Generating Natural Language IsiZulu Text From Mathematical Expressions

Project Proposal

Shannon Smith
Computer Science
University of Cape Town

1 PROJECT DECRIPTION

There is a global effort to make textbooks available online for visually impaired people – screen readers can turn these texts into synthesised speech, but stumble over LaTeX mathematical formulae. The South African Centre for Digital Language Resources (SADiLaR) supports research on all aspects of natural language processing – automatic speech recognition, text-to-speech systems (TTS), and spell checking and much more. The text-to-speech systems in their repository can convert text into synthesised speech for most of South Africa's official languages. However, listening to mathematical formulae in a LaTeX format is inefficient for several reasons. It requires some knowledge in LaTeX, it is slow to listen to, it sounds unnatural, and is ambiguous due to its typographical nature (Mazzei et al, 2019).

Natural language generation (NLG) could be a solution to this problem. NLG is a subfield of computer linguistics and artificial intelligence that maps some input data to readable, natural language text (Reiter and Dale, 1997). The input data is usually in a non-linguistic format, like numerical data, graphs, or images, but the output is always text. The mathematical formulae can be translated by an NLG system into natural language text, which is more understandable when read aloud than its LaTeX equivalent. For this reason, an NLG system will be built.
This project cannot tackle all 11 official languages, so it will focus on isiZulu. Although isiZulu is spoken by the majority (23% of the population), it is still under-resourced in software applications (Keet and Khumalo, 2017). The biggest barriers for developing applications in isiZulu is the lack of resources and the complexity of its morphology. Currently there are no systems in place to translate mathematical formulae into isiZulu natural language text.

The techniques used in existing maths verbalisers are language dependent and existing tools are suited to Indo-European languages. These tools fail for isiZulu's agglutinating grammar. Unlike Indo-European languages, which express the tense, negation etc. in separate components, agglutinative languages use prefixes and suffixes. It is the 17 noun classes and complex verb that restricts how the grammar can be implemented, this complexity does not allow for realisation methods like templates or grammar engine tools (Keet and Khumalo, 2017).

Instead grammar-infused templates can be used, as shown by (Keet and Khumalo, 2017) for a different domain. Grammar-infused templates are templates that have grammar rules on top of them (Mahlaza and Maria Keet, 2019).

2 PROBLEM STATEMENT

Human Language Technology (HLT) already have text-to-speech systems for multiple official languages, but these systems cannot produce useful or understandable descriptions of mathematical formulae. This leaves visually impaired users unable to efficiently listen to mathematical formulae. The central issue we are confronting is that there are no systems in place to verbalise mathematical expressions for visually impaired, isiZulu speakers.

2.1 Aims

The goal of this project is to bridge the gap in mathematical verbalisation for visually impaired isiZulu speakers, so that users have improved access to mathematics. This can be possible by building an NLG system that can generate understandable and useful isiZulu text from mathematical expressions. These descriptions must be natural language sentences with standard mathematical semantics.

2.2 Research Questions

- Can an NLG system be built that generates understandable and accurate isiZulu text descriptions of mathematical expressions?

3 PROCEDURES AND METHODS

This section discusses the methods and implementation strategies used to achieve the project's aims and answer the research questions.

3.1 Collect and Clean Mathematical Expressions

The first step is to collect mathematical expressions from Wikipedia dumps. There are around 356 000 formulae available, which is an impossible number to implement in this project's timeline. Therefore, to make this project feasible, we will select around nine operations to verbalise. This selection will be based

on how popular the operation is (How frequently it occurs in the Wikipedia repository).

The LaTeX formatted mathematical expressions are ambiguous and need to be cleaned into a less ambiguous form before the verbaliser can use them. The MathML format makes the semantics more explicit for the NLG system, by using tags for the operators (like <power>, <inverse>, etc.), which solves the problem of ambiguity. The LaTeX expressions will be translated into MathML with a tool called LaTeXML.

3.2 Gather Data for a Corpus

We will compile an isiZulu corpus of mathematical semantics, so the verbaliser can use it to construct sentences. This data resource will also contribute to improving the availability of resources for South African languages.

We are taking a corpus-based approach to NLG, where we ask domain experts to handwrite examples of appropriate output texts (Reiter and Dale, 2000). In this case, the domain experts are isiZulu linguists or teachers of mathematics who speak isiZulu. We will recruit these people with the assistance of Dr. Langa Khumalo, who is a linguist. Once this data has been gathered, we can manually design templates for the verbaliser.

