HONOURS PROJECT REPORT

DEPARTMENT OF COMPUTER SCIENCE

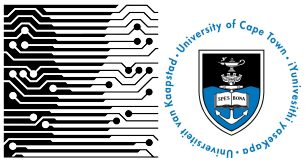
UNIVERSITY OF CAPE TOWN

Afri-Web

Author:
Katlego Moukangwe
Partner:
Nkosana Malumba

Supervisor:
Dr. Hussein Suleman

October 29, 2014



	Category	Min	Max	Chosen
1	Requirement Analysis and Design	0	20	0
2	Theoretical Analysis	0	20	0
3	Experiment Design and Execution	0	20	10
4	System Development and Implementation	0	15	15
5	Results, Findings and Conclusion	10	20	15
6	Aim Formulation and Background Work	15	20	20
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

Zulu is one of South Africa's indigenous languages understood by over 16 million speakers. There is a large and increasing amount of Zulu documents on the Internet, however there is no easy and effective means to get access to them. This creates a need for a Zulu information retrieval system. This paper outlines procedures followed to develop a language identification and crawling system for Zulu documents. The project managed to obtain a collection of over 60,000 Zulu documents and developed a Zulu identification system capable of identifying Zulu strings with an accuracy of 98.4% . The documents collected were subsequently indexed and a search platform provided for efficient querying.

Acknowledgements

I would first and foremost like to thank my supervisor, Hussein Suleman, for his commitment to the project. I would like to thank him for sharing his knowledge and providing guidance. I would also like to thank my project Partner Nkosana for being all-round helpful. Lastly, I would like to thank the laws of Physics for setting the parameters right, enabling me to finish on time.

CONTENTS

List of Figures

List of Tables

1 INTRODUCTION

Zulu is one of South Africa's largest indigenous languages, spoken by about 10-11 million native speakers and understood by more than 16 million speakers(Ethnologue, 2009). Multiple attempts have been made in order to preserve South African languages and culture, however not a lot of progress has been made. There is currently very little research and developments made on South African language information retrieval. One of the pioneering works performed in South African language information retrieval, was carried out by Erica Cosijn, Ari Pirkola, Theo Bothma and Kalervo Jrvelin from the University of Pretoria in a paper entitled Information access in indigenous languages: a case study in Zulu (Cosijn et al., 2002).

Information access in indigenous languages: a case study in Zulu ,focused on the digital accessibility of indigenous knowledge with Zulu being the main language(Cosijn et al., 2002). Cross-Lingual Information Retrieval (CLIR) and metadata were discussed as possible means of facilitating access(Cosijn et al., 2002). Popular CLIR approaches and their resource requirements were analyzed. The critique concluded by showing that there are multiple problems and difficulties encountered when implementing CLIR for African languages(Cosijn et al., 2002). Ambiguity, incorrect stemming, paraphrasing in translations, untranslatability and mismatching made it difficult to search full text on Indigenous Knowledge databases(Cosijn et al., 2002). These difficulties presented unique research opportunities for research in CLIR for African Languages.

1.1 AFRIWEB

Zulu has more than 10 million speakers, however digital documents in Zulu are very rare; as of October 2014, there are 682 Zulu-Wikipedia documents(ZuluWiki, 2014). However, the number of such documents has the potential to increase as more speakers of the language become digitally literate, more government documents are produced, more documents on the Web are translated and more books are produced for teaching and learning and popular consumption. AfriWeb is a simple information retrieval system for Zulu. The AfriWeb project entailed the gathering of Zulu documents on the Internet using a focused crawler, indexing and providing a search platform for crawled documents. It also provided a Cross-Lingual query option where users can submit queries and then have the response translated for them via a Google's translation API (APi, 2014).

SYSTEM REQUIREMENTS

The final software system is able to perform the tasks outlined below.

- Index documents containing Zulu text.
- Search Zulu text from indexed documents.
- Recognize Zulu text using a language model.
- Crawl for Zulu documents on the Internet.
- Perform morphological analysis on Zulu text or queries.
- Submit queries and translate returned documents to English.

1.2 Motivation and Research questions

There is a risk that South-African languages and cultures will never be digitally preserved because of the increasing number of South Africans adopting English as the sole means of digital communication. AfriWeb attempts to bridge the digital divide by creating a search platform that contains Zulu documents only. This will promote the creation of Zulu websites and documents on the Internet. This will also contribute to South African language information retrieval research. Accurate Cross-Lingual searching will also provide access for non-Zulu speakers to Zulu literature.

Problem Statement

The main aim of the project is to investigate the feasibility of harvesting Zulu Web documents, indexing and subsequently searching the harvested documents. This can be achieved by being able to automate the harvesting, indexing, searching and retrieval processes into one coherent process. One of the research tasks is to determine if it is possible to use a focused crawler to find Zulu documents on the Web. One of the major problems is recognizing Zulu text. There is not a well developed corpus for Zulu at the moment. Working with poor corpora for language identification is one of the major topics of research in information retrieval. Solving the problem of Zulu language identification enables one to find the content to index when developing a Zulu information retrieval system.

In order to determine the most effective way to search through Zulu data, different searching techniques were investigated. Stemming and removing stop-words are the main techniques that were researched. Being able to effectively and efficiently search Zulu text is one step closer to eliminating social and digital divides amongst South Africans. Is it possible to bridge the digital divide through African language information retrieval? Can African culture and language be preserved through African language search engines? A morphological parser was used to perform stemming and to find stop-words. The system was made specifically for people who are looking for information written/stored in Zulu. The users were primarily people who understand Zulu and are searching for information currently stored in Zulu. People who do not understand Zulu, however need access to the indigenous knowledge were aided by a translation service.

Research Questions

Language identification is central to a lot of pre-processing applications in information retrieval. Language identification is an essential pre-processing step for language processing techniques such as stemming, machine-translation, et cetera. **Is it possible to develop a language identification system capable of identifying Zulu text or strings?** As mentioned in the problem statement, there is not a lot of good Zulu corpora available on the Internet. The language identification system should perform fairly accurately in order to be used in other research developments that require language identification such as part-of-speech tagging, machine-translation, stemming, focused crawling,et cetera.

Is it possible to develop a focused crawler capable of identitying and harvesting Zulu documents on the Internet? Collecting Zulu documents is one step closer to building a Zulu search engine. A Zulu search engine will provide an easy way for locating Zulu literature, news or other forms of Zulu media that could otherwise be difficult to find in a general purpose search engine. AfriWeb attempts to answer the research questions highlighted.

1.3 Ethical, Professional and Legal Issues

AfriWeb evaluation involves using human subjects. There are multiple ethical concerns when working with human subjects for research. The experiments conducted should be safe and scientific. Subjects should be provided informed consent and the option of remaining anonymous. All of the above issues have been dealt with through obtaining a Ethical Clearance. University of Cape Town's ethical clearance application checks if the experiments are ethically justifiable i.e does the experiment provide informed consent, does not exploit users personal information, experimenter has enough permissions to perform the experiment, e.t.c

The tools and technologies used in this project were all obtained from open source projects. The projects were all licensed under GNU-GPL and BSD licences. Those licences provide us with the freedom to share and change all versions of a program and make sure it remains released under the same licence. Using the licences stated above ensures that we do not run into any copyright infringement problems.

1.4 Work Allocation

The AfriWeb project was split into two portions: The front end and related technologies, and document collection and language identification.

Katlego Moukangwe was responsible for the development of a language identification system. The language identification system entailed development of a statistical language model. He was responsible for developing a focused crawler to identify Zulu documents on the Internet and provide them in an indexable format.

Nkosana Malumba was responsible for the design and implementation of the search engine. This included the integration of existing technologies such as an open source search platform apache solr with a user interface. Additionally, two algorithms were developed that were used in the customization of the indexing and querying of the search engine.

1.5 Summary of report

This report provides the information required to be able to perform Zulu language identification. It provides the background and literature required to understand how language identification is performed, and also the method and instructions used to build a language identification system. The report also provides information and literature pertaining to developing a focused crawlers. Methods of implementing a focused crawler for identifying documents that contained Zulu was also detailed in this report. Both the language identification, and Zulu documents focused crawling were evaluated in order to make certain the entire software system works as planned and consistently.

2 BACKGROUND

There has been a lot of research done on creating language models for language identification. Language models can be used for other problems such as machine-translation, parts-of speech tagging, et cetera. There are multiple techniques that can be employed to build a language model. This chapter outlines the theoretical ground work for the language model built for the AfriWeb software system.

2.1 LANGUAGE IDENTIFICATION

Language identification of text has become increasingly important as large quantities of text are processed for tasks such as information retrieval or machine translation. Existing methods of language identification use various information available in the text domain such as, probabilities of certain character combinations or word combinations, ngrams of words, ngrams of letters, syntax, syllable characteristics et cetera(Vatanen et al., 2010). The language identification process can be divided into two steps: First a document model is generated for the document and language model for the language;second the language of the document is determined on the basis of the language model(Grothe et al., 2008).There are several language identification methods available.

WORD BASED APPROACHES

The short-word based approach uses words of up to a specific length to determine the language of a document independent of the word's frequency(Grothe et al., 2008). (Hull and Grefenstette, 1996) used one million characters of text and for each language, tokenized them and stored the words with less than five characters. The reason behind this approach is that significant words of a language such as conjunctions have mostly short lengths(Grothe et al., 2008). The language models developed contained words between 980 words to 2750 depending on the language(Grothe et al., 2008). Given a document, the language model goes through the document and tries to find matching words in the language model. This is the most Naive language identification method. However the idea of using language specific features such as common words or conjunction is the central idea amongst all the language models.

The frequent word approach generates the language model using the most frequent words in a language(Grothe et al., 2008). These words have the highest frequency of all words occurring in text. Word based language model are useful for classifying reasonably sized documents that contain strings in a single language. Using one hundred high frequency words per language extracted from training data of nine languages, 91% of provided documents where correctly classified (Grothe et al., 2008). The documents where of minimum 500 bytes in length.

