Language identification for South African Bantu languages Using Rank Order Statistics

Meluleki Dube

Department of Computer Science
University of Cape Town
South Africa

June 2018

Abstract

A lot of research has been done in the field of language identification. However, only a small proportion of these methods used for classification have been tested with the Bantu languages. In this paper we then look at one of the methods, and how it performs for the Bantu languages. The method used in this research is n-gram counting using rank orders. Using this method, we investigated how varying the testing chunk size and learning size affected the accuracy of correctly identifying the languages. The highest accuracy obtained was 99.3% with testing size of 495 characters and learning size of 600000 characters. The lowest accuracy obtained was 78.72% when the testing size was 15 characters and learning size was 200000 characters.

Keywords N-grams, N-fold cross validation, Rank Order statistics

1. Introduction

Language identification is the process of automatically determining the language of any electronic text (McNamee, 2005). This automation of determining languages that electronic text is written in, is an important problem when processing natural languages (McNamee, 2005). This is because steps taken after being able to identify the language are sometimes language dependant thus the need be able to identify the language (Duvenhage, et al., 2017) It is currently a hard task to identify text written in Bantu languages due to the lack of support for these languages on the Web. This, in turn, has led to the Bantu languages being excluded from services like translation services and voice synthesis services and also means that processing of these languages is made harder. Even though there are currently technological advancements with increased open information promoting multilingualism on the Web, some languages including the Bantu languages, are excluded from the benefits of these advancements (Chavula & Suleman , 2016).

The South African Bantu languages used in this paper include: isiZulu, isiNdebele, isiXhosa,siSwati, Setswana, Sesotho, Pedi, Tsonga and Venda. The first four (isiZulu, isiNdebele, isiXhosa and siSwati) all belong to the same family, which is the Nguni languages (Barnard & Botha, 2012). Most of these languages share a lot of vocabulary and Botha, et, al tried to quantify the measure by which these languages differ (Botha, et al., 2008).The results gave an indication as to how these languages are in actual sense, similar. The purpose of this paper is to determine if a computer system is still able to distinguish among these languages so as to be able to identify text written in these languages, given that they share a lot of vocabulary.

There are currently different techniques to address this problem. These include: Naïve Bayesian, normalized dot-product, centroid-based, and relative entropy and n-gram counting using rank order statistics(Barnard & Botha, 2012). In this paper, the latter is used to see if we can come up with a computer system which uses n-gram counting using rank order statistics to distinguish between the South African Bantu languages that have a lot of vocabulary similarities. The experiments conducted will evaluate the accuracy that can be achieved by the system. The accuracy depends on three factors:

1. The Testing size is one factor. Decreasing testing size text may decrease the accuracy of the system as was the case for (Barnard & Botha, 2012). However, most applications that employ language identification are concerned with identifying the languages for small amounts of text (Botha, et al. 2006). For example, if a social media site like Facebook were to prohibit bad language and bullying on their application, and they checked every post by all of their users to see if it contained any of the above mentioned, they

will first need to be able to identify the language that the posts are in. Secondly, most of the posts on social media are not made up of a large number of words, they are likely to contain posts with the number of characters below 1000. Therefore, when performing language identification for such posts, the testing data size will usually be small and thus this paper looks into the different accuracies we have for different test sizes with a language identifier that uses n-gram counting using rank orders.

2. The training size is another factor. Decreasing the size of the training text could also in turn decrease the accuracy of the system. Decreasing the training data set will mean excluding some part of the text from being used to train the system. This means that there is not much diversity left in the remaining text hence there is not the full variety of words within the training dataset. This then means that the system will be expected to generalize less successfully (Botha, et al., 2006). We, however, need the system to perform well for small training data sizes of hundreds of thousands of characters because, if the system is going to be implemented for all the Bantu languages, it will be hard task to get text corpora with above millions of characters for all the South African languages as some of these languages do not have much content available online.
3. The accuracy will also be affected by the similarity between the languages. The effects of these were evaluated by Botha, et al.,(2006) where they attempted to come up with a text-based language identifier for the eleven South African Languages. They used n-gram statistics as a feature for classification together with Support Vector Machine (SVM).

The size of testing and training dataset is calculated as the number of characters present. The methodology implemented is adapted from Cavnar and Trenkle, (Cavnar & Trenkle, 1994). The ultimate goal is to find out the optimum combination of the testing data size and training data size that will yield the optimum accuracy for the system. In Section 2, we present the different works that have already been done in the field of language identification and the different techniques that can be employed. Section 3 focuses on the methodology and experimental design for this paper. Section 4 presents the results and the analysis of the results and Section 5 is the conclusion and future work.

2. Literature Review

2.1 Theoretical Background

As stated above, language identification in this paper is taken as the automation of determining the language that a certain electronic text is written in (McNamee, 2005). This is a problem that has been addressed by a lot of people for the most major languages. Not much of the research conducted has focused on the Bantu languages. Notable works that looked at language identification with a focus on Bantu languages include the works by (Barnard & Botha, 2012) which looked at the factors that affected language identification with a focus on the South African languages. It therefore also included English and Afrikaans. Another project on language identification with a focus on South African Bantu languages presented a statistical approach to text-based automatic language identification, which focused on discrimination of language models as opposed to the representation used (Combrinck & Botha, 1995). The Bantu languages used in this research included isiZulu, isiXhosa, Tswana, Sepedi and Swazi (Combrinck & Botha, 1995)

2.2 N-gram Count Using Rank Orders

N-gram is a letter of size n which is a subsequence of characters from a word with the value of n usually being 1, 2, and 3 (Balone, et al., 2016). These special forms of n-grams are usually called unigrams for when n is 1, bigrams when n is 2 and trigrams when n is 3 (Balone, et al., 2016). e.g. trigrams for the Ndebele word, *'bhala'* (*'write'*) are *bha, hal, ala*.

