Generating Natural Language isiZulu Text from Mathematical Expressions

Shannon Smith
Department of Computer Science
University of Cape Town
May 2020

ABSTRACT

There is a lack of mathematical verbalisation, specifically in isiZulu, for the visually impaired. The problem can be solved by generating natural language descriptions of math expressions, so they can be used as input for existing text-to-speech systems. Current realisation tools do not apply to isiZulu because of the language's agglutinative morphology and lack of resources. We discuss the approaches to natural language generation and see that the most appropriate realisation method for isiZulu is grammar-infused templates. Previous work done in developing a controlled natural language of isiZulu shows it can be accomplished through verbalisation patterns and a context-free grammar for the verb. Previous work in verbalising mathematical have used template and grammar-based approaches and translated the ambiguous LaTeX expressions into a less ambiguous MathML format. However, there is still room for improvement for LaTeX to MathML translation.

KEYWORDS

Natural language generation, Controlled natural language, Niger-Congo B languages, isiZulu, Verbalisation of Mathematics.

INTRODUCTION

There is a global effort to improve the availability of online textbooks for visually impaired learners. In South Africa, there are already systems in place to generate text-to-speech in most of the home languages. The South African Centre for Digital Language Resources (SADiLaR)[1] researches and develops all aspects of natural language processing – automatic speech recognition, text-to-speech systems, grammar, and spell checking and much more. However, there is a gap in the verbalisation of mathematical expressions in South African languages. Text-to-speech systems can convert text into synthesized speech, but they can only read plain text and not mathematical formulae. Natural language generation (NLG) is critical in the solution to this problem. The mathematical formulae can first be translated into fluent text descriptions, which is then fed into a text-to-speech system, that visually impaired students can hear. NLG is a subfield of computer linguistics and artificial intelligence [4], that maps some input data to natural language text. This input data depends on the application, but some possible inputs include images, graphs, signals, ontologies, or numeric data [1]. It is used for a range of systems, the most common include generating reports: weather reports, financial reports, sports reports, news reports; dialogue systems; summarisation of text; and language translation. These systems all have different underlying methods/architecture, as their input can vary immensely but their overall purpose is the same: the automation of translating data into understandable text for human reading. The underlying methods are language dependent and must accommodate for the fact that languages have different linguistic structures. Existing NLG frameworks and tools are suited to Indo-European languages and fall short for handling other languages with different grammar structures.

The biggest issue for developing applications for South African languages is the lack of resources, like corpora. IsiZulu is one of South Africa's 11 official languages, it is a Niger-Congo B language, in the group of Nguni languages. The first isiZulu grammar was published by Grout (1859) and then much later the first dictionary by Colenso (1905) [15]. Niger-Congo languages share a lot of their linguistic structure, meaning the languages closely resemble one another. Although isiZulu is spoken by majority of SA (23% of the population), it is under-resourced in software applications [2] and there is much to be done to improve resources for isiZulu students. Currently there are no systems in place to translate mathematical formulae into isiZulu natural language text.

Therefore, the goal is to successfully bridge this gap for visually impaired isiZulu learners, so their math literacy can be improved. This can be done by building an NLG system that generates understandable text descriptions of mathematical expressions in the isiZulu language. There is a massive repository of LaTeX mathematical formulas on Wikipedia that can be cleaned and used as input for our NLG system.

This review discusses the morphology of isiZulu and proposes which of the NLG designs will be most suited to its complex grammar, specifically geared towards the translation of math expressions. The paper documents what has been done before in natural language generation in isiZulu and verbalisation of mathematical expressions, and assesses their strengths and weaknesses, so we may address any gaps.

[1]For more on SADiLaR: https://sadilar.org/

1 BASICS OF ISIZULU GRAMMAR

The morphology of isiZulu is rich and complex - the reason for its complexity is the agglutinative nature of Niger-Congo B languages [2]. Unlike Indo-European languages, which express the tense, negation etc. in separate components, agglutinative languages use prefixes and suffixes. In its simplest form, the structure of an isiZulu sentence is subject-verb-object. In the next two sections, we will briefly outline the noun classes and the complex verb structure.

1.1 The Noun Class

Like other Nguni languages, each noun belongs to a noun class. In isiZulu, there are a total of 17 noun classes, with a single and plural form in each class. The noun class is what dictates the agreement between the words in a sentence. The structure of a noun consists of a prefix and a stem. The prefix denotes if the noun is single or plural, and which noun class it is attached to. Sometimes a prefix has a pre-prefix (also called an augment).