3.3 Implement the Verbaliser

The verbaliser will run as a PC application from the terminal. The features are simple: a user inputs the name of an XML file containing a MathML expression; the system then outputs a new file containing a corresponding isiZulu text description. The user can specify the name of this output file on the command line. To ensure the system runs correctly, we will perform unit tests.

The implementation strategy is an iterative and incremental approach. We implement the verbaliser in cycles, starting with three initial mathematical operations, and adding new operations in each cycle. An iteration involves the following:

- Three mathematical operators are implemented.
- Once the operators are implemented, the generated text will be checked by our supervisor, who is an isiZulu speaker.
- In the next iteration, any feedback and changes are added, and another three operations are implemented.

This is so any problems can be caught and fixed early on. There will be three iterations over two months, producing a total of nine operations.

The system's architecture is a modular pipeline (See figure 1). It will be designed with distinct, well-defined, and easily integrated modules. This approach was chosen, as it is easier to develop, modify and debug the code separately. We will build a module for the text planner, sentence planner and linguistic realiser. The text planner will take the XML snippet as input, parse the MathML expression and choose the appropriate isiZulu words for that operation. The sentence planner takes these words as input and arranges them in a correctly ordered sentence. The linguistic realiser module will take the output from the sentence planner and generate a grammatically correct and complete sentence using the isiZulu templates. Lastly, the realiser ensures agreement between the nouns, verbs, and tenses by following the isiZulu morphological rules.

Figure 1: The Pipeline Architecture for the Mathematics Verbaliser, Adapted from (Reiter and Dale, 1997)

3.4 Text Evaluation

This project concentrates on whether the resulting system produces useful text, where we define useful as understandable and an accurate description. Therefore, the text must be evaluated in relation to how useful it is for the users.

3.4.1 Human-based Evaluation

Once the system is implemented, an online human evaluation will be performed to evaluate how understandable and accurate the text descriptions are.

The evaluation will consist of two questions per mathematical operation (with a total of 9 operations). For question one, the participants will read the generated text description of an expression and type out the corresponding formula into the Google Doc. For question two, the participants will rate the understandability of the generated text. A Likert scale can be used with ratings of 1 to 5 for each expression, where 1 is strongly disagree, 2 is disagree, 3 is unsure, 4 is agree and 5 is strongly agree. An example of the Likert scale is found in Appendix. A.

The participants will be emailed a link to a Google Doc, where they can fill out the online questionnaire. The participants can communicate over email when they have completed the evaluation, or if they have any questions during the evaluation. The questionnaire will keep the participants anonymous, as no personal or identifying information is required with the submission.

After the evaluation, the finding will be analysed. The formulae obtained from the participants for each description can be compared to the expected formula, to see how many are an exact match. A high number of matches indicates the text is an accurate description. The answers from the Likert scale questionnaire can be tallied for each rating (1 to 5). The frequency of each rating for an expression will give an idea of how understandable that operation is. The overall understandability of the text can be determined by tallying the ratings across all the operations.

3.4.2 The Participants

The end users of the proposed system are isiZulu speakers in the blind community. However, this evaluation focuses on the verbalisation of mathematics, rather than the vocalisation, so non-blind people will be used as participants.

The target population are students/teachers/academics in a mathematics field. Therefore, the test participants must be fluent in isiZulu and have completed at least first-year mathematics. Knowledge in higher level mathematics will be the criterion, to ensure they will have knowledge of all the operations.

Test users can be recruited through UCT, friends and snowball sampling. To draw attention to our study, we can put up adverts around campus and send an email to the Science Faculty students. We are aiming to recruit between five and ten people to join the test.

Participants will also be offered a cash incentive of R50 for their time. After the evaluation, they can email us their banking details and the money will be electronically transferred to them. Once transferred, they will be emailed a proof of payment and their details will be deleted.

