UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

COMPUTER SCIENCE HONOURS FINAL PAPER 2015

Title: AFRICAN LANGUAGE SPELLCHECKER

Author: BALONE NDABA (NDBBAL001)

Project Abbreviation: AFRISPEL

Supervisor: PROF. HUSSEIN SULEMAN

Category	**Min**	**Max**	**Chosen**
Requirement Analysis and Design	0	20	0
Theoretical Analysis	0	25	0
Experiment Design and Execution	0	20	20
System Development and Implementation	0	15	10
Results, Findings and Conclusion	10	20	20
Aim Formulation and Background Work	10	15	10
Quality of Paper Writing and Presentation	10		10
Adherence to Project Proposal and Quality of Deliverables	10		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	0	10	0
Total marks		**80**	**80**

Afrispel: An isiZulu spellchecker

Balone Ndaba (NDBBAL001)
University of Cape Town
Rondebosch, Cape Town, 7700
balonendaba@gmail.com

ABSTRACT

A piece of writing should have good content accessibility and readability for readers. This is done by using correct spelling. With little to no spellcheckers to support writing in African languages, we completed a project aimed at building an IsiZulu spellchecker. The spellchecker was developed using a data driven statistical language model. The system was evaluated on accuracy and speed. The system comprised of the pre-processing, error detection and post-processing stage. To test the system, a word was fed to the system and taken through appropriate stages to predict whether it is correctly spelled or not. There were three corpora to test the system. The corpora were tested on four thresholds. For both trigrams and four-grams, the 3 corpora gave accuracy rates above 50% for all thresholds when tested using n-fold cross validation. One of them contained old lexical content and two contained relatively new content. Testing the corpus containing old lexical content with contest from the corpora containing relatively new content gave accuracy rate above 50% only for the two smallest thresholds. Testing the latter with content from the former gave accuracy rates below 50% for all thresholds. The system performed well for most frequent words in the corpora. The results also showed that spellchecker training data should be changed as language expands. Testing a spellchecker that was trained with outdated data with outdated content gave positive results but it performed poorly with data containing newly emerging words in the language. This simply meant that spellcheckers should be updated as language evolves. The spellchecker performed better with trigram than with four-grams. The system was implemented using Java and MySQL was used to manage data access in the system.

1. INTRODUCTION

1.1 Problem statement

The aim of the project is to have a fully functional spellchecker. The spellchecker should be able to predict whether an IsiZulu word is wrongly spelled or correctly spelled. Whether a user provides a single word or a sentence the system should be able to help. Spellcheckers are used in word processing programs, emails, cellphone, blogs, forums and several other computer applications. Emerging computer technologies support languages other than English but they have limited support for African languages including isiZulu. The large number of people speaking, using or learning these language are disadvantaged. Spell checkers are a key component that is needed to move towards greater levels of multilingual support in word processors, search engine interfaces and optical character recognition systems. If a computer application can be run in any language, then spellcheckers are needed to support such applications. IsiZulu is not an exception here. Good spelling can also improve a web page and increase its ranking. Correct spelling will make a good impression to the page viewers and will increase online credibility for the page.

1.2 Proposed system

Some spell checking systems use the dictionary partially, for instance, extracting n-gram statistics [14] and some do not use a dictionary at all [8]. The proposed system here is the former. It uses a data driven statistical language model. The only way to run the system is to associate it with some data. The decision taken on a word, whether it is wrongly or correctly spelled, is not based on personal intuition but on data. The lexical content of a dictionary or corpus drives the spellchecking process. The system will have a corpus of words. Statistics will be computed from the corpus that will guide the decision of whether a word is correctly spelled or wrong spelled.

2. LITERATURE REVIEW

In this section we will take a look at the existing spellcheckers developed using a data driven statistical language model. We investigate the types of errors and the way these spellcheckers detect and correct errors. Spelling error detection and correction techniques are designed on the basis of different kinds of errors. Studies were previously performed to analyze various spelling error types [7].Some authors split spelling errors into non-word errors, isolated-word errors and real-word errors. These errors occur as typos or when the pronunciations of a misspelled word are assumed to be of an intended correct word [3].

2.1 Spelling errors

There are two main type of errors most spellcheckers aim at finding, which are non-word errors and real-word errors. Non-word errors are spelling errors resulting from words that do not appear in the reference dictionary and real-word errors are words that are in the reference dictionary but are actually erroneous spellings of some other words [17]. A spelling checker will detect a misspelled word and, depending on the level error, fine tune the word to provide a set of suggestions [13]. These suggestions are a set of words a user probably intended to type. Non-word errors are relatively easier to detect and eradicate. Real word errors are more intricate ones. Usually, such an error affects the syntax and semantics of the whole sentence, which in some cases requires human involvement for detection [1]. The use of spelling correctors should be handled with care. This is because some errors are attributed by auto-correctors, a feature of some word processing software [10]. A person may input a word they are not sure of and an auto-corrector can give a completely wrong feedback, especially if that person is foreign to the language in use.

2.2 Error Detection

The detector module is responsible for determining if a word is considered misspelled or not. N-gram analysis can be used in a detector.

N-gram analysis

An n-gram is an n letter subsequence of a string, where n usually is 1, 2, or 3[1]. N-grams are unigrams for one letter, bigrams for two letters and trigrams for three letters respectively. In general, n-gram analysis techniques check each n-gram in an input and compare it against an existing table of n-gram statistics. Spellcheckers using N-gram analysis makes use of N-gram statistics in a text corpus [19]. N-gram statistics is the frequency counts or probability of occurrence of N-grams. N-gram analysis is used to determine correctness of words in a mass of text [5]. N-gram based techniques used without the dictionary can be used to find the position at which in the misspelled word an error occurred. This is in some cases achieved by employing character-based N-gram language models [20] but can also use word-based N-gram language models. N-gram analysis is a statistical approach and statistical approaches are mainly influenced by corpora, their size and correctness. Lexical diversity is also an essential factor in a corpus. The frequency statistics for each word is compared with the system threshold. This threshold differs from one spellchecker to the other [4]. Normally, if the frequency is below the threshold the word is identified as wrongly spelled. This means n-grams that do not occur or with infrequent occurrences are considered to be misspellings.