NGRAM BASED TEXT CATEGORIZATION

An n-gram is an n character slice of a string, for example 'ba' and 'ad' are two possible 2-grams for the word 'bad'. Although in literature the term can include the notion of any co-occurring set of characters in a string (e.g., an N-gram made up of the first and third character of a word), in this paper the term is used for a contiguous slice only(Cavnar and Trenkle, 1994). For each word, you have to prepend and append a backspace character to signify the end of a word. According to this definition, all possible bigrams,trigram and quad-grams for the word 'BAD' are

bigrams <s>B,BA,AD,D</s>

trigrams <s>BA,BAD,AD</s>

quadgrams <s>BAD,BAD</s>

where '<s> ' and '</s>' are start and end characters respectively.
Human language inevitably contains some words or characters that are more common than others. The distribution of characters of words can be described using Zipf's law, which can be stated as

The nth most common word or character in a language occurs with a frequency inversely proportional to n(Cavnar and Trenkle, 1994)

This laws simply implies that there are characters and words in natural language that occurs more frequently than others. Furthermore there is a smooth continuum of dominance from the most common ngrams to the least common(Kingsley, 1949). The smoothness of words or ngram distributions means rankings for most common characters can be obtained without ambiguity, i.e not 3 letters can be the most occurring letters in a language. Using the distribution of ngrams or words, a class or category to classify languages is created languages(Kingsley, 1949). Ngram distributions can be considered a unique fingerprint of a language. We expect to have documents from the same category to have the same ngram frequency distribution. The procedure followed in order to classify document samples into their respective categories is illustrated in the figure below.

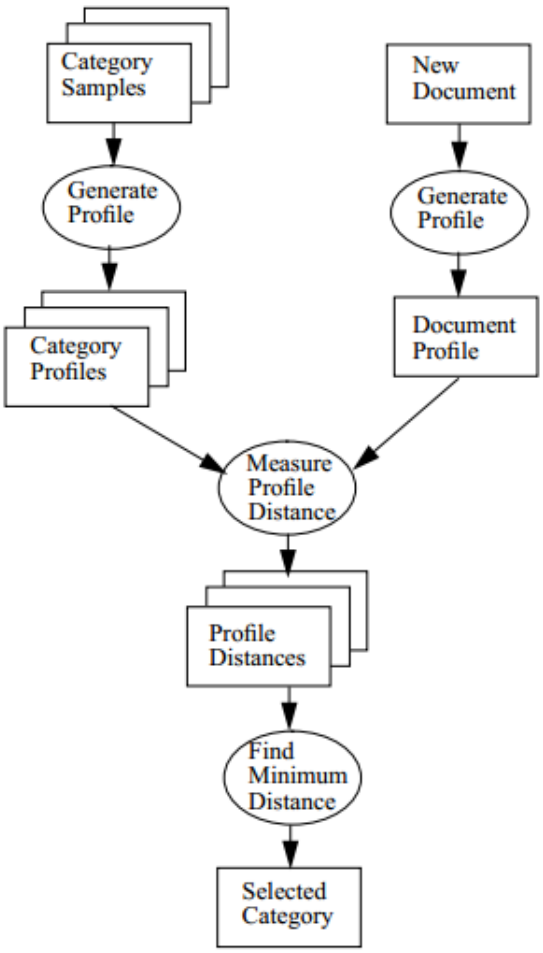


Figure 2.1: Classifying documents into ngram frequency categories

Firstly obtain sample documents for the categories you wish to classify. For language detection, this is usually a reasonably sized language corpus. Generating a language profile involves breaking the text from the category sample into ngrams and counting the occurrence of each ngram(Cavnar and Trenkle, 1994). The category profile can be any order ranking of ngrams in the language profile. An ngram of rank 1 is the most occurring ngram, of order 2 is the second most occurring ngram et cetera. A 300 ngram profile contains the 300 most occurring ngrams. There are multiple metrics used to measure the distance between two ngram profiles. The one explained below can be found in Cavnar's and Trenkle paper(Cavnar and Trenkle, 1994).

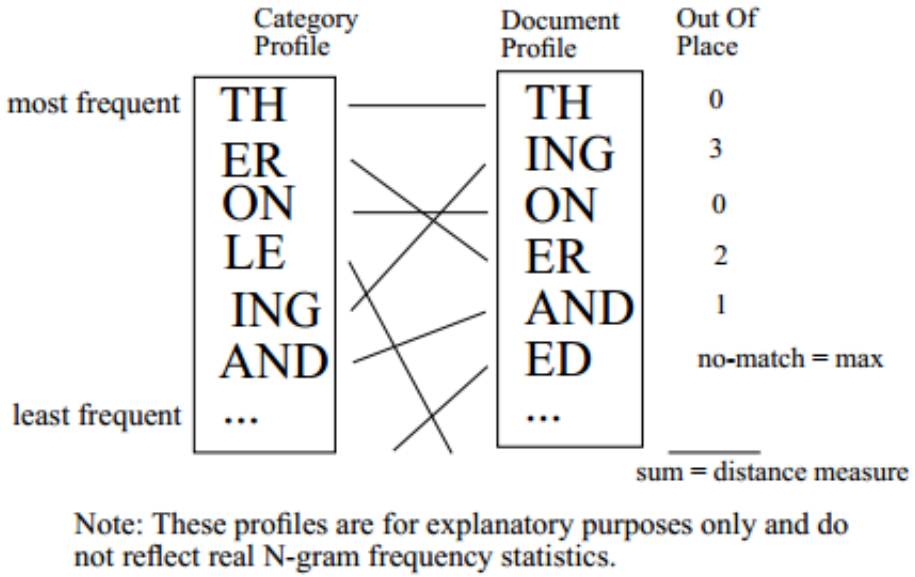


Figure 2.2: Out-of-place distance metric between two langauge profiles.

The out-of-place metric measures the distance between where the ngram needs to be in the language profile vs where it is found in the document profile. For example, if the most frequent ngram is 'TH' in the category profile and document profile, then distance between the category profile and document profile total distance is added by 0. If 'TH' is two places out-of-place, the total distance is added by 2

Using the out-of-place metric for language identification works very well for large documents(Cavnar and Trenkle, 1994). Test documents of larger than 300 bytes were tested and an accuracy of 95% was obtained(Cavnar and Trenkle, 1994). The out-of-place metric makes it easy to identify a language for large documents cantaining the same language. The documents AfriWeb works with are from the Internet and not necessarily homogeneous or of the correct size.

NGRAM SHORT TEXT LANGUAGE IDENTIFICATION

The language identification task is usually broken into two steps: the language modelling and classification. The ranking method explained previously does not consider conditional probability if a different word has been observed, that is

$$P(\text{York}|\text{New}) = P(\text{York}|\text{Rabbit})$$

where $P(a|b)$ is the probability that a is observed given that b has already been observed. The probability of the word "York" is the same regardless of what it is preceded by.

$$P(\text{York}|\text{New}) > P(\text{York}|\text{Rabbit})$$

However it is know that it is more likely to find the "New York" word combination than it is to find the "Rabbit York" word combination. The same principle applies to ngrams. Certain ngrams are more likely to follow each other. To deal frequently occurring combinations of ngrams, the language profile can be

redefined as an $n-1$ Markov chain. That means that the probability of a character or word depends on the preceding ngrams or words. The naive way to propagate the probability to higher order ngrams is using the Maximum likelihood estimate.

Definition 1. *Given ngrams or words w_1, w_2, then the maximum likelihood of ngram w_2 given w_1 is*

$$P(w_2|w_1) = \frac{count(w_1, w_2)}{count(w_1)} \tag{2.1}$$

Definition 2. *In Probability theory, the* **Chain Rule** *or* **general product rule** *is a calculation of any member of a set of joint random variables using conditional probabilities. A random variable is a variable whose value is subject to chance, i.e it can take on a set of possible different values. Joint random variable are Random Variables where chances a result belongs to either one of the Random Variables is the same. The chain rule states for two sets A_1 and A_2*

$$P(A_1, A_2) = P(A_1|A_2) \cdot P(A_2) \tag{2.2}$$

where $P(A_1, A_2)$ is the probability that A_1 and A_2 happen simultaneously.

Given a training set, a maximum likelihood estimate of words tends to over-learn the training data, i.e knows too much about the training data with very little prediction power(Vatanen et al., 2010). The estimate can be improved by using various smoothing methods that move the probability from the ngrams that occur frequently to those that rarely occur or have not been seen before(Vatanen et al., 2010). For the language identification task, models are created from character ngrams. A corpus is used as training data and maximum likelihood estimates are determined for ngrams in the language profile. Given an ngram, its maximum likelihood is determined and the class that has the maximum likelihood is chosen. This is regarded as a *Bayesian Classifier* with the ngram maximum likelihood probabilities as feature variables.

ADDITIVE SMOOTHING

The simplest smoothing used in practice is the additive smoothing technique. To avoid zero probabilities for unseen ngrams, it is assume that the ngram is seen δ more times than it actually does. Let $c(w_i)$ be the count of w_i, $|V|$ be the total number of ngrams, w_i^j be a slice of w from i to j. Equation 2.1 can be reformulated using additive smoothing as

$$P(w_j|w_i^{j-1}) = \frac{c(w_j) + \delta}{\delta|V| + \sum c(w_i^{j-1})} \tag{2.3}$$

where $0 < \delta < 1$ and $\sum c(w_i^{j-1})$ is the total number of ngrams of length $j-1-i$. (Chen and Goodman, 1999). Additive smoothing eliminates zero probabilities for ngrams not previously encountered. For example the Europarl bigram corpus[1] contains 867,00 distinct words, $86700^2 = 7,516,890,000$ possible bigrams are expected however only 30,000,000 words in the corpus. There are way too many unseen bigrams than there are seen bigrams. This explains why an overestimation of seen ngrams is required.