(Cavnar & Trenkle, 1994) devised an approach for text categorization which is tolerant of textual errors, achieving in one of the tests 99.8% accuracy (correct classification rate). The system also achieved 80% accuracy rate in classifying articles from a different computer-oriented newsgroup (Cavnar & Trenkle, 1994).The approach taken by Cavnar and Trenkle is adopted in this research to investigate whether for classifying Bantu languages it is possible to achieve high accuracy ratings. The system entails calculation and comparisons of profiles of N-grams frequencies, where a profile of N-gram frequency refers to the n-gram frequencies ranked in descending order (Cavnar & Trenkle, 1994). The reason this is possible is due to Zipfs, principle of least effort, (Zipf, 1965), which states that the frequency of occurrence of a word is almost equal to the inverse of its rank (Li, 1992). This then suggest that for 2 separate texts in a particular language, say Ndebele, both will have almost the same n-gram distribution. Thereby if we can compare 2 profiles and get that they are not far apart by some measure, there is then a high probability that the 2 profiles belong to the same classification groups. The comparison that were performed by (Cavnar & Trenkle, 1994), was measuring the distance between 2 profiles. The exact method of computing the distances is described within the methodology section as this is the same method used in this study.

2.3 Naïve Bayesian Classifier

The naïve bayesian approach involves writing a statistically based program which is capable of learning to distinguish between languages(Dunning, 1994). The classifier is based on the hypothesis that the role of a natural class is to predict the values for members of that class using the idea that if an agent knows the class it is then capable of predicting the values of the other members and if it does not know the class, in that case then Bayes rule will be used to predict the features of the value (Poole & Mackworth, 2017). With this approach,a small amount of data is needed for both training and testing(Dunning, 1994).The variant of a program such as the one used in the naïve bayesian classifier is in biochemistry in the application of genetic sequences, thus equating language classification to species application (Dunning, 1994).This method of language identification was then implemented by Barnard & Botha(,2012) where they aimed to determine the factors affecting the accuracy of identification of text-based langauges. The languages that were of focus in the study were the eleven official South African langauges. N-gram statistics were still used as features in the study (Barnard & Botha, 2012).The 11 official languages include the nine Bantu languages that are of interest in this research. The study showed that the size of the test chunk is one of the factors affecting the accuracy of text-based language identifiers as the results state that an accuracy of 99.4% was achieved for a chunk size of 100 characters and above while an accuracy of 83% was achieved for a test size of 15 characters. These results concur with the statement by (Dunning, 1994), that such will work moderately well for a testing size of 10 characters and will work very well for a testing size of 50 and above. Thus, one of the investigations in this research will focus on how the size of testing chunk size in terms of number of characters will affect the accuracy of the system in order to come up with an optimum test size.

2.4 Support Vector Machines

Support Vector Machine (SVM) first introduced by Boser, Guyon and Vapnik is a non-linear discriminant function that performs classification in a high dimension space(Botha, et al., 2006). This technique can use n-gram statistics, which are contained in a vector for each of the samples. The disadvantage of this method is the fact that the training of the system takes many hours as the size of test data is increased. This is due to the exponential growth in the feature space(Botha, et al., 2006). Therefore, if large test data sizes are being used, the algorithm will not be desired for situations where time is of essence.

2.5 Vocabulary Similarities within the Bantu Languages

As already mentioned in the introduction, most of the South African Bantu languages can belong to the same families. For instance, isiNdebele, isiXhosa, siSwati and isiZulu all belong to the same sub-family which is, Nguni languages (Barnard & Botha, 2012). This then has implications that these languages will share a lot of vocabulary and are very similar. This was proven by Botha, Barnard and Zulu(2008) where they investigated if it is possible to quantify the extent to which the South African languages differ. The results for these languages, where when 50 words were used to measure, were that Ndebele and Zulu had a distance of 232 between them, Ndebele and Xhosa had a distance of 279 between them(Botha, et al., 2008) and Ndebele and siSwati had a distance of 257 between them. Zulu and Xhosa had a distance of 276 between them(Botha, et al., 2008) and Zulu and siSwati had a distance of 194 between them. These numbers can be compared to other results for other languages that do not belong to the same family as these . For example English to Ndebele was measured to have a distance of 437, English to Zulu was measured to have a distance of 444, English to siSwati was measured to have a distance of 450 and English to Xhosa was measured to have a distance of 437(Botha, et al., 2008). It can also be seen that even for languages outside their group, the languages still have almost the same differences with them.

3. Methodology

As mentioned above, the methodology employed in this paper is adapted from the work by Cavnar and Trenkle(1994). Given that their system managed to have a maximum accuracy of 99.8% for classifying languages, the goal is to find out if we can have a combination of the training size and testing size that can result in this accuracy. The resulting system was then implemented in java.