2.3 Error correction

The corrector module is responsible for providing a set of possible corrections for a misspelled word. After a word is flagged as wrongly spelled, if possible a set of suggestions is availed. Studies point out that most misspellings involve at most one character change from the intended word [12]. This means the distance between the correct word and the misspelled word is ±1 character difference. According to some authors [10], n-grams can also be used for error correction. This is done by assuming certain n-grams within a word are correctly spelled and fix the remaining n-grams. A list of words is established as suggestions. It is also important to rank the words and lift the closest suggestion to the top of the list and presumably trim it. To organize this we need an algorithm that computes the minimum edit distance. A Levenshtein distance can accomplish this. It will suffice as it will show the shortest distance between suggested words and the word with the shortest distance will be considered as the best suggestion.

2.4 CORPORA

A corpus is a large collection of texts [15]. A spellchecker is as good as its corpus. Therefore, we need to be careful in selecting and using corpora. Availability of such corpora is of the essence as well. There already exists an open source morphological isiZulu corpus-Ukwebalana [16]. The corpus uses actual language, which enables future research and use. Statistical approaches are mainly influenced by corpora, its size and correctness. A corpus can be built from collected documents in public archives, libraries, consulting language experts and using already existing dictionaries [18]. Language evolves, so should the corpora to yield expected results. Outdated corpora means currently introduced words to the language will not be recognized.

3. SYSTEM DESIGN AND IMPLEMENTION

This section shows the design and implementation of the system. It gives full details on how the system algorithms were established and organized. It also outlines the stream of events within the system and the aim of each and technologies used in the system.

3.1 Preprocessing algorithm

The algorithm will take in the input text, tokenize the input text and search for these words in the exception list. Tokenizing the input text is the process of extracting unique words in the input text. The words are individually searched for in the exception list collected from the corpus. The exception list is a list containing known correctly spelled words which should not be spellchecked. It helps to minimize the number words taken through the error detection algorithm. Only words absent in the exception list are taken to the error detection algorithm. The exception list should be a hash table. Hashing [2] is a technique used for searching for an input string in a precompiled hash table via a key or a hash address associated with the word and retrieving the word stored at that particular address. The hashing algorithm is fast as it minimizes the number of comparisons for the lookup.

3.2 Error detection algorithm

The system aims at detecting non-word errors. Non-word errors are spelling errors resulting in word forms that do not exist in the language. The design proposes a statistical method based on a language model that is a combination of the character-trigrams or character-four-grams and the probability of having each character-trigram or four-gram in a particular training dataset.

3.2.1 Character-trigram Model

A character-trigram model is a probabilistic model and a trigram is a set of 3 consecutive characters taken from a string. In the model, each word is composed of 3 character sequences. The next step is to obtain frequencies for each 3 character sequence from the database.

The model then considers each 3 character sequence as a word. It then finds the probability of each word w_i out of the total number of words in the training dataset N, given as,

$$P\left(\frac{w_i}{N}\right) = \frac{w_i frequency}{N\ frequency}$$

We set the threshold. Any probability less than the threshold means that the word based on statistics is a wrong word form.

3.2.2 Character-four-gram Model

A character-four-gram model is a probabilistic model and a four-gram is a set of 4 consecutive characters taken from a string. In the model, each word is composed of 4 character sequences. The next step is to obtain frequencies for each 4 character sequence from the database. The model then considers each 4 character sequence as a word. It then finds the probability of each word out of the total number of words in the training dataset N. This means if a non-existent or rare trigram or four-gram is detected, then the word is flagged as a misspelled word, otherwise not.

These models are language independent. They requires no prior knowledge of the language. All misspelled words are then taken to the best suggestion algorithm. The best suggestion algorithm is computationally intensive. Therefore, efficiency of the error detection algorithm is important to avoid creating suggestions for correctly spelled words. The process of filtering the best suggestion can result in thousands of suggestions and we are looking for a few best suggestions. The process should include ranking to only retrieve the highly ranked suggestions. Most of spelling errors have one error letter, because of transposition of two letters (transposition error), adding extra letter (insertion error), omitting one letter (deletion error) and mistyping one letter [5]. Minimum edit distance helps to filter for a few best suggestions. The Minimum Edit Distance algorithm [18] is defined as the minimum number of edit operations needed to transform a string of characters into another one.

3.3 Postprocessing algorithm

The post processing algorithm marks all incorrectly spelled words on the input text. It helps to give presentable feedback. Words move from one stage to the other. Known correctly spelled words by the system are no taken to the error detection stage. See figure 1 below for the system architecture.

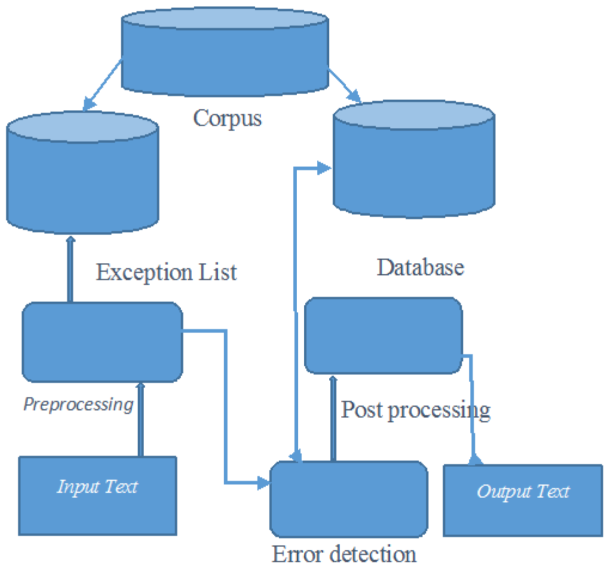

Figure 1. System architecture

3.4 Search algorithm

The search algorithm is important in the design as the system should be fast. Speed and accuracy are key in the design. The faster the search algorithm, the faster the system. Therefore, it is imperative to choose the faster algorithm for searching the exception list and the database. To help speed up the design the trigram statistics and four-gram statistics used in the system is precompiled and store in the database.