[1] http://www.statmt.org/europarl/

GOOD-TURING ESTIMATE

The Good-Turing estimate is central to a lot of smoothing techniques(Chen and Goodman, 1999). It states that if an ngram occurs r times in the training data, then assume it occurs r^* times where

$$r^* = (r+1) \cdot \frac{n_{r+1}}{n_r}$$

n_r is the number of ngrams that occur r times in the training data(Chen and Goodman, 1999). The Good-Turing estimate can be converted to probability by normalizing the total number of ngrams occurring r-times as $N = \sum_{r=0}^{\infty} n_r \cdot r^*$

$$P\left(w_i^j\right) = \frac{r^*}{N}$$

(Koehn, 2009) tested The Europarl corpus comparing adjusted counts using the Good-Turing estimate with observed counts. A test set of 7,514,941,065 bigrams was tested and the results presented below.

Table 2.1: Good-Turing for 2-Grams in Europarl

Counts	Counts of counts	Adjusted counts	Tested counts
r	n_r	$r*$	t
0	7,514,941,065	0.00015	0.00016
1	1,132,844	0.46539	0.46235
2	263,611	1.40679	1.39946
3	123,615	2.38767	2.34307
4	73,788	3.33753	3.35202
5	49,254	4.36967	4.35234
6	35,869	5.32928	5.33762
8	21,693	7.43798	7.15074
10	14,880	9.31304	9.11927
20	4,546	19.54487	18.95948

t is the average counts of n-grams in test set that occurred r times in corpus. Table 2.1 shows that the adjusted count is fairly accurate when compared against the test count.

In practice the Good-Turing estimate is never used standalone for smoothing because it does not include contributions from lower-order and higher-order ngrams. However it is used by several smoothing techniques(Chen and Goodman, 1999).

ABSOLUTE DISCOUNTING

Absolute Discounting is created by subtracting a fixed discount D from ngrams with non-zero count. The probability expression is given below.

$$P\left(w_j|w_i^{j-1}\right) = \frac{max\{c(w_i^{j-2}) - D, 0\}}{\sum c(w_i^j)} + (1 - \lambda_i^{j-1}) P\left(w_j|w_{i+1}^{j-1}\right) \tag{2.4}$$

where λ_i^{j-1} is a smoothing parameter calculated from the training data. Absolute counting can be motivated using the Good-Turing estimate (Chen and Goodman, 1999). It has been shown that the average Good-Turing discount $r - r^*$ remains constant for ngram with $r \geq 3$(Chen and Goodman, 1999).

KNESER-NEY SMOOTHING

Kneser-Ney smoothing is an extension of Absolute counting where the lower order ngram distribution is combined with higher order ngram distribution in a different manner(Chen and Goodman, 1999). Kneser-Ney smoothing adjusts equation 2.4 so that the first term dominates if the discounted ngram count is near zero i.e $c(w_i^{j-2}) - D \approx 0$(Koehn, 2009). Similarly if high order ngrams have more weight or have a high relative discount, then the second term is adjusted to carry very little weight(Koehn, 2009). This means the probability is moved from the high order ngrams to lower-order ngrams. Suppose your training data had a very common word "FRANCISCO" that only occurs after the word "SAN". A bigram model using Absolute discounting or maximum likelihood would assign a high probability for the word "FRANCISCO". The word "FRANCISCO" only appears after the word "SAN" so it should have a low unigram probability because of how unlikely it is to find it alone as a unigram. Kneser-Ney smoothing tries solve the issue of unigrams being over represented by calculating the probability using not only the number of occurrences a word appears, but the number of different words it appears after(Chen and Goodman, 1999). The number of different units that follow an ngram is written as

$$N\left(\bullet x_i^j\right) = |\{x_{i+1}^j : C(x_{i+1}^j) > 0\}| \tag{2.5}$$

Similarly,

$$N\left(\bullet x_i^{j-1}\bullet\right) = \sum_{x_j} N\left(\bullet x_i^{j-1} x_j\right) \tag{2.6}$$

For an ngram the probability can be estimated as(Vatanen et al., 2010)

$$P\left(x_i | x_k^{j-1}\right) = \frac{N\left(\bullet x_k^i\right)}{N\left(\bullet x_k^{j-1}\bullet\right)} \tag{2.7}$$

Kneser-Ney is the smoothing technique used to set up the language model for the AfriWeb project.

2.2 FOCUSED CRAWLING

The World Wide Web is growing at an extremely fast rate. This poses enormous challenges for general-purpose crawler because of the sheer amount information they have to process(Chakrabarti et al., 1999). The goal of a focused crawler is to selectively seek out pages that are relevant to a pre-defined set of topics(Chakrabarti et al., 1999). The topics used by focused crawlers are usually not just keywords by documents exemplifying what kinds of documents to crawl. Instead of collecting and indexing all possible Web documents to answer queries, a Web crawler analysis its crawl boundary and determines links that are likely to be relevant for the defined set of topics(Chakrabarti et al., 1999).

Examples of task accomplished by focused crawlers can be to; crawl for pages with a .jp domain, collect all documents with the highest PageRank[2], collect all pages about "swine flu" et cetera. Focused crawlers use all aspects of a Web page to be able to classify it as relevant or irrelevant.

[2]Page ranking algorithm used by Google

FOCUSED CRAWLING TERMINOLOGY

Classifier is what evaluates the relevance of a hypertext document with respect to the specified topics(Chakrabarti et al., 1999).

Distiller identifies hypertext nodes which are great access points to more relevant pages within a few links(Chakrabarti et al., 1999).

Crawl Prioritization are rules that affect the order in which pages are visited.

Seed defines initial domains to be crawled.

Whitelist Strategy starts from a list of high quality seed websites and limits the crawl scope to the domain of those URL's.

Web Scraping is a technique for extracting information on websites. This usually involves conversion of unstructured information into structured information.

Term frequency is the number of times a term appears in a document.

Inverse document frequency is a measure of how much information a term provides. It is calculated as $idf = \log \frac{N}{df}$, where df document frequency, and N the total number of documents.Document frequency of a term t measures the number of documents term t appears.

Term Frequency–Inverse Document Frequency $td.idf$ is a numerical statistic that measures how important a word is in a document. It is calculated as the product between term-frequency and inverse-document frequency.

OVERVIEW OF FOCUSED CRAWLING ALGORITHMS

The Visible Web is the part of the Internet that is accessed by following links. The vast majority of the Internet's information can only be accessible through submitting queries or Web forms(Novak, 2004). A Web crawler usually has access to the visible Web. The process a focused crawler follows in order to get the information from the Visible Web, is to start with a set of seed pages that indicate the type of content a users is interested in. Links from the seed pages are then sorted by page relevance and then inserted into a priority queue(Novak, 2004). Evaluation of page relevance can range from simple keyword matching to complex machine learning classification schemes. Links obtained from the relevance page algorithm are then inserted into a filter(Novak, 2004). A filter removes links that are considered completely undesirable. Examples of undesirable links are those that have a "do not follow" meta tag. This are links the Web master requires not to be indexed. This is called the Robot Exclusion protocol(Novak, 2004). The Robot Exclusion protocol is not mandatory and can be ignored or overridden. However it is advisable to adhere to it in order to avoid crawler traps(Novak, 2004).

Link or crawl prioritization are rules that determine which links are more likely to have Web pages relevant to the focused crawler's topic. The most widely used and known metric to order Web pages is the PageRank. PageRank tries to objectively measure human interest and devotion to a Web page(Page et al., 1999). PageRank naively works by counting the number of links to a page to estimate the importance of a webpage. The idea is that important webpages are more likely to receive links from other pages. (Cho et al., 1998) proposed using PageRank for crawl prioritization. Calculating the PageRank(Page et al., 1999)

score on the graph induced by pages downloaded so far and then using this score as a priority of URLs extracted from a page(Novak, 2004). (Cho et al., 1998) investigation shows that there is an improvement for using PageRank for link prioritization as opposed to using a breadfirst or depth-first approach(Cho et al., 1998). The improvement, however was not considerably better because of the fact that the page score is calculated on a small subset and non-random portion of the Web(Novak, 2004). PageRank is hardly used used for topic-driven tasks, however performs well for general-use(Novak, 2004). Figure 2.2 shows a visual representation of PageRank with a webpage's PageScore.

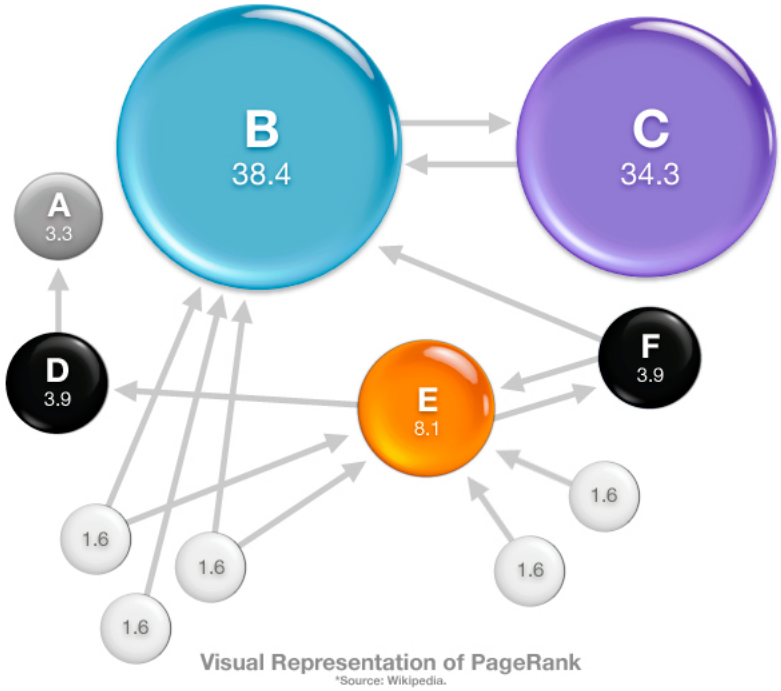


Figure 2.3: Visual representation of PageRank

Evaluating Focused crawlers

Focused crawlers are usually evaluated by their 'harvest rate'. Harvest rate is the ratio between the number of relevant documents retrieved to the total number of documents retrieved(Novak, 2004). This is easy to calculate if the set of relevant documents can be easily obtained. Identifying a set of relevant documents can be easily accomplished by a user based evaluation. Unfortunately user experiments involving users accessing Web crawls are extremely problematic(Novak, 2004). The growing number of crawls and the scale of the Web suggests that to have a reasonable notion of a focused crawlers effectiveness, a large number of user tests are required. Although a large scale user based evaluation would be ideal, it is important to look into user independent evaluations(Menczer et al., 2001). Below is an outline of three topic-driven metrics for measuring crawler effectiveness.

