



Faculty of Sciences
Computer Science & Information Systems

A Context-Aware Lemmatization Model for Setswana Language Using Machine Learning

by

Kgosiyame Ditiro Bafitlhile

Reg. No: 16100260

BSc (Computer Systems Engineering), Sunderland, UK

A Thesis Submitted to the Faculty of Science in Partial Fulfillment of the Requirements
for the Award of the Degree of Masters of Science in Computer Science of BIUST

Supervisor(s):

Dr. Dimane Mpoeleng

Department of Computer Science & Information Systems,
Faculty of Sciences, BIUST

E-mail: mpoelengd@biust.ac.bw

Dr. Zhivko Nedev

Department of Computer Science & Information Systems,
Faculty of Sciences, BIUST

E-mail: nedevz@biust.ac.bw

Dr. Keletso Letsholo

Department of Computer Science & Information Systems,
Faculty of Sciences, BIUST

E-mail: letsholok@biust.ac.bw

August 2022

DECLARATION REGARDING THE WORK AND COPYRIGHT

Candidate Kgosiyame Ditiro Bafitlhile

Student ID: 16100260

Thesis Titled: A Context-Aware Lemmatization Model for Setswana Language Using Machine Learning

I **Kgosiyame Ditiro Bafitlhile**, certify that the Thesis is all my own original work and that I have not obtained a degree in this University or elsewhere on the basis of any of this work.

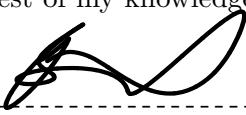
This thesis is copyright material protected under the Berne Convention, the Copyright and Neighbouring Rights Act, Act. No. 8 of 2000 and other international and national enactments, in that behalf, on intellectual property. It must not be reproduced by any means, in full or in part, except for short extracts in fair dealing; for researcher private study, critical scholarly review or discourse with an acknowledgement, without the written permission of the office of the Postgraduate School, on behalf of both the author and the BIUST.

Signed: 

Date: 25/08/2022


Primary Supervisor(s) Dr. Dimane Mpoeleng, Dr. Zhivko Nedev and Dr. Keletso Letsholo

I **Dr. Dimane Mpoeleng**, hereby confirm that I have inspected the above titled thesis and, to the best of my knowledge, it is based on the original work of the candidate.

Signed: 

Date: 25-Aug-2022

I **Dr. Zhivko Nedev**, hereby confirm that I have inspected the above titled thesis and, to the best of my knowledge, it is based on the original work of the candidate.

Signed: 

Date: 25-Aug-2022

I **Dr. Keletso Letsholo**, hereby confirm that I have inspected the above titled thesis and, to the best of my knowledge, it is based on the original work of the candidate.

Signed: 

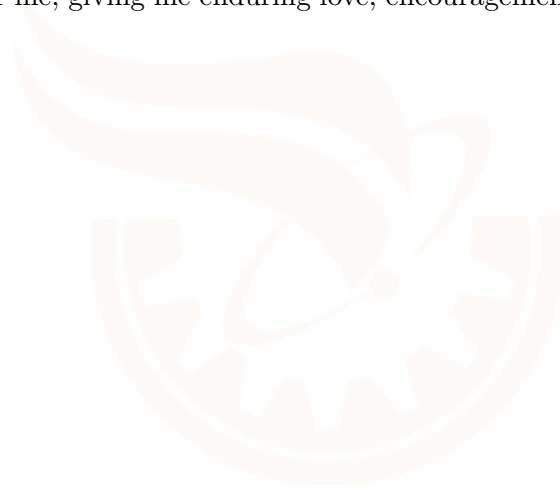
Date: 25-Aug-2022

Acknowledgements

The work presented in this thesis was conducted jointly with my supervisor, Dr. Keletso Letsholo. Through good and bad times, he has guided and supported me, I am forever thankful for that. I would also like to acknowledge my co-supervisors Dr. Zhivko Nedev and Dr. Dimane Mpoeleng for their constructive discussions, which yielded valuable insights into various aspects of the research.

I would also like to thank Lotsane Senior Secondary School Setswana teachers with their expertise in Setswana language. I also thank my Masters Colleagues, Nonofu Mautle, Kabelo Madise, Ronnie Nkhorie, Thamang Madile and Godfrey Mothonjeni for their companionship, and words of encouragement.

Lastly, I want to express my gratitude to my parents Modise Bafitlhile and Sebonile Bafitlhile for always being there for me, giving me enduring love, encouragement and support through my education process.



Abstract

Lemmatization is an important task which is concerned with making computers understand the relationship that exists amongst words written in natural language. It is a prior condition needed for the development of natural language processing (NLP) systems such as machine translation and information retrieval.

In particular, Lemmatization is intended to reduce the variability in word forms by collapsing related words to a standard lemma. There is a limited research on lemmatization of Setswana language. A large part of the available research on Setswana lemmatization relies on rule driven strategy, which takes time to construct, lacks context of how words are used, and needs extremely qualified language skills. Moreover, it has been discovered that the treatment of language with hand coded regulations lacks generalization component as it requires a continual redesign every time new data appears and this complicates the scalability of systems. With such rich vocabulary and complex morphology, lemmatization of Setswana cannot be easily unraveled using explicit rules developed by programmers.

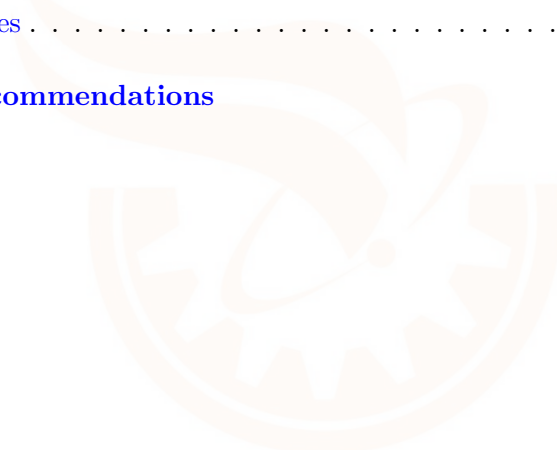
In this thesis we describe how a supervised machine learning approach that employs the use of Naive Bayes algorithm can solve Setswana lemmatization with regard to how words are used in sentences. The contribution of this study includes; first, context aware lemmatization model, that handles most of the morphologically productive classes. Second, we experiment with the strongest multi-class algorithm Naive Bayes, which to our best knowledge has never been used to address lemmatization in Setswana. The accuracy of the lemmatization model obtained from the experiments reached 70.32%. The model shifts from entirely hand programmed rules and is able to lemmatize words based on the context how they are used. In Setswana lemmatization should be done according to sentence intension, the model again ensures that as long as the data is a good example of the goal concept the generalization is simultaneously created, which allows the model's future performance to continue improving.

Furthermore, given that this is a young area of research with no standard datasets for training and testing, we also contribute with a considerable medium sized dataset which remains a coveted resource for research community. The experimental results obtained from this study shows that machine learning approaches are more reliable than rule based approaches in lemmatizing Setswana inflectional words with regard to the context of how they are used.

Contents

Contents	v
List of Figures	v
List of Tables	vi
1 Introduction	1
1.1 Introduction	1
1.2 Statement of the Problem	4
1.3 General Objective	4
1.4 Specific Objective	4
1.5 Research Questions	4
1.6 Expected Outcome	5
1.7 Justification of the Study	5
1.8 Outline of the Thesis	6
2 Background	7
2.1 Natural Language Processing	7
2.2 Morphology	8
2.3 Context in Natural Language	9
2.4 Setswana language	10
2.5 Naive Bayes (NB) algorithm	11
3 Related Work	13
3.1 Rule-based approach	13
3.2 Statistical machine learning approach	14
3.3 Available Setswana lemmatizers	16
3.3.1 Machine-Learning approaches	16
3.3.2 Rule-based approaches	16
3.4 Existing lemmatizers for other languages	17
3.4.1 Machine-Learning approaches	17
3.4.2 Rule-based approaches	18
4 Methodology	22
4.1 Dataset	22
4.2 Setswana Tagset	23
4.3 Annotation	28
4.3.1 Lemma layer	28
4.3.2 POS layer	29

4.3.3	Named Entity Recognition	30
4.4	Data Preparation	31
4.4.1	Data Integration	31
4.4.2	File Transformation	32
4.5	Model building	33
4.6	Technologies & Environments	35
4.6.1	Programming Language	36
4.6.2	Overview of libraries	36
5	Experimental Results & Discussions	38
5.1	Experimental setup	38
5.1.1	Import libraries	38
5.1.2	Dataset loading	39
5.1.3	Text pre-processing	40
5.1.4	Vectorization & Feature Extraction	40
5.1.5	Training Model	42
5.1.6	Evaluation	43
5.1.7	Test Cases	46
6	Conclusions & Recommendations	52



List of Figures

1.1	NLP text pre-processing pipeline	2
2.1	Artificial intelligence contains machine learning and NLP	8
3.1	Example of a linguistic rule to perform lemmatization	13
3.2	A diagram of a typical learning problem	15
4.1	Sample plain text document to be used in development of dataset collected from Mmegi.	22
4.2	Workflow of an annotation project in WebAnno	28
4.3	Example of word reduced to their lemma.	29
4.4	Distribution of POS tags in the annotated dataset.	30
4.5	Distribution of NER tags in the annotated dataset.	31
4.6	Script to integrate multiple TSV files into one TSV file.	32
4.7	Dataset format	33
4.8	A flowchart of lemmatization model.	35
5.1	Experimental process setup	38
5.2	Some used libraries	39
5.3	Making the dataset available into the project	39
5.4	An example of token frequency as vector encoding generated under countvectorizer scheme	41
5.5	Dataset splitting	43
5.6	Confusion matrix table of n classes	44
5.7	Results of experiments	45
5.8	lemmatization procedure	46
5.9	input example	47
5.10	example of the model prediction output	47
5.11	Bonang as noun (i.e. NC1a)	48
5.12	Bonang as verb	49
5.13	Dimpho used at the beginning of a sentence	49
5.14	Dimpho used in the middle of a sentence	50
5.15	Metseng used as a noun (i.e. villages)	50
5.16	Metseng used as a verb (i.e. to swallow)	51

List of Tables

3.1	Summary of existing works	20
4.1	A tag-set for Setswana Nouns	23
4.2	A tag-set for Setswana Pronouns	24
4.3	A tag-set for Setswana Adjectives	24
4.4	A tag-set for Setswana Verbs	24
4.5	A tag-set for verbal prefixes	25
4.6	A tag-set for Adverb	25
4.7	A tag-set for Ideophone	25
4.8	A tag-set for Conjunctive	25
4.9	A tag-set for Interjective	26
4.10	A tag-set for Particles	26
4.11	A tag-set for Punctuation	26
4.12	A tag-set for Residual Words	27
4.13	A tag-set for Named Entities	27
4.14	Statistical parameters of the dataset showing the number of tagged words and untagged words	29

1. Introduction

This chapter outlines a brief overview of this research. The chapter entails the introduction, the problem statement, objectives, research questions, justification of this study as well as the expected outcome of the study. The last part of this chapter presents a short description of how the rest of the document is organised.

1.1 Introduction

Natural language is essentially a means of communication across the globe. It plays a key role in the exchange of ideas, help and thoughts between members of society. Therefore, all natural language should carry equal importance. A computer can be exposed to a large amount of textual data that is not very computer friendly i.e., can not be easily processed by computer. Making computers to effectively process text in a variety of applications is an important issue, and to do this, we are required to go through different modules. The basic and first module is morphological analysis, which is concerned with understanding the structure of words.

While significant progress in the field of Natural Language Processing (NLP) is available for several of the world's major languages such as English, which is considered to have a simple morphology [2], most African languages including Setswana are still lagging behind in the field of natural language processing. This is mostly due to the nature of the language's morphology, (i.e., the structure of words) which is considered to be complex [1]. Most of Bantu languages including Setswana are less-resourced, and have a partial presence in the world wide web as they lack the resources required for the implementation of human language technologies [3]. The digital divide, a situation where there is inequality concerning access in digital resources can also be attributed to a lack of human language technology applications [4].

For Setswana to experience an explosion of NLP applications like other languages, basic tools such as morphological analyzers need to be developed. These tools will eventually serve as a highly effective bridge to ensure that all Batswana can enjoy the benefits of improved human-machine interaction. The usefulness of text-based computational applications such as grammar and spelling checkers, search engines, question answering systems, automatic summarization systems and platforms such as the internet which is predominantly a textual medium still do not benefit all Batswana. This is due to lack of effective core language technologies that can simplify the processing of Setswana language in machines and extract relevant knowledge from it.

When developing natural language understanding systems, researchers have adopted a text pre-processing pipeline module, where an operation is divided into independent tasks that cooperate at different levels of language comprehension [5]. This pre-processing pipeline helps to deal individually with components that are important in the complete system. One of the most

needed component in the pipeline is a lemmatizer. Figure 1.1 shows lemmatization in relation to other components in the text pre-processing pipeline.



Figure 1.1: NLP text pre-processing pipeline [6]

In morphological analysis, lemmatization is a normalization technique defined as the transformation of all inflected word forms to their dictionary look-up form [7]. It is the process of obtaining the normalized form from a given word. This process is very important as lemmas have special significance in extremely inflected languages such as Setswana. Much of the lemmatization traditional routines depend on the conditional reality that most of the inflectional variations found in some languages such as English take place in desinence [8]. However, it should be noted that languages such as English are morphologically simple [9], as compared to Setswana. In Setswana lemmatization reaches a higher level of complexity because a word may go many transformations to reach its lemma. For example in English, the word village can appear as villages, the equivalent of Setswana word is *motse* which can appear in several forms such as *metse*, *metseng*. In English, the task is as simple as to remove *-s* that indicate the plural form in villages, to remain with village but to reduce *metseng* to *metse* the morphological process becomes complex, because many adjustments have to be made. Firstly, we have to remove *-ng* to remain with *metse* then change *-me* to *-mo* to get *motse* which is the lemma for *metseng*. Nevertheless, there are other English words that are morphologically complex for examples, goose→geese, mouse→mince, but in English this cases are very few as compared to Setswana making Setswana language to be morphologically rich and more complex than English.

Another technique in morphological analysis that is closely related to lemmatization is stemming. Stemming reduces a word to its stem which does not necessarily have to be a lemma, but a trait that is shared amongst a group of words. It is not always possible to use stemming to find the meaningful root of a word while lemmatization will always return the root word [10]. In this study, we particularly focus on lemmatization rather than stemming.

Some of the benefits of lemmatization is found in information retrieval and language translation systems [11], [12]. In these systems it is necessary for words which are morphological variants of each other to be recognized and treated similarly.

For search result improvements, a search engine utilizes the capability of a lemmatizer to understand human language and ensure that the greatest number of relevant matches is included in search results [11]. The lemma of a word can be used to increase search relevance as most natural language processing systems are able to work with normalized words more effectively

rather than having to individually handle all surface variants of a word. One good example of the use of a lemmatizer in information retrieval is its integration in Google search engine, which can automatically lemmatize inflected words like “ works, working ” to also produce hints for the word-form “work”.