4 RELATED WORK

Previous work done for generating text in isiZulu for other domains has used pattern-based methods for verbalising ontologies (Keet and Khumalo, 2017). The verbaliser uses a grammar-infused template, with embedded rules for agreement between words and partially attached rules for noun pluralisation. The isiZulu verb is represented as a CFG and their production rules cover subject and object concords, negation, present tense, aspect, and mood. These grammar-infused templates enable the NLG system to use complex grammar rules where it is needed, and simpler, less expensive templates where it is not (Reiter, 1995). Templates on their own are not applicable to isiZulu's complex morphology and developing a full grammar system is too time-consuming for this project. Therefore, this project's verbaliser will make use of grammar-infused templates to create isiZulu text.

(Keet and Khumalo, 2017) evaluated their text with isiZulu speakers. Their outputs were understandable for simpler sentences, but sometimes become ambiguous for more complex sentences. This may present some challenges for our verbaliser, as the complex math descriptions need to unambiguous and understandable.

As far as we know, there are no existing math verbalisers for isiZulu. Previous work in math verbalisation for other languages have used NLG techniques that are not applicable to isiZulu's grammar. However, their methods of collecting and cleaning mathematical expressions, and evaluating the generated text are relevant to this project.

4.1 Collect and Clean Mathematical Expressions

(Mazzei et al, 2019) and (Ferres and Sepúlveda, 2011) collected LaTeX expressions from Wikipedia and cleaned them into a MathML format. The reason for this, is that the typographical language of LaTeX leaves ambiguity for some expressions. Like f^{-1} can be read as:

(1) the variable f to the power of -1.
(2) the inverse of function f.

A way to unambiguously represent the presentation and semantics of mathematics is through Mathematical Mark-up Language (MathML), a W3C standard format. (Mazzei et al, 2019) translated the LaTeX formulae into MathML using the tool LaTeXML (Miller, 2007), and this project will use this tool too.