Assessment via Classifiers. The first approach to user independent evaluation is to use a classifier. 100 topics were selected and a classifier built for each(Menczer et al., 2001). A positive set for a topic P consist of all pages corresponding to P's seed URL's(Menczer et al., 2001). Negative topics are the seed of the

remaining 99 topics. The one exception happens if topic P and Q share URL's in their seed URLs. Then Q is removed from P's negative set and vice verse. 50 best features of the seed websites were selected. To determine if a page is relevant, its features are compared to that of the seed websites and then classified by topic(Menczer et al., 2001). In this analysis, classifiers are used only for evaluation and not in the crawl algorithm(Menczer et al., 2001).

Assessment via a Retrieval System. The second evaluation strategy uses an independent retrieval system to rank pages according to topic(Menczer et al., 2001). A crawler is assessed by looking at when good pages were crawled. The assumption is that a good crawler will retrieve high ranking pages earlier than the lower ranked ones(Menczer et al., 2001). Note that using the temporal position of crawl documents is only useful if the crawl is ran live or once. If the crawl is intended to add to a cumulative index such as a search engine, then the rank is not interesting or useful(Menczer et al., 2001).

Assessment via Mean topic similarity. The third method assumes that a good crawler should remain in the vicinity of the topic *vector space*(Menczer et al., 2001). A topic vector space is a mathematical model defining topics that are similarly. To figure out if the topic of a page is relevant, the cosine similarly angle between the $tf * idf$ vector of the topic and the $td * idf$ of the crawled page is calculated(Menczer et al., 2001). If the cosine similarity angle is highest, the topics are considered similar and thus relevant.

2.3 Discussion

It has been shown that a large number of research has been made on language identification and focused crawling. The concepts and methods described in this chapter play a vital role in the development of AfriWeb's language identification and focused crawling system. Kneser-Ney smoothing is the language modelling technique used to develop the language model. The focused crawling algorithm used is based on a combination of all approaches that were discussed. Being able to crawl for Zulu documents is a step closer to creating a Zulu search engine.

3 Design and Implementation

3.1 Overview

The aim of the language identification and focused crawler components of AfriWeb is to accurately identify Zulu text and download Zulu documents from the Internet. This section details how each component was developed or how they progressed from start to the final software product.

Language identification overview

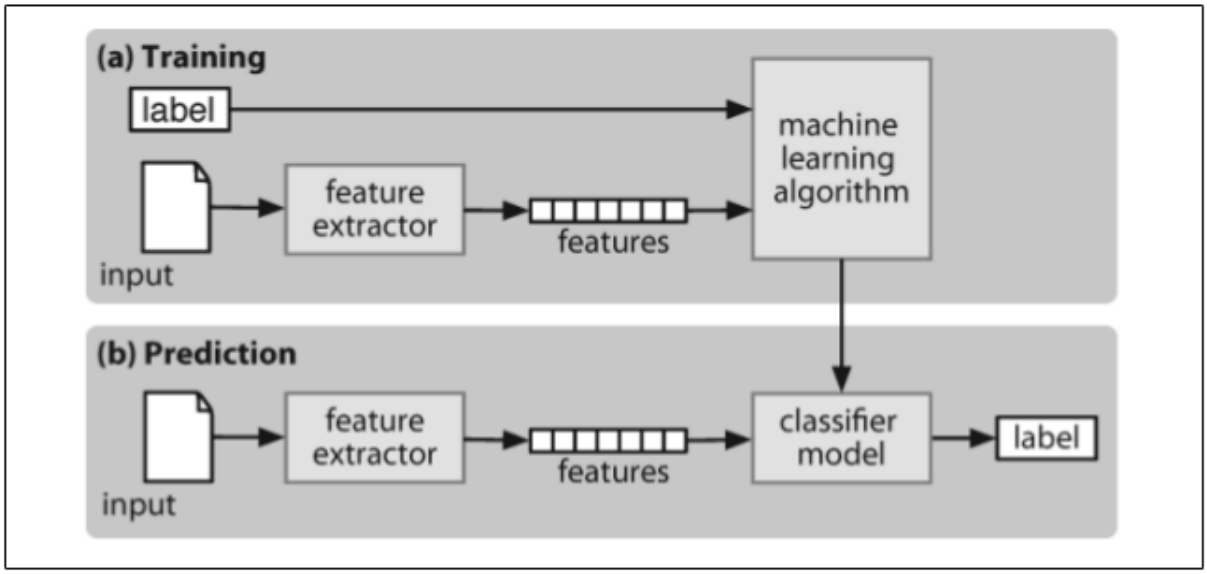


Figure 3.1: Supervised classification for language identification

Supervised classification is choosing the right label for a given input. The language model is suppose to decide, given a string if it is Zulu or non-Zulu. The language model is trained using a corpus during the training phase as illustrated in Figure 3.1. A feature extractor is a tool to obtain the corpus ngram distribution. A classifier is what decides if input provided belongs to Zulu or non-Zulu classes. Figure 3.1 shows the result expected after the implementation.

Focused Crawling overview

The crawling starts from a set of seed websites. The content of a page is loaded and then preprocessed so that the core information can be obtained. The preprocessing stage involves the removal of punctuation, HTML tags, e.t.c . The content is then forwarded to the classifier to decide if it is relevant or not. If a page is relevant, all its links are visited to search for more relevant pages.

3.2 Technology and Tools

NLTK-*Natural Language Processing With Python* is a Python platform used to work with human language data. It contains over 50 corpora and lexical resources such as text processing for classification, tokenization, stemming, tagging, parsing et cetera. *VariKN* is a modelling toolkit that provides tools for training

ngram language models(Siivola, 2007). It provides methods to perform absolute discounting smoothing, Kneser-Ney smoothing and Revised Kneser pruning amongst many others. *VariKN* is written in C++ however provides a Python wrapper for those who require to use it in Python(Siivola, 2007).

Ukwabelana Corpus was the Zulu corpus used(Spiegler et al., 2010). It was split into two sets to perform language model training and testing.

Scrapy is an open source Web crawling framework with support for Web scraping. It is written in Python. It provides methods for easy extraction of meta-data on websites. It is easily customizable. All Python dependencies and subprograms where downloaded using pip, a software management package.

Heritrix is a large scale Web crawler designed for Web archiving. It is written in Java and provides command line and Web interface tools to initiate and manage crawls.

3.3 PREPROCESSING

The data passed from the focused crawler to the language model is assumed to be preprocessed. Methods to perform the following preprocessing were implemented.

- Remove punctuation. Punctuation marks were replaced by an empty character. Words such as "John's" become "Johns". Removing punctuation removes statistical analysis of punctuation. It does not matter that fullstops are more common in Zulu than there are in non-Zulu.
- All strings are in lower-case. Word such as "Something" become "something". This is to avoid same strings being processed differently since Python treats lower and upper case strings differently.
- Non-ASCII characters are replaced by an a space character. All Zulu characters are within the ordinal range of ASCII characters. If a characters in non-ASCII, it is considered non-Zulu. Strings such as "Change language to 文章内容。" become "Change language to ".
- All numbers are considered relevant, however not included in the total number of Zulu words. If the number "1994" is encountered, then it is saved to the indexable document, if and only if the page is deemed relevant by the classifier.
- HTML tags have been removed using lxml.html.cleaner. The cleaner removes embedded javascript, comments, meta-tags, frames, post-forms, unknown-tags, embedded multimedia and scripts. Removing the HTML elements makes it easier to get the textual content on a webpage.
- Multiple whitespace is reduced to an empty string. Whitespace provides formatting information and does not provide any more information about the language of a webpage. Multiple tabs will be reduced to an empty string.

3.4 ITERATION I

The first development iteration was concerned with rapid prototyping and development of a proof of concept system. A horizontal prototyping strategy was employed. The language model was prototyped first because it forms part of the focused crawler. The prototype for the language model used the ranking method as described in the background chapter. The ranking method was implemented in Python. It

used NLTK to generate unigrams, bigram and trigrams. The ngrams where stored in a Python dictionary to enable constant time lookups. Finding the most common ngram involved sorting the list of ngrams by counts. Below is the pseudocode of the algorithm used for the language identification prototype.

Algorithm 3.1: IDENTIFYLANGUAGE(input)

```
procedure CREATEPROFILE(corpus)
  ngram distribution ← all possible unigrams, bigrams and trigrams from corpus
  profile ← 300 most occurring ngrams
  return (profile)

procedure DISTANCE(language profile, document profile, ngram)
  comment: calculates the out-of-place distance for ngram
  distance ← 0
  for ngram ∈ document profile
  do { if ngram ∈ language profile
       then { lang index ← index of ngram in language profile
              document index ← index of ngram in document profile
              A ← absolute value(lang index, document index)
              distance ← distance + A
       else distance ← distance + 300
  return (distance)


main
  non-Zulu profile ← CREATEPROFILE(non-Zulu corpus)                    (i)
  Zulu profile ← CREATEPROFILE(Ukwabelana corpus)
  input profile ← CREATEPROFILE(input)
  if DISTANCE(non-Zulu profile, input profile) ≤ DISTANCE(Zulu profile, input profile)
    then return (non-Zulu)
    else return (Zulu)
```

The *Ukwabelana Sentences* corpus was used to train the language profile(Spiegler et al., 2010). The *Ukwabelana Corpus* has a list of Zulu sentences obtained from Zulu fictional writing and the Bible. A language profile was developed and the top 300 common ngrams selected. A Naive Bayers classifier was used to determined which class or set a test string belonged to. The Naive Bayers classifier worked by determining the out-of-place distance between the test string. The out-of-place distance calculation was implemented as described in the background chapter using a maximum distance of $d = 300$. The class which had the lowest out-of-place distance was considered the correct class. Using the out-of-place distance showed that Zulu can be detected to a reasonable accuracy given strings of sufficient length(Cavnar and Trenkle, 1994).

