Literature Review for isiXhosa SpellChecker*

Siseko Neti
NTXSIS001@myuct.ac.za

ABSTRACT

This paper looks closely at the work and research that has been done in the field of spell checking. Given that a spell checker's main task is to check whether the word given as input is correctly/incorrectly spelled, it is necessary to understand the structure (words, meaning and context) of the actual language in order to make these decisions. The review mostly uses the concepts used/defined in linguistics to describe the language structure since linguistic is the study of language. This paper also look at attempts that have been made in producing a spelling checker for a highly agglutinative and conjunctive language such as the Nguni language isiXhosa. Finally, from all the work done in this field we can tell that the best way to implement the most effective and efficient spell checker is to use some lexicon together with an automatic morphological analyser, where the morphological analyser will be be constructed using finite state networks.

CCS CONCEPTS

•**Computing methodologies** → **Phonology / Morphology;** *Language resources;* Lexical semantics; •**Computation Linguistics** → Morphological Analyser;

1 INTRODUCTION

A spelling checker is a computer program that determines incorrectly spelled words from an input text. It is mostly incorporated with a spelling corrector where for all the incorrect words the program produces a number of correct suggestion and then automatically rank the suggestions before displaying them to the user [Miangah 2014]. Many spell checkers are available for European languages but only a few exist for Nguni languages. Since isiXhosa is the second largest South African language [Mzamo et al. 2015], this project will research on how to use the rule-based approach for the isiXhosa language in order to produce the most efficient and effective spell checking technique or mechanism. IsiXhosa was chosen amongst all the other languages since it is closely related to all the other Nguni languages which are IsiZulu (the first largest language in South Africa), siSwati and isiNdebele, and we hope that any work done for this language can easily be bootstrapped to one of these languages which would then cover the majority of South Africa's population [Mzamo et al. 2015].

In this paper we start by describing some concepts from linguistics which will be used throughout the paper. We then look at technique's that can be used in performing spell checking and then dwell further on the most effective technique for a high agglutinative language as to how it works. Finally we then look at the work done in the spell checking field for other agglutinative languages and then conclude our review for this field.

2 THE BODY

This section describes the theory and applications involved in developing spell checkers. We start by describing the most useful concepts that we will use throughout this review which are defined in linguistics studies.

2.1 Linguistics Concepts

Linguistics is one study which is mostly concerned with language research and development thus since we are working in the field of language spell checking we will borrow and use some concepts from it throughout this review. It is thus necessary to understand what some of these concepts mean before we even use them. One of the most important concepts that we will use is Morphology. Morphology refers to the study of the internal structure of words, and the systematic form-to-meaning correspondence of words, which deals with ways in which words are formed [Booij 2012].

Some words can be splitted into smaller units that could also have their own meanings. Therefore a new concept named *morpheme* is defined as the minimal linguistic unit with a lexical or grammatical meaning, for example the word buyer has the morphemes, *buy* and *ey* where *buy* is called a *lexical morpheme* (morpheme that can occur as a word on its own) and *er* is called an *affix* (morpheme that cannot function as a word on its own) [Booij 2012]. With that in mind, a language is then called agglutinative if its words can be formed by a combination of different morphemes, where these morphemes are not modified in spelling or phonetics (speech sound) prior their use in any word [Prószéky and Kis 1999].

The functions of morphology that we are most concerned with are, the creation of new words (lexeme) and spelling out the appropriate form of a lexeme in a particular syntactic context. These words mostly belong to a grouped set containing multiple words such as dictionaries or corpora, which are further described in the following subsection.

2.2 Text Corpus

A corpus is a collection of text from the same or different subject domains which could be used to better understand the language morphology especially in that subject domain [Miangah 2014]. Therefore using a corpora (many corpus) or one large corpus can better guarantee the accuracy of the spelling checker with regard to the input needed to be checked because a linguistic corpus

*This is an honours Project proposed by Dr Maria Keet

ALSPEL, Rondebosch, Cape Town, South Africa
 123-4567-24-567/08/06…$15.00
DOI: 10.475/123_4

based study is known for providing an accurate description of a language [Miangah 2014].