1.2 The Complex Verb

The isiZulu verb structure consists of a core called the verb root (VR), which is extended with prefixes and suffixes. The prefixes reflect subject and object concords, tense/aspect, mood, and negation. IsiZulu verbs are very complex in that they have five different tenses: remote past, recent past, present, immediate future, remote future. On top of the prefixes, there are also suffixes that represent stative, applicative, reciprocal, and passive verbs. Finally, the verb ends with a final vowel depending on negation [11]. The complex verb structure with all its affixes is shown in (1).

(1) <NEG> <SC> <T/A> <MD> <OC> <VR> <Extension><FV>

Where, NEG=negative, SC=subject concord, T/A=tense/aspect, MD=mood, OC=object concord, VR=verb root, FV=final vowel.

2 DESIGNING A NATURAL LANGUAGE GENERATION SYSTEM

(Gatt and Krahmer, 2018) discusses three approaches to NLG architecture: Modular, Planning-based, and Other Stochastic approaches. A modular architecture is a system designed so that the individual modules perform their own clear-cut tasks [1]. The planning-based approach is more unified across the tasks. The stochastic approach requires a sufficient data resource to build the statistical models, which a low-resource language like isiZulu does not have. All three architectures roughly follow the same 6 tasks: Content determination, text structuring, sentence aggregation, lexicalisation, referring expression generation (REG) and realisation.

A traditional modular architecture is (Dale and Reiter's, 1997) pipelined approach (see figure 1.). There are three stages, that each perform their own specific tasks [4]:

1. *Text planning* → Content determination and text structuring (or discourse planning).
2. *Sentence planning* → Sentence aggregation, lexicalisation and REG.
3. *Linguistic Realisation* → Realisation.

Figure 1: The Pipeline Architecture of a Natural Language Generation System (Reiter and Dale, 1997)

Our project will involve developing the sentence planner and the linguistic realiser of the pipeline. The primary task of the text planning stage is content determination, that is, deciding what information we want to convey from the given data. This is dependent on the application and is mostly filtering or summarizing, however, this task is not needed for this project. The content for our NLG is determined by the mathematical expression, so no filtering is necessary.

The sentence planner takes the output of the text planner, 'what to say', and decides how to order these words into a structured sentence, 'how to say'. The first task is sentence aggregation, the phase in which sentences are made more natural sounding by removing any redundancy, and similar subjects are grouped together into one sentence. Next is the lexicalisation task, where the system can decide on sentence variation. For example, in the context of math verbalisation, there are several ways to word an expression: $a^2 + b^2 = c^2$. This can be worded as " a to the power of 2 plus b to the power of 2 equals c to the power of 2" or "The sum of a squared and b squared equals c squared". It depends on the domain of the system whether sentence variation is important or not. The final task of the sentence planner is REG, this is where the system identifies the domain entities (also called domain concepts) and decides on which words/phrases it will use to represent these entities.

Linguistic realisation is the final step of generating a grammatically correct and complete sentence after the choosing the appropriate words and phrases. This entails filling in any gaps that were not present in the input, such as adding articles, prepositions, auxiliary verbs, and punctuation. The realiser ensures agreement of nouns, verbs, and tenses, by following some form of morphological rules. The set of tasks performed during sentence planning and realisation is called 'tactical generation'.

There are many approaches to realisation [1]:
1. Templates
2. Grammar-based systems
3. Statistical approaches
4. Grammar-Infused Templates

2.1 Templates

This is an early technique of filling in the slots of a predefined sentence with data entries [1]. An example of a template for a basic addition equation is:

<$OP1 plus $OP2 equals $OP3>, where OP1, OP2 and OP3 can be substituted for any operand. It can generate a sentence such as, *"Two plus three equals five".*

In this approach, lexicalisation becomes obsolete, instead the content determination task will choose an appropriate template. There may be template alternatives to choose from, to simulate language variation. There are many advantages to templates if the application's domain of inputs is small and syntax variation is not needed. They are much easier to understand, faster to construct and yield similar performances to more complex models [6]. They also have a predetermined quality that is decided on by the developer, avoiding any grammatically incorrect outputs. However, this approach is not appropriate if linguistic variation is important or if the domain is large – as the time of constructing these templates by hand is significantly increased [1]. It is also language dependent. For languages like English, templates are a sufficient solution, but for more complex morphologies, templates do not suffice [9].