The focused crawling was firstly attempted using the Heritrix Crawler. Heritrix is primarily designed for Web archiving(Mohr et al., 2004). It provided seamless ways to crawl certain domains and workarounds when encountering unexpected input or Web objects. Adjusting Heritrix for focused crawling would

involve stripping away all its Web archiving settings. It would involve overriding Heritrix's crawlers frontiers, pre-fetch chains, extract chains et cetera(Mohr et al., 2004). The best alternative was to use a framework that made it easy to obtain meta-data information from websites and is easily customizable. Scrapy framework was the alternative chosen.

3.5 ITERATION II

The language identification task can be improved by the aid of a dictionary. the *Ukwabelana Words* corpus was used as a dictionary for Zulu by storing the words in a Python set. A set is the best data-structure for implementing a dictionary. It facilitates constant time lookups for membership. That means before line i of algorithm 3.1, a check was added to see if the word is part of the Zulu dictionary or not. Different ranking distances were investigated however there were not considerably better or worse than the 300 ngram distribution ranking.

Scrapy was setup and installed using pip - a software management system to install and manage Python packages. The strategy for visiting nodes was breadfirst. Nodes which contained Zulu were considered relevant. Nodes that contain above 50 Zulu words were considered relevant and all its links will be followed to look for more Zulu documents. It is assumed that documents that contain Zulu are more likely to be connected to other Zulu documents than documents which are not. This assumption of topical relevance is the reason why all the links of a relevant page where visited and not just links which contained Zulu text in the link description.

Algorithm 3.2: CRAWL(seed websites)

```
procedure PREPROCESS(link)
  comment: Remove HTML structure information and obtain a list of sentences from URL
  return (URL content)

main
  seen URLs ← {}
  queue ← {seed websites}
  while queue is not empty
    do { link ← pop link from queue                                   (i)
         if link is seen
           then continue
         add link to seen URLs
         link content ← PREPROCESS(link)
         if IDENTIFYINGLANGUAGE(link content) == Zulu
           then { download Zulu portion of link content
                  en-queue all the links from link
```

The above pseudocode shows how the first prototype of the focused crawler worked. If Zulu was detected on a Web page, it was regarded as relevant and subsequently downloaded. This method managed to obtained a fair amount of Zulu documents, however a considerably large number of non-Zulu documents

as well. A refinement to avoid non-Zulu documents was required.

3.6 ITERATION III

The last iteration. The *Ukwabelana Sentences* corpus was transformed using a Python script so that it can be given as input into *VariKN* language modelling toolkit(Siivola, 2007; Spiegler et al., 2010). *VariKN* requires character in sentences to be separated by a space and start and end of sentence to be indicated by a '<s>' and </s> symbols respectively.*VariKN* was ran to perform Kneser-Ney smoothing on the input. It was limited to unigrams, bigram and trigrams. The results were outputted in ARPA format, which is shown in figure 3.6. The ARPA format (Paul, 2006) contains the number of ngrams of each kind $n1, n2, \cdots, nN$ and the smoothed probability p of an ngram $w1 \cdots wN$. The probabilities were stored in a Python defaultdictionary for constant time access. Given input, the language model broke the input into words and applied the language detection on the words. Performing the language detection on the words enables dictionary lookups for words already know to be part of the Zulu language. Furthermore an English dictionary was used to reduce the number of English strings considered to be Zulu. The English dictionary was obtained from Python's *enchant library*. Let $W = w_1, w_2, w_3, \cdots, w_n$ be a word, then using the chain rule the probability of a word W consisting of ngrams $w_1, w_2, \cdots, w_n$ can be calculated.

```
\data\
ngram 1=n1
ngram 2=n2
...
ngram N=nN

\1-grams:
p       w                [bow]
...

\2-grams:
p       w1 w2            [bow]
...

\N-grams:
p       w1 ... wN
...

\end\
```

Figure 3.2: Ngram ARPA format

$$P(W) = P(w_1)P(w_2|w_1)P(w_3|w_1,w_2)\cdots P(w_n|w_1,w_2,\cdots,w_{n-1})$$

In a trigram language model, the above equation becomes

$$P(W) = P(w_1)P(w_2|w_1)P(w_3|w_2,w_1)P(w_4|w_3)P(w_4|w_3,w_2)\cdots P(w_n|w_{n-1},w_{n-2}) \tag{3.1}$$

For each word encountered, its membership was checked against the two implemented dictionaries. If the word appeared in both dictionaries, then it was classified as non-Zulu. If a word did not belong to any of the implemented dictionaries, then its class was obtained through the statistical language model. From the list of possible unigrams, bigrams and trigrams obtained from *VariKN*, the probability of a word was calculated. Ngrams where generated using NLTK. The probability that the word belonged to Zulu was calculated using equation 3.1. The probabilities for both Zulu and English were calculated, and the one with a highest probability was chosen as the class of choice. If the difference between the two probabilities was less than 10^{-3}, non-Zulu was chosen as the class of choice. The additional check of subtracting the obtained probabilities was to deal with languages that were neither English nor Zulu. The pseudocode for the language identification using smoothing is illustrated below.

Algorithm 3.3: IDENTIFYLANGUAGEWITHSMOOTHING(input)

```
global english-pdf, Zulu-pdf, english-dict, Zulu-dict
procedure LOADLANGAUGEMODEL(smoothedARPAfile)
  comment: pdf is a Python defaultdictionary
  pdf ← {}
  for each ngram ∈ smoothedARPAfile
    do pdf[ngram] ← ngram probability
  return (pdf)

procedure CALCULATEPROBABILITY(pdf, word)
  p ← 1
  for each ngram ∈ word
    do p ← p * pdf[ngram]
  return (p)

main
  if input ∈ Zulu-dict and input ∈ english-dict
    then return (non − Zulu)
  if input ∈ Zulu-dict
    then return (Zulu)
  if input ∈ english-dict
    then return (non − Zulu)
  english-pdf ← LOADLANGAUGEMODEL(english-arpa-file)
  Zulu-pdf ← LOADLANGAUGEMODEL(Zulu-arpa-file)
  english-probability ← CALCULATEPROBABILITY(english-pdf)
  Zulu-probability ← CALCULATEPROBABILITY(Zulu-pdf)
  if |Zulu-probability − english-probability| ≤ 10^3
    then return (non − Zulu)
  if Zulu-probability < english-probability
    then return (non-Zulu)
    else return (Zulu)
```

In the last iteration for the development of the focused crawler, methods of avoiding non-Zulu text were

enforced. The HTML tag lang was restricted to Zulu pages and English pages. English includes all kids of English such as "en_US", "en_AU", e.t.c This was to avoid irrelevant pages that contained other languages. The additional language check was inserted after line i of algorithm 3.2.If the language is "en" then a threshold of 50 Zulu words was required in order to be classified a Zulu document. If the documents language parameter is "zu" then a threshold of 20 Zulu words was required for the document to be classified relevant. The reason for a lower threshold for documents with lang="zu" is because most Zulu documents on the Web contain very little Zulu. If the language is already Zulu, very little false-positives are expected. websites such a Zulu-Wiki contain pages with very little Zulu content.

3.7 Issues and Challenges

There were a lot of implementation challenges and design decision challenges faced. The first was deciding on which corpus and what sized corpus to use. There was not a lot of corpora to choose from, however from the hand-full of Zulu corpora available, there was not a clear cut method to objectively find the best one. There is a lot of literature on corpus building and analyzing the pros and cons of using different corpora. *Ukwabelana* was chosen because it had about 28000 sentences of which Python can handle reasonably well. However there is a possibility that other corpora might have resulted in better results. The list of available Zulu corpora can be found at https://corplinguistics.wordpress.com/tag/Zulu/.

Finding seed websites to start crawls. The crawler follows almost a whitelist crawling strategy. It finds most of its relevant pages from the seed websites. Searching for Zulu documents from Google.com does not provide enough unique websites.

3.8 Portability and Maintainability

The language model and focused crawler were implemented in Python. The only way for anyone else to run the project is to have an environment similar to what has been set up on the computer used for development. It would not be considered portable because of the need to install the dependencies. Packages were installed using different package management systems. Instructions for setting up the environment can be provided, however it is up to the user to handle any system dependant errors. This is in contrast to a Java developed project which can be compiled once and ran everywhere where there is a Java Virtual Machine.

The project is maintainable because of how the components have been separated and well documented. Defects can be detected in isolation because the language model is different from the focused crawler. Methods and other components where implemented in an Object-Orientated manner which makes debugging, error detection and unit testing easier.

3.9 Discussion

This section described how the language identification system and the focused crawler were implemented. The details of this section were to provide information on how the project evolved from the initial prototyping stages to the final deliverable. It also provides information on how the project can be re-implemented using the same tools or different tools. The project can be also be easily adjusted to experiment with different parameters and settings. Input for loading corpora is read from a file, probabilities are read in ARPA format and seed websites can easily be changed.

4 EVALUATION

4.1 LANGUAGE IDENTIFICATION EXPERIMENT

The language identification system is responsible for identifying a language given a piece of text. The evaluation of this component assess how well it classifies given test sets into their correctly labelled classes. Given a string, tf can fall into any of the classes shown below

True-positives Zulu strings that are correctly classified as Zulu.

True-negatives Zulu strings that are wrongly classified as non-Zulu.

False-positives non-Zulu strings that are correctly classified as non-Zulu

False-negatives non-Zulu strings that are wrongly classified as Zulu.

A good language identification system is one that has as very little to no True-negatives and False-negatives.

HYPOTHESIS

Language detection becomes more accurate with longer input strings.

ENVIRONMENT SETUP

1. The Language model.
2. Zulu strings obtained from the *Ukwabelana Sentences* Corpus(Spiegler et al., 2010).
3. Zulu strings mixed with English strings obtained from the *Ukwabelana Sentences* corpus and NLTK *Brown Corpus*
4. Non-Zulu and non-English strings. Sentences obtained from *Paiśa* Italian Corpus.