Below is a MathML expression that shows how the ambiguity of the inverse function problem can be solved. 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>
```

We are selecting nine operations to implement, based on how frequently they appear in the Wikipedia repository. We adapted this approach from (Ferres and Sepúlveda, 2011), who determined that most frequent keywords in LaTeX formulae found on Wikipedia are subscript, superscript, fractions, square roots, sum, sin, cos and partial derivatives. They created their templates based on these popular operations.

4.2 Text Evaluation

(Mazzei et al, 2019) tested the understandability of their generated text, with visually impaired participants with knowledge in mathematics. This method of involving real end users is a meaningful way to measure usefulness. (Reiter and Dale, 2000) recommends including human evaluations when the usefulness of the system needs to be measured, rather than using some metric.

(Mazzei et al, 2019) created a survey of 25 questions – for each question, an audio description of an expression was played, and the participants were required to input a corresponding LaTeX representation. The test measured their understandability based on the exact match or similarity between the expected answer and the participant's answer.

Asking users to input a LaTeX representation is unnecessary (it's hard to find users with LaTeX knowledge), instead users can read the descriptions and just write down a formula. Their user evaluation determined that 71% of simple expressions were precisely understood, meaning participants had a fairly good understanding of simple expressions. Since, our project's evaluation is under similar conditions, we want to compare our evaluation results with this paper's result.

5 ETHICAL, PROFESSIONAL AND LEGAL ISSUES

Ethical considerations are necessary as human beings are involved in the evaluation tests, and thus we require an ethics clearance. Participation is entirely voluntary, and participants are free to withdraw at any time. Prior to the evaluation tests the participants will be informed on what the purpose of the evaluation is and then we will obtain the participant's written consent. They will remain anonymous as we do not need their personal details and there will be no video or audio recordings during the evaluation.

All research and development will be conducted in compliance with the third-party use specifications of the software libraries. All code written will be released under the University of Cape Town's policy for the creative commons license.

6 ANTICIPATED OUTCOME

6.1 System

The anticipated outcome is a fully functional mathematical verbaliser. The system will take a MathML expression (an XML file) as input and produce an output file containing natural language isiZulu text.

There are some expected challenges. The first challenge is that several LaTeX expressions may not translate into a MathML format. These can be translated by hand, but if the difference is significant another translator should be considered. The second challenge is making sure the grammar of the output text is correct enough, to ensure that it does not affect the understandability of the description.

Another challenge we anticipate is that more complicated mathematical expressions may produce ambiguous or unintelligible descriptions. This is for two reasons, (1) The descriptions become harder to follow as the sentences become longer. (2) Evaluation of the grammar rules in another context, showed they became slightly misunderstood for more complex sentences.

6.2 Impact

Since there are no other isiZulu math verbalisers, we cannot predict an expected result. However, previous attempts at math verbalisation in other languages, showed that on average 70% - 79% of the generated text is understandable for users. We will be interested to see how our results compare to that.

We hope this project will be beneficial to isiZulu speakers in the blind community, by helping them to improve their mathematical literacy. It contributes to the global effort in improving resources for visually impaired. It also contributes to improving resources for otherwise low-resourced languages in South Africa.

6.3 Key Success Factors

We care about whether the system produces useful, accurate, and understandable texts that genuinely help the blind community. Therefore, we will judge the success of the NLG system based on whether it is understandable for the end users.

7 PROJECT PLAN

7.1 Risks

There are several possible risks to the success of this project. These risks and their associated mitigation, monitoring and management strategies have been listed in a risk matrix. This matrix can be seen in Appendix. B. The risks are rated (low, medium and high) by their probability of occurring and their impact if they do occur.

7.2 Timeline and Milestones

The project runs from the 13th of May until the 19th of October, a timeline of this project is represented as a Gantt chart in Appendix. C. The timeline covers all the methods and procedures and the estimated time to complete them. The project's milestones and deliverable deadlines are also shown.

7.3 Resources Required

The resources required for this project are minimal. No extra equipment is needed, just a regular PC will suffice. Third party software such as the LaTeX to MathML convertor, LaTeXML will be used.

We require data resources, like the LaTeX formulae taken from Wikipedia repositories and the linguistic data collected from teachers of mathematics or isiZulu linguists. The isiZulu verbalization patterns and CFG verb developed by (Keet and Khumalo, 2017) will be extended to fit our domain.

7.4 Deliverables

There are several deliverables that need to be produced throughout the project's timeline. These deliverables are listed in table 1. with their respective deadlines.

Table 1: A List of Deliverables and their Due Dates.

Deliverable	Due Date
Project Proposal	2nd June
Ethics Clearance Application Submission	8th June
Initial Software feasibility Demonstration	3rd – 11th August
Final Complete Draft of paper	4th September
Project Paper Final Submissions	14th September
Project Code Final Submission	21st September
Final Project Demonstration	5th – 9th October
Poster Due	12th October

Web Page	19th October

REFERENCES

[1] 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

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

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

[4] 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

[5] 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.

[6] 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.

[7] 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

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

[9] 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.

[10] 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.

[11] R. Dale and E. Reiter. “Building natural language generation systems”. In: Cambridge University Press (2000).

APPENDIX

Appendix A – Likert Chart

Figure 2: An Example of a Questionnaire During the Human Evaluation Test

		Strongly Disagree	Disagree	Unsure	Agree	Strongly Agree
"The square root of x"	1. This expression is understandable.	1	2	3	4	5

Appendix B – Risk Management

Table 2: Risk Matrix Detailing Project Risks and Associated Mitigation Strategies.

Risk	Probability	Impact	Mitigation	Monitoring	Management
Failure to finish project before deadline.	Medium	High	Regular meetings and updates with supervisor to ensure tasks are done in time.	Check that Gantt chart tasks are being finished in time, milestones are being met and that the schedule is being followed	Discard any unnecessary features and focus on core functionality.
Text requirements are not being met before human evaluations.	Medium	High	Allow enough time to be able to make changes in the next iteration if requirements are not being met.	Use feedback from supervisor to judge if changes need to be made.	Reduce the number of operations to be implemented and focus on perfecting a few.
Inadequate Developer Skills	Low	High	Regularly ask the supervisor for guidance during the development stage.	Check if the developer is unable to handle the demands of the software implementation.	Propose alternative, easier development route.
Failure to acquire adequate test users	Low	Medium	Start looking for users early in the project.	Unable to find test users or users drop out of the evaluation.	Use automated evaluation instead of human evaluation.
Failure to translate some LaTeX expressions into MathML	Low	Low	Start translating LaTeX expressions early so the project can recover from delays.	More than 30% of the chosen expressions do not have an MathML translation.	Translate the expressions by hand, or if there are too many, use another translator.
Scope creep	Low	Low	Refer to original aims, to ensure unnecessary features are not being added.	Report increasing activities that are not in original scope.	Reduce scope and concentrate on finishing the bare requirements.

Appendix C – Gantt Chart

Figure 3: Gantt Chart Showing Project Timeline and Milestones