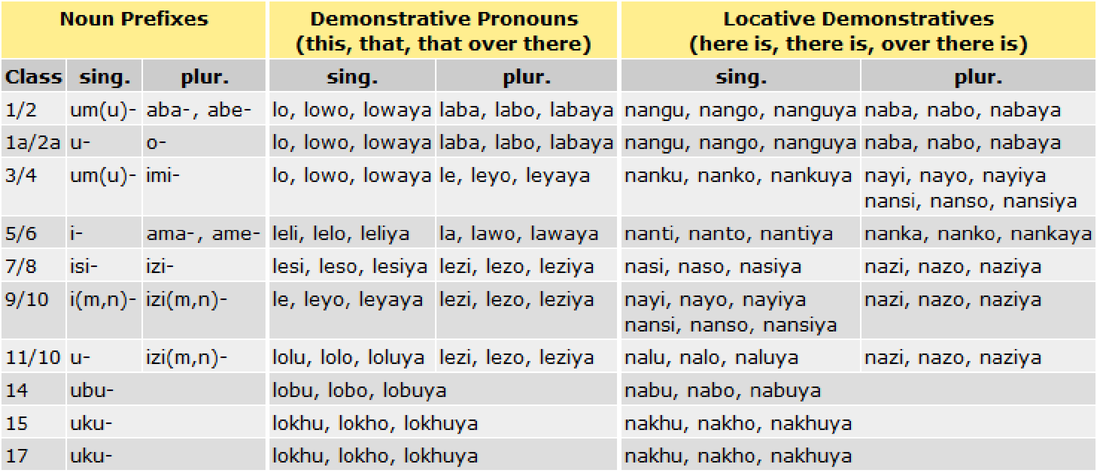
Absolute pronouns lose the trailing -na when used with a prefix. E.g.: nga- + mina = ngami (about me).

-**dwa:**
only/solely (adv.)
-**nke:**
sg.: all, the whole
pl.: all, every

-**dwa:**
one/only (adj.)
-**nye:**
it/there is one ...

The whole truth about -dwa

Inkosi yodwa	Only the chief. The chief alone.
Inkosi eyodwa	One chief
Eyodwa inkosi	The only chief



Noun Prefixes			Demonstrative Pronouns (this, that, that over there)		Locative Demonstratives (here is, there is, over there is)	
Class	sing.	plur.	sing.	plur.	sing.	plur.
1/2	um(u)-	aba-, abe-	lo, lowo, lowaya	laba, labo, labaya	nangu, nango, nanguya	naba, nabo, nabaya
1a/2a	u-	o-	lo, lowo, lowaya	laba, labo, labaya	nangu, nango, nanguya	naba, nabo, nabaya
3/4	um(u)-	imi-	lo, lowo, lowaya	le, leyo, leyaya	nanku, nanko, nankuya	nayi, nayo, nayiya nansi, nanso, nansiya
5/6	i-	ama-, ame-	leli, lelo, leliya	la, lawo, lawaya	nanti, nanto, nantiya	nanka, nanko, nankaya
7/8	isi-	izi-	lesi, leso, lesiya	lezi, lezo, leziya	nasi, naso, nasiya	nazi, nazo, naziya
9/10	i(m,n)-	izi(m,n)-	le, leyo, leyaya	lezi, lezo, leziya	nayi, nayo, nayiya nansi, nanso, nansiya	nazi, nazo, naziya
11/10	u-	izi(m,n)-	lolu, lolo, loluya	lezi, lezo, leziya	nalu, nalo, naluya	nazi, nazo, naziya
14	ubu-		lobu, lobo, lobuya		nabu, nabo, nabuya	
15	uku-		lokhu, lokho, lokhuya		nakhu, nakho, nakhuya	
17	uku-		lokhu, lokho, lokhuya		nakhu, nakho, nakhuya	

APPENDIX C: EXPERIMENT ADDITIONAL RESULTS

Table C- 1: A table showing the average query length, Morphological parser average and Stemming algorithm average

Participant No:	Average Query Length	MPAv	SAAv
Part001	5	0,25	0,12
Part002	4	0,1	0,04
Part003	3	0,05	0,06
Part004	2	0,18	0,13
Part005	1	0,09	0,09
Part006	1	0,18	0,19
Part007	3	0,17	0,13
Part008	3	0,2	0,11
Part009	1	0,11	0,1
Part010	3	0,035	0,08
Part011	4	0,1	0,03
Part012	3	0,19	0,14

Table C- 2: The queries that the participants in experiment 2 entered in the search engine

Participant	**Query 1**	**Query 2**	**Average Length**
Part001	amacici awesifazana ayegqokwa ngenkhathi zakudala	impahla ethandwa ngabantu besifazana abancane	5
Part002	isizwe ngobukhulu eMzansi Afrika	izizwe ezaqala ngesikhathi sikaShaka	4
Part003	amabhebesi aseMzansi Afrika	izilwane zaseMzansi Afrika	3
Part004	umkhumbi kaNoah	izehlakalo zikaNoah	2
Part005	amakhansela	uhulumeni	1
Part006	Izikole	imifundo	1
Part007	amazwi amahle okucula	amaculo okuqala esontweni	3
Part008	indlela engafika ekhaya	ikhaya lase zulwini	3
Part009	uShaka	abeZulu	1
Part010	imithandazo eyehlukile ebhayibhelini	imithandazo engekhoyo emfundweni	3
Part011	ukwehlukana kwabantwana abancane	umehluko phakathi kwabantu abancane nabadala	4
Part012	indawo yokuhlalala	izindlu ezanikezwa abantu abamnyana	3

APPENDIX D: EXPERIMENT 3 RESULTS

Participant No	1) Difficult of using the system	2) Was the text in a clear and readable manner?	3) Where the features of the interface easy to use?	4) Submitting a query to the search engine is straight forward	5) Were the feedback messages useful when completing tasks?	6) How does this search engine compare to the more popular ones such as Google?	7) Did you make use of the "English" option when entering a query in the search engine	How likely are you to recommend this search engine to other people
Part001	9	Very clear	5	9	7	7	No	9
Part002	7	Very clear	7	10	9	9	Yes	9
Part003	9	Very clear	9	9	4	8	Yes	8
Part004	6	Very clear	7	7	6	8	No	6
Part005	7	Very clear	8	8	1	6	No	5
Part006	9	Very clear	7	5	5	7	No	4
Part007	6	Visible, but not very clear	6	7	2	3	No	1
Part008	7	Very clear	6	8	3	8	No	4
Part009	8	Very clear	5	9	9	5	No	6
Part010	7	Visible, but not very clear	8	10	7	7	No	5
Part011	9	Visible, but not very clear	6	7	8	9	No	8
Part012	7	Very clear	4	6	5	5	Yes	6
	7,583333333		6,5	7,916667	5,5	6,833333		5,916666667

Table D- 1: The results from experiment 3