In terms of natural language processing, particularly computational morphology, the Bantu languages certainly belong to the lesser-studied languages of the world [Pretorius and Bosch 2003]. Due to this, previous projects for spell checking had to look or even develop language text corpora [Jones et al. 2005] because of the limited freely available corpora for Nguni Languages [Pretorius and Bosch 2009]. This was done since many spell checker's at that time were using a lexicon, a lexicon being a repository of all the information concerning the established words and other established expressions of a language [Booij 2012], for deciding whether a word is correctly/incorrectly spelled, this is described further in the next section.

Due to the fact that most official languages in South Africa belonged to the lesser-studied languages of the world, the Government of South Africa has decided to have an open source repository for data/information about all the official languages in South Africa which then will also make a major contribution to language research and development. The repository is called RMA (http://rma.nwu.ac.za/) and currently it has four corpora with a total of 86 1026 tokens and three corpora with a total of 65 000 words for the isiXhosa language. Now with projects like RMA, there is a much more text corpora available for the isiXhosa language, thus we do not need to stress much about this step.

These corpora are mostly used in forming or creating lexicons for the language since they can have information about all the different subject domains. The next section thus describes how these corpora can/could be used in the actual development of the spelling checker.

2.3 Spell Checker

In this paper since we conduct a research for providing a spell checker, we will use some of the conclusions made from the morphological study of the isiXhosa language. Firstly isiXhosa has many morphemes per word [Mzamo et al. 2015] and it is a highly agglutinative and conjunctively written language, which means that there are literally millions of possible words that can be derived from a limited number of roots and stems through the use of affixes [Bosch and Eiselen 2005]. Due to this it is then regarded as one of the complicated languages.

Now, given that the main function of a spell checker is to determine whether the word given as input is a correctly spelled word of the target language [Bosch and Eiselen 2005]. In the past as briefly stated in the section above, the most obvious and most widely used spell checker method is the use of a lexicon of correctly spelled words against which an input word is compared. Since isiXhosa is a highly agglutinative and conjunctively written language this cannot be the case because the lexicon would be too large and the amount of physical memory needed to load such a lexicon would be unreasonably large [Bosch and Eiselen 2005].

Therefore the spell checker will need to include some form of automatic morphological analysis, which will be the enhancement to a lexicon that we may possibly already have. This means that this analysis will make it possible for a spell checker to accept correctly spelled words that are not contained as entries in the lexicon. Therefore, this will make it possible for the spelling checker to recognise a large number of words without increasing the size of the lexicon [Bosch and Eiselen 2005].

So far we have only talked about the morphology of the language but looking only at the morphology of the language is not enough for a spell checker that performs morphological analysis because if some syntactic and semantic information about the language is missing, then the spell checker may regard valid words as incorrect [Miangah 2014]. To avoid these kind of failures we will also concentrate on the syntax (a collection of principles defining how to put together a sentence) and semantics (meaning of certain terms and/or sentences) of the language.

Now since morphology won't be enough on its own, it is also advantageous to include lemmatizers as components of the spelling checker, which are modules that find the linguistic normalised form of a word [Eiselen and Puttkammer 2014]. Lemmatizers are known for reducing the volume of the system database [Miangah 2014] thus we hope that they might help in reducing the volume of the lexicon, which would then make the spelling checker to use the smallest possible physical memory. Computationally these morphological analysers and lemmatizers are represented as finite state networks.

2.4 Finite State

Given that Finite State Automata (FMAs) and Finite State Transducers (FSTs) both operate on strings (a sequence of symbols). Recent methods for optimizing speed or accuracy suggest that we can rely on finite dictionaries or acyclic finite automata as language models, a language model being a one-tape finite-state automaton recognising valid word forms of a language [Pirinen et al. 2010]. Finite state automata can be extended to transducers. Where we can have finite-state sequential string-to-string transducer which are used in the representation of large-scale dictionaries, computational morphology, and local grammars and syntax [Mohri 1997]. Since a language can have a very large lexicon, using a single transducer may lead to time and space problems, thus it is advantageous to use multiple transducers [Karttunen et al. 1996].