2.2 Grammar-based Systems

These systems construct sentences based on the grammar rules of the natural language being used. These grammars are usually restricted in some way to make them easier to develop – full natural language grammars are usually under-resourced and complex. Grammar-based systems require a great deal of detail specific to the language, therefore, they do not work well as "plug-and-play" modules [1]. This issue has prompted the use of realisation engines that give users morphology APIs. There are several of these programming tools for defining natural language grammars, like Grammatical Framework (GF) and SimpleNLG.

GF is a popular open source grammar engine for multilingual grammar applications (it supports around 30 languages), using functional programming. It needs a grammar library as a resource, so it will not work for under resourced languages like isiZulu [14]. Another issue for modelling isiZulu with the GF engine is that GF can only handle grammatical structures similar to English, hence, there has not been any implementations of isiZulu grammar using GF.

SimpleNLG is an NLG realisation engine based on the architecture in figure 2. It has a Java API, is well documented and was originally developed for English. The benefits of using SimpleNLG is that it is now portable for other languages and allows simple implementation of linguistic operations [12]. SimpleNLG is not applicable to Niger-Congo B languages, as its encoding was made to suit Indo-European languages (like English).

Although grammar-based systems may be more challenging to develop, they have a few advantages over templates. The advantages of using these realisation engines is that they are domain independent and the quality of the output is higher than templates [1]. Maintainability and making changes to the system is also significantly easier, templates require being rewritten to handle any change [6]. The short-comings of these realisation engines are that they are not suited for languages not structurally similar to English; and they require a corpus of the grammar's rules, which may not be available for low-resourced languages, like isiZulu.

2.3 Statistical Approaches

NLG can also be accomplished by training a statistical model using machine learning. A stochastic approach during realisation is very promising for future NLG development [1]. NLG was not a popular approach in the past as the techniques are computationally expensive (text alternatives are generated in full before the stochastic model selects it) and the output grammar was not good quality [18]. (Gatt and Krahmer, 2018) describes two approaches to statistical methods, (1) A set of grammar rules generate a variety of text alternatives, then a single text is chosen as the optimal realisation by a statistical model. (2) Instead of being introduced at a later stage, a statistical model is involved during the generation decisions, hence, removing unnecessary generation of alternatives. The advantage of (2) is that it is not as computationally expensive as (1). The (1) approach is seen in the HALOGEN/NITROGEN systems (Langkilde and Knight), relying on statistical knowledge from a corpus in the form of n-grams. (Belz, 2008) developed a system, pCRU, that uses the (2) approach, to generate weather reports. The pCRU system uses context-free grammars (CFG) to formalise the language, then generates the most statistically likely derivation of a sentence, based on some corpus. The statistical model drives which generation rules of the CFG are expanded, making sure the optimal text is derived. Belz evaluated her system against traditional-handcrafted pipelined systems, and HALOGEN-style systems. Belz concluded that there is an improvement in development time and reusability for pCRU. There was also evidence that probabilistic generation is more efficient than the n-gram techniques used post-selection in HALOGEN-style systems.

However, the disadvantage of both stochastic methods is that they rely on corpus-based statistical information. Therefore, to use these statistical methods, we need a comprehensive data resource (such as a corpus), which is not applicable for isiZulu.

2.4 Grammar-infused Templates

The leading approach to NLG is the use of templates with perhaps a few simple rules for template selection. For example, an English system needs a rule to select a template with the article 'an' instead of 'a' if the object noun starts with a vowel. However, as the complexity of a language increases, it becomes infeasible to use this method – isiZulu has 17 noun classes each with their own pre-fix and the verb conjugation needs to agree with both the

subject and the object in the sentence, meaning there would have to be a rule for how each class effects the verb. One solution is to attach the language's grammar rules to the templates.

We now see that there are two kinds of templates – the traditional templates that have predefined words and fill-in slots; and templates that use a natural language grammar on top of the templates (introduced as grammar-infused templates [17]). This can improve the grammar quality of the generated text and enable more flexibility without needing to use a complete grammar-based realiser like GF. Grammar-infused templates are useful when modularity is needed, as they allow for detachable grammars and the grammar can be reused across domains. This is significant when developing for low-resourced languages, like isiZulu, as existing grammar rules are scarce and need to be reused.