The performance of language translation systems, which is perhaps the most important way in which computers aid communication can be limited by word variations in natural language and a lemmatizer can improve this as it can use the lemma to reference variations and reduce vocabulary size [12].

Setswana lemmatization with regard to context needs more investigation. Whereas the currently proposed methods [1], [4], [13], [14], [15] solve this problem, nonetheless, it is clear that researchers still tend to pay insufficient attention to the context of how words are used. Thus, they do not provide information on the performance of the model when faced with words that are similar but used in different context; since the concepts of sentences do not exist in their studies. In Setswana, considering the context of the surface word is an important factor for effective lemmatization. This is because on varying context, a word may originate from different roots, and mistakenly lemmatizing words out of context can lead to wrong results; as a motivating example, the word *Dimpho* could appear as a noun of different classes, as a result lemmatizing the word *Dimpho* will require correctness of context in detail.

- *Dimpho di ne di abiwa ke tautona.* (The gifts were awarded by the President.)
- *Dimpho o ile masimo.* (Dimpho went to the lands.)

The fact that we can remove prefix *-Di* in the first sentence to lemmatize *Dimpho* to *mpho* is not the case with *Dimpho* in another sentence. A large part of lemmatization works on Setswana language [4], [13], [14], [15] involves the use of rule based approaches. The problem with these techniques is that they are exhaustive to implement and fails to scale well with increase in data. Alternatively, Groenewald [1] reported a machine learning Setswana lemmatization model. However, all these cited works do not bother about the context of word occurrence. One challenge in Setswana lemmatization is the challenge of lexical ambiguity, that is, words having different meanings. Noting this, it is of interest to consider a statistical-based approach that can lemmatize words based on the context of how they are used. In this research, we describe the first attempt at building a data-driven morphological analyzer for Setswana.

One of the most interesting developments in computational linguistics has been the emergence of statistical machine learning algorithms [16]. They have proven to be effective in addressing natural language processing tasks by learning from pre-annotated data. Moreover, they are reported to achieve greater efficiency, more robustness, and better coverage than rule-based approaches [17]. Thus, making them well suited to deal with irregularities and complexities of natural languages better than language specific rules as they use statistical frequency information as opposed to string patterns in words [18]. We aim to develop the Setswana lemmatization model by taking advantage of the best-supervised machine learning algorithm Naive Bayes (NB).

1.2 Statement of the Problem

Research on context-aware lemmatization for Setswana language is lacking. Up to present, lemmatization methods have been proposed [1], [4], [13], [14], [15]. However, these works have limitation as they do not consider the context of how words are used in sentences. In these lemmatizers a word will always have the same lemma representation regardless of the context in which it occurs, while in reality a word may acquire different lemma depending on its usage. This models can suffer performance when it comes to realistic conditions. Setswana is a language with complex and rich morphology, (i.e. word forms pertaining to a single lemma). Its lexical variety is very high and this hinders the development of text-based natural language processing systems for Setswana language. Success in the development of natural language processing applications, such as machine translation and information retrieval depends on a proper understanding of the structure of words hence the need for context-aware morphological analyzers for Setswana language.

1.3 General Objective

To develop a context-aware lemmatization model for Setswana language using a machine learning algorithm.

1.4 Specific Objective

The following specific objectives have been set to achieve the above general objective:

1. To design a dataset and tagset for Setswana language.
2. To implement the Naive Bayes algorithm to lemmatize Setswana words.
3. To train the Naive Bayes lemmatization model using annotated Setswana dataset.
4. To evaluate the performance of the model using accuracy, precision, recall and F1-score.

1.5 Research Questions

This study seeks to address the following research question:

1. What tools can be used to design and tag Setswana dataset ?
2. What libraries can be used to implement Naive Bayes algorithm ?
3. What machine learning training strategy can be used to train the lemmatization model?
4. What is the performance of the developed lemmatization model on new data?

1.6 Expected Outcome

The expected outcome of this study will be a data driven lemmatization model for Setswana language (i.e., through the use of the Naive Bayes machine learning algorithm). The model should be able to recognize inflectional words in Setswana and return the lemma for those words considering the context of how the words are used. The benefits of having such a model are:- It will serve as an addition to existing works to enable pre-processing of Setswana language for subsequent use in larger application frameworks such as machine translation. This model could have great impact on increasing the amount of text available in Setswana Language making it more of a literary language as well as helping proofreader, writers and editors.

1.7 Justification of the Study

From the literature review (section 3.3) it is clear that research on the lemmatization of Setswana language has not received much attention, and to the best of our knowledge no work has implored the use of statistical machine learning with context understanding. The number of natural language processing works are growing for many languages but Setswana is lagging behind. Therefore, there is an urgent need to develop computational models that can be used in the development of various kinds of applications such as grammar and spelling checkers, search engines, question-answering systems, and automatic summarization.

This work indicates new directions for research and development. It is necessary to conduct this research because, when applications related to Setswana language are developed, end-users who seek information stored in Setswana language (i.e., text documents) can be benefited. This research will also form part of the efforts to bridge the digital divide and make languages to be more accessible to each other and help Batswana to enjoy the benefits of improved human-machine interaction.

1.8 Outline of the Thesis

The remainder of the thesis is organized as follows:

Chapter 2: Introduces the necessary background underlying this study, by providing a quick introduction to the context in natural language, morphology, and the overview of Setswana language. Readers with more advanced knowledge in natural language processing can skip this chapter.

Chapter 3: Discusses related works that motivate the work on the development of a context aware lemmatization model for Setswana language.

Chapter 4: Presents in detail the proposed work of the Setswana context-aware lemmatization model, design of experiments, and the evaluation of the proposed work.

Chapter 5: Presents the experimental executions and associated results based on the evaluation of the proposed model.

Chapter 6: Gives conclusion remarks, by discussing the overall results of the proposed study, it provides a summary of the study, research-based outcomes and gives a direction to future work that can emerge from the study.

2. Background

This chapter presents the essential background to natural language processing, the morphology of the Setswana language, as well what is meant by context in the scope of our work. The main aspects of interest are briefly introduced by means of linguistic examples. Readers with more advance knowledge in natural language processing can skip this chapter.

2.1 Natural Language Processing

Natural Language Processing(NLP) is the main research area of this thesis. NLP began in the 1950's and has always been an active research in the field of Computer Science. It is a sub-area of artificial intelligence and linguistics devoted to making computers understand words written in natural language [19], [20]. The ultimate objective of natural language processing is to support human-machine interaction by making sense of human language and derive meaning in a valuable manner.

In order to understand NLP we need to bear in mind that we have different kinds of languages: artificial languages such as C, C++, Java, Python etc., that are well defined and invented for easy communication of instructions to computers [21]. On the other hand, natural languages are languages which are not invented but evolved overtime like English or Setswana. These languages are not understood by the computer and it is, therefore, necessary for us to translate text written in natural language into what machines can understand [22].

The broader picture in Figure 2.1 shows natural language processing put in relation to other fields. Therefore, the development of state of the art natural language processing systems requires an adequate knowledge of other disparate fields such as machine learning and artificial intelligence.

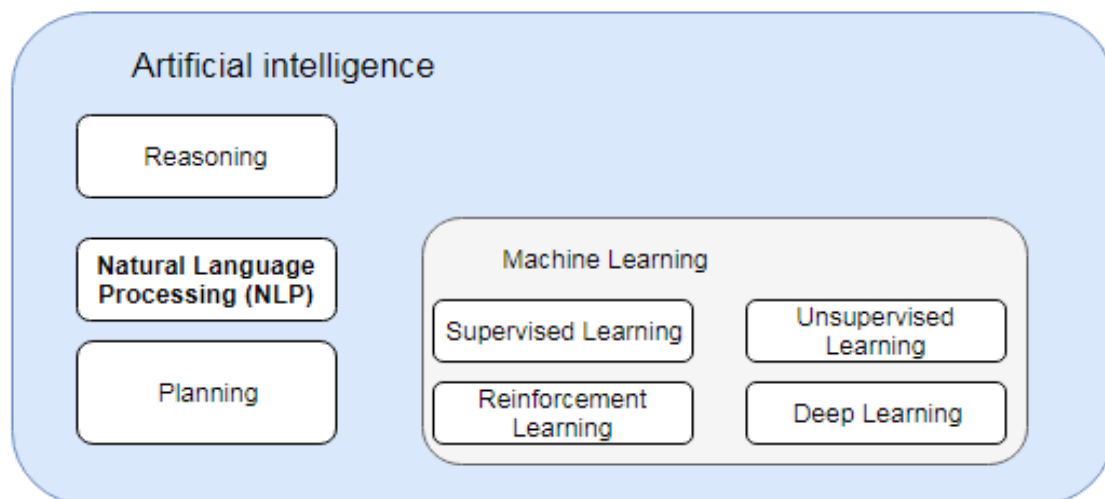


Figure 2.1: Artificial intelligence contains machine learning and NLP [23]

While humans can readily master a language, computers on the other hand do not use natural language the same way that humans do. The ambiguity and imprecise characteristics of natural languages make it hard for machines to perform natural language processing [22]. This is because machines do not have the understanding of the concepts presented in textual data.

Natural language is hard for machines to understand for several reasons; words can have different meanings, these meanings vary depending on context and they can acquire new meanings over time. Natural language is also changing every day and new words are being created making words too much to deal with.

The best way to get over this technological hurdle is to constantly develop low-level task tools for Setswana language such as lemmatizers that can ultimately work in greater systems [24]. NLP can be applied into various areas like, question answering, email spam detection, information extraction, summarization and machine translation [25]. This field is important as it is the driving force behind many applications such as Google translate, personal assistance applications such as Ok Google, Siri, Cortana and Alexa, even word processors employ NLP techniques to verify grammatical accuracy of the texts [26].

2.2 Morphology

Morphology is one of the most important part in automatic treatment of natural language. It refers to the study of the structure of words, how they are formed and their relationship to other words in the same language [27], [28]. At the mention of natural language the first thing that appears in our mind are words and then how they are connected to deliver an intended message. Morphological analysis helps in the study of the structure of words and parts of words such as stems, roots, prefixes and suffixes. All these provide a wealth of linguistic information that becomes useful when processing textual data. i.e. word analysis is essential in most development of natural language processing applications.

To develop NLP applications that can process Setswana text in computers we must build components that can understand morphemes and how these morphemes are composed to form complete words [29]. A morpheme is the smallest meaningful unit of a language [27], for example, the morpheme *-ng* can be attached to the Setswana verb *leba* (to look) to form *lebang* which does not change the meaning of the word but alters it to suit the case in which the word is used.

Many forms of a word may be related to the one lemma. For example, the lemma *leba*/look can appear as *lebang*, *lebile*, there are several forms for this type of verb. Morphology comes in two classes, inflectional and derivational morphology [30]. Inflectional morphology only modifies a word to mark grammatical function without changing the meaning of a word. Derivational morphology usually changes the meaning of a word and it is beyond the scope of this study.

2.3 Context in Natural Language

To understand the key area of this thesis, the term context in the scope of our work will be explained. According to Dash [31] context refers to “linguistic environment (rarely detached or isolated) in which a particular word occurs”. Still regarding context, Requejo [32] emphasized the importance of context in natural language when he states that “the complete meaning of a word is always contextual, and no study of meaning apart from a complete context can be taken seriously”. In natural language the significance of a word often differs owing to the particular context surrounding it. This makes context an important component of the literary text as it illuminates the true meaning and relevance of the text. If context is ignored the true meaning of words may be overlooked.

Words which look the same are likely to be exposed to different context and lemmatizing them will require the knowledge of how they are used rather than specifying rules that search inflectional prefixes and suffixes without bothering about how they are used. We strongly argue that context direct lemmatization, for instance, consider a scenario in which the Setswana word *legong* /wood, which is already in its base form, appears alongside the verb *bonang*/look. When using a linguistic rule to lemmatize *bonang* to *bona* by removing the suffix *-ng*, the rule can erroneously lemmatize *legong* to *lego* which may not be an actual word because *-ng* in *legong* is not a suffix but forms part of the lemma of the word *legong*. Therefore, complex exception rules should be created to cater for such situations. Furthermore, development of exception rules is complex and time-consuming so the proposed machine learning approach reduces the development time as well. Another limitation of the approach that lacks context understanding can be demonstrated by the example below:- The word *Bonang* could be:

- A noun: *Bonang o ile masimo*. (Bonang went to the lands).
- A verb: *Bonang, maru a thibile*. (Look, it is cloudy).

From the above example, the word *Bonang* is exposed to different contexts. How will the lemmatizer learn to lemmatize the word *Bonang* which appears in a different contexts? This entails more than just knowing which characters to remove to lemmatize the word. Failure to deal with such situations can affect the meaning of a word and create more problems in systems such as machine translations.

Looking at these problems, linguistic features such as part of speech (POS) and named entity (NE) information are important to capture characteristics of words and ensure good accuracy in restoring words to their base forms. These features act as meta-data i.e., patterns that provide useful information about words for precise lemmatization. Without these features, we argue that lemmatization becomes a heuristic process that chops off the ends of words in the hope of achieving good results most of the time, and this compromise good precision as these features are important beyond those in string pattern matching [33]. The main contribution of our thesis is therefore to fill the gap by adding context in lemmatization of words in Setswana, ultimately leading to good precision rate in lemmatization.

The pioneering works [1], [4], [13], [14], [15], to develop Setswana lemmatizer focused on verbs and nouns only and ignored other word classes, as they are considered closed i.e., morphologically unproductive. However, a coherent Setswana sentence is made of different word categories. Moreover, the noun class has sub-categories along morphological dimensions, (see Table 4.1), and to our best knowledge, this has not been addressed in the past studies on lemmatization of Setswana. Adjectives and adverbs are also able to display some level of inflections on words. Although the number of inflected words from these categories is very few, to our best knowledge, this has never been addressed in Setswana. Words play different functions depending on how they are used, therefore, special attention needs to be given when most part of speech categories are presented in a text.

A lot of work has been accomplished for the resource-rich languages such as English, and little work has been done in Setswana. Work done on resource-rich languages is not directly applicable to Setswana because languages differ from country to country. Languages that are morphological complex as Setswana are not easy to be understood and processed by a computer.

Based on this background, there is still some potential to expand upon their previous works in terms of lemmatizing words based on the context of how they are used, as well as reducing development time and the ability to include most parts of speech categories, rather than nouns and verbs only.

By including other word categories we make the proposed model more robust to be able to deal with a variety of user inputs in form of coherent sentences rather than being restricted to nouns and verbs only, as it is the case with previous studies.

2.4 Setswana language

Setswana is a national language of Botswana which is used by 80% of the country's population estimated at 2.1 million [34], despite it not being widely discussed in the literature mainly due to the unavailability of digital resources. As a result, much needs to be done to boost the digital resources available to Setswana speakers. Setswana is also one of the Bantu spoken languages, it makes up one of South Africa's 11 official languages and forms the fifth-largest language group in South Africa. It is more spoken in north-western parts of South Africa, where the country borders with Botswana.