Language models were created to represent the different languages used in the experiment. An extra model for testing was created. Language Models also referred to as Category profiles by Cavnar and Trenkle (1994), are simply the rankings of the frequency counts of the n-grams of the corpus for that language. These rankings are in descending order meaning that the most frequent n-grams will appear on top of the list while the least frequent n-grams appear at the bottom of the list. This was so, because the least frequent n-grams that appear at the bottom of the list will usually be a result of misspelt words and words with grammatical errors that do not represent the languages and can be discarded. Cavnar & Trenkle, (1994) described that since Zipf's law can be restated to say that the word that is the

n^{th} most common word in a language will occur with a frequency that is inversely proportional to n, the implication will then be that for a given language, there will be a set of words that will dominate most of the other words for each language in terms of the frequency in which they are used. Further, if a distribution graph of the n-gram ranks/words frequency can be drawn, the graph will have a smooth continuum of dominance from frequent to least frequent which can help to select parts of the language to disregard when processing the languages (Cavnar & Trenkle, 1994). These are the parts that we stated earlier that are as a result of spelling errors, grammar errors or just that they are not really important in the language. The threshold rank order in their paper was 300 characters, meaning that any n-gram that has a rank greater than 300 will be disregarded. The 9 South African Bantu languages where then tested to see if this law. To do so a model sizes of 100000 where used for all the languages that is 100000 characters weere taken from each of the corpus. A list of trigrams grams was produced for each of the language and their frequencies were computed to determine how may times each trigram occurred in the text corpus. The result of the frequency distributions are shown in Figure 1.

As Figure 1 shows, the nine South African Bantu languages used in the experiment obey the law stated by Zipf, and also the extension to it added by Cavnar and Trenkle(1994). This then allows for the experiment done by them to be adapted and still leave the 300th rank as a threshold rank beyond which we do not consider the n-grams.

3.1 Chosen technique for language identification

The chosen procedure for language identification uses n-gram counting using rank orders. The flow chart of how the system will work given below is taken from the paper presented by Cavnar and Trenkle(1994). The flow chart for the procedure is shown by Figure 2.

Category samples in this case are the South African Bantu languages that are going to be supported in this system. Therefore, the possible category that a testing file could fall into will be one of these languages. The new Document will then be the data for the testing sample. The profiles and mentioned above are in this paper referred to as the models. Therefore, we will have the testing models, which is the model that consists of ranked n-grams from the testing data sample represented as the document profile in the above flow chart. We will also have the language models which are the different category models which will also have the n-grams in their rank order. Below we explore how the different steps of the flow chart were conducted to come up with the results.

3.2 Generation of the different Models

To generate the models, the same procedure was applied for both the language models and the testing models. Since the goal of the experiment required varying the model sizes we needed to keep information to know when to stop adding more data to the models. The steps below describe how this was completed:

1. While there is still something to read from the file and the model limit has not been reached:
   (a) Read the entire line from the file
   (b) Break the string read from the line into the different characters
   (c) Create a temp variable,
   (d) While the number of character used to create the model is less than or equal to the limited number:
      i. Add the character to the temp variable to create a larger string
      ii. Move to the next character
   (e) Create the different n-grams from using the string temp variable.
   (f) Add the different n-grams to a Hash-map stored in the respective profile class on updating the count value for the different n-grams.

After creation of the different profiles, we can then compare the profiles to one another by means of measuring the distance between the profiles.

3.3 Measuring of the profile distances

This is the decision process, were given a testing model, we can match it with one of the language models to determine which of the models have the least distance between it and the testing profile. The distance between two language models according to Cavnar and Trenkle (1994), is how far apart the two models are from each other. This calculation is presented below in Table 2. The distance represented as Max indicates an n-gram that was found in the testing model but could not be found with a range of the top 300 n-grams in the language model. These maximum values in the experiment are replaced by the maximum difference, which is 300. The procedure to calculating the distance is given in Algorithm 1: This is repeated comparing all languages to the testing profile.
Figure 3 shows how the the distance are going to be calculated.