(Mahlaza and Keet, 2019) developed a model for the various ways templates and grammar rules can be combined, based on their relationship. There are two types of relationships, embedding and attachment (either partial attachment or compulsory attachment). A grammar rule is embedded if the grammar rule is deleted when the template is deleted; a grammar rule is attached if it remains after a template is deleted. Partial attachment means grammar rules are only attached to certain templates; compulsory attachment means there must be at least one rule attached to each template. An example of embedded and partial attachment is seen in section 3.2, (Keet and Khumalo, 2017).

These hybrid systems enable the NLG system to use complex grammar rules where it is needed and simpler, less expensive templates where it is not [6]. This combines the benefits of both approaches – templates are simple to construct but lack the ability to handle complex grammars; full grammar systems are expensive to develop but can accommodate for grammar complexity.

3 CONTROLLED NATURAL LANGUAGE OF ISIZULU

The biggest challenges we face in developing a CNL for isiZulu is that the language is under-resourced, under-researched and has a complex morphology. The complexity of the isiZulu morphology makes it difficult to present it as a controlled natural language (CNL) for the NLG system [2]. The agglutinative nature of the noun class and verb does not allow for implementation with existing verbalisation tools. In this section we will detail how isiZulu can be formalised as CNL so that an NLG system can use it. Previous works done in this field are evaluated and discussed in the context of what methods will/will not be suited to isiZulu.

3.1 Ontology Verbalisation

A CNL is a subset of a natural language that has restricted grammar or vocabulary. It is a way to formalise and unambiguously represent a language so a machine can understand it. An ontology provides a way to do this – it formalises knowledge as "concepts" and models the relationships between these concepts. Ontology verbalisation is used in NLG to generate text from these knowledge bases and aims to be domain independent. Web Ontology Language (OWL) and other Semantic Web logic-based languages are becoming the preferred syntax for NLG systems, due to their tool infrastructure [10]. For example, a popular CNL using OWL ontologies, in English, is Attempto Controlled English (ACE) (Fuchs et al., 2010). It is a restricted subset of the English language that can be automatically and unambiguously represented by first-order logic.

3.2 Developing Grammar Rules for Agglutinating Languages

Previous works done for generating text in Niger-Congo languages have used pattern-based methods for verbalising ontologies and a Context-Free Grammar (CFG) to represent the verb, in isiZulu [2][9][11][19], Runyankore [14] and isiXhosa [5][21].
(Keet and Khumalo, 2017) showed that due to the elaborate rules of the noun classes and verb conjugation, using templates on their own would not be feasible and a more detailed grammar-based system is needed [2]. They approached the problem of knowledge-to-text in isiZulu with verbalisation patterns. The verbaliser uses a grammar-infused template, with embedded rules for agreement between words and partially attached rules for noun pluralisation [19]. The patterns verbalise universal quantification ("For all"/"each"), existential quantification ("at least one"), conjugation ("and"), subsumption ("is a") and negation ("not") [9]. The logic foundation used for isiZulu was OWL 2 EL (A W3C standard of OWL 2) – even simple verbalisation using OWL 2 EL requires a grammar engine for isiZulu's complex grammar [9]. They only accommodated for one of the five tenses (present tense), but this should be enough for generating math descriptions. In [11] the isiZulu verb is represented as a CFG and their production rules cover subject and object concords, negation, present tense, aspect, and mood. To evaluate the grammar quality and understandability of the verbalisation patterns, a survey was conducted on 32 isiZulu speakers. The participants indicated that their outputs were mostly 'grammatical and acceptable' for simpler sentences, but 'grammatical and ambiguous' for more complex sentences. They concluded that the ambiguity could be from the participant's varying dialects. Overall, the results revealed that the generated text was grammatically correct, though sometimes tricky to understand. The grammar rules developed by (Keet and Khumalo, 2017) are limited but applicable to a variety of domains.

(Byamugisha, Keet and DeRenzi, 2016) developed a verbalisation pattern for Runyankore (A Niger-Congo language spoken in Uganda), by bootstrapping from the existing isiZulu patterns [14]. Because they are from the same family of languages, they have similar structures – the verb and noun class affect sentences in the same way. Therefore, the isiZulu verbalisation patterns could act as a base for Runyankore. This bootstrapping approach greatly reduces time for developing grammar systems and helps under-resourced languages like Runyankore.