Since there are already existing finite state tools used in computational linguistics, in order to compute these finite state transducers we will/might use some of these tools. Finite state tools used in linguistics applications belong to the *Xerox* project, amongst all these tools the ones that we are mostly interested on are *lexc* and *xfst*. Where Lexc is a compiler for morphological analyser [Karttunen 2010] and xfst is a general purpose utility for computing with finite state networks which enables the user to create simple automata and transducers from text and binary files, regular expressions and

other networks by a variety of operations. The xfst and lexc languages are development tools intended for constructing finite state network [Beesley and Karttunen 2003], which would play a major role in the development of the spelling checker.

Similarly, others have used morphological decomposers which split up a token/word into constituent, to perform this they use stemming where stemming is a process which reduces morphologically variant of a word to a single root or stem of the variant with this obtained by removal of suffixes from that particular word. Stemming and lemmatization make it possible to compare a query in one morphological form with a word in a document in another morphological form. These decomposers are different from morphological analysers where the individual morphemes are identified and assigned tags based on their grammatical function. This is quite useful for complicated languages such as isiXhosa where a word can have many morphemes.

Now since we would like to apply/bootstrap the morphological analysis method to the other Nguni languages, it is known that Bantu Languages differ in terms of their phonological features implying that each Bantu language requires an independent morphological analyzer [KATUSHEMERERWE and Hanneforth 2010]. Unlike other approaches that had been implemented prior the release of the paper in 2010, the paper [KATUSHEMERERWE and Hanneforth 2010] focused on analysing nouns using Finite State Methods (fsm2). From that, they were able to show that the actual implementation of fsm2 using a context free grammar and replacement rules is applicable to a morphologically complex Bantu language [KATUSHEMERERWE and Hanneforth 2010] which isiXhosa is one of them.

The next section gives us more information about the method that has been used and/or succeeded in most projects regarding spell checking for agglutinative languages.

3 RELATED WORK

Spelling checker attempts have been made for the language Quechua, which is a strongly agglutinative, suffixing language. The first attempt used the xfst tool. Another atempt has been made since the xfst spelling checker was slow. This was named the *foma* spell checker which uses minimal edit distance search to calculate the minimum deviation of a given input string from the recognised strings of the regular language implemented by the automaton. Since xfst was large the foma spell checker peforms its tasks much faster that the xfst [Rios 2011].

The paper [Theron and Cloete 1997] described the acquisation of two-level rules for isiXhosa noun locative pairs. They were taking two lexemes, a source and a target word, then identify the prefix, suffix and source morphemes between the two words. Their rules for determining the suffixes and prefixes would not work if there were a different set of source/input nouns as there can be a huge difference between nouns. Thus projects that try to treat isiXhosa as a simple non-agglutinative language will only work for the subsection/subject-domain that they are looking at, it cannot work for the language as a whole.

The project described in [Jones et al. 2005] was not a success because they checked each and every word in the spelling checker against the lexicon, which then resulted in words that were not in the lexicon to be regarded as incorrect even though their lexicon was too small (101 265 words) to decide for the entire isiXhosa language. As this [Jones et al. 2005] was the first attempt in developing a spelling checker for the isiXhosa language, inconsistencies relating to spelling errors, hyphenation, capitalisation, dialectal variants and offensive words and use of apostrophes all contributed to the development of the SpellChecker to be not successful. Mostly these inconsistencies were caused by the lack of the semantics and syntectic aspects of the human language in the spell checkeker.