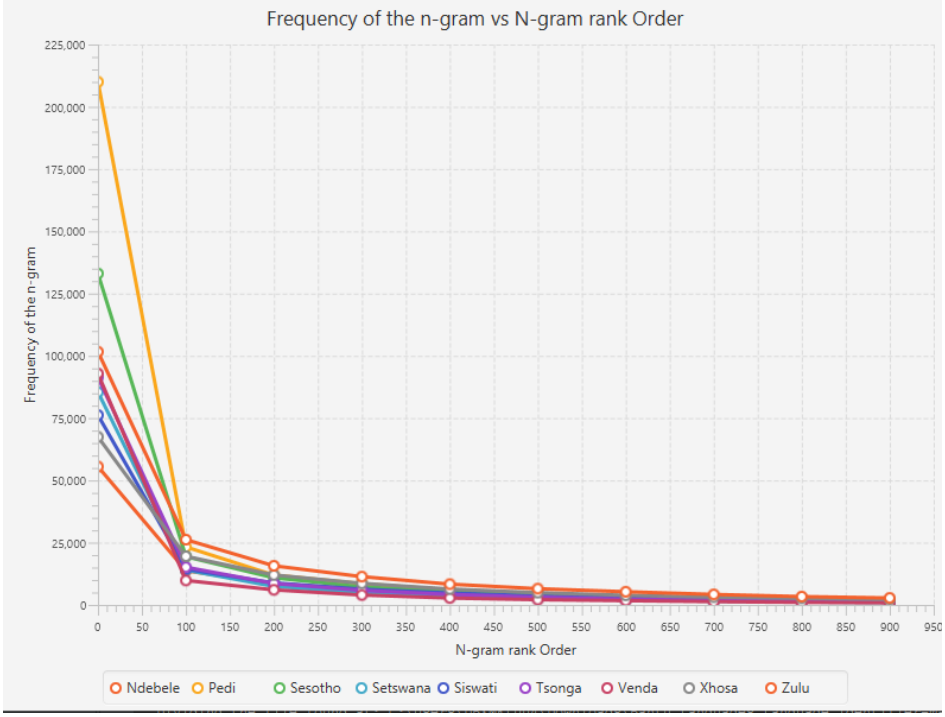

Figure 1. Showing the distribution graph for the different n-gram ranks and their frequency at that rank.

```
1:  procedure MEASURE PROFILE DISPLACEMENTS
2:      distance ← max int
3:      function PROFCOMPARE(Profile1, Profile2)
4:          tempDistance ← 0
5:          counter ← 0
6:          if sizeOf profile1 < 300 then
7:              limit ← sizeOf profile1
8:          else
9:              limit ← 300
10:         for counter < limit do:
11:             prof1Ngram ← profile(counter)
12:             i ← 0
13:             found ← FALSE
14:             for i < sizeOf profile2 do:
15:                 if profile1[counter] == profile2[i] then
16:                     found ← TRUE
17:                     tempDistance ← tempDistance + (counter − i)
18:                                                          ▷ Break
19:             if found == false then
20:                 tempDistance ← tempDistance + 300
21:             counter ← counter + 1.
```

Algorithm 1: Algorithm for measuring distances between profiles

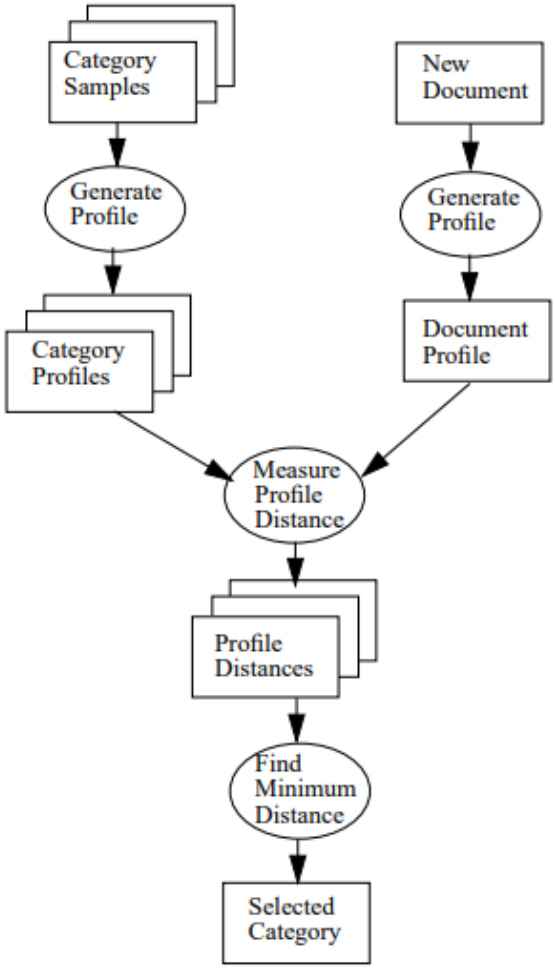

Figure 2. Showing the steps taken to determine a language (Cavnar & Trenkle, 1994)

Trigrams for the testing data that is in isiNdebele arranged in the order of their frequencies (highest to lowest)	**Trigrams for the training data (model for isiNdebele language) arranged in the order of their frequencies (highest to lowest)**	**Out of order number for the model and the testing data given by the absolute value of the difference between rank in mode- rank in testing data**
nga	nga	|0-0|=0
la	oku	|1-2|=1
oku	la	|2-1|=1
a n	ela	Max
ana	ana	|4-4|=0
enz	nam	|5-6|=1
ela	enz	|6-3|=3
		$\therefore distance = 0 + 1 + 1 + Max + 0 + 1 + 3 = 6 + Max$

Figure 3. Showing the distribution graph for the different n-gram ranks and their frequency at that rank.

When all language models have been compared with the testing model, the language model with the shortest distance between it and the testing model will most likely be the language and will thus be presented as the language

3.4 Data Sets

The first step in this research was to gather text corpora for the 9 Southern African Bantu Languages. These were collected from the South African Centre for Digital Language Resources.The data was then manually cleaned to remove the unnecessary descriptions of the individual corpora. The corpora were for all the 9 South African Bantu languages including: isiZulu, isiNdebele, and isiXhosa, siSwati, Pedi, Sesotho, Setswana, Tsonga and Venda. Table 2 presents the number of characters that were present in each of the files. Since the results from this paper are going to be contrasted with the results by Barnard & Botha (2012), we kept the training data size small with the maximum tested training size as 600000 characters used to create the model. This is because according to (Dunning, 1994), a NaÃŕve Bayesian classifier should be able to work well with a training size of as few thousand words for training size.

Language corpus	**Number of characters in file (includes whitespaces)**
isiZulu	13708706
isiNdebele	7337788
isiXhosa	10322727
siSwati	7496674
Pedi	11798404
Sotho	10044806
Setswana	7024305
Tsonga	7591259
Venda	5527459

3.5 Experimental Design

In evaluating the system, N-fold cross validation was employed. Since we only had one corpus for each language, we needed a technique to produce the training dataset and the testing dataset for the different languages. With this technique, the datasets must be divided into N separate parts, where N is a number greater than 2. In this case N was taken to be 10, meaning we had 10 subdivision of the data. The data was divided in terms of number of characters. Therefore, the number of characters in one division was a tenth of total number of characters in the corpus. From the divisions, one of the divisions will be removed before the training begins (Schneider, 1997). Therefore, the system will only be trained using $\frac{9}{10}^{\text{th}}$ of the total corpus and the $\frac{1}{10}^{\text{th}}$ is then left out to be used for testing the performance of the system (Schneider, 1997). With N fold-cross validation, the procedure is repeated N times (Schneider, 1997). At each iteration, one of the subsets will be used for testing; this has to be a subset that has not been used for testing before. The other N-1 subsets will be used to for training the system (Schneider, 1997).