3.5 Database management System (DMS)

The choice of DMS is also crucial. The system needs a robust system, capable of handling multiple simultaneous reads and write. We introduced escaping to all strings containing single quotes as Java does not allow storing or retrieving strings with single quotes unless escaped.

3.6 Tools and technology

Java

Java is a single root, single inheritance object orientated language. It has vast libraries to support and ease development. It provides clear error messages, supports distribution and has built-in support for multithreading. It is relatively easy to integrate Java with other programming platforms. Applications built in Java are platform independent. They can run on both Linux and Windows. This is a platform that was used to implement all algorithms in the system. The interaction and communication within Java is plain [11].

MySQL

MySQL is a relational database management system. We chose MySQL because it stores data efficiently and data is quick to retrieve. MySQL is robust and not easily accessible to security threats. It was used to store trigrams and four-grams. Speed is critical so we needed a fast response storage.

4. EXPERIMENT DESIGN AND EXECUTION

4.1 Experimental dataset

Three corpora were used in the experiment. The corpora are used to provide both the training dataset and the testing dataset.
Table 1 before shows characteristics of corpora used in the spellchecker.

Table 1. Characteristics of corpora

Corpus description	Size of corpus	Number of unique words
Ukwebalana corpus	288 106 raw words.	87033
Corpus compiled by Prof. Langa	538 732 raw words	33020
The news items corpus	20000 raw words	9587

4.2 Experimental design

We used 10-fold cross validation to partition a corpus into the training data set and the testing dataset. This is done by randomly breaking the corpus into ten sets of equal size. We carried out 10 experiments, and used 9 folds for training and unique words from the remaining one for testing. Each data split is used for testing once. Assuming that the corpus contains zero incorrect words, we intentionally introduced 46 known wrongly spelled words to each testing dataset on each experiment. The spell checker was fed with the same wrongly spelled words as testing rotates across 10 folds. The performance of the system on each test was evaluated using the confusion matrix. The average was computed from the measures of all 10 tests to find accuracy rate.
The experiment was carried out on all the corpora. The second phase of our experiment was to test with the entire unique word set for each of the other 2 corpora and state the results for each corpus separately. The spell checker was trained with each corpus and tested with the other 2 corpora (unique words). The performance of the system on each test was also evaluated using a confusion matrix.

4.3 Evaluation metrics

The system was evaluated using the confusion matrix. The confusion matrix classifies the results of the test into true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). See figure 1 for the classification. True positives are cases in which the spell checker predicted a correct spelling and the word is actually correctly spelled, True negatives are cases in which the spell checker predicted a wrong spelling and the word is actually wrongly spelled, False positives are cases in which the spell checker predicted a correct spelling and the word is actually wrongly spelled and False negatives are cases in which the spell checker predicted a wrong spelling and the word is actually correctly spelled.

	Correct spelling	Wrong spelling
Correct spelling	TP	FP
Wrong spelling	FN	TN

Figure 2. Classification of results

After classification of results we want to compute the accuracy of the spell checker. The accuracy rate of the spell checker is (TP+TN)/Total number of words in the test dataset. We might want to check misclassification-how often the spell checker is wrong-which is (FP + FN) / Total number of words in the test dataset. True positive rate is (TP)/ Total number of correctly spelled words in the test data set and True negative rate is (TN)/ Total number of wrongly spelled words in the test data set.

All tests will be run on different thresholds. The threshold is the probability at which we still consider character compositions within the word to be correct. These tests were run on both trigrams and four grams to see where the spellchecker performs the best.

4.4 Hypotheses

Test 1: The spell checker is accurate using Ukwebalana corpus.
Test 2: The spell checker is accurate using Prof. Langa's corpus.
Test 3: The spell checker is accurate using news items corpus.
Test 4: The Ukwebalana corpus is the most accurate corpus of the corpora mentioned.

5. RESULTS AND FINDINGS

5.1 Results analysis

To analyze the results we plot accuracy rate measures on the line graph. These measures are probabilities and are between 0 and 1. Since the first part of the experiment uses n-fold cross validation, we first compute average measures for these 10 folds. We will end up with a single measure for each threshold as if there was no n-fold cross validation. We do not test with results for individual folds but averages over these 10-folds on each of the thresholds. Each probability shows the probability or percentage of accuracy for that threshold. The horizontal axis represents threshold and the vertical axis represents accuracy rate. This enables us to view the behavior of the system with different corpora and thresholds. The combination of the corpus and the threshold that yields the best results can be identified. If the accuracy rate is 0.50, it means that the spellchecker is 50% accurate. Examples of word segmentation into trigrams and four-grams. An IsiZulu word "isenzo" trigrams and four-grams are "ise","sen","enz","nzo" and "isen","senz","enzo" respectively.

5.2 Results for trigrams

5.2.1 Confusion matrices

The matrices below are an overall view of the confusion matrices for trigram. To achieve this, each matrix was converted to percentages first and for each threshold, added over the 10-folds and converted to percentages again.