(Mahlaza and Keet, 2019) took advantage of the linguistic similarity between isiXhosa and isiZulu's grammar. They developed CFG rules to generate verbs for weather reports in both isiZulu and isiXhosa (the two languages shared 42 of the rules). Since they concentrated on verbs only, they were able to cover more than one tense, including past, present and future. The evaluation of their CFG showed that the syntax and semantics of the generated strings were mostly correct for both languages. A corpus was developed, however, their corpus is specific to a weather domain and does not apply to mathematics.

3.3 Corpora

There is a small isiZulu Definite Clause Grammar (DCG) and parts-of-speech (POS) tagger, Ukwabelana (Spiegler *et al*, 2010) [15]. The DCG does not cover the same complexity of the verb that [11] does, but covers more POS. The Ukwabelana is an open-source isiZulu corpus that contains 100 000 word-types (morphologically labeled words) and 30 000 POS tagged sentences, from the isiZulu Bible. It is the most comprehensive corpus available but it's outdated and has limitations. The Human language technologies (HLT) project is currently working on a corpus for African Languages.

4 VERBALISATION OF MATHEMATICS

4.1 The Syntax and Semantics of Mathematical Expressions

Mathematics has its own specific, scientific language, that is a subset of a natural language. Just like any natural languages, the linguistics are tricky to formalise – sentences can be phrased in many ways and they may contain ambiguity. This section addresses the challenges in verbalising mathematical expressions and what solutions are possible.

Currently there exists thousands of LaTeX formatted math formulae on Wikipedia, (over 350 000) that can be translated into text descriptions. Below is an example of a LaTeX formatted formula [12]:

```
\gamma =\frac{1}{\sqrt{1-\frac{v^2}{c^2}}}
```

Which represents this expression:

$$\gamma = \frac{1}{\sqrt{1-\frac{v^2}{c^2}}}$$

Blind people can already access these mathematical expressions but listening to the LaTeX format is inefficient for several reasons [13]. It requires some knowledge in LaTeX, it is slow to listen to, it sounds unnatural, and is error-prone due to its typographical nature. The typographical language of LaTeX leaves ambiguity for some expressions, like f^ {-1}.

f^ {-1} can be read as:

(1) the variable f to the power of -1.

(2) the inverse of function f.

These LaTeX formatted formulae can be extracted, translated into a less ambiguous format, and inputted into an NLG system to produce a more natural sounding text description. A way to unambiguously represent the presentation and semantics of mathematics is through Mathematical Markup Language (MathML), a W3C standard format. MathML is an XML application for describing mathematical expressions - more specifically, the semantics of the expression are handled by Content MathML (CMML). There are a few LaTeX to MathML translators, like LaTeXML (Miller, 2007) and SnuggleTex. Below is an example of a LaTeX to MathML conversion by LaTeXML that solves the ambiguity of the inverse function problem. Where (1) is the inverse function and (2) is the exponent of -1; *ci* tags are variables and *cn* tags are numbers.

(1)
```
<math>
  <apply>
    <inverse/>
    <ci> f </ci>
  </apply>
</math>
```

(2)
```
<math>
  <apply>
    <power/>
      <ci> f </ci>
      <cn> -1 </cn>
  </apply>
</math>
```

(Stamerjohanns *et al*, 2009) conducted a comparison study on five different LaTeX to MathML convertors. The features that were tested included: documentation, coverage, percentage of incomplete conversion, percentage of errors, percentage of successful conversion, and quality. Out of the 5 convertors, LaTeXML scored well in the coverage category, with the highest rate of successful conversions (89%), and the third lowest percentage of errors (35%). It also had a higher scoring quality of the resulting output and had an advantage of ease of use.