When we are testing if the system will be able to correctly recall a language, for instance, Ndebele, the evaluation of the system performance will be as follows. As mentioned above, there will be 10 iterations of this evaluation process. At the first iteration, the first subdivision for the Ndebele corpus is used for testing and the other 9 subdivisions will be used for training the system to be able to predict Ndebele. The other languages will have their 10 subdivision used for training. On the second iteration, the above process is repeated but now only the 2nd subdivision is used for testing and the rest of the corpus for Ndebele is used for training. This will continue until all the 10 subdivisions have been used for testing during 1 iteration. The number of times the system is able to correctly classify a testing model to the corresponding language is counted. Hence, at every iteration, if the system manages to correctly recall the language, the count is incremented. The system was evaluated in terms of accuracy. To get the accuracy of the system for being able to correctly identify the language represented in the testing profile the following simple formula is then used:

$$Accuracy = \frac{count}{9} * 100 \tag{1}$$

Count is the count variable. The above equation assumes that at each iteration all the dataset was used for testing thus we are dividing by 9 which is the number of iterations that will be made.

3.6 Experiment

The steps, taken in performing the experiment are as follows: Given that the corpus is around 5 million characters, that means $\frac{1}{10}^{\text{th}}$ of that will be around 500 thousand characters. Therefore, from this, we can get many testing profiles that use a large proportion of the testing subdivision. To do this, we read testing size characters at an index stored it into a list, then skipped 2000 characters and read more testing size characters. This was repeated 100 times. The sequences of chunks saved to the list were then used to create the testing models for the language. When comparing the testing Profile with the other profiles, All the profiles created using the Strings in the lists will be compared with each language and the average accuracy for that testing size will be computed and saved as described by Algorithm 2. Since there are 100 training profiles each going to be iterated changed 9 times using N-cross validation,

1: **procedure** CARRYING THE EXPERIMENT
2: *Divide each language corpus into 10 subdivisions*
3: *i ← Subdivision to be used for testing*
4: *accuracy* ← 0 ▷
i is a number either 1, 2, 3.....,9, 10 which are the different subdivision ofthe corpus dat was divided to 10 segments
5: *Select language for testing*
6: *Select training data size*
7: *Select testing data size*
8: *Generate language models except for the language being used for testing. Limit the number of character to the model size*
9: *Generate language model for the language being used to test the system with number of character limited to the model siz*
▷ *This time leave out the ith subdivision as its. going to be used for testing*
10: *Generate the testing models for the language being used to test the system using the ith subdivision* ▷
How these are created is explained above
11: **for** $testingProfilecreated$ **do**:
12: *compare it to the language profiles*
13: *Find language profile with least distance*
14: **if** *Language profile with least distance is language used to test* **then**
15: $accuracy \leftarrow accuracy + 1$
16: *Calculate the accuracy of the system using Equation (2)*
17: *Return the Accuracy value calculated.*

Algorithm 2: Algorithm for carrying the experiment

The accuracy of the system can be calculated using the equation:

$$Accuracy = \frac{accuratePrediction}{900} * 100 \quad (2)$$

Algorithm 2 will be repeated for the different model sizes and for different test data sizes. In this experiment, the test size was varied from 15 characters to 510 characters and the training data size was varied from 100000 characters to 600000 characters.

4. Results and Analysis

The results obtained from the experiments are presented together with the discussion of the observed behaviour of the system. We first discuss the general performance of the system, and then look at how the system performed for some of the individual languages. The reason for choosing some of the languages and not all is because the same behaviours were seen with all the languages and thus we discuss the language with less accuracy, language with high accuracy and the language with a median accuracy obtained.

4.1 Average performance

The maximum achieved accuracy was 99.3%. This was when the data used to create the language model sizes was 600000 characters and the data used to create the testing chunk size was 450 characters. The minimum accuracy achieved on the other hand was 78.72% which was obtained when the training data size was 200000 and the testing chunk size was 15 characters.

The results, to a certain extent, adhere to the hypothesis. Looking at the graph presented in Figure 4, it can be seen that both increasing the testing data size and the data size for creating the models will effectively increase the accuracy of the system. The increase in accuracy due to increase in training data size is seen in the shift to the right of the different graphs. The system never reaches a 100% accuracy and this the 100% accuracy line acts as an asymptote of the graph. However, after a certain point in time, increasing the testing data size becomes not useful as there is no increase in the overall accuracy of the system. Therefore, the combination that yields the optimum results will be taken as the combination that yields the maximum accuracy.

The research by Botha, Barnard and Zulu(2008), can be used to explain the low results obtained when the model size and the test size were all at their minimum used in this experiment. Consider the matrix shown below which shows the number of times the system mistakenly recalled one language as the other for when the model size was 100000 and testing models size was 15.

As seen from Figure 5, the reason for the low average accuracy for the system is the failure to properly identify some of the Nguni languages, in particular Ndebele. The system managed to identify Ndebele 57.67% within the 900 trials. While other languages achieved 81.89% and above accuracy for the same testing chunk size and model size.