10-fold cross validation
Ukwebalana corpus

<u>Threshold 0.003</u>

84.07	0.01
15.49	0.03

<u>Threshold 0.004</u>

75.84	0.01
22.73	0.03

<u>Threshold 0.005</u>

65.72	0.01
33.85	0.04

<u>Threshold 0.006</u>

57.53	0.01
42.03	0.04

Prof. Langa corpus

<u>Threshold 0.003</u>

64.96	0.03
34.08	0.12

<u>Threshold 0.004</u>

58.30	0.03
41.16	0.12

<u>Threshold 0.005</u>

48.14	0.02
51.33	0.13

<u>Threshold 0.006</u>

434.4	0.02
560.21	0.13

News items corpus

<u>Threshold 0.003</u>

82.80	0.05
14.01	0.27

<u>Threshold 0.004</u>

76.50	0.05
19.92	0.27

<u>Threshold 0.005</u>

70.98	0.05
25.82	0.28

<u>Threshold 0.006</u>

61.17	0.05
33.50	0.28

<u>Training dataset: Ukwebalana corpus</u>
Testing dataset: Prof. Langa corpus

<u>Threshold 0.003</u>

53.97	0.03
45.89	0.11

<u>Threshold 0.004</u>

51.05	0.03
48.82	0.11

<u>Threshold 0.005</u>

46.78	0.03
53.08	0.11

<u>Threshold 0.006</u>

44.12	0.03
55.75	0.11

Testing dataset: News items corpus

Threshold 0.003

70.01	0.10
29.51	0.37

Threshold 0.004

58.14	0.10
41.38	0.37

Threshold 0.005

56.76	0.09
42.76	0.38

Threshold 0.006

51.45	0.09
48.07	0.38

Training dataset: News items corpus

Testing dataset: Prof. Langa corpus

Threshold 0.003

88.56	0.02
11.30	0.12

Threshold 0.004

85.25	0.02
14.61	0.12

Threshold 0.005

84.45	0.02
15.41	0.12

Threshold 0.006

83.31	0.02
16.56	0.12

Testing dataset: Ukwebalana corpus

Threshold 0.003

26.94	0.01
73.01	0.04

Threshold 0.004

22.95	0.01
79.99	0.04

Threshold 0.005

21.58	0.01
78.37	0.05

Threshold 0.006

20.17	0.01
79.78	0.05

Training dataset: Prof. Langa corpus

Testing dataset: News items corpus

Threshold 0.003

88.93	0.10
10.59	0.37

Threshold 0.004

77.40	0.09
22.12	0.38

Threshold 0.005

74.17	0.08
25.35	0.39

Threshold 0.006

68.06	0.08
31.47	0.40

Testing dataset: Ukwebalana corpus

Threshold 0.003

41.43	0.01
58.52	0.05

Threshold 0.004

35.07	0.01
64.87	0.04

Threshold 0.005

28.05	0.01
71.90	0.04

Threshold 0.006

24.01	0.01
75.93	0.04

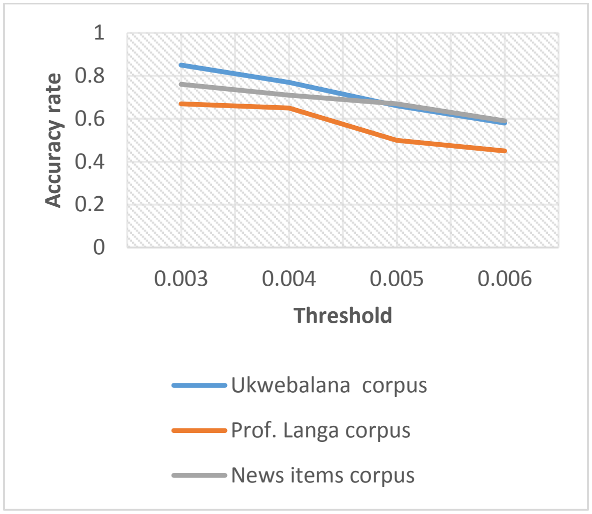

Figure 3. Performance of n-fold cross validation for Ukwebalana corpus, Prof. Langa corpus and news items corpus.

5.2.2 Ukwebalana corpus

Below are results for Ukwebalana corpus in average probabilities (accuracy rate) from 10-fold cross validation and their interpretation.

Threshold 0.03

0.85

Threshold 0.004

0.77

Threshold 0.005

0.66

Threshold 0.006

0.58

Looking at the chart of Ukwebalana corpus, the conclusion is that the smaller the threshold, the better the outcome. That is not entirely true because testing was mainly done with correctly spelled words. A proportion of correctly spelled words in the testing data set was higher than of wrongly spelled words. What the results showed before averages were computed was that the smaller the threshold, more of correctly spelled words were flagged as correct spelling but also more of wrongly spelled words were flagged as correct spelling. This means the threshold should not be too high or too low but despite the unbalanced quantities of correctly spelled words and correctly spelled words the performance of the spellchecker tested with Ukwebalana corpus was still acceptable with accuracy rate above 50% on all thresholds. This is because the corpus contains highly frequent words.

5.2.3 Corpus compiled by Prof. Langa

Below are results for Prof. Langa corpus in average probabilities (accuracy rate) from 10-fold cross validation and their interpretation.

Threshold 0.003

0.67

Threshold 0.004

0.65
Threshold 0.005
0.50
Threshold 0.006
0.45

Looking at the chart of Prof. Langa corpus, the spellchecker performs best at the threshold of 0.003 and had 67% accuracy rate. This is a big corpus and could perform better but it mostly contains infrequent words which affected performance.

5.2.4 News items corpus

Below are results for news items corpus in average probabilities (accuracy rate) from 10-folds and their interpretation.

Threshold 0.003
0.76
Threshold 0.004
0.71
Threshold 0.005
0.67
Threshold 0.006
0.54
Looking at the chart of news items corpus, the spellchecker performs best at the threshold of 0.003 and had 76% accuracy rate.

5.2.5 Second phase of the experiment

5.2.5.1 Testing Ukwebalana corpus with the other two corpora

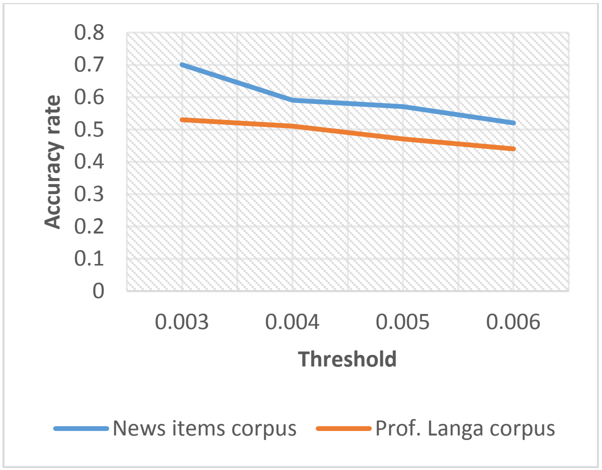

Figure 4. Performance of Ukwebalana corpus tested with Prof. Langa corpus and news items

5.2.5.1.1 Results for testing with Prof. Langa corpus

Training dataset is Ukwebalana corpus and testing dataset is Prof. Langa corpus.
Threshold 0.003
0.53
Threshold 0.004
0.51
Threshold 0.005
0.47
Threshold 0.006
0.44
Looking at the chart of Prof. Langa corpus, normally for low thresholds the spellchecker will flag a lot of words as correctly spelled words. This is not the case here because the training dataset lexical content is outdated and is tested mostly with new words that are being introduced as language evolves. The performance of the spellchecker for the given training dataset is low with accuracy rate of 53% at the threshold of 0.003.