There is also the issue of ambiguity in the precedence of expressions, when reading the expression out loud. $x + \frac{y}{z}$ is read aloud as "x plus y over z". To a listener, this could be interpreted as $\frac{x+y}{z}$. (Mazzei *et al*, 2019) suggested a solution of vocalising brackets, "x plus open parenthesis y divided by z close parenthesis". (Ferres and Sep´ulveda, 2011) suggested pauses instead of brackets. As well as being ambiguous, sentences can also be phrased differently. As described in section 2, during the lexicalisation task the NLG system handles the choice between alternative words. The same applies to mathematical expressions, "*a quarter"* is synonymous with "*one over four".*

For the NLG system we will need an isiZulu lexicon with all the vocabulary for math verbalisation. As far as we know there is not such a lexicon available, so we need to compile one ourselves. The largest corpus available is Ukwabelana, which is not applicable to mathematics since it is sourced from Biblical stories. Phrases like "a quarter", "a logarithmic function", "x cubed", "the integral of" and "the square root of" are examples of what is not in the Ukwabelana and needs to be translated into isiZulu.

4.2 Natural Language Generation of Mathematical Expressions

There has been work done in mathematical verbalisation using the NLG realisation approaches discussed in section 2 for Spanish [12] and Italian [13].

(Ferres and Sep´ ulveda, 2011) approached the problem of math verbalisation in Spanish, with a template-based system, called MathAcc. When deciding on the wording of the templates, they surveyed a group of 60 people, asking them to describe expressions that were given to them from several mathematical categories. (Ferres and Sep´ ulveda, 2011) then chose their template's wording based on the most popular response in each category. They cleaned the LaTeX formulae from Wikipedia into a CMML format, with the tool SnuggleTex. There were some shortcomings, they could only verbalise formulae for which there is a LaTeX to CMML translation. There was failure to translate some works (like the binomial coefficient), so they only generated descriptions for around 66% of Wikipedia's formulae. Of the generated test outputs, roughly 10% were incorrect. Their findings concluded that a better translation of LaTeX to CMML or another format completely, would have yielded better results. One fault with the paper is that they never evaluated their results with visually impaired users, and instead used automated evaluation.

(Mazzei et al, 2019) used a grammar-based system to generate Italian descriptions of math expressions, using SimpleNLG (described in section 2.2) for the realisation engine. They translated the LaTeX formulae into CMML using LaTeXML. They organized the expressions as following: treat all numbers and variables as nouns, and all operators as verbs.

- Relational operators as copula sentences ("a is equal to b")
- Algebraic operators as declarative sentences ("a plus b").
- Logical operators as conjunctions ("a or b"),
- Elementary operators ("The sine of"), sequence ("The limit of"), calculus ("The integral of") as noun phrases.
- Pairs and conditional sets as reduced relatives ("The point x and y").

SimpleNLG already has a basic Italian vocabulary but it does not include specialised mathematical terms. They fixed this by appending their own constructed lexicon for mathematics to the existing lexicon. To test the generated text's understandability, they created a survey of 25 questions – for each question, an audio description of an expression was played, and the participant was required to input a corresponding LaTeX representation. The participants were visually impaired and had knowledge in mathematics. The test measured their understandability based on the exact match or similarity between the expected answer and the participant's answer. The simple expressions mostly held exact matches, but more complex expressions were slightly misunderstood. Their evaluation determined that 71% of simple expressions were precisely understood, meaning participants had a fairly good understanding.

DISCUSSION

This review discussed the pipeline architecture proposed by (Reiter and Dale, 1997) and the numerous ways to implement the linguistic realisation stage. Templates are suitable when there is no need for linguistic variation and the domain is small. Grammar-based systems rely on a languages grammar rules for surface realisation - it is time-consuming and requires sufficient data to develop but has the trade-off of a better quality of generated text. Statistical approaches use a statistical model to optimally generate text, by relying on a data resource like corpora. Grammar-infused templates are templates that have grammar rules encoded into them, they work well for low-resourced languages as it enables grammar detachability. However, not all these approaches to realisation are applicable to isiZulu for two reasons:

(1) IsiZulu has a morphologically complex grammar.
It is the noun classes and complex verb that restricts how the grammar can be implemented - grammar engines, like GF and SimpleNLG were encoded for Indo-European languages, which cannot handle the complexity of isiZulu's grammar. Therefore, none of the grammar tools are applicable to this project. Using templates on their own will not suffice.

(2) isiZulu is an under-resourced language.
Without data sources like corpora, statistical approaches cannot be used. The Ukwabelana corpus is the largest isiZulu corpus available but is limited in its applicability to certain domains.