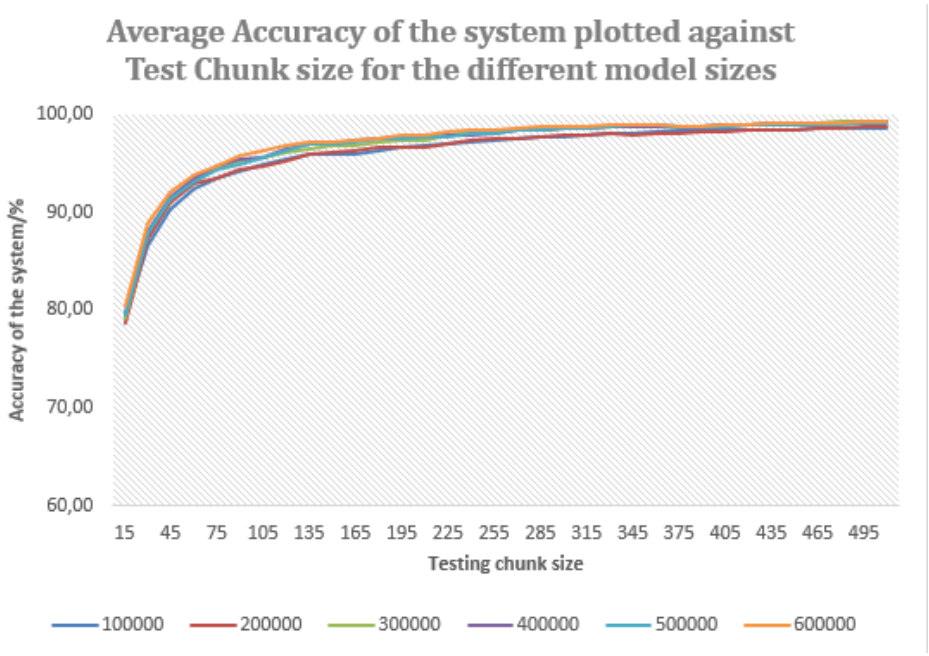

Figure 4. Showing the graph of the average accuracy of the system plotted against the Testing chunk size for the different training data sizes.

	Ndebele	Pedi	Sotho	Tswana	Swati	Tsonga	Venda	Xhosa	Zulu
Ndebele	519	1	1	3	16	10	5	98	247
Pedi	3	786	22	80	4	2	1	2	0
Sotho	9	7	783	90	5	0	2	2	2
Tswana	0	51	100	737	1	5	2	0	4
Swati	40	3	4	2	788	6	3	27	27
Tsonga	11	2	2	9	8	854	11	0	3
Venda	4	1	2	1	2	10	873	3	4
Xhosa	84	1	5	1	41	6	5	519	238
Zulu	105	1	2	1	63	2	3	156	567

Figure 5. Showing predicted languages when different languages were used as testing model the first row is for the predicted languages and the first column reflects the languages used for testing. Testing data size used was 15 characters and training data size used was 100000 characters.

This behaviour also persisted for larger training data sizes. This can be seen in the next section where we discuss the system performance when the training data data was in isiNdebele. As already stated, the low accuracies are mostly due to failure to correctly classify the Nguni languages, which are very similar. This is because, the testing size for these languages are relatively small. Since they share a lot of vocabulary, it becomes hard to tell them apart

4.2 System performance for the individual languages

4.2.1 Ndebele

Figure 6, shows a graph depicting the performance of the system in classifying a testing model that was made up of the Ndebele language.

As can be seen the graph displays the same behaviour displayed for the Average system performance. However, the graph starts at a lower point, that is for a testing chunk size of 15, while the average graph has accuracies around 78.9% for all the different model sizes, now for the different models sizes, we have an accuracy

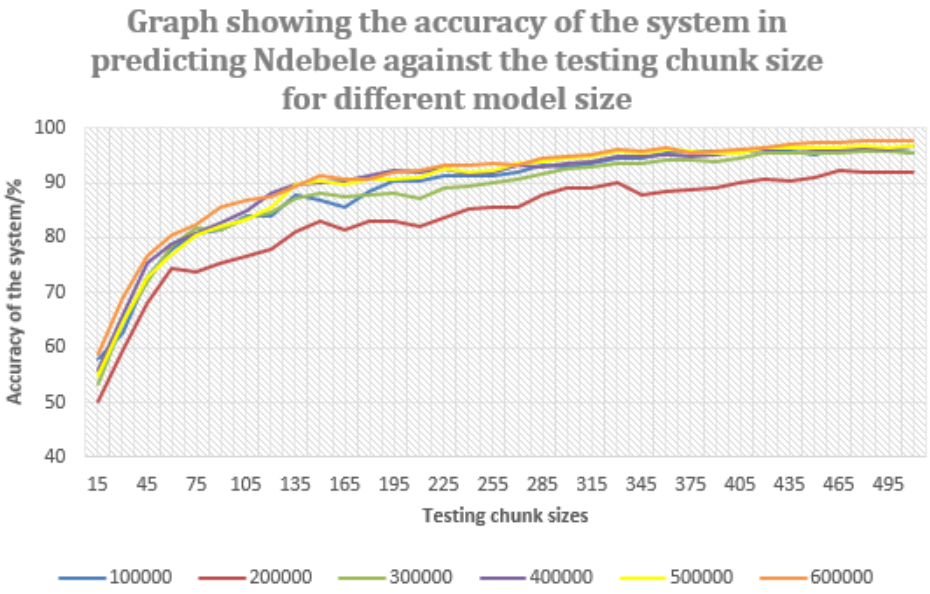

Figure 6. Showing a graph of how system accuracy plotted against testing chunk size for when the testing model was in Ndebele.

that is around the of 58.9%. Thus, the system seems to have had a hard time trying to predict Ndebele for a small testing chunk size. The reason for this, as already stated in the hypothesis, and when analysing the average system performance, is due to the shared vocabulary between Ndebele and the other Nguni languages. The highest achieved accuracy for correctly classifying a testing model in Ndebele was 97.78% for a testing chunk size of 495 and a training size of 600000 characters. The lowest accuracy was 50.11% obtained when the system testing model size was 15 characters and the model size for the languages was 200000.