5.2.5.1.2 Results for testing with news items corpus

Training dataset is Ukwebalana corpus and testing dataset is news items corpus.
Threshold 0.003
0.70
Threshold 0.004
0.59
Threshold 0.005
0.57
Threshold 0.006
0.52
Looking at the chart of news items corpus, the performance of the spellchecker is acceptable. It stays above 50% for all tested thresholds. The spellchecker performed best at the threshold of 0.003 and had 70% accuracy.

5.2.5.2 Testing news items corpus with the other two corpora

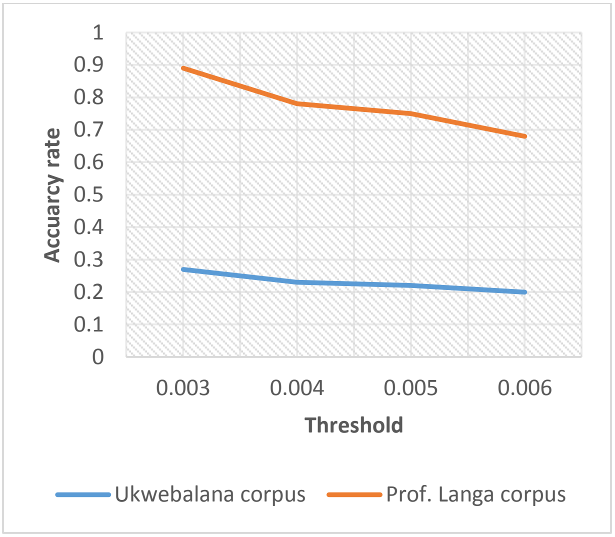

Figure 5. Performance of news items corpus tested with Ukwebalana corpus and Prof. Langa corpus.

5.2.5.2.1 Results for testing with Prof. Langa corpus

Training dataset is news items corpus and testing dataset is Prof. Langa corpus.
Threshold 0.003
0.89
Threshold 0.004
0.85
Threshold 0.005
0.85
Threshold 0.006
0.83
Looking at the chart of Prof. Langa corpus, the spellchecker is accurate for the given training dataset and testing dataset. This is because the spellchecker yields above 80% accuracy for all thresholds. The other factor is possibly the lexical content of the testing corpus. Most of words are short. This reduces the chance of finding a rejected sequence on characters. On top of that the

content of both the training dataset and the testing dataset is new evolving content in isiZulu.

5.2.5.2.2 Results for testing with Ukwebalana corpus

Training dataset is news items corpus and testing dataset is Ukwebalana corpus.

Threshold 0.003
0.27
Threshold 0.004
0.23
Threshold 0.005
0.22
Threshold 0.006
0.20

Looking at the chart for Ukwebalana corpus, the performance is poor. The accuracy rate is below 50% for all tested thresholds. This is because the testing dataset contains relatively outdated content which is not represented in the training dataset.

5.2.5.3 Testing Prof. Langa corpus with the other two corpora

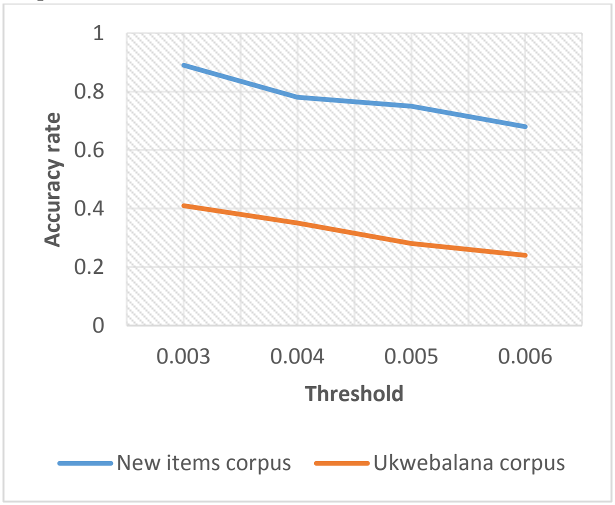

Figure 6. Performance of Prof. Langa corpus tested with news items corpus.

5.2.5.3.1 Results testing for with news items corpus

Training dataset is Prof. Langa corpus and testing dataset is news items corpus.

Threshold 0.003
0.89
Threshold 0.004
0.78
Threshold 0.005
0.75
Threshold 0.006
0.68

Looking at the chart for news items corpus, the spellchecker is accurate for the testing dataset. Testing with this dataset gave good results but because of the small size of the training dataset, the performance decreases with increase in thresholds.

5.2.5.3.2 Results for testing with Ukwebalana corpus

Training dataset is Prof. Langa corpus and testing dataset is Ukwebalana corpus.

Threshold 0.003
0.41
Threshold 0.004
0.35
Threshold 0.005
0.28
Threshold 0.006
0.24

Looking at the chart for Ukwebalana corpus, the performance is poor. The accuracy rate is below 50% for all tested thresholds. This is because the testing dataset contains relatively outdated content which is not represented in the training dataset.

.

5.3 Results for Four-grams

5.3.1 Confusion matrices

The matrices below are an overall view of the confusion matrices for four-grams. To achieve this, each matrix was converted to percentages first and for each threshold, added over the 10- folds and converted to percentages again.