The nearest relatives of Setswana are Pedi and Southern Sotho. These three languages, with a total of about 16 million speakers, are so near to each other that they should be regarded as three variants of one language from a linguistic point of view [34]. However, the analysis in this work is based on Setswana used in Botswana. In Botswana, Setswana language is used for both spoken and written communication. The Setswana language is characterized by rich vocabulary and complex morphology, where a word can occur in a large number of inflected forms, depending on its syntactic and semantic role in the sentence, this makes affixations very prominent [35], [36].

Setswana has several part of speech categories with major ones being (nouns, verbs, pronouns, adverbs, particles, interjections and idiophones) and most of the words which belong to categories of nouns and verbs are full of inflectional morphology [4]. That is, the words are attached with affixes that express some grammatical functions regarding tenses, mood, gender and numbers. These affixes are functional words or particles which have no meaning but their occurrence in words is for emphasizing. The affixes can come in the form of:- prefix when they precede the root or suffix when they follow the root. In addition to prefix and suffix inflectional processes can cause Setswana words to undergo infixational modification where affixes are inserted within the root. For examples:

- Prefix

$pod\dot{i} \Rightarrow \mathbf{di} + podi = \mathbf{dipodi}$

- Suffixation

$tsamaya \Rightarrow tsamaya + \mathbf{ile} = tsama\mathbf{ile}$

- Infix

$ngwaga \Rightarrow ngwa + \mathbf{ge} + ng = ngwa\mathbf{geng}$

Therefore, as a pre-requisite to building a state of the art electronic language aids for Setswana such as grammar/spell checkers, automated translation systems, questionnaires dictionaries and information retrieval systems we need to develop good morphological analyzers.

2.5 Naive Bayes (NB) algorithm

The NB algorithm is the defector standard text classifier which comes under supervised machine learning and uses frequencies in the data to make decisions [37]. This algorithm is based on Bayes theorem with independence assumption between attribute, despite the strong assumptions that features are independent, it performs very well in many applications [38].

Lemmatization is a very complex task because of all the irregularities and exceptions that are there in natural languages. Establishing language specific rules for lemmatization is not an easy task and this can be attributed to these irregularities and exceptions. The Naive Bayes algorithm examines the conditional probabilities generated in training data and this simplifies

the building of the model as well as the interpretation. The Naive Bayes algorithm is simple to implement and has a very simple structure [39] based on the statistical feature weighting method. This algorithm is one of the state of the art pattern classifier, suitable for classification with discrete features, for example, word counts for text classification.

Outside lemmatization, the Naive Bayes algorithm has been successful in solving text-based problems, for example, sentiment classification [40], named entity recognition in textual data [28], text classification [41], [42], part of speech tagging [43], [44], as well as spam detection in emails [45], [46].



3. Related Work

In this chapter, we discuss previous work that this thesis directly builds upon. In a nutshell, we will describe the most two prominent approaches being rule-based methods and statistical machine learning methods that are commonly used for solving problems and are well documented in the literature for the research community. We also give the relevant literature from other researchers and the techniques they used to develop lemmatizers for Setswana language as well as other languages other than Setswana.

3.1 Rule-based approach

Rule-based approaches have been around for centuries, and can be regarded as the earliest efforts of problem solving which follows a conventional programming approach. In these approaches, programmers have to explicitly define the rules covering all possible data patterns [47]. Unlike machine-learning which follows a black box technology, this approach is very declarative leading to highly transparent and expressive models [47].

Based on their declarative nature, it is easy to control the resulting precision in a smaller training dataset of high quality and face challenges as the task widens. See Figure 3.1 below shows a simple illustration of lemmatization rule implemented in java to lemmatize Setswana inflectional words that have a patterns *-di*, *-ng*, e.g. *dikhuting* (holes) to its lemma..

```
String[] patterns = { "di", "ng"};
lemmatize("dikhuting", patterns);
}

public static String lemmatize(String token, String [] patterns)
{
    String lemma= token;

    for (String pattern : patterns)
    {
        if(lemma.contains(pattern))
            lemma=lemma.replace(pattern, "");
    }
    return lemma;
}
```

Figure 3.1: Example of a linguistic rule to perform lemmatization

The lemmatizer has a function that takes a word i.e., token and array of patterns, if the word contains the pattern, the if-else rules are applied to return the lemma of the word. The output of the above function will be *khuti* which is the lemma for *dikhuting*.

Besides their strength, rules have drawbacks that can be addressed by modern machine learning techniques [48]. It is easy to define the rules that search for string patterns in words, but the complexity increases as we put this to test in the real world when we deal with a lot of textual data [49].

Developing rules is challenging and time-consuming for some domains [48], [23], especially for natural languages as it requires a great effort to capture all complex patterns. This can result in a large number of rules to represent a large number of possible situations. Consequently, as the rules continue to grow, introducing a new knowledge for example adding a new rule, we might introduce contradiction with the previous rules which can have negative effects on the performance of the application [48], [50]. Rule-based solution in natural language operates on character level by specifying some elements to be matched and the corresponding action, therefore, they can result in elevated computational expenses in solving natural language problems.

The rules explicitly provided by programmers have no automatic learning due to their declarative nature [49]. As a result, they perform badly on minor variation in data which was not part of the training set, every time the rules have to be developed for newly introduced data. Often rules emerge from efforts of knowledge engineering in which the knowledge an expert brings to solve a problem is codified in a form of “if..then else..” form. The left (if) side of the rule is the preconditions under which the rule can be activated and eventually fired. The right (then) side specifies situational-specific activities that moves the system nearer to a solution when firing the rule.

Therefore, there has to be a good prior knowledge about the problem and the data, thus using rules to solve NLP problems requires a deeper knowledge of the language to identify patterns in words. Rule-based solutions are very exhaustive, time-consuming and generally impossible to accomplish for real world problems [48]. This approach is very useful and performs well on a small sample of data. If we want to build scalable systems to handle large data it is not feasible to stick with rule-based approaches.

3.2 Statistical machine learning approach

Machine learning is defined as the automated detection of meaningful patterns in data [51]. It is a form of artificial intelligence with two phases involved i.e training and testing. This enables a computer to learn from data and make rational decisions rather than through explicit programming. Once the learning has been done it can be used to predict the patterns on future data and improve automatically without the need for further programming.

When statistical techniques and machine-learning are combined they become a powerful approach in solving problems that require information extraction from data sets [23], [52]. In the past decade, statistical machine learning has been a key instrument in solving problems ranging

from natural language processing, image processing as well as in fundamental Sciences such as Biology [53], [54] and Astronomy [55], [56].

Machine-learning relies on expert annotated training data to derive decision rules by means of statistical regularities between the observable representation of data and output labels and less linguistic knowledge when solving NLP related problem [23]. The amount of knowledge available on certain tasks might be too large for conventional programming by humans [51]. The same idea applies to the problem of lemmatization, and it cannot be easily unraveled by hand-coded rules developed by programmers.

The development of robust learning models for natural language processing is of paramount need looking at how complex textual data is and it cannot easily be solved using the conventional programming techniques. Statistical machine-learning uses frequencies in the data to produce a reasonable output without having to know much about the rules of language.

The most studied problem in statistical machine-learning is the problem of classification [23], which deals with two kinds of spaces, the input space X (space of instances) and the output space y (labeled space). Statistical machine-learning tasks are different and are mostly categorized into two types of learning problems depending on the knowledge the algorithm has about the classes in the data i.e. supervised and unsupervised learning.

Supervised machine-learning is whereby each training data of input observation is associated with the correct output [57]. The learned expertise will then be applied to the unseen data. In supervised learning, the structure of the data is already known and the aim is to assign new data to correct classes. In contrast, unsupervised learning has no access to output values and the algorithm find structures in the data by creating classes on their own. The approach implord in this research focuses on a supervised learning method. See Figure 3.2 for a general overview of a machine learning-problem.

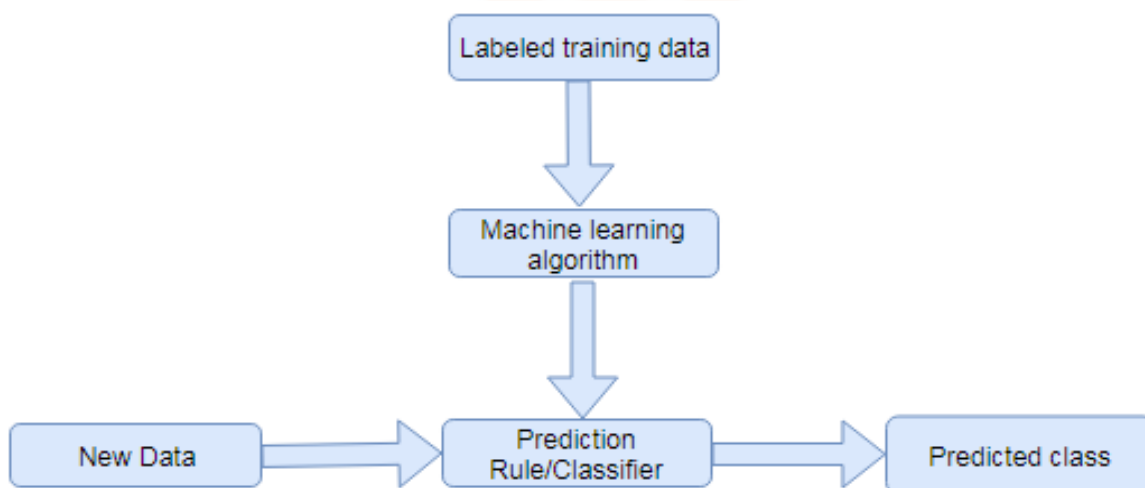


Figure 3.2: A diagram of a typical learning problem

3.3 Available Setswana lemmatizers

3.3.1 Machine-Learning approaches

Groenewald [1] developed a machine learning lemmatizer for setswana language based on K-NN classification algorithm. The lemmatizer was evaluated on a dataset containing 2,947 lemma annotated Setswana words. The dataset was split into a training dataset consisting of 90% of the data, with test-set consisting of 10% of the data. The lemmatizer obtained an accuracy of 64.06%. In this work [1] the author did not consider the context of how words are used. The work was done on isolated words annotated with their lemmas. However, lexical ambiguity is common problem in Setswana language, therefore ignoring word occurrence in sentences can suffer under practical situations as words can have multiple interpretation. This will require the lemmatizer to decide which interpretation is appropriate in a certain context.

3.3.2 Rule-based approaches

Most studies on Setswana word lemmatization, focuses on the use of rule based approaches. The first working prototype to explore Setswana lemmatization was put forward by Brits et al. [4]. They developed a rule-based lemmatizer that utilizes the use of finite state automata (FSA) using regular expression to represent linguistic rules.

FSA are machines that performs computations by moving through a series of states to check for patterns and activate functions in the form of regular expressions. An example of a regular expression rule to lemmatize the Setswana verb *Bonang* (look) can be `[?; n,[g]:[]]`, which means in a string ending with *-ng* change *-ng* to empty character to remain with *Bona*.

Regular expressions (REs) are very helpful in extracting regular information i.e., the information provided in a limited number of known formats. Setswana is not a regular language and the use of regular expression cannot be a good idea because Setswana includes too many special cases. Inflectional words may need many transformations regarding addition and subtraction of characters both prefix, infix and suffix to reach their basic form. In addition, REs are brute force approach and it cannot learn, they do not care about the language nor the meaning of what they find. In a set of 500 verbs, the lemmatizer had a precision of 94% and in a set of 500 nouns, it had a precision of 93%. While this are good results, in reality lexical ambiguity is a problem in Setswana, a word may have different lemmas depending on its usage therefore considering lemmatization on isolated words may face challenges under realistic scenarios. No information provided on how the model deals with this problem of lexical ambiguity.

Similarly, notable work was done by Malema et al. [13], who has a great influence on the development of morphological analyzers for Setswana language. Malema et al. [13], developed a ruled-based Setswana verb lemmatizer. Their work was conducted on verbs only. The rules were implemented using python. Their experiments were done on a dataset consisting of 6000 verbs and they achieved a performance of 87%.

Furthermore, Malema et al. [14], [15] did a series of work on morphological analyzer for Setswana verbs and nouns. They developed a Setswana verb analyzer and generator [14] implemented in

java. They used a dictionary as a list of Setswana verbs in their basic form put in a hash table together with a database of word transformation rules. The analyzer will then receive a word and apply transformation rules to reduce the word to its lemma. The test data for verb analyzer contained 2000 verbs and achieved a performance of 87%, a similar approach was done to develop a noun analyzer and generator [15] using java. The test data for the noun analyzer and generator contained 2000 nouns and had a performance rate of 79%.

Despite this good attempt from existing works [1], [4], [13], [14], [15], to develop a morphological analyzer for the Setswana language, there is limited evidence on how to account for the context of the words to be lemmatized. It is very important to lemmatize words based on the context of how they are used.

Moreover, Setswana is very complex and sometimes a word has to go through many alterations to be reduced to its basic form. How easy is it to develop rules to recognize patterns and alter a word to its basic form? One can also argue that these works have been assessed to a very limited scope in terms of the size of the dataset used in previous studies. How about if we increase the size of the dataset? Due to the complexity of patterns that needs to be detected on the word possibly, there can be some contradictions as new texts are worked on i.e., new discoveries are made.

3.4 Existing lemmatizers for other languages

Other researchers have made significant attempts to develop morphological analyzers for their indigenous languages, as a way of making them enjoy the benefits of improved human-machine interaction.

3.4.1 Machine-Learning approaches

A novel approach was put forward by Chakrabarty et al. [58] who proposed a context-aware lemmatization model for the Bengali language which belongs to a group of major Indic languages. Their method implored the use of feed-forward neural networks. The representation of words in a neural network based framework was handled through a vectorization technique, where words were transformed into numerical forms. To evaluate the lemmatizer, two supervised dataset were manually built, one for training and the other for testing. The lemmatizer obtained an accuracy of 69.57% on the test dataset.

Another study was carried out by Akhmetov et al. [59] who developed an open source lemmatizer for Russian language based on regression models. The lemmatizer was trained on the open grammatical dictionary of Russian language. On train/test accuracy scoring, the lemmatizer achieved an accuracy of 68% on test dataset using the decision tree regressor model.

Moreover, Akhmetov et al. [60] presented a supervised machine learning lemmatizer based on decision trees and random forest classifier method. The authors mentioned that the lemmatizer support 25 languages, which are publicly available under the Open Data Base License (ODbL)

of which the low resourced languages such as Setswana are not part of. The accuracy score of the model is different per language.

Kanerva et al. [61] developed a sequence to sequence lemmatization model using artificial neural networks. The evaluation of the lemmatization model was done on 52 different languages with varying lemmatization complexity and training data sizes. This model was tested on languages that have their dataset available under universal dependency treebanks. The universal dependency is an international cooperative project aimed at creating treebanks of the world's languages. The performance of the model varied according to different languages.