The general trend of the graph, still shows that for increased testing model sizes and for increased model size for the languages, the system has better performance. This low performance, in turn affected the overall performance of the system as it can be seen during the analysis that classifying other languages did not have such mediocre performance. The same behaviour is true for when the testing model was representing Xhosa

4.2.2 Zulu

The system had a rather better accuracy when it came to correctly classifying a testing model that was in isiZulu.The highest accuracy for classifying a testing model in Pedi was 99.89%, which was achieved for different combinations of testing model size and the model size for the languages. The lowest recorded accuracy was 62.78% for when the testing model size was 15 and the models size for the languages was 300000. This was the average performance the system had for predicting the languages. Again the graph shows behaviour that where hypothesized already. Figure 7 is a graph depicting the above statements.

4.2.3 Tsonga

The best performance of the system was for when the testing chunk was in Tsonga. The system had a lowest accuracy of 94.89% when the testing chunk size was 15 and the learning size 100000. The highest accuracy was 100% for all different combinations of testing size and learning data size. Therefore, the system can easily classify Tsonga as compared to other languages. Venda, and Pedi had graphs similar to the Tsonga Graph.

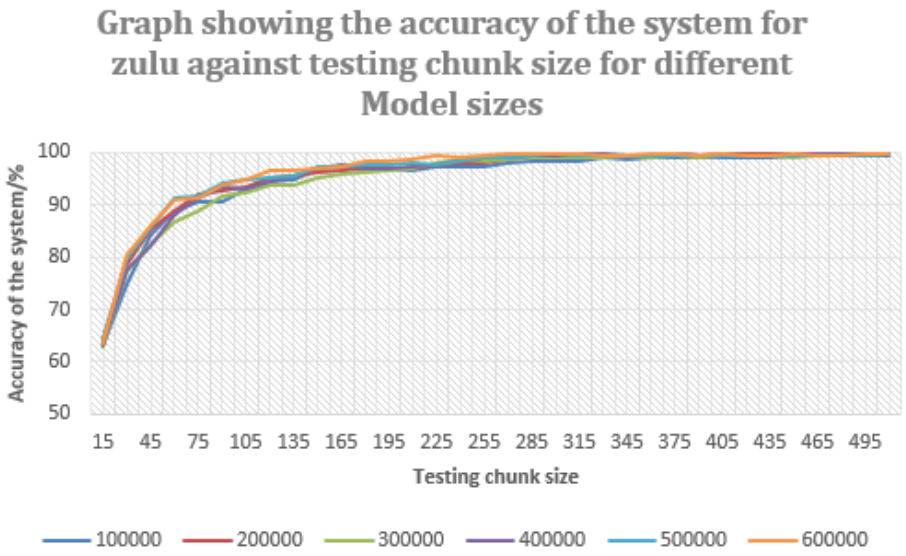

Figure 7. Showing a graph of how system accuracy plotted against testing chunk size for when the testing model was in Zulu.

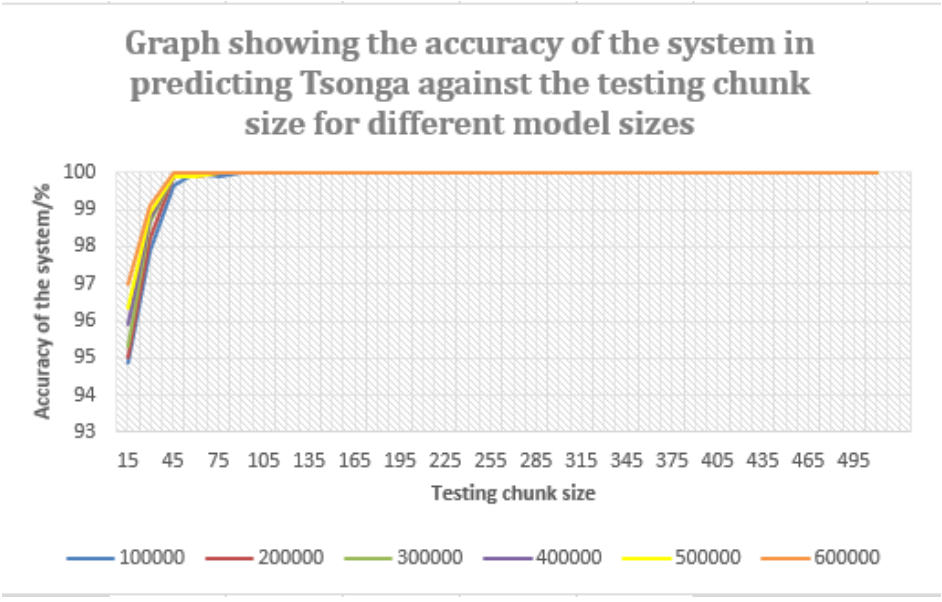

Figure 8. Showing a graph of how system accuracy plotted against testing chunk size for when the testing model was in Tsonga.