METHOD

The accuracy of the language model can be defined as the percentage of Zulu strings identified from a set of strings, or equivalently the amount of non-Zulu strings correctly classified as non-Zulu from a set of non-Zulu strings. To investigate how the language model behaves with input strings of different lengths, accuracy for sentences of varying lengths will be determined and graphed to explore the behaviour.

1. Generate multiple stings or sentences of Zulu and observe what fraction is correctly classified as Zulu.
2. Generate strings of mixed Zulu and non-Zulu strings and note what fraction of that is classified as Zulu and what fraction is classified as non-Zulu.
3. Generate strings of non-Zulu and observe what fraction of that is classified as non-Zulu.

MEASUREMENTS

For each data-set, the amount of true-positives, true-negatives, false-positives and false-negatives were measured. The language identification will be considered accurate if it has very little false-negatives and true negatives.

4.2 FOCUSED CRAWLER EXPERIMENT

The crawler's effectiveness is subject to the accuracy of language detection. The crawler's accuracy can be defined as the percentage of Zulu documents crawled from the total set of crawled documents. Accuracy for larger data-sets is desirable.

HYPOTHESIS

The lower the fraction of non-Zulu strings classified as Zulu, the better the accuracy of the focused crawler.

ENVIRONMENT SETUP

1. User population consisting of 10 human subjects.
2. Interface to easily classify documents.

METHOD

1. Three sets of crawled documents will be used to measure the accuracy of the focused crawler.
2. Experimenter provides the first set of crawled documents.
3. Experimenter randomly selects 10 documents using the program evaluate-crawler.py.
4. User classifies the provided documents as Zulu or non-Zulu using the provided interface..
5. Experimenter provides another set of crawled documents and then repeat step $2-5$. If accuracy for all sets have been measured, the experiment is complete.

MEASUREMENTS

10 randomly chosen documents were provided to each subject. They were required to classify how many of the selected documents contained Zulu. The total number of documents classified as Zulu was noted.

4.3 EXPERIMENT RESULTS

ZULU DETECTION RESULTS

The language model was given five different test sets and the observed results used to calculate its accuracy. The sets were labelled ($P_{Zulu}, P_{English}, P_{mixed}, P_{Italian}$) were the composition of each set is explained below

P_{Zulu} Contained 29423 variable length sentences with 100% Zulu words.

$P_{English}$ Contained 28000 variable length sentences which had 100% English words.

P_{Zplus} Contained 25000 variable length sentences of which 40% was Zulu and 60% was English.

$P_{Italian}$ Contained 25215 variable length sentence with 100% Italian words.

The set of Italian sentences was obtained from the *Paiśa Italian Corpus*[3], the English sentences were obtained from NLTK's *Brown Corpus* and the Zulu sentences were obtained from the *Ukwabelana Zulu Corpus*.

Each of the sets was run through the language model and the accuracy was calculated. The accuracy is defined as the ratio between the *number of words identified as Zulu* over the *total number of Zulu words*, or equivalently *number of non-Zulu* over the *total number of non-Zulu* expected. The results for each of the four classes are presented in the table below. The inputs from the sets where classified into four sets as shown below.

Table 4.1: Percentage of the input data-set belonging to each class

Dataset	True-Positive(%)	True-Negative	False-Positive	False-Negative
P_{Zulu}	98.4	1.6	0.0	0.0
$P_{English}$	0.0	0.0	98.8	1.2
$P_{Italian}$	0.0	0.0	87.6	12.4
P_{mixed}	97.5	2.5	98.3	1.6

The graph below shows that there is a large fraction of words with high accuracy and very low fraction of words with low accuracy.

[3] http://www.corpusitaliano.it/en/

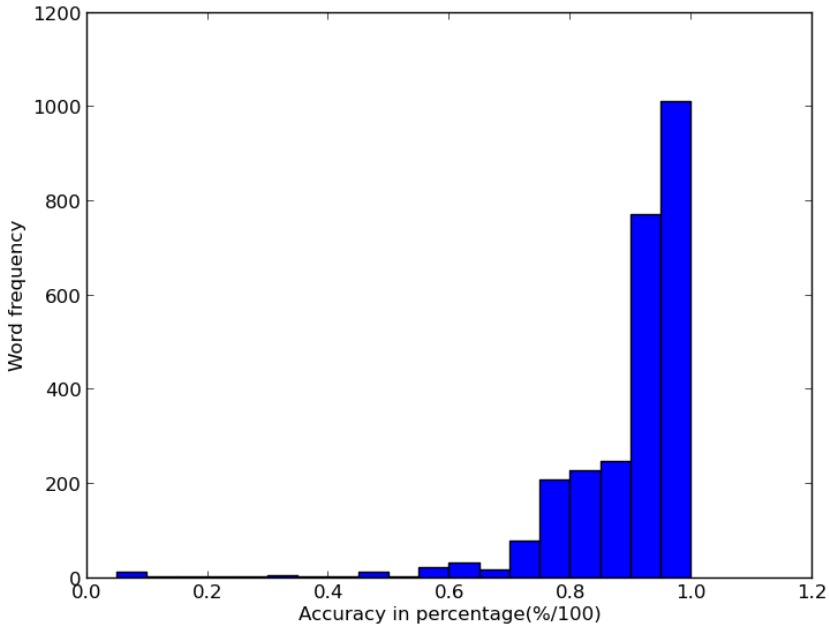


Figure 4.1: Bar-chart of Zulu Accuracy vs word frequency for P_{Zulu}

The effects of varying sentence length for the rest of the data-sets was investigated and the results are presented below.