3.4.2 Rule-based approaches

Gupta et al. [62] designed a rule-based lemmatizer for the Urdu language which is one of the constitutional languages of India. In their approach, various rules for removal and addition of suffixes along with a database for exceptional words i.e. words which do not require any removal or addition of character was created. Any input word to be lemmatized was checked against the database if it was found in the database it was returned as the root word else if it was not found the algorithm proceeded to apply rules to return lemma. The experiments for this work were done on 1000 Urdu words and obtained an accuracy of 90.30%.

Another rule-based approach was proposed by Prathibha and Padma [63] for Kannada inflectional words, which is the official and administrative language of the state of Karnataka in South India. In their proposed work they only considered nouns and verbs. They created a dictionary containing prefixes and suffixes for both nouns and verbs. For any input word a search was done to check for suffixes and prefix in the input of inflectional words and a set of rules were framed in such a way that as the prefix or suffix were found and removed if necessary, the addition of characters to return the base word was done. In their work, they did not specify the number of words used. However they mentioned that they created four different datasets that were simply words taken from four different sources to be fed as inputs to their developed rules. They did not provide exact accuracy for each dataset, rather they specified that in each dataset the rules obtained an accuracy that was over 85%.

A ripple down rule algorithm was proposed by Plisson et al. [64] for Slovene words. Their approach focused on word endings only i.e. what suffix should be removed and/or added to get the normalized form of a word. However inflections can also contain a change to the internal buildup of the words, as opposed to straightforward suffixation, i.e. suffixes are not the only cause of variations in natural language, their ripple down rule algorithm performs very well but in a limited scope as it only considers suffixes. They did their experiments by running the algorithm in five different dataset. The variation of performance in the dataset was small and the overall experimental results achieved an accuracy of 77%.

Suhartono et al. [65] explored lemmatization by building a rule-based web application to lemmatize Bahasa which is the Indonesian language. The presented method uses an Indonesian dictionary and a set of rules to remove affixes. The input word is first checked if it is listed as a lemma in the dictionary, if it succeeds the algorithm stops and returns the word as the lemma.

If not it proceeds to apply rules that satisfies the conditions of returning the base form of the word. The test data contained 57, 261 valid words taken from kompas.com, one of the biggest news companies in Indonesia, and it was supplied as one word per lemmatization process. The rules were built using PHP and MySQL database and achieved an accuracy of 98%.

Paul et al. [66] created a lemmatizer for Hindi, a commonly spoken language in India, using rule-based approach. They created a knowledge base for storing grammatical features of the Hindu words along with linguistic rules. The rules worked by removing suffixes from the input word. After the removal of suffixes, if the words provide a proper meaning it is displayed as the root, otherwise, particular characters are added to the stripped word to make it a proper meaningful word. The lemmatizer was evaluated for accuracy with 500 words. Among those 500 words 456 words were correctly lemmatized and 44 words were incorrectly lemmatized because they violated the rules. This gave an accuracy of 91%.

We conclude our literature review by summarizing all the existing works on the lemmatization of Setswana language and other languages in Table 3.1.



Author	Title	Approach	DataSize	Results
Brits et al. [4]	Automatic lemmatization in Setswana: Towards a prototype	rule based	500verbs, 500nouns	94%verbs, 93%nouns
Malema et al. [13]	A rule based Setswana verb lemmatiser	rule based	6000 verbs	87%
Malema et al. [14]	Setswana Verb Analyzer and Generator	rule based	2000 verbs	87%
Malema et al. [15]	Setswana noun Analyzer and Generator	rule based	2000 nouns	79%
Gupta et al. [62]	Design & development of rule based Urdu Lemmatizer	rule based	1000 words	90.30%
Prathibha and Padma [63]	Design of rule based lemmatizer for Kannada Inflectional words.	rule based	-	+85%
Plisson et al. [64]	A rule based approach to word lemmatization	rule based	-	77%
Suhartono et al. [65]	Lemmatization Technique in Bahasa: Indonesian Language	rule based	7,839 unique valid words	98%
Paul et al. [66]	Development of Hindu Lemmatizer	rule based	500 words	91%
Chakrabarty et al. [58]	A Neural Lemmatizer for Bengali	Machine Learning (feedforward neural network)	2,126 test instances	69.57%
Groenewald [1]	Using Technology Transfer to Advance Automatic Lemmatization for Setswana.	Machine Learning (K-NN algorithm)	2,652words	64.06%
Akhmetov et al. [59]	Open-Source lemmatizer for Russian Language based on tree regression models.	Machine Learning (Decision trees)	-	68%
Akhmetov et al. [59]	Highly Language-Independent word Lemmatization using a Machine- Learning classifier.	Machine Learning (Decision trees)	-	different according to languages
Kanerva et al. [61]	A Sequence to Sequence model for lammatizing Universal Dependencies Treebanks.	Machine Learning (Neural Networks)	-	different according to languages

Table 3.1: Summary of existing works

Lemmatization tasks are different according to the nature of languages. For Setswana a few research work has been done. To the best our knowledge, a fully unified morphological analyzer for Setswana language has not been discovered except the ones specializing only on nouns and verbs. This lacks coverage as it can not explain how it deals with words which are not in categories of nouns and verbs. Our proposed work is different in the sense that it tries to deal

with lemmatization in coherent Setswana text (i.e. sentences) therefore, considering most of all part of speech categories. The lemma of a word do not only depend on the considered word, but also on its role within syntactical construction, and thus on the context in which it is used.

As mentioned earlier in chapter 2 (i.e., background), Setswana has several part of speech categories with major ones being (nouns, verbs, pronouns, adverbs, particles, interjections and idiophones). Even though the words that belong to categories of nouns and verbs are the ones that pose a serious challenge in lemmatization, there is a need to have a unified model that can handle lemmatization in a coherent Setswana text (i.e. sentences). Sentences in Setswana are not only made of verbs and nouns but all part of speech categories in Setswana language. Developing a specialized morphological analyzer that only focuses on verbs and nouns is not enough as it does not give a clear account of how it will perform when faced with coherent sentences that include other words outside these categories.

As an observation from literature (see table 3.1), most researchers follow a rule-based approach that is very dependent on the language knowledge. However, this can result in complexities as the dataset continues to increase because rules also increases significantly with an increase in dataset size. Therefore we argue that the moment the datasets continues to grow scalability problems may start to surface hence degrade in performance. Small datasets used in rule-based solutions automatically reduce the size of operation and the nature of complexities involved in the lemmatization process hence resulting in higher accuracies.

Another observation from the literature [4], [13], [14], [15], is that, the most common challenge reported by researchers who used rule-based methods are conflicting rules which yielded wrong results. This can have severe effects when morphological analyzers are incorporated in larger systems such as language translation systems and summarization tools.

Lastly, researchers who follow a rule-based approach developed lemmatizers that take input as a single word per lemmatization process and this makes it difficult to consider the context in which words are being used. Rule-based methods depend on some words satisfying the conditions of certain hand-coded rules that remove affixes and add required characters to produce the valid roots of inflected words without paying attention to how words are being used.

With this in mind, we propose a model that can lemmatize words based on the context of how they are used, and at the same time being able to improve in performance with regard to increase in the dataset.

4. Methodology

This chapter focuses on describing the nature of the research, a description of methods used in carrying out the study, the research design as well as their suitability for the study. The methodology is broken down into two main parts. The first part outlines data collection procedures, dataset and tag-set development. The second part presents the tools and technologies used in developing our model. The main aim of this study is to develop a context-aware lemmatization model for Setswana language using statistical machine learning techniques.

4.1 Dataset

A crucial part of every machine learning application is data. Dataset development is a critical process in the development of a data-driven lemmatizer for Setswana language. The prominent problem since way back has been that, linguistic resources such as dataset are limited and the existing resources are either far from being open and standard hence making it difficult to develop data-driven models for Setswana language. Even though this is true, we have successfully developed a dataset for this study. The dataset is built out of past news articles from Botswana containing Setswana sentences. The news articles were collected from Mmegi and Botswana Press Agency (Bopa). A Setswana Magazine is hard to find, however Kutlwano magazine which is mainly written in English has stories written in Setswana which we were able to include in the dataset. All these texts were in plain text format. The collected documents were well written and edited by professional reporters with minimum noise. Some words were already in their normalized forms and several other words in inflected forms. Figure 4.1 shows a sample of the original text document collected to be used in the development of the dataset.

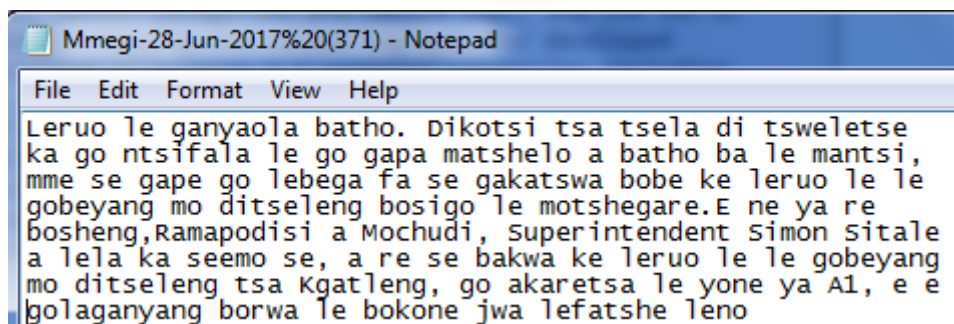


Figure 4.1: Sample plain text document to be used in development of dataset collected from Mmegi.

The text document shows an error where the word *mantsi* is supposed to read *bantsi*. these are some of the noise we corrected during dataset preparation.

4.2 Setswana Tagset

A tag-set consists of labels/tags used for adding information on each word in a sentence. To tag the dataset POS tagset tailored for Setswana spoken in Botswana was developed with the help of Setswana teachers from Lotsane Senior School in Palapye. The development of the tagset was made with reference to Bertus van Rooy & Rigardt Pretorius [67]’s POS tags for Setswana which is fully in line with the guidelines stated by the Expert Advisory Group on Language Engineering Standards (EAGLES) group. Some of the word categories stated by the EAGLES group are relevant to Setswana language. The developed tagset accounts for fifty five (55) POS tags which are utilized as features to know the local context of words before we can lemmatize them. We summarized the used tagset, which provides contextual information in our dataset in Tables 4.1-4.12. However, some of the tags are not relevant to the problem we are addressing as they are closed word categories. Regardless of this, they must be used to correctly tag the dataset and maintain the structure of sentences.

Noun: We used 16 noun tags to distinguish between the different noun classes. All locative classes (16, 17, and 18) are tagged as NLOC.

Table 4.1: A tag-set for Setswana Nouns

Tag category	Label	Example (s)
Noun class 1 (mo-)	NC1	motho, monna, modisa, mmusi
Noun class 1a	NC1A	Thato, Kgama, Bonang, Dimpho
Noun class 2 (ba-)	NC2	batho, basadi, Basotho, babusi
Noun class 2a (bo-)	NC2A	boKhama, boDimpho, boThato
Noun class 3 (mo-)	NC3	motse, morafe, mmu, mmidi
Noun class 4 (me-)	NC4	megala, mesese, mebutla, mere
Noun class 5 (le-)	NC5	lefoko, leuba, letlalo, lephutse
Noun class 6 (ma-)	NC6	maeba, masilo, manyena, maina
Noun class 7 (se-)	NC7	sediba, selo, setlhare, selepe
Noun class 8 (di-)	NC8	dilo, ditlhare, dikobo, dirope
Noun class 9 (N-)	NC9	ntho, nko, podi, kgomo, tlhogo
Noun class 10 (diN-)	NC10	dipodi, dikoloi, ditsebe, ditlhapi
Noun class 11 (lo-)	NC11	lore, loso, lotlhaka, logong
Noun class 14 (bo-)	NC14	bosigo, botlhoko, bogobe, bogadi
Noun class 15 (go-)	NC15	goja, goitse, goratwa, goaga
Noun class 16, 17, 18	NCLOC	fatshe, godimo, moseja, gare, gae

Pronoun: These are words that stand in place of nouns. There are three main types of pronouns in Setswana, Personal, Demonstrative and Qualificative. The following tags are used:

Table 4.2: A tag-set for Setswana Pronouns

Tag category	Label	Example (s)
Absolute singular	PP1	nna, wena, one, lone, ene
Absolute plural	PP2	rona, lona, bone, tsone
Demonstrative singular	NC2	yo, yoo, oo, seo, eo
Demonstrative plural	PD2	bao, bale, tse, a, tseno
Qualificative singular	PN1	yothe kgamelo
Qualificative plural	PN2	botlhe

Adjective: Adjectives are words in a sentence that modifies or describe a noun, pronoun or a thing.

Table 4.3: A tag-set for Setswana Adjectives

Tag category	Label	Example (s)
Adjective Qualificative	QAJ	setala, sentle, kima, khutshwane
Relative Qualificative	QRV	yoo, baba, sese
Possessive Qualificative	QPS	yame, gagwe, tsame, tsarona
Quantitative Qualificative	QQN	sotlhe, tsotlhe, botlhe, rotlhe
Enumerative Qualificative	QEN	bangwe, bafe, mongwe, nngwe

Verb: is a word that signifies the occurrence of an action.

Table 4.4: A tag-set for Setswana Verbs

Tag category	Label	Example (s)
Verb	VERB	mmotsa, ithuta, ntshwenya, direla
Copulative Verb	VCOP	-le, -se, -na, -nna
Auxiliary verb	VAUX	-setse-,

Verbal Conjugations/ Verbal Prefixes

Table 4.5: A tag-set for verbal prefixes

Tag category	Label	Example (s)
Negative morpheme	VPNE	-ga-, -se-, -sa-
Subject concord	VPSC	-ba-, -tsa-, -o-
Object concord	VPOC	-di-, -re-, -go-, -a-
Present tense prefix	VPPR	-a-
Future tense prefix	VPFT	-tla-,
Past tense prefix	VPPT	-kene-, -ale-
Progressive aspect prefix	VPPA	-santse-,
Potential aspect prefix	VPPO	-ka-

Adverbs: are words that describes or modifies verbs, adjective with respect to manner, pace or time.

Table 4.6: A tag-set for Adverb

Tag category	Label	Example (s)
Locative adverb	ADLOC	nokeng,taung, Gaborone, Botswana
Temporal adverb	ADTEM	bosigo, mariga, kamoso
Manner adverb	ADMAN	fela, jaana, ruri, sentle
Interrogative adverb	ADINT	Kae?Leng?Mang?Eng?
Cardinal numeration adverb	ADNUC	gangwe, tlhano, masome a mabedi
Ordinal numeration adverb	ADNUO	santlha,sabobedi, sabofelo

Ideophone: is a lexical class of words that conveys ideas or expressions by means of sound, colour, manner, smell, state, action or intensity.