10-fold cross validation
Ukwebalana corpus

Threshold 0.003

79.14	0.03
20.44	0.04

Threshold 0.004

73.94	0.03
25.59	0.03

Threshold 0.005

64.12	0.03
36.52	0.03

Threshold 0.006

56.23	0.02
44.99	0.04

Prof. Langa corpus

Threshold 0.003

62.15	0.02
36.40	0.13

Threshold 0.004

54.66	0.02
43.69	0.13

Threshold 0.005

47.15	0.02
52.51	0.13

Threshold 0.006

40.61	0.02
57.89	0.14

News items corpus

Threshold 0.003

76.80	0.04
19.96	0.28

Threshold 0.004

69.50	0.04
26.78	0.28

Threshold 0.005

63.51	0.03
33.29	0.29

Threshold 0.006

56.18	0.03
40.63	0.30

Training dataset: Ukwebalana corpus
Testing dataset: Prof. Langa corpus
Threshold 0.003

49.75	0.01
50.11	0.13

<u>Threshold 0.004</u>

48.76	0.01
51.10	0.13

<u>Threshold 0.005</u>

46.11	0.01
53.75	0.13

<u>Threshold 0.006</u>

43.37 0.01
56.49 0.13

Testing dataset: News items corpus

<u>Threshold 0.003</u>

68.52	0.04
31.00	0.44

<u>Threshold 0.004</u>

56.20	0.03
43.32	0.45

<u>Threshold 0.005</u>

55.63	0.03
43.89	0.45

<u>Threshold 0.006</u>

49.84	0.02
49.68	0.46

<u>Training dataset: News items corpus</u>

Testing dataset: Prof. Langa corpus

<u>Threshold 0.003</u>

88.10	0.02
11.76	0.12

<u>Threshold 0.004</u>

84.71	0.02
15.16	0.12

<u>Threshold 0.005</u>

84.11	0.02
15.75	0.13

<u>Threshold 0.006</u>

83.03	0.01
16.84	0.13

Testing dataset: Ukwebalana corpus

<u>Threshold 0.003</u>

26.69	0.01
73.25	0.05

<u>Threshold 0.004</u>

22.61	0.01
73.25	0.05

<u>Threshold 0.005</u>

21.21	0.01
78.74	0.05

<u>Threshold 0.006</u>

19.93	0.003
80.02	0.05

<u>Training dataset: Prof. Langa corpus</u>

Testing dataset: News items corpus

<u>Threshold 0.003</u>

85.40	0.05
14.12	0.43

<u>Threshold 0.004</u>

73.67	0.05
25.85	0.43

<u>Threshold 0.005</u>

73.33	0.04
25.19	0.43

<u>Threshold 0.006</u>

65.68	0.03
33.84	0.44

Testing dataset: Ukwebalana corpus

<u>Threshold 0.003</u>

41.15	0.01
58.80	0.05

<u>Threshold 0.004</u>

34.95	0.01
65.00	0.05

<u>Threshold 0.005</u>

27.93	0.01
72.02	0.05

<u>Threshold 0.006</u>

23.78	0.003
76.16	0.05

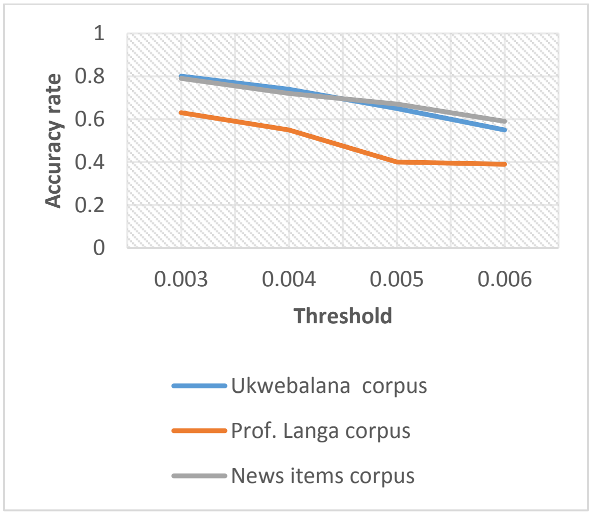

Figure 7. Performance of n-fold cross validation for Ukwebalana corpus, Prof. Langa corpus and news items corpus

5.3.2 Ukwebalana corpus

Below are results for Ukwebalana corpus in average probabilities (accuracy rate) from 10-fold cross validation.

<u>Threshold 0.003</u>

0.80

<u>Threshold 0.004</u>

0.74

<u>Threshold 0.005</u>

0.65

<u>Threshold 0.006</u>

0.55

5.3.3 Prof. Langa corpus

Below are results for Prof. Langa corpus in average probabilities (accuracy rate) from 10-fold cross validation.

<u>Threshold 0.003</u>

0.63

<u>Threshold 0.004</u>

0.55

<u>Threshold 0.005</u>

0.40

<u>Threshold 0.006</u>

0.39

5.3.4 News items corpus

Below are results for Ukwebalana corpus in average probabilities (accuracy rate) from 10-fold cross validation.

Threshold 0.003
0.79
Threshold 0.004
0.72
Threshold 0.005
0.67
Threshold 0.006
0.59

5.3.5 Second phase of the experiment

5.3.5.1 Testing Ukwebalana corpus with the other two corpora

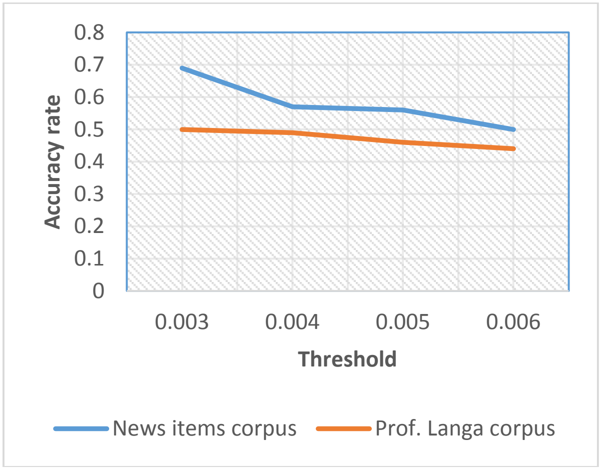

Figure 8. Performance of Ukwebalana corpus tested with Prof. Langa corpus and news items.