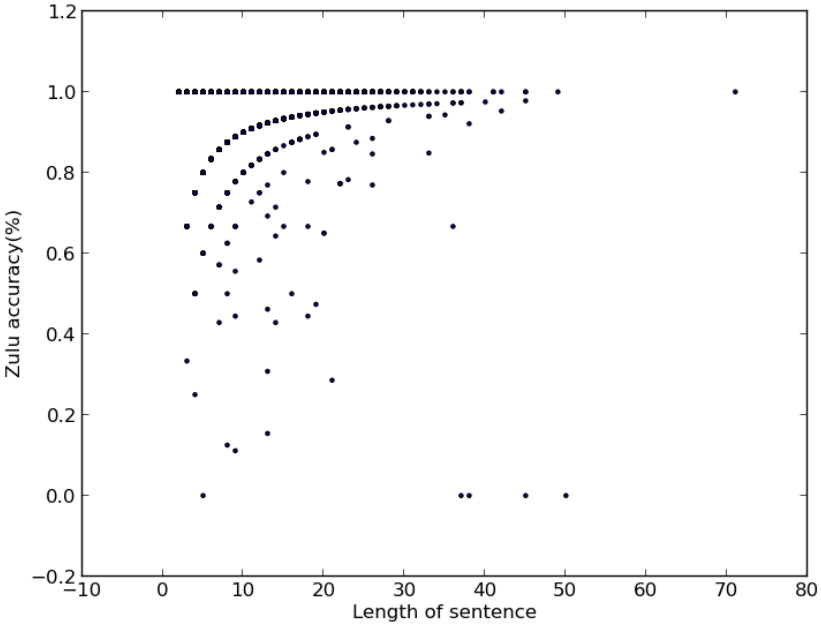


Figure 4.2: The effects of varying sentence length on P_{Zulu}

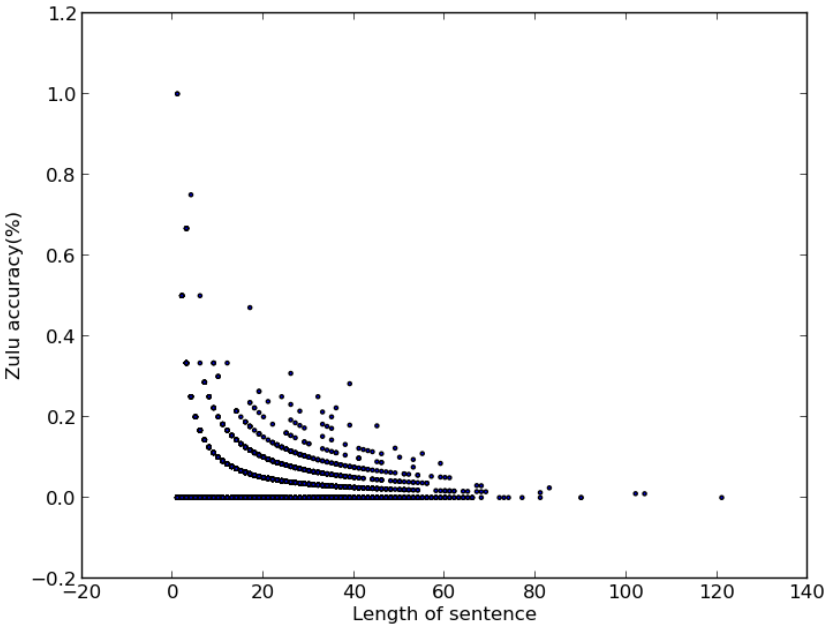


Figure 4.3: The effects of varying sentence length on $P_{English}$

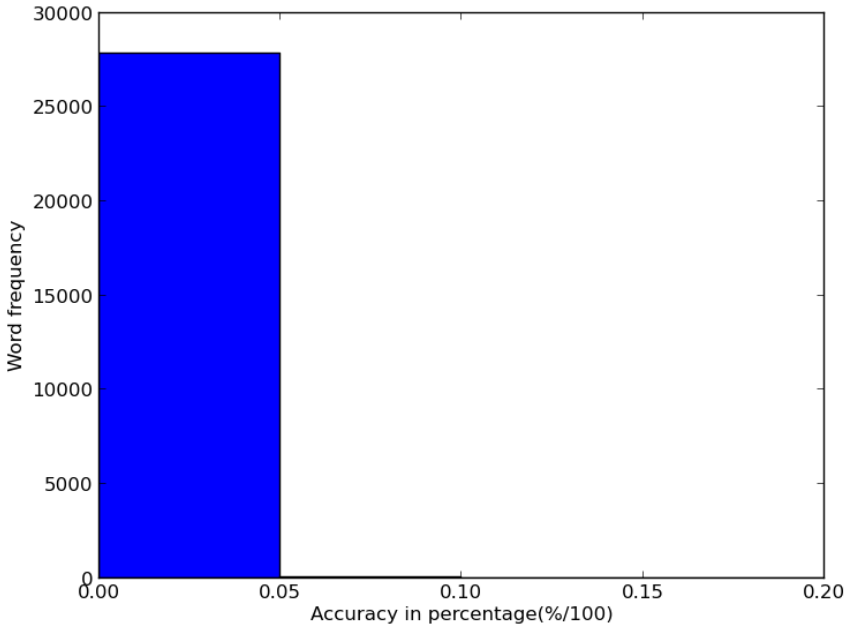


Figure 4.4: Bar-chart of Zulu inaccuracy vs word frequency for $P_{English}$

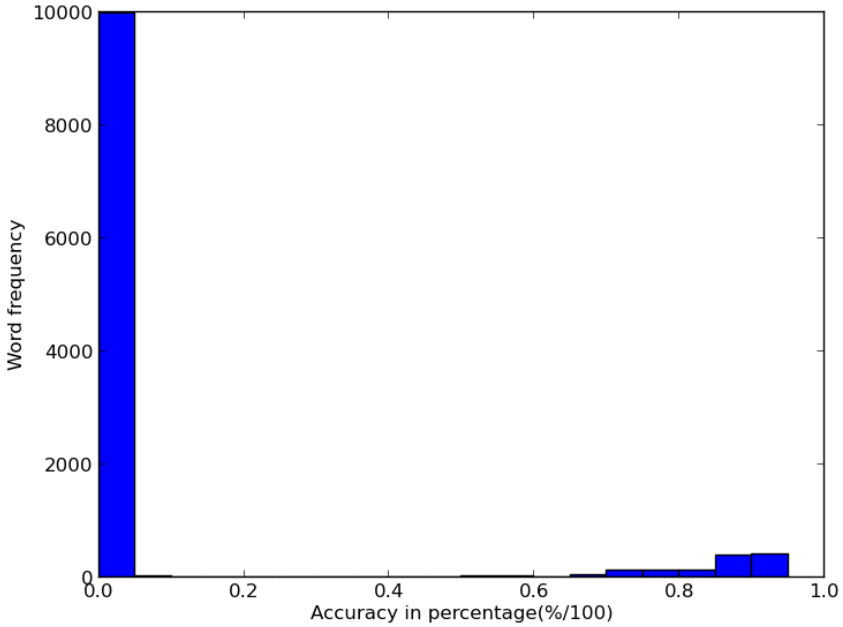


Figure 4.5: Bar-chart of Zulu accuracy vs word frequency for P_{mixed}

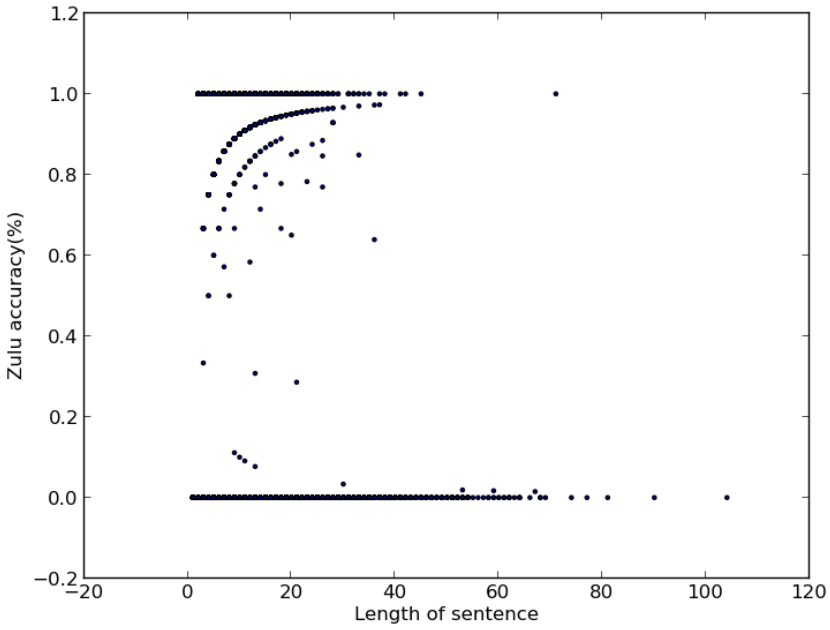


Figure 4.6: The effects of varying sentence length on P_{mixed}

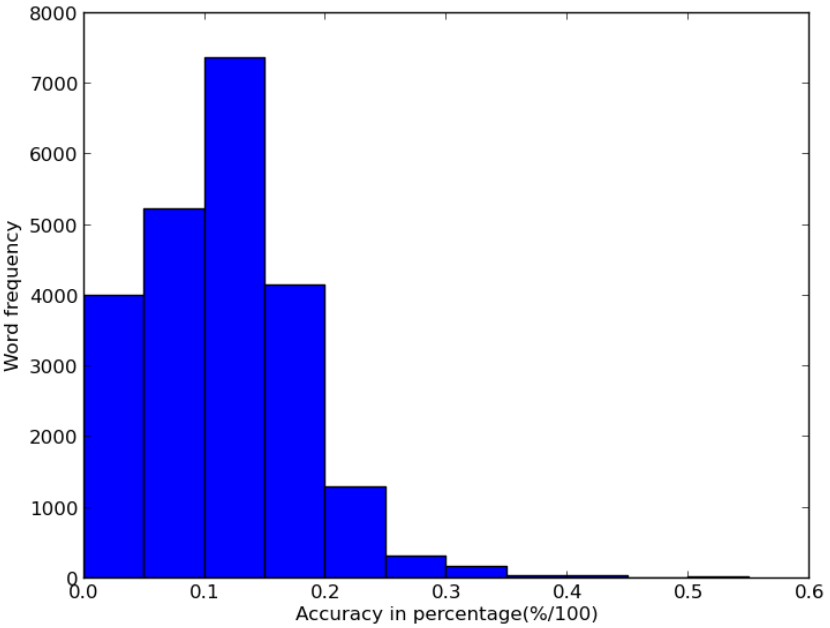


Figure 4.7: Bar-chart of Zulu accuracy vs words frequency for P_{Italia}

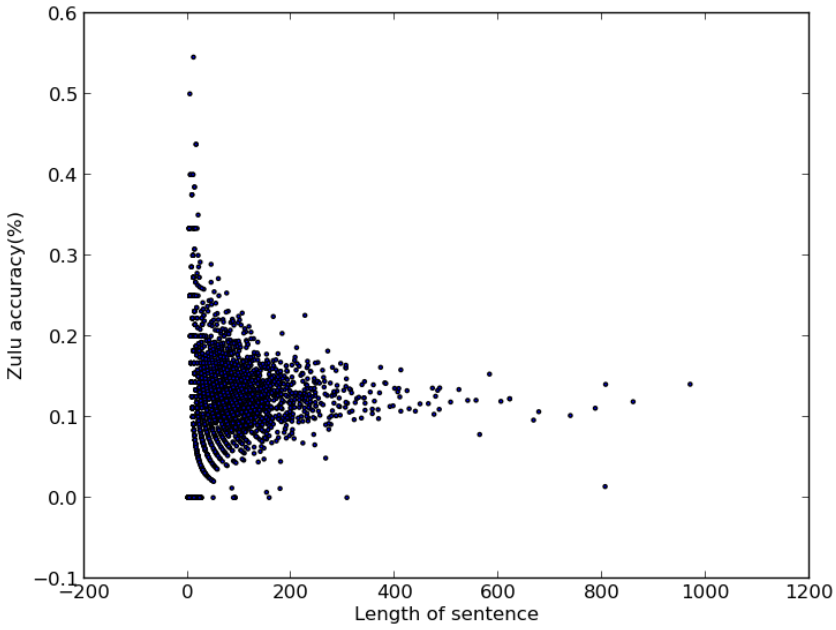


Figure 4.8: Graph of accuracy vs word frequency for the Italian data-set

4.4 Discussion

Language model Performance with set P_{Zulu}

A portion of the *Ukwabelana Sentences* corpus was used as a test set. The aim was to determine what fraction of the sentences does the language identification correctly classify as Zulu or non-Zulu. The bar-chart 4.3 show that there are very little words classified as non-Zulu. The bar-chart also shows that there is a trend when considering correctly classified words. The language identification can confidently classify a larger proportion of the words found. From figure 4.3 it is evident that the proportion of True-positives outweighs that of True-negatives.

The top of figure 4.3 has a solid line for sentences that have an accuracy of 100%. This is due to the dictionary look-ups performed for words known to be Zulu. The sentences which have 0% Zulu accuracy corresponds to words that are suppose to be Zulu, however classified as non-Zulu. These include words that are both Zulu and English such as " *lento, into, bake,* $\cdots$ ". The language model classifies words that belong to Both Zulu and non-Zulu as non-Zulu. The image belows display Zulu words from *Ukwabelana Sentences* that were classified as non-Zulu. The size illustrates their relative frequencies.

Figure 4.9: Words from *Ukwabelana Sentences* classified as non-Zulu

From the image above it can be concluded that the *Ukwabelana* sentences corpus didn't contain 100% Zulu sentences as expected. It contains words like " *assurance* , *standards* , *training* , $\cdots$ " which are not Zulu. From figure 4.3 we can see that these words contribute a negligible portion of the entire sentence corpus and the errors due to that can be ignored.

It was hypothesized that longer strings will result in higher accuracy. Figure 4.3 confirms this. Figure 4.3 shows that there is a gradual increase in accuracy as the length of the sentences increases. The long sentences have more ngrams which provides more information to the statistical language model to perform a better classification. Overall we can conclude that the language model performs very well given enough reasonably lengthened Zulu strings.