Table 4.7: A tag-set for Ideophone

Tag category	Label	Example (s)
Ideophone	IDEO	phatsi, phamo, tuu, rago

A conjunctive is a word that introduces a sentence or links up words, phrases, clauses or intensity.

Table 4.8: A tag-set for Conjunctive

Tag category	Label	Example (s)
Conjunction	CONJ	mme (and), gonne (because), kgotsa (or)

An interjective is a word, exclamatory in character and generally isolated, that is used to express some emotion, to convey assent or dissent, or to call attention or give a command.

Table 4.9: A tag-set for Interjective

Tag category	Label	Example (s)
Interjection	INT	ao!, ija!, ijoo!, nnyaa! wena!

Particles are morphemes that has grammatical functions in sentences, this words are associated with phrases or other words to impart the meaning.

Table 4.10: A tag-set for Particles

Tag category	Label	Example (s)
Infinitive class prefix	PAINF	go (to)
Demonstrative particle	PADEM	yo, seo, fale
Possessive particle	PAPOS	sa gago (of/belonging to you)
Qualificative particle	PAQUA	se segolo (that is big)
Instrumental particle	PAINST	ka (with)
Locative particle	PALOC	go (at)
Associative particle	PAASS	le (and/with)

Punctuation: This tag is for special symbols that are used in text to indicate division between sentences and phrases. These include commas, colons, full-stops, semi-colons, quotation-marks, hyphens, exclamation-marks, brackets, etc.

Table 4.11: A tag-set for Punctuation

Tag category	Label	Example (s)
Punctuation	PU	, () “

Loan words (i.e. words from other languages) that were present in the dataset were not removed, instead, they were tagged as residuals. The suggestion of the EAGLES Group is that this category be reserved for words which do not easily fit into any of the known values. Table 2.12 below shows the residual word-classes for tagging Setswana.

Table 4.12: A tag-set for Residual Words

Tag category	Label	Example (s)
Foreign word	RFW	Sorry
Formula	RFM	E = mc ²
Symbol	RSYM	°C
Acronym	RACR	AIDS
Abbreviation	RABR	Moh.
Unclassified	unkw	Hesitation phenomena

Another relevant feature that provides important information to the local context of words is the named entities (NE). NE are unique identifiers that can give useful information before lemmatization of words takes place. The proposed model should be able recognize the named entities i.e., person, organizations and other relevant entities (where possible) before performing lemmatization. The tagset for the named entity was developed with reference to George A. Miller & Florentina Hristea [68] and A. Kilgarriff & C. Fellbaum [69]. We summarize the used named entity in Table 4.13.

Table 4.13: A tag-set for Named Entities

Tag category	Label	Example (s)
Animal	ANI	Kgomo,Koko,Phuduhudu, di-tonki,dikgomo,dipodi
Arifact	ART	Nkgo,Phate,koloi,ntlo ya bojang
Attribute	ATT	senatla,mokima, botala jwa lewapi
Body	BODY	nko, molomo,tsebe
Event	EVENT	dikhwaere, letlhafula, kirisimose
food	FOOD	legapu,bogobe,nama ya podi
Group	GROUP	Bakwena, Mapodise, Batswana
Location	LOC	Gaborone, noka ya metsimothabe
Object	OBJ	Lewatle, Sefako,Letlapa
Organization	ORG	University of Botswana, SADC
Person	PER	Lepodise, Kabelo G. Madise
Phenomenon	PHENO	Legadima, Sefefo, Tladi
Plant	PLANT	Mosetlha, Tlhaga, Mosu
Quantity	QUA	Masome a mabedi, Lekgolo
Relation	REL	Rakgadi,malome, Ntsalake
State	STATE	Thokgamo, Bodutu,Kagiso
Time	TIME	Bosigo, labone, nako ya bobedi

4.3 Annotation

The collected articles were loaded into the WebAnno tool for manual annotation. The WebAnno is a linguistic annotation tool with its functionality accessible over the internet through a web browser [70]. This annotation tool is mostly used in the natural language processing community as well as in the digital humanities for the development of linguistic dataset.

In the WebAnno we can read and annotate each sentence for different documents. The tool supports several pre-defined annotation layers including the lemma, POS, NER, etc. through a web-based user interface. Furthermore, it allows additional configuration of custom layers as they may be required for the annotation task.

The WebAnno also supports different roles such as annotator, curator, and project manager, [71], [72], [73]. The process and quality of annotation in the dataset were monitored and measured in terms of the agreement from an annotation team lead by our Supervisor Dr Keletso Letsholo. The annotation team also included Setswana teachers from Lotsane Senior School in Palapye and fellow Masters students from the Computer Science Department.

The annotation process was satisfied as shown in Figure 4.2.

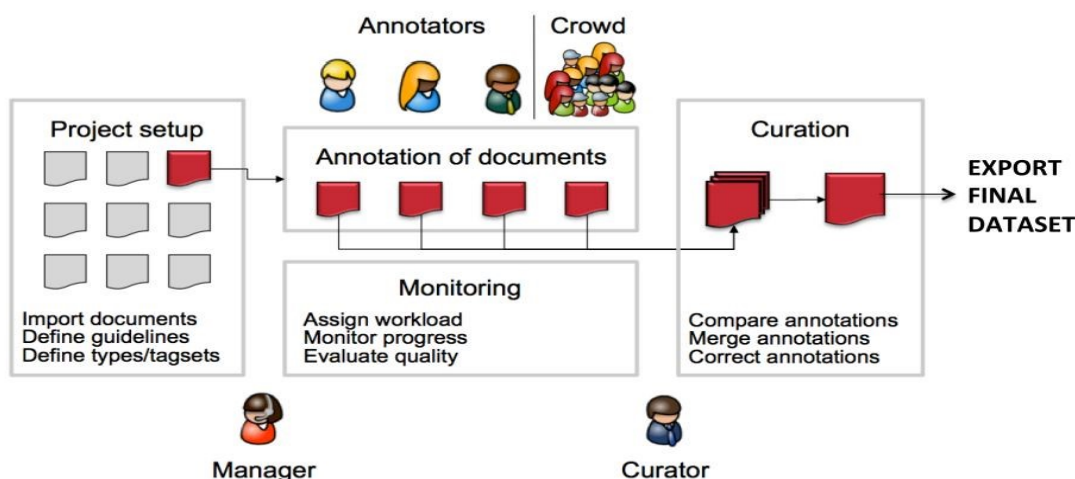


Figure 4.2: Workflow of an annotation project in WebAnno

[74]

The lemmatization model was trained in a supervised fashion via the maximum likelihood method, using a dataset that is hand-tagged with lemmas, POS tags and NE tags. We therefore, annotated the dataset over three layers being lemma, POS, NER. These layers will play a major role, as they will provide the algorithm with metadata features as to what is relevant about the dataset. Table 4.1 shows the statistical parameters of the annotated dataset.

4.3.1 Lemma layer

A supervised approach to machine learning requires fully labeled examples to allow the algorithm to learn an inferred function that can later be used to map new examples. Annotation on the

Layer	No of words	Tagged words	Untagged words
POS	15,516	15,516	0
NER	15,516	15,516	7192
Lemma	15,516	15,516	0

Table 4.14: Statistical parameters of the dataset showing the number of tagged words and untagged words

lemma layer aims to identify and annotate words with their uninflected forms (i.e. headword-form or base-form) in the dataset. Words are either in their base form or inflected forms. Thus the lemma for the verb *thusitse*/helped would be *thusa*/help, while the lemma for the noun *mosadinyana*/young woman would be *mosadi*/woman. The lemma tagging process was done with reference to Thanodi ya Setswana [75]. Figure 4.3 shows an example of words tagged with their base forms. Every token is followed by a / and its assigned lemma.

Iketleleng/iketle boloto/boloto Tshireletso/Tshireletso ./ Banana/monana bangwe/bangwe mo/mo lefatsheng/lefatshe leno/leno b
a/ba kgaletswe/kgala mokgwa/mokgwa o/o o bosula/bosula wa/wa go/go rata/rata go/go ikgatla/ikgatla ba/ba re/re bone/bone
ba/ba tlhakanela/tlhakanela dikobo/kobo "/" boloto/boloto "/" kgotsa/kgotsa ba/ba sa/sa dirise/dirisa dikhondomo/khondomo ./

Figure 4.3: Example of word reduced to their lemma.

4.3.2 POS layer

In linguistics part of speech (POS) tagging refers to grammatical tagging, which is a method of marking a word in a given text as corresponding to a specific part of speech category [76]. In this project, POS tagging is important because task such as lemmatization (i.e., lemma identifier) require to know the part of speech of each token before lemmatizing them. A word's role in a sentence is closely tied to its part of speech. The same inflectional word in one sentence can elucidate a certain lemma while in another sentence it means something different. Thus, POS provides information of essential importance to text comprehension before lemmatization takes place. Abbreviations for part of speech categories are called tags, and the set of tags used for part of speech tagging are called tag-set.

Our developed Setswana tag-set as explained in Section 4.2 contains categories including nouns, verbs, pronouns, adverbs, particles, interjections, idiophones, etc. A detailed description of individual tag categories and labels with examples has been provided in Tables 4.1 to 4.11. Word categories can either be open or closed. An open word category accepts new words while a closed word categories has a fixed set of tokens. The tag-set used for annotating the POS layer contained close to 55 tags, and the highest frequency of words in our dataset are verbs. Figure 4.4 shows the distribution of POS tags in the annotated dataset.

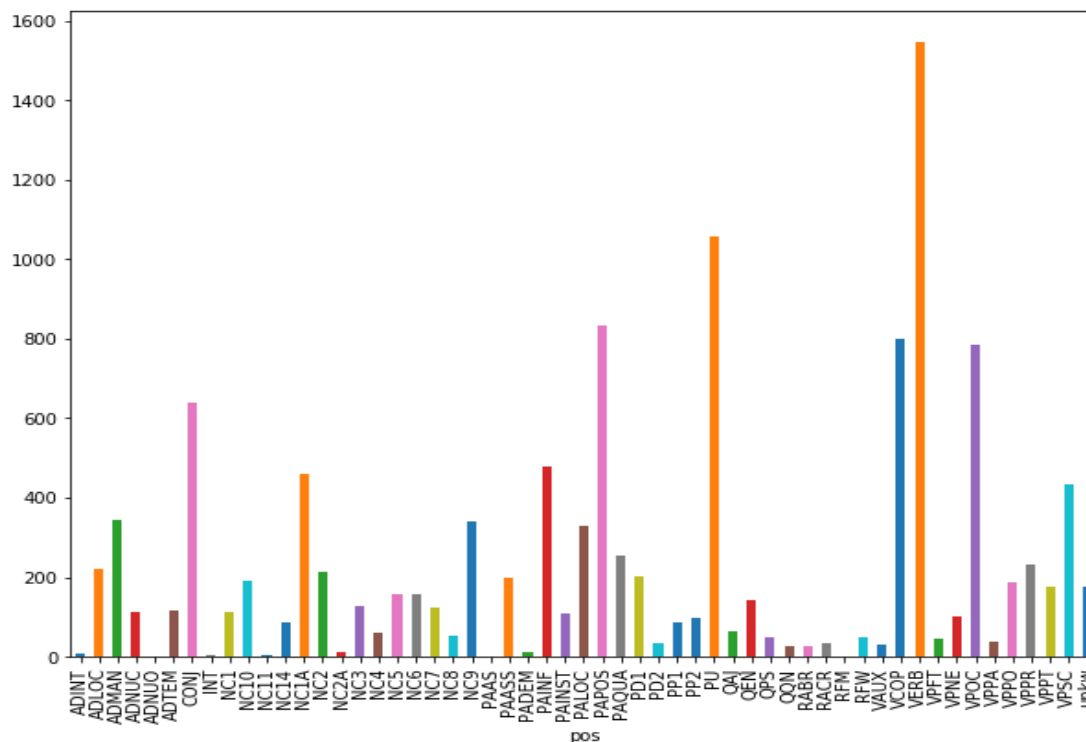


Figure 4.4: Distribution of POS tags in the annotated dataset.

4.3.3 Named Entity Recognition

Named entity recognition (NER) is a standard NLP task that aims to locate named entities in text and assign them some categories. These are usually proper nouns like name of a person, locations, organizations, time, etc [77]. Named Entities (NE) plays an important role in natural language text. The identification of entities before lemmatization is of paramount importance, since the lemmatization of NE requires special treatment because it is difficult to lemmatize similar words appearing in a different context. NE provides inference abilities for our model to know how to lemmatize word with regard to context, acting as a special feature to tackle the disambiguation problem.

Abbreviations for named entity types are called tags, and the set of tags used for named entities are called tag-set. A detailed description of individual tag categories and labels with examples have been provided in Table 2.13. Figure 4.5 shows the distribution of named entity types in the annotated dataset.

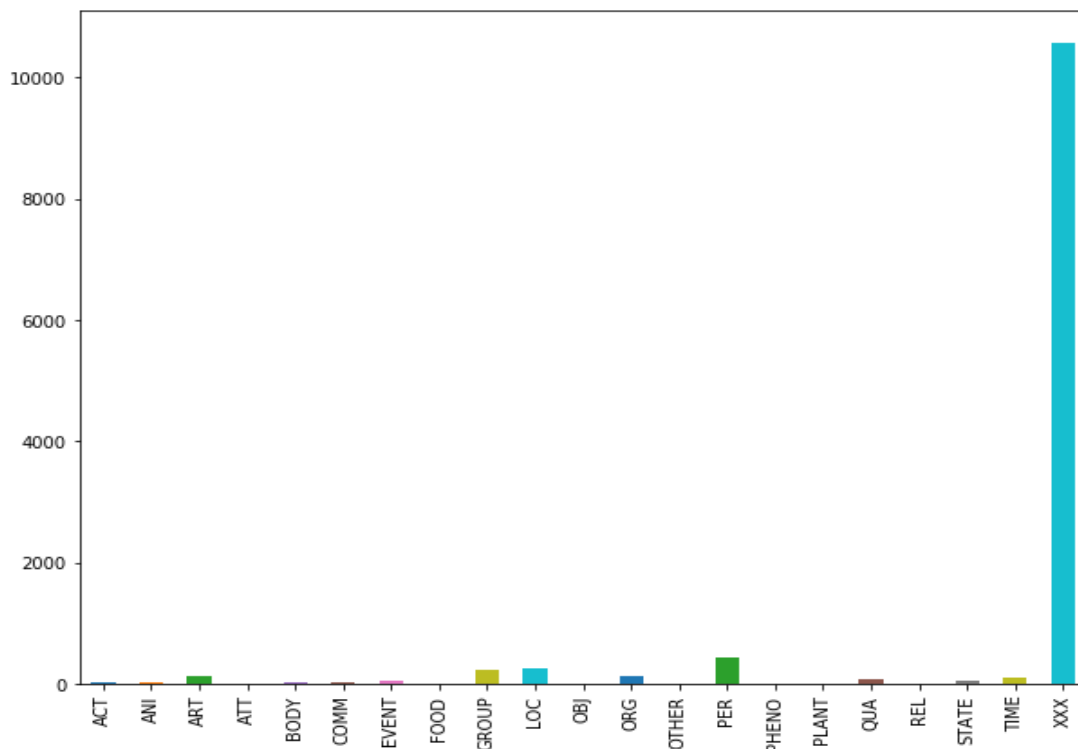


Figure 4.5: Distribution of NER tags in the annotated dataset.