5.3.5.1.1 Results for testing with Prof. Langa corpus

Training dataset is Ukwebalana corpus and testing dataset is Prof. Langa corpus.

Threshold 0.003
0.50
Threshold 0.004
0.49
Threshold 0.005
0.46
Threshold 0.006
0.44

5.2.5.1.2 Results for testing with news items corpus

Training dataset is Ukwebalana corpus and testing dataset is news items corpus.

Threshold 0.003
0.69
Threshold 0.004
0.57
Threshold 0.005
0.56
Threshold 0.006
0.50

5.3.5.2 Testing news items corpus with the other two corpora

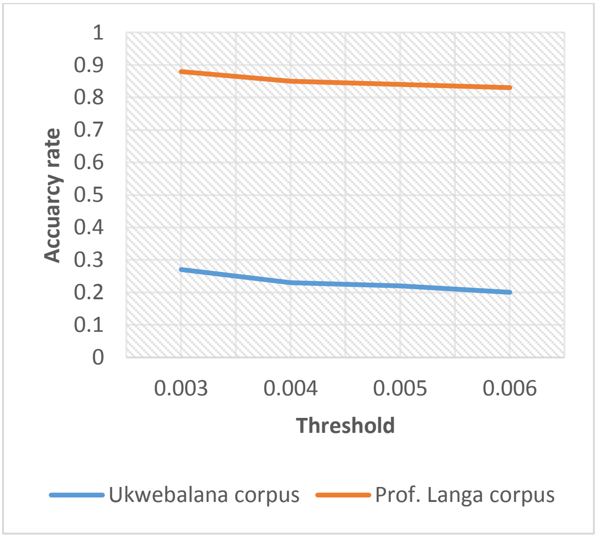

Figure 9. Performance of news items corpus tested with Ukwebalana corpus and Prof. Langa corpus.

5.3.5.2.1 Results for testing with Prof. Langa corpus

Training dataset is news items corpus and testing dataset is Prof. Langa corpus.

Threshold 0.003
0.88
Threshold 0.004
0.85
Threshold 0.005
0.84
Threshold 0.006
0.83

5.3.5.2.2 Results for testing with Ukwebalana corpus

Training dataset is news items corpus and testing dataset is Ukwebalana corpus.

Threshold 0.003
0.27
Threshold 0.004
0.23
Threshold 0.005
0.22
Threshold 0.006
0.20

5.3.5.3 Testing Prof. Langa corpus with the other two corpora

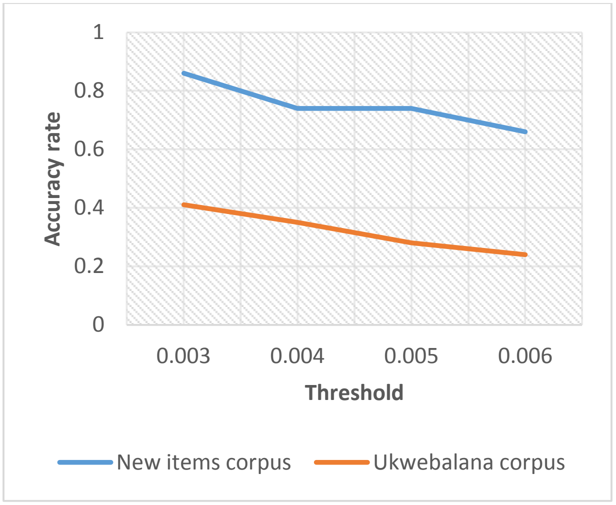

Figure 10. Performance of Prof. Langa corpus tested with news items corpus.

5.3.5.3.1 Results testing for testing with news items corpus
Training dataset is Prof. Langa corpus and testing dataset is news items corpus.

Threshold 0.003	0.86
Threshold 0.004	0.74
Threshold 0.005	0.74
Threshold 0.006	0.66

5.3.5.2.2 Results for testing with Ukwebalana corpus
Training dataset is Prof. Langa corpus and testing dataset is Ukwebalana corpus.

Threshold 0.003	0.41
Threshold 0.004	0.35
Threshold 0.005	0.28
Threshold 0.006	0.24

6. CONCLUSION

The spellchecker is accurate in detecting words that do not occur in the training corpus. All three corpora used in n-fold cross validation performed well tested with their fragments. For trigrams, Ukwebalana corpus gave accuracy rate of 85% at the threshold of 0.003, Prof. Langa corpus gave accuracy rate of 67% at the threshold of 0.003 and news items corpus gave accuracy rate of 76% at the threshold of 0.003. They all had accuracy rates above 50%. Ukwebalana corpus had the best results. Testing them with each other reflected something else. It is imperative to update corpora for spellcheckers. An outdated corpus can lead to poor performance. Testing both corpora with Ukwebalana gave accuracy rate below 50%. Ukwebalana corpus gave 53% accuracy rate when tested with Prof. Lang corpus and gave 70% when tested with news items corpus; News items corpus gave 89% accuracy rate when tested with Prof. Langa corpus and 27% tested with Ukwebalana corpus; and lastly, Prof. Langa corpus gave 89% accuracy rate when tested with news items corpus and 41% accuracy rate when tested with Ukwebalana corpus. It also showed that a spellchecker is as good as its corpus. The newly composed corpora performed really badly when tested with old words from the Ukwebalana corpus. We also discovered that because Ukwebalana contained highly frequent words, it managed to still perform well when tested with other corpora. This means most of the trigrams in the testing corpora were present in Ukwebalana corpus. We then deduced that the larger the corpus the better the performance. It will show highly frequent words and the spellchecker will target those. The most updated corpora are preferable and the 0.003 threshold. For four-grams, Ukwebalana corpus gave accuracy rate of 80% at the threshold of 0.003, Prof. Langa corpus gave accuracy rate of 63% at the threshold of 0.003 and news items corpus gave accuracy rate of 79% at the threshold of 0.003. Testing both corpora with Ukwebalana gave accuracy rate below 50%. Ukwebalana corpus gave 50% accuracy rate when tested with Prof. Lang corpus and gave 69% when tested with news items corpus; News items corpus gave 88% accuracy rate when tested with Prof. Langa corpus and 27% tested with Ukwebalana corpus; and lastly, Prof. Langa corpus gave 86% accuracy rate when tested with news items corpus and 41% accuracy rate when tested with Ukwebalana corpus. The spellchecker performed slightly better with trigrams than with four-grams. The probability of finding four-grams of a word was lower than the probability of trigrams. This increased the number of false negatives for each test.