Author/Year	Methods/Frameworks	Suitability for isiZulu
Keet and Khumalo (2017) [2]	A grammar-infused template approach using verbalisation patterns with OWL ontologies.	**Yes.** These grammar rules are available for use and can be extended for mathematical terms.
A. Belz (2008) [18]	A statistical method for realisation.	**No**. A statistical approach requires a sufficient data resource.
Ferres and Sepúlveda (2011) [12]	A template-based approach.	**No**. The template approach is not feasible for isiZulu's rich morphology.
Mazzei, *et al* (2019) [13]	A grammar-based system, using a SimpleNLG realisation engine.	**No**. Grammar engines are not suited to isiZulu's grammar.
Spiegler *et al* (2010) [15]	An IsiZulu POS tagger and small DCG.	**Yes**. The corpus can be a starting point, but needs mathematical terms appended.

Table 1: Articles are compared in relation to their ability to support our work, isiZulu text generation.

Table 1 summarises the discussed article's frameworks and NLG methods, and states whether their methods suit the generation of isiZulu text.

Grammar-infused templates can accommodate for isiZulu's grammar, (Keet and Khumalo, 2017) generated isiZulu text using verbalisation patterns and a CFG to represent the verb. The generated text was understandable for simple sentence construction but became ambiguous for more complex sentences. However, the ambiguity was a consequence of varying dialects. The grammar rules are available for use but are limited to one tense (present tense). (Mahlaza and Keet, 2019) developed more than one tense for the verb, but this is not necessary for verbalising mathematical expressions, as they are conventionally written in present tense. A useful technique for developing low-resourced languages is bootstrapping, where instead of starting development from scratch, existing patterns can be tailored to fit other languages with similar linguistics.

Overall, there have been several successful attempts at verbalising mathematical expressions for the visually impaired. There is ambiguity in how mathematical expressions are formatted in LaTeX, which is an issue for generating a comprehensive description. This can be fixed by converting the LaTeX expressions into a MathML format. However, there is still room for improvement in translating from LaTeX to MathML, as some LaTeX files did not translate. When comparing (Ferres and Sep´ ulveda, 2011) and (Mazzei et al, 2019) we need to consider that their evaluation methods were different: (Ferres and Sep´ ulveda, 2011) used automated evaluation on their generated text, to check grammatical correctness and coverage, but did not involve end-users. (Mazzei et al, 2019) concentrated more on understandability for end-users, rather than just coverage. They played audio clips of the text to check how well participants understood. Their percentage of correct output is lower than (Ferres and Sep´ ulveda, 2011), however, (Mazzei et al, 2019) involved visually impaired people, which are the real end-users, deeming their results more meaningful. The results indicate for both papers that simple expressions were understandable but complex expressions were ambiguous and need improvement. (Mazzei et al, 2019) used a method of categorizing the mathematical expressions as noun phrases, declarative and such, this approach will be useful to our project, as the sentence components need to be well defined as POS for the grammar rules to construct them. (Mazzei et al, 2019) appended mathematical terms to an existing lexicon, which is another method we could use for isiZulu, to reduce development time.

CONCLUSIONS

There has been ample research in the general field of NLG and in generating Niger Congo languages, such as isiZulu, isiXhosa and Runyankore. Most traditional NLG methods fail for these agglutinating languages, but they can be successfully modelled through verbalisation patterns with OWL ontologies and using a CFG for generating the verb. All the existing isiZulu grammar rules displayed good results in understandability during evaluation tests. The grammar rules have been used for a variety of applications, namely weather bulletins, but there has been no research done on verbalising mathematics for isiZulu.

Low-resourced languages are challenging to develop for software applications, so any grammar rules that are developed should be reusable across many types of domains. Grammar-infused templates cater for this reusability, by allowing grammar detachability.

Data sources for isiZulu are scarce and have limitations. The open-source isiZulu corpus, Ukwabelana, is not applicable to mathematics but can act as a starting point to append mathematical terms.

We looked at the LaTeX to MathML translators SnuggleTex and LaTeXML; LaTeXML proved to be a good contender for understandability and coverage. For future work, perhaps a better LaTeX to MathML converter can be investigated further.

REFERENCES

[1] A. Gatt and E. Krahmer: Survey of the State of the Art in Natural Language Generation: Core tasks, applications, and evaluation. J. Artif. Intell. Res. 61: 65-170 (2018))

[2] C. M. Keet and L. Khumalo: Toward a Knowledge-to-Text Controlled Natural Language of isiZulu. Language Resources and Evaluation 51.1 (2017), pp. 131–157

[3] C. Maria Keet: NLP for African (Nguni) languages. Guest lecture NLP course Poznan University of Technology, (2018): http://www.meteck.org/teaching/NLPAfricaPoznan18.pdf

[4] E. Reiter and R. Dale: Building Applied Natural Language Generation Systems (1997).