Currently there is one spellchecker for the Nguni languages, which is for the IsiZulu language using the statistical approach, currently the spellchecker only peforms error detection [Ndaba 2015]. We believe that this approach can be easily bootstraped for another Nguni language since they are all similar. In this project we will mainly focus on using another spellchecking technique, which is the rule-based approach in order to compare the two approaches for deciding which is more efficient and faster.

4 CONCLUSIONS

From the results and the discussion above, it can further be conluded that a spell checker can be as accurate as possible if it uses a large monolingual lexicon and all the human language aspects which are semantics, syntactics and morphology are carefully studied and implemented in the spell checker. Also for highly agglutinative and conjuctively written languages it is impossible to use a lexicon which has all the words in that language and thus morphological analysis may be usefull for spell checking purposes in such languages.

REFERENCES

Kenneth R Beesley and Lauri Karttunen. 2003. Finite-state morphology Xerox tools and techniques. *CSLI, Stanford* (2003), 30–54.

Geert Booij. 2012. *The grammar of words: An introduction to linguistic morphology.* Oxford University Press.

Sonja E Bosch and Roald Eiselen. 2005. The effectiveness of morphological rules for an isiZulu spelling checker. *South African Journal of African Languages* 25, 1 (2005), 25–36.

Roald Eiselen and Martin J Puttkammer. 2014. Developing Text Resources for Ten South African Languages.. In *LREC*. 3698–3703.

Jackie Jones, Kholisa Podile, and Martin Puttkammer. 2005. Challenges relating to standardization in the development of an isiXhosa spelling checker. *South African Journal of African Languages* 25, 1 (2005), 1–10.

Lauri Karttunen. 2010. Update on finite state morphology tools. *Ms., Palo Alto Research Center* (2010).

Lauri Karttunen, Jean-Pierre Chanod, Gregory Grefenstette, and A Schille. 1996. Regular expressions for language engineering. *Natural Language Engineering* 2, 04 (1996), 305–328.

FRIDAH KATUSHEMERERWE and Thomas Hanneforth. 2010. fsm2 and the morphological analysis of Bantu nouns–first experiences from Runyakitara. *International Journal of Computing and ICT research* 4, 1 (2010), 58–69.

Tayebeh Mosavi Miangah. 2014. FarsiSpell: a spell-checking system for Persian using a large monolingual corpus. *Literary and Linguistic Computing* 29, 1 (2014), 56–73.

Mehryar Mohri. 1997. Finite-state transducers in language and speech processing. *Computational linguistics* 23, 2 (1997), 269–311.

Lulamile Mzamo, Albert Helberg, and Sonja Bosch. 2015. Introducing XGL-a lexicalised probabilistic graphical lemmatiser for isiXhosa. In *Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), 2015.* IEEE, 142–147.

Balone Ndaba. 2015. Afrispel An isiZulu spellchecker. *AFRICAN LANGUAGE SPELL CHECKER* 1 (2015), 2–12.

Tommi Pirinen, Krister Lindén, et al. 2010. Finite-state spell-checking with weighted language and error models. In *Proceedings of LREC 2010 Workshop on creation and use of basic lexical resources for less-resourced languages*.

Laurette Pretorius and Sonja Bosch. 2009. Exploiting cross-linguistic similarities in Zulu and Xhosa computational morphology. In *Proceedings of the First Workshop on Language Technologies for African Languages*. Association for Computational Linguistics, 96–103.

Laurette Pretorius and Sonja E Bosch. 2003. Finite-state computational morphology: An analyzer prototype for Zulu. *Machine Translation* 18, 3 (2003), 195–216.

Gábor Prószéky and Balázs Kis. 1999. A unification-based approach to morpho-syntactic parsing of agglutinative and other (highly) inflectional languages. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. Association for Computational Linguistics, 261–268.

Annette Rios. 2011. Spell checking an agglutinative language: Quechua. In *5th Language and Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*. 51–55.

Pieter Theron and Ian Cloete. 1997. Automatic acquisition of two-level morphological rules. In *Proceedings of the fifth conference on Applied natural language processing*. Association for Computational Linguistics, 103–110.