After the annotation team had collaboratively annotated the dataset on different layers it was then passed to the curation process to check for any annotation mistakes. Then the tagged documents were exported as tab-separated values (TSV) files and merged into a single document. At this stage, the dataset contained 386 annotated sentences equating to 15,516 words with 2,371 unique words.

4.4 Data Preparation

Most researchers describe data preparation as the most intensive step in model building [78]. Machine-learning algorithms use data as a sequence of samples, so file formats for machine learning models should be in proper layout. The dataset was annotated with WebAnno in an unstructured text format (.txt).

Several text file documents that were collected from different sources such as, Mmegi, Bopa and Kutlwana, and then loaded into WebAnno for annotation, after annotation the annotated files were exported to our workstation in tab-separated values (TSV). WebAnno did not provide us with the service of automatically merging documents into one large file, therefore, after the annotation process, we performed data integration and file transformation.

4.4.1 Data Integration

The first problem to be addressed in data preparation stage is data integration. This is a method of merging data information from different sources into a single information unit used for information mining. Out of all the annotated files which we exported to our workstation

from WebAnno, we wanted to have a single statistical data matrix i.e. a database table, where columns of the table represent features and rows corresponds to a given member of dataset in question.

The Windows Operating System already has some methods to join multiple files together using the command line by running the script as shown in figure 4.6.

```
C:\Users\kgosiyame\Desktop\corpus>copy *.tsv setswana_corpus.tsv
```

Figure 4.6: Script to integrate multiple TSV files into one TSV file.

The command will read all the contents of .tsv files and output it all to the combined .tsv file called setswana_corpus.

4.4.2 File Transformation

The second consideration in data preparation is file transformation. A file format describes the structure and encoding of the data stored in it and it is mostly identified by its file extension. One of the most important file formats to work with in machine-learning is comma separated values (CSV) format. However, this file format was not provided by our annotation tool i.e WebAnno.

We, therefore, exported the annotated document from the WebAnno as a tab separated values (TSV) and converted them to a comma separated values (CSV), which is the most common format for machine learning projects. CSV files are popular for machine-learning as they are easy to view, debug and process by any application. Both the TSV and CSV file formats of our dataset are illustrated in Figure 4.7.

Along with this process of putting together data in an optimal format we manually corrected minor spelling errors to improve our data quality.


```

#FORMAT=WebAnno TSV 3.2
#T_SP=de.tudarmstadt.ukp.dkpro.core.api.segmentation.type.Lemma|value
#T_SP=webanno.custom.SemanticEntity|category
#T_SP=webanno.custom.SetswanaPOS|PosValue

#Text=Rra Gaone o bolokile Setswana.
1-1 0-3 Rra _ PER[1] NC1A
1-2 4-9 Gaone _ PER[1] NC1A
1-3 10-11 o _ VPSC
1-4 12-20 bolokile boloka _ VERB
1-5 21-29 Setswana _ _ NC7
1-6 29-30 . _ _

```

a Example of initial dataset from WebAnno in TSV Format

1	#FORMAT=WebAnno TSV 3.2						
2	#T_SP=de.tudarmstadt.ukp.dkpro.core.api.segmentation.type.Lemma value						
3	#T_SP=webanno.custom.SemanticEntity category						
4	#T_SP=webanno.custom.SetswanaPOS PosValue						
5							
6							
7	#Text=Rra Gaone o bolokile Setswana.						
8	1-Jan 0-3	Rra	_	PER[1]	NC1A		
9	2-Jan 9-Apr	Gaone	_	PER[1]	NC1A		
10	3-Jan 11-Oct	o	_	_	VPSC		
11	4-Jan 20-Dec	bolokile	boloka	_	VERB		
12	5-Jan 21-29	Setswana	_	_	NC7		
13	6-Jan 29-30	.	_	_	_		

b Example of the dataset after being converted into CSV format

Figure 4.7: Dataset format

4.5 Model building

Classification is an outstanding task in data analysis. To solve a wide range of problems such as NLP related tasks most scientist use classifiers. This classifiers maps a set of input attributes x into its class labels y [17]. In the same way, the principal approach to our lemmatization model takes the structure of a machine learning classifier. In this approach inflected words are assigned lemmas based on the likelihood suggested by the training dataset of labeled documents.

Instead of hard-coding the solution to a particular linguistic problem in a collection of hand-crafted rules, our data driven method attempts to extract the necessary linguistic classification properties from the annotated dataset.

In this master thesis, we implemented the Naive Bayes (NB) algorithm to learn lemmatization of Setswana words. The NB algorithm is adopted because it gives a way of counting the cases

(from Setswana training dataset) to develop tables regarding the probabilities of certain lexical items as well as pattern usage of words in the formation of sentences. This gives the Naive Bayes algorithm the ability to model very complex functions because it describes the probability of an event, based on prior knowledge of conditions that might be related to the event [40].

By adopting the use of Naive Bayes algorithm the proposed model obtains better ratios and reasoned deduction of events based on prior knowledge of observation on how to lemmatize words using the method of maximum likelihood. To decide on how to lemmatize a word, the algorithm is informed by lexical features encoded in the form of tags (i.e., POS tags and NE tags).

Using estimations from training data, the Naive Bayes algorithm can calculate the probability of each potential lemma. To find the optimal lemma (l_{opt}) of a new data instance equation 4.1 is applied [79].

$$\left\{ \begin{array}{l} l \\ l_{opt} \end{array} \right. = \arg \max_{l_j \in l} P(l_j) \prod_{i=1}^n P(x_i | l_j) \quad (4.1)$$

Given a sequence of words in a sentence the task of the proposed lemmatization model is to use predictor variables x_1, \dots, x_n to produce a sequence of associated lemmas l_1, \dots, l_n for each word in a sentence. Now consider Figure 4.7 which gives an abstract representation of the pipeline specific to our approach.

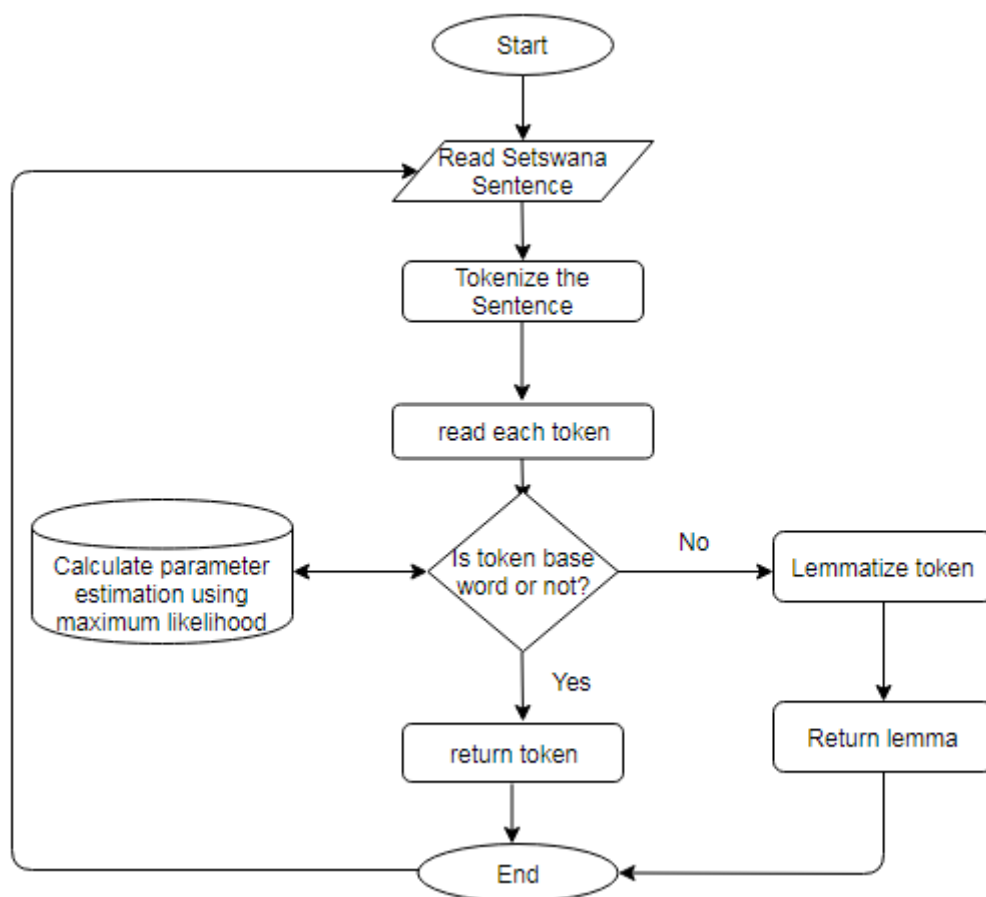


Figure 4.8: A flowchart of lemmatization model.

The model receives a sentence as an input, tokenizes the sentence and outputs for each word in the sentence a candidate lemma without using any external resources such as dictionaries. We pre-annotated the dataset with linguistic features of part of speech (POS) and named entity (NE) tags to enrich it with contextual information, and then employed the feature identification process of n-gram to find what sequence of tags are most common.

4.6 Technologies & Environments

Natural language is largely computational, we need the right tools to do it. To build our lemmatization model and carrying out the experiments of the study, we utilize the availability of open source libraries. To build the model, we first set up python on our windows machine using Anaconda. Anaconda is a full distribution software that installs Python, IDEs (Integrated Development Environment), and other packages needed for any data science project. Among IDEs installed with python is the Spyder and Jupyter Notebook. For developing our model and carrying out experiments we used Jupyter Notebook, which runs via a web browser. In this environment, we can easily mix code with results of that code, including graphs and other data visualization in one file.

4.6.1 Programming Language

The source code for this project was written in Python. Python is an open source, high level programming language and it is among the fastest growing programming languages in the world due to the easy of learning involved and its surprising capabilities, code written in python are considerably shorter and simpler than other high-level languages such as Java and C++. In addition, python has well designed built-in features and standard libraries that make programming in python more efficient. These open-source libraries helps developers to automate their tasks very easily.

4.6.2 Overview of libraries

This project uses several libraries to support the coding part, these include:-

- Scikit-learn
- Pandas
- Numpy
- Matplotlib
- Natural language tool-kit (NLTK)

Scikit-learn is one of the most gigantic python module that provides tools for data mining. The library has algorithms that support classification, regression and clustering. Through a consistent interface, the scikit-learn offers a good amount of supervised and unsupervised algorithms together with evaluation metric that helps developers to construct models very easily in a consistent manner. For the implementation of this thesis, the Naive Bayes algorithm was taken from Scikit-learn library.

Pandas :- this library provides powerful, expressive and flexible data structures that make data manipulation and analysis easy for python programming language. Pandas is suited for many different kinds of data including tabular data, ordered and unordered data (i.e. time series data), arbitrary matrix data with rows and columns and any form of observational/statistical dataset. With the help of this library we can load, prepare manipulate, model and analyze data.

Numpy:- (short for Numerical Python), the library has been specifically designed to support multi-dimensional array, matrix structures, as well as mathematical functions to operate on these arrays, by using numpy we can perform important mathematical and logical operations on the dataset.

Matplotlib:- This is one of the most popular python package used for data visualization. It is a plotting package that is useful for providing good quality graphics to plot 2D and 3D figures from our dataset.

Natural Language Tool-Kit:- The Natural Language Tool-Kit (NLTK) is an open-source program providing ready to use computational linguistic resources [80]. It offers different methods of pre-processing such as tokenization. NLTK has a module tokenize that comprises of sub-modules sentence_tokenize and word_tokenize. In this project, the word_tokenize module was used.



5. Experimental Results & Discussions

In this chapter, we provide and discuss the experimental setup for our project. We discuss the steps carried in engaging the developed dataset as described in the previous chapter, into building the proposed lemmatization model, i.e., the pre-processing, feature selection method, the training process and lastly we discuss the results of our work.

5.1 Experimental setup

Below is the work flow of the steps followed in order to build the lemmatization model:-

- Import libraries
- Dataset loading
- Text pre-processing
- Vectorization and feature selection
- Train model (Train/Test set)
- Model Evaluation

The figure below illustrates the big-picture view of our experimental process:

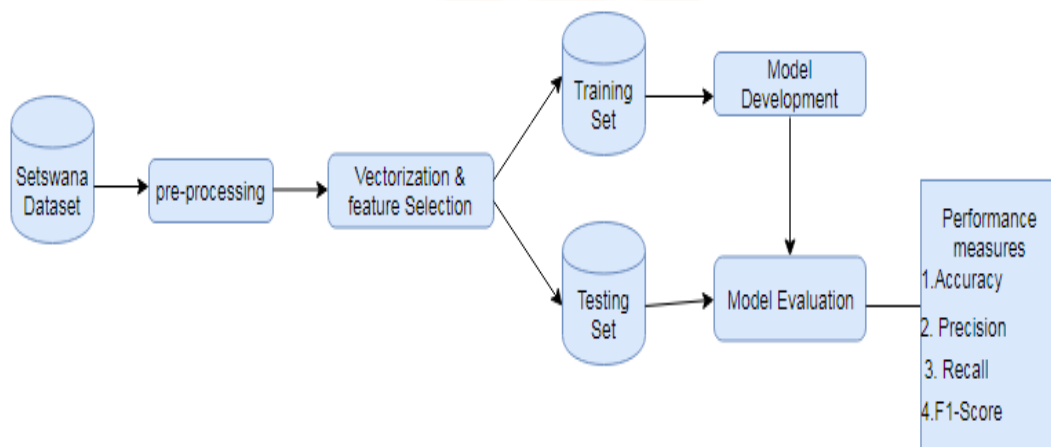


Figure 5.1: Experimental process setup

5.1.1 Import libraries

As mentioned earlier in Section 4.5, this project make use of available libraries such as Scikit-learn, Pandas and Numpy. All these libraries had to be made available in our working space.

Figure 5.2 shows a simple python code executed to import the required libraries.

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.naive_bayes import GaussianNB
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
```

Figure 5.2: Some used libraries

5.1.2 Dataset loading

The goal of this project is to develop a data-driven lemmatization model for the Setswana language. The model works with data to learn patterns on how to lemmatize Setswana words. Once the data has been prepared and tagged using the WebAnno tool we exported it into our workstation and merged it to form a single document, sitting in our local system we load it directly into our application during execution. The dataset is read as a pandas dataframe object, using the readcsv function to make it available in our project.