[5] Zola Mahlaza. "Grammars for Generating isiXhosa and isiZulu Weather Bulletin Verbs". MA thesis. Cape Town, South Africa: University of Cape Town, 2018

[6] E. Reiter. NLG vs. templates. In Proceedings of the 5th European Workshop in Natural Language Generation, pages 95–105, Leiden, NL, May 1995.

[7] A. Gatt and E. Reiter: SimpleNLG: A realisation engine for practical applications - Proceedings of the 12th European Workshop on Natural Language Generation, Athens, Greece, 30 – 31 March 2009. pages 90–93.

[8] E. Reiter: An Architecture for Data-To-Text Systems. Proceedings of the Eleventh European Workshop on Natural Language Generation, 2007

[9] C. M. Keet and L. Khumalo. "Basics for a Grammar Engine to Verbalize Logical Theories in isiZulu". In: Rules on the Web. From Theory to Applications - 8th International Symposium, RuleML 2014, Co-located with the 21st European Conference on Artificial Intelligence, ECAI 2014, Prague, Czech Republic, August 18-20. 2014, pp. 216–225.

[10] C. M. Keet and L. Khumalo. "Toward Verbalizing Ontologies in isiZulu". In: Controlled Natural Language - 4th International Workshop, CNL 2014, Galway, Ireland, August 20-22, 2014. Proceedings. Ed. by B. Davis, K. Kaljurand, and T. Kuhn. Vol. 8625. Lecture Notes in Computer Science. Springer, 2014, pp. 78–89.

[11] C. M. Keet and L. Khumalo. "Grammar rules for the isiZulu complex verb". In: Southern African Linguistics and Applied Language Studies 35.2 (2017), pp. 183–200.Conference Name: ACM Woodstock conference

[12] L. Ferres and J.F. Sepúlveda: Improving accessibility to mathematical formulas: the Wikipedia math accessor. Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, pp. 1-9. 2011.

[13] A. Mazzei, M. Monticone, C. Bernareggi: Using NLG for speech synthesis of mathematical sentences. Proceedings of the 12th International Conference on Natural Language Generation, pages 463–472. 2019.

[14] J. Byamugisha, C. M. Keet, and B. DeRenzi: Bootstrapping a Runyankore CNL from an isiZulu CNL. Controlled Natural Language - 5th International Workshop, CNL 2016, Aberdeen, UK, July 25-27, 2016, Proceedings. Ed. by

B. Davis, G. J. Pace, and A. Z. Wyner. Vol. 9767. Lecture Notes in Computer Science. Springer, 2016, pp. 25–36

[15] S. Spiegler, A. van der Spuy, P. A. Flach: Ukwabelana - An open-source morphological Zulu corpus - Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 1020–1028, Beijing, August 2010

[16] K. Van Deemter, Emiel Krahmer, and M. Theune. “Real vs. Template-based Natural Language Generation: A False Opposition?” In: Computational Linguistics 31 (1 2005), pp. 15–23.

[17] Z. Mahlaza and C. Maria Keet: A classification of grammar-infused templates for ontology and model verbalisation, 2019.

[18] A. Belz: Probabilistic Generation of Weather Forecast Texts - Proceedings of AACL HLT 2007, pages 164–171, Rochester, NY, April 2007.

[19] Keet, C.M., Xakaza, M., Khumalo, L.: Verbalising OWL ontologies in isiZulu with Python. In: The Semantic Web: ESWC 2017 Satellite Events. LNCS, vol. 10577, pp. 59–64. Springer (2017), 30 May - 1 June 2017, Portoroz, Slovenia

[20] H. Stamerjohanns, D. Ginev, C. David, D. Misev, V. Zamdzhiev, M. Kohlhase: MathML-aware Article Conversion from LATEX, A Comparison Study, (2009), Towards a Digital Mathematics Library, pp. 109–120.

[21] Z. Mahlaza and C. M. Keet: A method for measuring verb similarity for two closely related languages with application to Zulu and Xhosa, (2019), pp 34 -54