Language model Performance with set $P_{English}$

The next property investigated was the percentage of false-positives obtained. The language was given a set of 100% English sentences. Figure 4.3 shows that false-negatives were not found. Words that belong to both English and Zulu are classified as Zulu. The corpus may consist of very common words, hence language identification can be accomplished solely on dictionary lookups.

Language model Performance with set P_{mixed}

The Internet doesn't contain Zulu strings only, therefore the language model's performance was investigated on mixed documents. The mixed documents data-set contained randomly ordered Zulu and English sentences. It was known before-hand that it contained 40% Zulu and the rest English. This particular data-set tries to figure out if the language model will behave differently given mixed language input. Figure 4.3 shows that it was able to discriminate Zulu and non-Zulu. The data-set behaved the same as it would

if the languages were separated. The accuracy increased for longer sentences for both languages. This meant that accuracy obtained for homogeneous documents applied to mixed documents

LANGUAGE MODEL PERFORMANCE WITH SET $P_{Italian}$

The language classification is accomplished using a Bayesian classifier. The classes composing the Bayesian classification are Zulu and non-Zulu(English). It was then tested with the Italian corpus *Paiśa* to observe how the identification system handled different languages. The accuracy was lower compared to that of Zulu sentences and English sentences. Figure 4.3 shows that the longer sentences resulted with higher accuracy as expected. The reason is that the ngram distribution becomes less similar to that of of the two classes and therefore classified as non-Zulu. An accuracy of 87.6% means that out of 10 million Italian words, 1.2 million of them will be classified as Zulu!

4.5 FOCUSED CRAWLING RESULTS

The human subjects were given 3 sets with the descriptions below.

Set one Documents were crawled from a single seed website {zu.wikipedia.org}. The HTML tag:lang was not limited. If a document had more that 20 Zulu words, it was classified as relevant and all its links visited. The set contained 3879 crawled documents.

Set two Documents were crawled started from 2 seed websites {zu.wikipedia.org , isiZulu.news24.com }. The HTML tag:lang was limited to 'zu' and 'en' or anything if it wasn't set. If a document had more that 20 Zulu words, it was classified as relevant and all its links visited. The set contained 5825 crawled documents.

Set three Documents started from 2 seed websites {www.jw.org/zu/,www.wordpocket.com/zu/index.htm}. The HTML tag:lang was limited to "zu" and "en" only. If a document with lang="zu" has 20 Zulu words it was considered relevant. If a document with lang="en" has 50 Zulu words it was considered relevant. The set contained 64001 crawled documents.

Table 4.2: Table showing results for documents considered to contain Zulu

User no	Set one(10)	Set two(10)	Set three(10)
1	3/10	9/10	10/10
2	1/10	8/10	10/10
3	2/10	3/10	10/10
4	2/10	7/10	10/10
5	1/10	8/10	10/10
6	2/10	10/10	10/10
7	1/10	7/10	10/10
8	1/10	8/10	10/10
9	2/10	10/10	10/10
10	4/10	8/10	10/10
Average	1.9/10	7.8/10	10/10

4.6 Discussion

The Internet has a large number of non-Zulu documents therefore it has been hypothesized that if the number of false-positives is reduced then the amount of crawled Zulu documents becomes cleaner i.e small fraction of false-negatives. An important fact to note is that the accuracy of Zulu detection is the same for all three sets below.

Crawl Parameters for Set one

The first set of documents crawled were from *Zulu Wiki*. From the results in Table 4.2, every set of 10 randomly chosen documents didn't have more than 2 Zulu documents on average. Consider what happens if the crawler goes through 10 million Italian documents on the Internet. 1.2 million of them will be classified an Zulu. Considering how much Italian is on the Internet, this event is possible. Regardless of how much Zulu documents the crawler obtained, having 80% of crawled Zulu documents being non-Zulu would not be acceptable.

Crawl Parameters for Set two

In order to avoid other languages that can be encountered, the HTML tag:lang was restricted to "zu" and "en" if it was set. The assumption is that there is probably very little to no Zulu sites using any other lang parameter other than "zu" or "en". There is possibly Zulu sites that have no lang parameter set. In a set of 5825 documents, it was observed that about 8/10 randomly chosen documents contained Zulu. This is a great improvement from 2/10 containing Zulu in a set of 3879 documents.

Crawl Parameters for Set three

Set two parameters showed a considerable improvement in the percentage of Zulu documents from Set one parameters. It was observed that after an extended crawl, Set two got a lot of non-Zulu documents. This was because there is a lot of non-Zulu sites that do not set the tag:lang parameter. This resulted in more false-positives. The crawler was then restricted to "zu" and "en" languages. If a document contained English, it had to have more than 50 Zulu words to be considered relevant. Setting the threshold for the number of Zulu words higher for English documents would reduce the number of false-positives. Out of a set of 64001 crawled documents, not a single one was observed to be non-Zulu.

It is worth noting that the crawler evaluation samples were small and might possible not reflect the crawlers effectiveness. However is has been mention in the background chapter that crawler evaluation is a topic of extensive research and there is no simple way to measure a focused crawlers effectiveness.

5 FUTURE WORK

The is a lot of room for improvement to the language identification system and focused crawler.

5.1 LANGUAGE MODEL IMPROVEMENT

Generating a language profile using Kneser-Ney smoothing might not be the best suited technique for Zulu strings. There are multiple smoothing methodologies such as Modified Kneser-Ney, Kneser-Ney with backoff counts et cetera (Chen and Goodman, 1999). There are more language identification solutions that can be employed on top of the the statistical language model. Analysis of lexical structure, identifying other unique feature variables of a language. The language identification can be continuously improved by investigating alternative methods to problems that have been solved.

5.2 FOCUSED CRAWLER IMPROVEMENT

The focused crawler can be improved to work with other textual formats found on a website such as text files or pdfs. A method to crowd-source seed websites can be implemented. This makes it easier for finding new websites that have recently been developed however not found yet by Google's crawler. A large scale evaluation system can be setup which enables users to have access to crawled documents. This improves the evaluation by working with a large sample of the crawled documents.

5.3 ALTERNATIVE TOOLS AND COMPARISONS

As mentioned earlier, there are multiple corpora alternatives and solutions that can be developed and tested. The effect of different corpora on accuracy can be investigated. The language model can be tested with languages morphologically similar to Zulu such as IsiXhosa or Ndebele.

6 CONCLUSION

The aim of the project was to develop a language identification system to detect Zulu strings, and a focused crawler capable of determining and harvesting Zulu documents on the Internet. Background research has shown that there are multiple techniques and methods to perform language identification and focused crawling. Kneser-Ney smoothing was the methods used in the project and forms an integral part of the language identification. Literature on focused crawling showed that there are multiple techniques that can be used to find topic specific documents.

There were multiple design, implementation and evaluation decisions made during the development of the project. The lack of a well developed Zulu corpus did not provide huge challenge, however showed the possibility that language identification can be accurately performed for resource scarce languages.

The language model developed managed to provide a 98.4% accuracy. Answering the research question *Is it possible to develop a language identification system capable of identifying Zulu strings?* The language identification system developed is capable of identifying 98.4% of provided Zulu documents. It can be concluded that it is possible to develop a language identification system capable of accurately identifying Zulu strings. *Is it possible to develop a focused crawler capable to find and download Zulu documents on the Internet?* More than 60,000 Zulu documents were successfully crawled. From the results of this project it can be concluded that it is possible to develop a focused crawler capable of finding and downloading Zulu documents.

BIBLIOGRAPHY

Google Translate APi. Google Translate API. https://cloud.google.com/translate/docs, 2014. [Online; accessed 21-October-2014].

William B. Cavnar and John M. Trenkle. N-gram-based text categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.

Soumen Chakrabarti, Martin Van den Berg, and Byron Dom. Focused crawling: a new approach to topic-specific web resource discovery. *Computer Networks*, 31(11):1623–1640, 1999.

Stanley F Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999.

Junghoo Cho, Hector Garcia-Molina, and Lawrence Page. Efficient crawling through url ordering. *Computer Networks and ISDN Systems*, 30(1):161–172, 1998.

Erica Cosijn, Ari Pirkola, Theo Bothma, and Kalervo Jarvelin. Information access in indigenous languages: a case study in zulu. *South African Journal of Libraries and Information Science*, 68(2):p–94, 2002.

Ethnologue. Zulu - A language of South Africa. http://archive.ethnologue.com/16/show_language.asp?code=zul, 2009. [Online; accessed 21-October-2014].

Lena Grothe, Ernesto William De Luca, and Andreas Nürnberger. A comparative study on language identification methods. In *LREC*, 2008.

David A Hull and Gregory Grefenstette. Querying across languages: a dictionary-based approach to multilingual information retrieval. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–57. ACM, 1996.

Zipf George Kingsley. *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*. Addison-Wesley Press, 1949.

Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

Filippo Menczer, Gautam Pant, Padmini Srinivasan, and Miguel E Ruiz. Evaluating topic-driven web crawlers. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 241–249. ACM, 2001.

Gordon Mohr, Michael Stack, Igor Rnitovic, Dan Avery, and Michele Kimpton. Introduction to heritrix. In *4th International Web Archiving Workshop*, 2004.

Blaz Novak. A survey of focused web crawling algorithms. 2004.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

Doug Paul. Ngram arpa format. *U.S. Department of Defense Advanced Research Project Agency (ARPA)*, 2006.

Vesa Siivola. VariKn - Language modelling toolkit. http://forge.pascal-network.org/docman/view.php/33/58/variKN_toolkit.html, 2007. [Online; accessed 21-October-2014].

Sebastian Spiegler, Andrew Van Der Spuy, and Peter A Flach. Ukwabelana: an open-source morphological zulu corpus. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1020–1028. Association for Computational Linguistics, 2010.

Tommi Vatanen, Jaakko J Väyrynen, and Sami Virpioja. Language identification of short text segments with n-gram models. In *LREC*. Citeseer, 2010.

ZuluWiki. Ikhasi Elikhulu. http://zu.wikipedia.org/wiki/Ikhasi_Elikhulu, 2014. [Online; accessed 21-October-2014].