5. Conclusions and Future Works

In this paper, we set out to investigate if we can come up with a language identifier system for the Bantu languages using the method of counting n-grams using rank orders as an identification technique. The main goal was to come up with a combination of testing chunk size and training data size that would give the optimum accuracy values. To do this, we adapted the procedure used by Cavnar & Trenkle(1994) to design the system. We then used N-cross validation to validate and test the system. The result from the experiment confirms the hypothesis made earlier. Since the South African Bantu languages share a lot of vocabulary, this will affect the system performance as it will be harder to tell the languages apart this was proven by the presented languages with much emphasis on the Nguni languages which had low accuracy rate recorded, excluding siSwati.

However, to answer the main research question, it turns out that Yes we can have a language identifier system that is based on n-gram counting using rank order statistics. When the size of training dataset is reduced, the results showed that the accuracy for most cases also reduced due to there being fewer data for learning and lack of variety of words in the learning language. However, this accuracy was still above 75%, meaning that 3 out of 4 times, the system will be able to tell the languages apart.

Finally, it was also seen that decreasing the testing data size, also for most of the cases, decreased the accuracy of the system as some of the testing data may appear many times in other languages since, as already mentioned, the South African Bantu languages have a lot of vocabulary similarities. The system works best if the testing chunk size is 495 and the training data size is 600000.

In future works, the Naïve Bayesian classifier method and SVM method of identification used by Barnard and Botha (2012) can be employed to see if machine learning algorithms can have a better performance if they are to be tested with only the South African Bantu Languages.

Acknowledgements

Many thanks to Professor Hussein Suleman for the guidance given throughout the research project.

References

T. W. F. . C. I. Agency. (n. *d.).Retrieved March*, 14, 2018.

N. Balone, H. Suleman, M. C. Keet, and L. Khumalo. The effects of a corpus on isizulu spellcheckers based on n-grams. *Durban.Retrieved March 2018*, 14, mar 2018. URL `http://www.meteck.org/files/afrispeIST16crc.pdf\`.

T. C. Bell, J. G. Cleary, and I. H. Witten. *Text Compression*, pages 77–98. T. C. Prentice Hall, Englewood Cliffs, N. J., 1990.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. *A training algorithm for optimal margin classifiers*. Proceedings of the fifth annual workshop on Computational learning theory; ACM Press, Pittsburgh, 1992.

G. Botha, V. Zimu, and B. Etienne. November). *Text-based Language Identification for the South African Languages.98(4)*, 98(4):141–146, 2018a. URL `http://hdl.handle.net/10204/951`.

G. R. Botha and E. Barnard. Factors that affect the accuracy of text-based language identification. *Comput. Speech Lang.*, 26(5):307–320, October 2012. ISSN 0885-2308. doi: 10.1016/j.csl.2012.01.004. URL `http://dx.doi.org/10.1016/j.csl.2012.01.004`.

G. R. Botha, E. Barnard, and N. P. Zulu. July 25). *Orthographic measures of language.Literator*, 29(1):185–204, 2018b.

W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. *roceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval.Retrieved*, 3, 2018.

C. Chavula and H. Suleman. *Assessing the Impact of Vocabulary Similarity on Multilingual Information Retrieval for Bantu Languages*, pages 16–23. Proceedings 8th Annual meeting of the Forum on Information Retrieval Evaluation (FIRE 2016). ACM New York, Kolkata, 2018. URL `http://pubs.cs.uct.ac.za/archive/00001161/01/FIRE_2016_paper_22.pdf`.

H. P. Combrinck and E. C. Botha. Text-based automatic language identification. *Proceedings of the 6th Annual Symposium of the Pattern Recognition Association of South Africa.Retrieved May*, 09, 2018. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.46.3801&rep=rep1&type=pdf`.

M. Damashek. February 10). *Gauging Similarity with n-Grams: Language-Independent Categorization of Text.Science*, 267:843–848, 2018. URL `http://www.jstor.org/stable/2886144`.

T. Dunning. March 210). *Statistical identification of language.pp.94-273.Retrieved May*, 08:94–273, 2018.

B. Duvenhage, M. Ntini, and P. Ramonyai. *Improved text language identification for the South African languages*. Pattern Recognition Association of South Africa and Robotics and Mechatronics; IEEE, PRASA-RobMech; Bloemfontein, 2017.

W. Li. November). *Random texts exhibit Zipf's-law-like word frequency distribution.IEEE Transactions on Information Theory*, 38(6), 1842.

P. McNamee. February 01). *Language identification: a solved problem suitable for undergraduate instruction.Journal of Computing Sciences in Colleges*, 20(3):94–101, 2018.

V. (n Jakkula. d.). *Tutorial on Support Vector Machine (SVM).Retrieved May*, 10, 2018.

D. L. Poole and A. K. Mackworth. Artificial intelligence: Foundations of computational agents. *Cambridge University Press.Retrieved*, 05, 2018. URL `http://artint.info/2e/html/ArtInt2e.html`.

D. K. Rycroft. Tone-patterns in zimbabwean ndebele. *Bulletin of the School of Oriental and African Studies, University of London*, 46:77–135, 2018. URL `http://www.jstor.org.ezproxy.uct.ac.za/stable/616704`.

J. Schneider. February 7). *Cross Validation.Retrieved*, 3, 2018. URL `http://www.cs.cmu.edu/~schneide/tut5/node42.html`.

G. K. Zipf. *Human behavior and the principle of least effort.* G. K. Hafner Publishing, Zipf, Human behaviour and the principle of least effort. New York, 1965.