```
# Import the dataset into working space
df = pd.read_csv('setswana_corpus.csv')
```

	Sentence #	word	pos	entity	lemma
0	Sentence: 1	iketleleng	VERB	XXX	iketle
1	Sentence: 1	boloto	ADMAN	XXX	boloto
2	Sentence: 1	Tshireletso	NC1A	PER	Tshireletso
3	Sentence: 1	.	PU	XXX	.
4	Sentence: 2	Banana	NC2	PER	monana
5	Sentence: 2	bangwe	QEN	XXX	bangwe
6	Sentence: 2	mo	PALOC	XXX	mo
7	Sentence: 2	lefatsheng	ADLOC	LOC	lefatshe
8	Sentence: 2	leno	PADEM	XXX	leno
9	Sentence: 2	ba	VPSC	XXX	ba

Figure 5.3: Making the dataset available into the project

5.1.3 Text pre-processing

Data cleaning is a very crucial step when building a machine learning model. One of the steps to improve the performance of our model was to apply some pre-processing on our textual data to prepare it for statistical modeling. The key procedure in this stage was to figure out how to deal with noise that comes in the form of unwanted spaces, unwanted characters and missing values. Depending on the nature of our problem we had to do some cleaning on white spaces, unwanted characters and null values. We made sure that this pre-processing did not cause any problems with the structure of our sentences.

Most of the special characters, for example functional symbols, such as Dr., LT, III which comes before a person's name were removed from our dataset as they were not important and did not add value to our results. Null values are a serious noise in machine learning algorithms, and one way of dealing with them was to replace them with a default value. In our case, all null values in the dataset were given character "XXX ". Most null values were found in the NER layer. We also removed excess white spaces from our dataset. Text pre-processing compressed the vocabulary of our dataset by 22.2% from 15,516 to 12100 words with 2366 unique words. We also did exploratory analysis on the dataset to understand the relationship vocabulary size and unique words, distributions of various N-grams and information that helped to prepare the data for statistical modeling.

5.1.4 Vectorization & Feature Extraction

The main hurdle in dealing with text-data is that it is non-numeric, it is not well structured and does not go hand in hand with computers. As a result, we had to turn the text in our dataset into a vector of numbers which are used as input in machine learning algorithm i.e., Naive Bayes.

The word vectors becomes the learned representation of the input that is used in the model. The most useful information from vector representations is that similar words i.e., those that show up in similar contexts end up with comparable vector representations. By using vectors, the model can extract relevant features which will help it learn from existing data and make predictions about text to come since they represent word's distributed weight proportional to the frequency in which a word shows up in our dataset.

To vectorize our dataset we made use of CountVectorizer provided by the Scikit-learn library.

CountVectorizer

CountVectorizer offers an easy way of both tokenizing a collection of text and building a matrix with the count of words in the dataset. It also has other helpful features particularly the n-grams range i.e., co-occurrence information of words.

A typical example of forming a vector is shown by example in Figure 5.3. Suppose a dataset contains two sentences of unique tokens extracted out of the dataset. The tokens will form our dictionary and the size of the count vector-matrix will be given by unique tokens in the dataset.

- Sentence 1: [*Bana ba a lela* / The children are crying]
- Sentence 2: *Bana ba ile sekoleng* / The children went to school]

From these sentences, a vocabulary of five words is learnt. The count matrix is of size 2*6 and it will be represented as shown in Figure 5.4.

	<i>Bana</i>	<i>ba</i>	<i>lela</i>	<i>ile</i>	<i>Sekoleng</i>
<i>Sentence :1</i>	1	1	1	0	0
<i>Sentence :2</i>	1	1	0	1	1

Figure 5.4: An example of token frequency as vector encoding generated under countvectorizer scheme

Now a column can be seen as a vector for the corresponding word in the matrix. For instance, in the above matrix, the term vector for sekoleng/school is [0, 1] and so on.

Figure 5.4 is an example to offer the reader an understanding of how to construct a vector representation of the text in the dataset. This text Vectorizer counts how many times each vocabulary item occurs, and makes a feature for each one. So we had to make sure that we have a knowledge of the relationship between words in the dataset, therefore, we used the N-gram technique.

N-gram

N-gram is one of the basic and efficient feature identification process used in some of the natural language processing systems to evaluate what sequence of words are most common in language modeling. It refers to the continuous sequence of words with length n [81]. N-gram started with Claude Shannon in 1948 when he investigated the process of predicting the next letter in a given sequence of letters. Since then the use of n-gram expanded into other applications such as machine translations.

Examples of n-gram commonly used include:- unigram($n=1$), bigram ($n=2$), trigram(3), etc. For example, the word based n-gram of the following Setswana sentence is:-

“Bonang, maru a thibile.”

Uni-gram \implies *Bonang, maru, a, thibile.*

Bigram-gram \implies *Bonang maru, maru a, a thibile.*

Tri-gram \implies *Bonang maru a, maru a thibile.*

Through the knowledge of n-grams, we add multiple variables that depend on each other to the patterns we use to detect information in text, this generates richer a pattern. In our model, we used the n-grams range of 2 (i.e bigram). The introduction of the n-gram embedding takes into account word order in a short context and can further enrich the text representation.

For feature selection, from the dataset shown in Figure 5.2, we created a target vector “y” and predictor “X” matrix. y is the feature we are trying to predict i.e., lemma in our model and X is the remaining matrix after selecting the target vector y .

5.1.5 Training Model

In this phase, a classifier model is generated using the Naive Bayes algorithm. There are several practices that we can embrace to train the proposed data-driven lemmatization model amongst them, k-fold cross-validation, or the Holdout validation (train/test split).

K-fold Cross Validation

K-fold cross-validation is a common practical technique to get a good estimate of the error of a learning algorithm. Here the data is divided into k-fold i.e. segment of the same size. The advantage of this practice is that ultimately all the instances in the dataset are used for both training and testing, as a result it is quite effective in decreasing overfitting by rotating validation continuously. An inconvenience with this practice is that it is computationally costly regarding processing power because of extra training that needs to happen. Moreover, an overlap between the training and validation set can lead to overestimation of the performance of the model.

Holdout Validation

In our work, we adopted the Holdout validation practice. Most machine learning techniques are well served by a train/test split (i.e Holdout validation) method of the Scikit-learn library. In this method, the dataset is divided into two parts i.e. training and testing set. The division of dataset into training and testing set was made by random division into two sets, in proportion 60% (training) and 40% (testing) as shown in Figure 5.5. Holdout validation method requires less computational resources as compared to K-fold cross-validation and at the same time performs well on our dataset. The training set holds the subset of the data to train the model (known data) while the test set holds the subset of the data that the model has not seen before (unknown data).

Procedure 1: Procedure to Create Training Set and Test Set in Holdout Validation

Input: Setswana DataSet

Output: TrainingSet, TestSet

begin

1. SET trainingSetSize = DataSet.Count / 2
2. SET testSetSize = DataSet.Count - trainingSetSize
3. SET TrainingSet = DataSet.Take(trainingSetSize)
4. SET TestSet = DataSet.Skip(trainingSetSize).Take(testSetSize)

end

For the training part we used the `.fit()` function to train the model by showing it a bunch of examples from the training dataset.

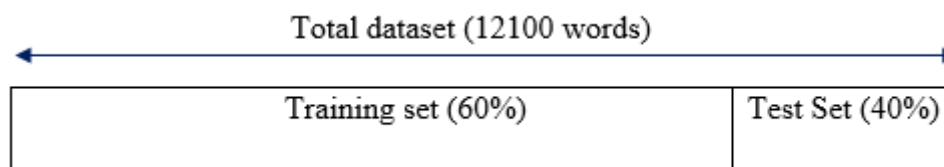


Figure 5.5: Dataset splitting

5.1.6 Evaluation

Intrinsic evaluations such as accuracy seem to be the general trend in evaluating the performance of natural language text based systems [82]. From the previous studies, as shown in literature in chapter 3, all the reported works on lemmatization use accuracy as a performance measure. The python programming language has a metric library i.e., *sklearn.metrics* which can be used to generate a report of the performance of the developed model based on precision, recall and f1-score.

In the same way, through accuracy, the quality of the developed model is evaluated based on the percentage of correct predictions over total instances. However, measuring accuracy of the model does not necessarily reflect its good performance [83]. For example, in situations where there is a large class imbalance, the model can predict the value of the majority for all predictions and achieve a high accuracy.

In addition to accuracy, we also report precision, recall, and F-1 score. These measures are not directly processed from the raw classifier output, but from a matrix known as confusion matrix. A confusion matrix is a grid containing information about the actual and predicted classes, and has as many columns and rows as there are classes [84]. In itself, a confusion matrix is not a performance measure but almost all the performance measures are based on the numbers within it.

The confusion matrix is based on four parameters being true positive, false positive, true negative and false negative. A true positive (TP) is a result where a positive class is properly predicted by the model, i.e when the actual lemma of a word is true, similarly, a true negative (TN) is a result where the model predicts the negative class properly. A false positive (FP) is a result in which the model predicts the positive class wrongly, and a false negative (FN) is a result in which the model predicts negative class wrongly.

Figure 5.5 below shows a sample format of a confusion matrix with n classes.

		Predicted Number			
		Class 1	Class 2	...	Class n
Actual Number	Class 1	x_{11}	x_{12}	...	x_{1n}
	Class 2	x_{21}	x_{22}	...	x_{2n}

	Class n	x_{n1}	x_{n2}	...	x_{nn}

Figure 5.6: Confusion matrix table of n classes

Accuracy:- Accuracy is the most intuitive performance measure, it refers to the ration of correctly lemmatized observation to the total observation. This can be indicated in mathematical terms as:-

$$\left\{ Accuracy = \frac{\text{total number of correct predictions}}{\text{total number of predictions}} \right. \quad (5.1)$$

Our model obtained an accuracy of 70.32%, which means our model is approx. 70% accurate.

Precision: also referred to as positive predictive value, tells how many words were predicted correctly out of the ones that were predicted as belonging to a given tag, it takes a number of words that were correctly predicted positive for a given tag i.e. lemma and divides it by the number of words predicted correctly and incorrectly as belonging to that tag i.e. lemma.

$$\left\{ Precision = \frac{TP}{TP + FP} \right. \quad (5.2)$$

Recall is the proportion of appropriate cases obtained over the complete number of appropriate cases irrespective of whether they have been predicted correctly. This measure states how many words were predicted correctly out of the ones that should have been predicted as belonging to a given tag, it is given by the formula below:-

$$\left\{ Recall = \frac{TP}{TP + FN} \right. \quad (5.3)$$

F1-score:- is the harmonic mean of precision and recall taking both metrics into account in the following equation:-

$$\left\{ F1 - score = \frac{2 * precision * recall}{precision + recall} \right. \quad (5.4)$$

Evaluating our model using accuracy, recall, precision and f1-score we achieved 70.32%, 70%, 65% and 66% respectively. Figure 5.7 below shows the results of all the performance measures used. These results are based on holdout validation practice.

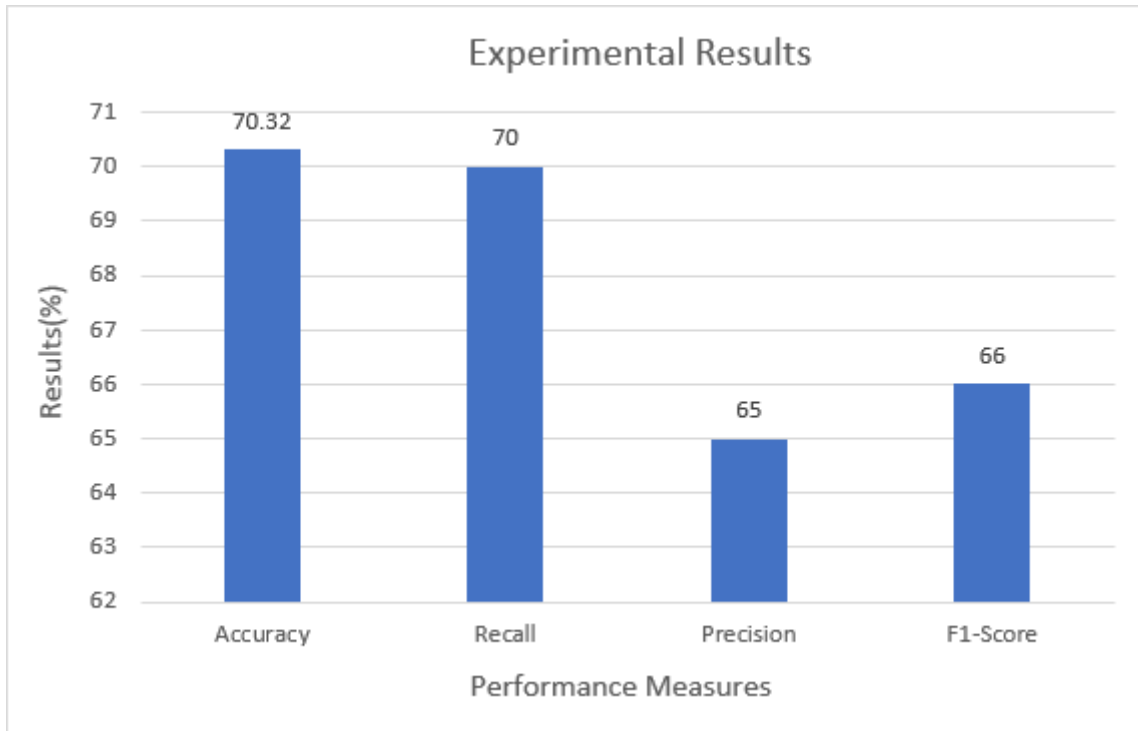


Figure 5.7: Results of experiments

The major concern of our work is to lemmatize words based on the context of how they are used. To further evaluate our model to see if it achieves this aim, a number of test cases were carried out by interactively typing Setswana declarative sentences to see how our model lemmatizes them.

Test cases is the process of validating a particular objective. This will inform us if the expected behaviour of our model is fulfilled. In our case, the purpose is to understand the performance of the lemmatization model in correctly identifying the lemma of a word in a given context by manually judging the produced lemmas. Our proposed lemmatization model is explained by pseudo-code in Figure 5.8.

Procedure 2: Pseudo-code of the model**Input:** Setswana Sentence**Output:** lemma of each token in the Sentence**begin**

```

| read  Setswana  test sentence
| foreach  test sentence  do
|   tokens=word_tokenize(sentence)
|   foreach  token in test sentence  do
|     if  token is a lemma  then
|       | return "token"
|     else
|       | lemmatimize token
|       | return "lemma"
|     end
|   end
| end
end

```

Figure 5.8: lemmatization procedure

A number of novel results as well as failures associated with the developed model have been discovered.

5.1.7 Test Cases

At the most core level, NLP systems are designed to achieve some task that can be evaluated in an input-output manner. Thus, evaluation boils down to a simple question: to what degree does the system input corresponds to correct desirable outputs? The interface to our model is a text field where the input is the sentence to be lemmatized, then the output is all the lemmas associated with each word in the sentence. Given sentence s , the task of the model is to predict the lemma $l_i \in l$ for each given word w_i where l is a set of all possible lemmas. In the first test case shown in Figures 5.9-5.10, we carried out experiment with a fairly simple sentence that does not contain any cases of polysemous words and therefore the lemmatization procedure becomes less difficult as there is only one lemma that can be assigned to a word.

In this test the model takes an input as a sentence and returns all the lemmas of the words in the sentence through the `.predict()` function, based on the training dataset.

An example of the result from the model is shown below:-

Based on accuracy measurement i.e. counting the number of words with correct lemmas and dividing by the total number of words in the sentence, our model produces a score of 12/14 in this sentence which indicate 85% accuracy. Trying to lemmatize this sentence using a rule-based approach will be complicated introducing enormous number of rules. The complexity to develop

INPUT