7. FUTURE WORKS

Only a few spellcheckers attempt to detect real-word errors, such as CRITIQUE [9]. It is a potential future addition to the spellchecker to improve its capabilities. The spellchecker also does not provide suggestions because of time constraints for the project and the fact that the main focus in a spelling checker is error detection. Before everything else, error detection should be working properly. We had to invest more time on experimenting the error detection. Another important observation was that the spellchecker flags all infrequent words as wrong spellings. A significant improvement could be using a data driven statistical language model together with a theory-driven linguistic model to build a spellchecker. This will help check whether infrequent words follow language rules. It will definitely improve performance. The spellchecker should also detect unlikely combinations of words.

8. ACKNOWLEDGEMENT

I would like to thank Prof. Hussein Suleman for his support in the supervision of the project.
Thanks to Dr. Maria Keet for also helping with the supervision. She also provided me with the news items corpus. The corpus that also strengthened and influenced my results and conclusions.
Thanks to Prof. Langa for providing me with another corpus. A much bigger corpus results in the clear results and conclusions.

9. REFERENCES

[1] Anthony, I., Charibeth., Alberto., Paul., C., Chan., Cheng and Querol, Vazir Joshua. SpellCheF : Spelling Checker and Corrector for Filipino. *Journal of research in science, computing, and engineering.* Vol. 4 No. 3 December 2007.

[2] Basheer, S., Sindhu, L. "Survey of Spell Checking Techniques for Malayalam: NLP". *International Journal of Computer Trends and Technology (IJCTT)* – volume 17 Number 4 Nov 2014.

[3] Bhatti, Z., Ismaili, I.A., Shaikh, A. A., Soomro, W. J. "Spelling Error Trends and Patterns in Sindhi". Journal of Emerging Trends in Computing and Information Sciences, Vol. 3, No.10, 2012.

[4] Bidyut B. Chaudhuri. A simple real-word error detection and correction using local word bigram and trigram. *Proceedings of the Twenty-Fifth Conference on Computational Linguistics and Speech Processing (ROCLING 201*3).

[5] Damerau, F.J., "A technique for computer detection and correction of spelling errors", *Comm. ACM* 7(3):171-176, 1964.

[6] Gupta, N., Mathur, P. Spell checking techniques in NLP: a survey. *Department of computer science, Banasthali vidyapith,India* .2012.

[7] Jurafsky, D. Martin, J. *Speech and Language Processing*, Pearson. 1992.

[8] Morris, Robert and Cherry, Lorinda L., "Computer detection of typographical errors," IEEE Trans Professional Communication, vol. PC-18, no. 1, pp. 54-64, March 1975.

[9] Heidorn, H., K Jensen, L A Miller, R J Byrd, and M Chodorow. 1982. The EPISTLE text-critiquing system. IBM Systems Journal, 21:305–326.

[10] Hirst, G and Budanitsky A. Correcting real-word spelling errors by restoring lexical cohesion. Natural Language Engineering, 11(1):87–111, March 2005.

[11] Kindler, E., Krivy, I. Object-Oriented Simulation of systems with sophisticated control. International Journal of General Systems, 2011, 313–343.

[12] Kukich, K., Techniques for Automatically Correcting Words in Text. *ACM Computing Surveys*. Volume 24, Number 4. pp. 377-439. December 1992.

[13] Pirinen, T and Hardwick, S. Effect of language and error models on efficiency of finite-state spell-checking. In Proceedings of FSMNLP, Donostia–San Sebastían, Spain, 2012.

[14] Riseman, Edward M. and Hanson, Allen R., "A contextual post-processing system for error correction using binary n-grams," IEEE Trans Computers, vol. C-23, no. 5, pp. 480-493, May 1974.

[15] Robin. Natural language processing. Article on natural language processing. Dec 8th, 2009.

[16] Spiegler, S., Spuy A and Flach P. Ukwabelana - An Open-source Morphological Zulu Corpus. Proceedings of the 23rd International Conference on Computational Linguistics (COLING, 2010), 1020–1028 Robin. Natural language processing. Article on natural language processing. Dec 8th, 2009.

[17] Tavast, A., Koit, M & Muischnek K. 2012. Human Language Technologies: The Baltic Perspective. *Proceedings of the Fifth International Conference Baltic HLT 2012.* IOS Pres BV. Netherland.

[18] Vale, O. A., Candido Jr, A., Muniz, M. C. M., Bengtson, C. G., Cucatto, L. A., Almeida, G. M.B., Batista, A., Parreira, M. C., Biderman, M. T., Aluísio, S. M. Building a large dictionaryof abbreviations for named entity recognition in Portuguese historical corpora. In LATECH2008. Paris: ELRA, v. 1. p. 1-10, 2008.

[19] Wagner, R.A. and Fischer, M. J, "The string-to-string correction problem", *Journal of the Association for Computing Machinery*, 21, 168-173, 1974.

[20] Wasala A, Weerasinghe R, Pushpananda R, Liyanage C & Jayalatharachchi E. A Data-Driven Approach to Checking and Correcting Spelling Errors in Sinhala. *The International Journal on Advances in ICT for Emerging Regions*. 2010 03 (01) : 11 – 24.

[21] Wu, J., Chiu, H., & Jason, S. Chang. Integrating Dictionary and Web N-grams for Chinese Spell Checking. *Computational Linguistics and Chinese Language Processing* Vol. 18, No. 4, December 2013, pp. 17-30.