```
Setswana_Sentence= str(input(""))
```

```
Dikotsi tsa tsela di tsweletse ka go ntsifala le g
```



```
Dikotsi tsa tsela di tsweletse ka go ntsifala le go gapa matshelo a batho
```

Figure 5.9: input example

OUTPUT

```
print(" lemmas=", clf2.predict(count_vect.transform((tokens))))
```

```
lemmas= ['kotsi' 'tsa' 'tsela' 'di' 'tswelela' 'ka' 'go' ',,' 'le' 'go' 'gapa'
'botshelo' ',,' 'motho']
```

Figure 5.10: example of the model prediction output

rules for such coherent sentence lies in the fact that words are too many and may contain similar characters which needs a lot of exceptions to result in good performance.

In the experiment above, the model was unable to predict the lemma for the word *ntsifala* and *a*. This may be due to various reasons among them:-

- Error due to variance:- Variance is the measure of how susceptible the model is to the training data subset. The training technique used, as explained in chapter 5, uses restricted examples from a bigger population. Therefore rare words (i.e. words with less frequency) may not be covered in training which leads to the lemmatizer mistakenly lemmatizing words because they receive less reliable estimates.
- Imbalanced dataset:- When machine-learning algorithms are trained on a dataset with a high degree of imbalance, they tend to be overwhelmed by the majority observations and incorrectly predict the minority observations. Imbalanced dataset is a problem in natural language processing because words present a highly skewed distribution [85].

Ideally, a good lemmatizer will lemmatize all words used in a certain context or semantic group to the same lemma. Due to irregularities that are there in all natural languages, all lemmatizers are prone to mistakes, therefore, we might say that no lemmatizer can be expected to work perfectly.

It is very important to understand that predictions made by machine-learning algorithms are advised by training data. The ultimate improvement to the aforementioned problems are the continuous improvement on the dataset, especially in areas of the input domain where the model fails to correctly predict lemmas for words. Machine-learning improves as the amount and quality of training data is increased [23].

To our best knowledge, given the progress that has been made in lemmatization of Setswana language [1], [4], [13], [14], [15], none of them addresses the issue of disambiguation. These lemmatizers lemmatize each word independently but not with its relation to other words. Through test cases, we demonstrate the contribution of our study, that is, to disambiguate the occurrences of polysemous words in the lemmatization of Setswana language. We are identifying the actual lemma of an ambiguous word based on distinct situations with sentences.

Figures 5.11-5.16 show how our proposed model handles lemmatization of the same words that are used in a different context. From this test cases, we observe cases of awareness in the context of how words are used. At first, we tested our model on the polysemous word *Bonang* (look) used in two different sentences:

- *Bonang o ile lesong.*
- *Bonang, maru a thibile.*

Figure 5.11 below shows an example of the results from the model when the word *Bonang* (look) is used as a person's name (i.e. noun class 1a NC1a).

INPUT

```
Setswana_Sentence= str (input(""))
```

```
Bonang o ile lesong
```



```
Bonang o ile lesong
```

OUTPUT

```
print("lemmas=", clf2.predict(count_vect.transform ((tokens))))
```

```
lemmas= ['Bonang' 'o' 'ile' 'o']
```

Figure 5.11: *Bonang* as noun (i.e. NC1a)

Figure 5.12 shows the result from the model when the word *Bonang* (look) is used in a different context as a verb.

INPUT

```
Setswana_Sentence= str (input(""))
```

```
Bonang maru a thibile
```



```
Bonang maru a thibile
```

OUTPUT

```
print("lemmas=", clf2.predict(count_vect.transform ((tokens))))
```

```
lemmas= ['Bona' 'leru' '.' 'thiba']
```

Figure 5.12: Bonang as verb

Figures 5.13-5.14 below shows an example of the result from the model when the word *Dimpho* is used at the beginning as well as in the middle of a sentence.

- *Dimpho di abiwa ke ntate.*
- *Mogokgo wa sekole rre Dimpho Bonang o kopile bana go ithuta.*

INPUT

```
Setswana_Sentence= str(input(""))
```

```
Dimpho di abiwa ke ntate
```



```
Dimpho di abiwa ke ntate
```

OUTPUT

```
print(" lemmas=", clf2.predict(count_vect.transform((tokens))))
```


```
lemmas= ['mpho' 'di' 'aba' 'ke' 'ntate']
```

Figure 5.13: Dimpho used at the beginning of a sentence

INPUT

```
Setswana_Sentence= str(input(""))
```

```
Mogokgo wa sekole rre Dimpho Bonang o kopile bana
```

Mogokgo wa sekole rre  Dimpho Bonang o kopile bana go ithuta

OUTPUT

```
print(" lemmas=", clf2.predict(count_vect.transform((tokens))))
```

```
lemmas= ['mogokgo' 'wa' 'sekole' 'rre' 'Dimpho' 'Bonang' 'o' 'kopa' 'ngwana' 'go' 'ruta']
```

Figure 5.14: Dimpho used in the middle of a sentence

The use of n-gram in our model can help to determine that a word preceded by the word “rre” has a high chance to be regarded as a person’s name.


We also tested our model on the ambiguous word *metseng* used in two different sentences. Figures 5.15-5.16 below we shows an example of the results from the model when a sentence contains ambiguous word that should be resolved before lemmatization. In two different sentences the model correctly disambiguated the word *metseng* which can take different lemmas depending on sentence scenarios.

- *mo metseng e gaufi.*
- *a re metseng dijo.*

INPUT

```
Setswana_Sentence= str(input(""))
```

```
mo metseng e gaufi
```

 mo metseng e gaufi

OUTPUT

```
print(" lemmas=", clf2.predict(count_vect.transform((tokens))))
```

```
lemmas= ['mo' 'motse' 'e' 'gaufi']
```

Figure 5.15: Metseng used as a noun (i.e. villages)

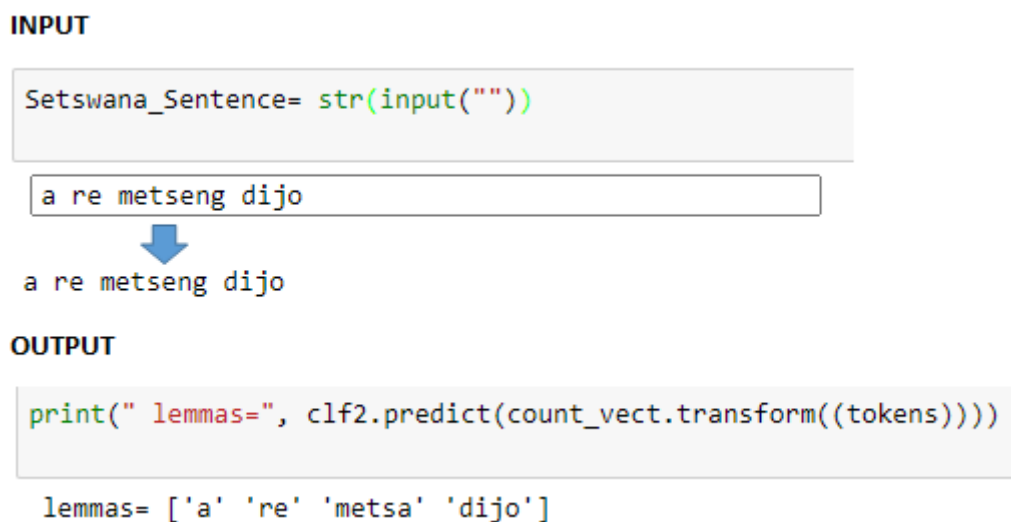


Figure 5.16: Metseng used as a verb (i.e. to swallow)

The novelty of our work lies in the fact that it can lemmatize polysemous words based on the contextual information, unlike the former works [1], [4], [13], [14], [15], that only produces one lemma for each word even though its lemmatization can vary in different contexts. This can be clearly shown by how the model treated the lemmatization of inflectional words *Dimphe*, *Bonang* and *Metseng* used in different sentences. (see Figures 5.11-5.16).

The input to our model is a sequence of words (i.e., sentence), unlike the prior works on lemmatization of Setswana language which used a rule-based methods on a single word per lemmatization process. This makes it hard to assess how they will perform when faced with a coherent sentence and are silent on the context of how words are used.

It is important to note that our model report lower accuracy as compared to the previous works on Setswana lemmatization. This could be attributed to variety of reasons, including the difficulty in dealing with disambiguation of words that our model has introduced. Previous lemmatizers lemmatize words without contextual information as they are can only produce one lemma for each word even though a word may have different lemmas. Furthermore, the difference in the size of the dataset used can contribute in different performances. Our model can deal with a much higher dataset than the previous works on Setswana lemmatization. In rule-based methods using a less dataset can favor the performance of the model but when the data increases problems in scalability will start to surface, hence the degradation in performance.

The previous works tend to focus on one aspect separately (i.e. nouns and verbs). This specialization makes it hard for them to assess how they will perform in cases where they are faced with words of different semantic categories.

6. Conclusions & Recommendations

In this thesis, to the best of our knowledge we made the first step into exploring context-aware lemmatization technique for the Setswana language using a statistical machine-learning approach. The work is motivated by the desire to push further the language tools for Setswana, to enjoy improved human-machine interaction. In the computer world, especially the internet content, Setswana language is less represented, and using Setswana in computerized systems is an issue due to its complex morphology and lack of language tools especially morphological analyzers.

The novelty of this work lies in the fact that the model can lemmatize words based on the context of how they are used by utilizing lexical features of POS tags and NE tags together with the n-gram approach. Results show that our lemmatizer performs impressively as it reaches 70.32%, given the morphological complexities of the Setswana language.

From a larger perspective, our work demonstrates that Setswana inflectional words can be lemmatized based on the context of how they are used. The machine-learning approach is more suited in context understanding, better than any set of expert rules. However, the study is not without limitations. One of the significant limitation was the dataset. The cases of polysemous words (i.e. one word with multiple meaning) in the dataset were very low. This resulted in the model having few examples to learn from. In the light of this finding, we therefore, need to develop dataset mostly targeted to such phenomena in order to have a more representative dataset in the future.

The other drawback is that the model performs well in lemmatizing words that appears more frequently in the dataset and has a high error rate on lemmatizing less common words due to less insufficient evidence to effectively learn the weights.

Moreover, the work done in this project provides a basis for future research in several areas. With the present approach we further intend to improve our work on the following future aspect:-

- Enrichment, modification, and expansion of the dataset to come up with a complete Setswana corpus that can be used to develop natural language models specifically machine-learning models.
- Incorporate the Smoothing technique in our lemmatization model. This technique will assist in addressing scenarios related to determining the likelihood estimates of out of vocabulary words.
- We also intend to experiment with different n-gram ranges. In this study the experiments were based only on n-gram size 2, i.e. bigrams.
- It could be interesting to explore the applicability of other types of machine-learning algorithms other than the Naive Bayes to see their strength and performance in addressing

Setswana lemmatization.

- One long term goal is to have a complete state of the art Setswana lemmatizer that can be used as a plug-in in other systems.

Summing up, it is clear that Setswana is nascent in the field of NLP. To our best knowledge, this research represents the first step towards context-aware lemmatization of the Setswana language, more profound research in lemmatization of Setswana language is needed. The results show a promising performance. However, there are still many open research questions such as: what if we explore other machine-learning algorithms e.g. decision trees, artificial neural networks, support vector machines, etc., to lemmatize Setswana? Such questions are hoped to shape future research direction for others interested in advancing Setswana in the field of NLP.



References

- [1] Hendrik J. Groenewald. Using Technology Transfer to Advance Automatic Lemmatization for Setswana. *Proceedings of the EACL 2009 Workshop on Language Technologies for African Languages – AfLaT 2009, pages 32–37, Athens, Greece, 31 March 2009.*, (March):32–37, 2009.
- [2] Los Angeles. The First Workshop on Statistical Parsing of Morphologically Rich Languages Proceedings of the Workshop. (Spmr1), 2010.
- [3] Vukosi Marivate, Tshephisho Sefara, Vongani Chabalala, Kemogetswe Makhaya, Tumisho Mokgonyane, Rethabile Mokoena, and Abiodun Modupe. and classification : Setswana and Sepedi. 2019.
- [4] J C Brits, R S Pretorius, and Gerhard B Van Huyssteen. Automatic Lemmatization in Setswana: Towards a Prototype, 2006.
- [5] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.
- [6] Roshan Adusumill and I. NLP in the Stock Market.
- [7] Bart Jongejan and Hercules Dalianis. Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike. *ACL-2009, Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, (August):145–153, 2009.
- [8] Simple Data-driven Context-sensitive Lemmatization. Simple Data-Driven Context-Sensitive Lemmatization .
- [9] Michal Toman, Roman Tesar, and Karel Jezek. Influence of word normalization on text classification. *Proceedings of InSciT*, pages 354–358, 2006.
- [10] Abhisek Chakrabarty, Onkar Arun Pandit, Utpal Garain, Computer Vision, and Pattern Recognition Unit. Context Sensitive Lemmatization Using Two Successive Bidirectional Gated Recurrent Networks. pages 1481–1491, 2017.
- [11] Vimala Balakrishnan and Ethel Lloyd-yemoh. Stemming and Lemmatization : A Comparison of Retrieval Performances. 2(3):262–267, 2014.
- [12] H Dejean, E Gaussier, C Goutte, and K Yamada. Reducing parameter space for word alignment. *Proceedings of the HLT-NAACL 2003 Workshop on Building and using parallel texts: data driven machine translation and beyond- Volume 3*, 3(June):23–26, 2003.

- [13] G A Malema, N P Motlogelwa, and M Lefoane. A Rule-Based Setswana Verb Lemmatizer. pages 38–45, 2015.
- [14] Gabofetswe Malema. Setswana Verb Analyzer and Generator. (7):1–11.
- [15] G Malema, Moffat Motlhanka, and Boago Okgetheng. Setswana Noun Analyzer and Generator Setswana Noun Analyzer and Generator. (July), 2018.
- [16] Prakash M. Nadkarni, Lucila Ohno-Machado, and Wendy W. Chapman. Natural language processing: An introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.
- [17] Walter Daelemans. Evaluation of Machine Learning Methods for Natural Language Processing Tasks. pages 755–760, 1999.
- [18] Wahab Khan, Ali Daud, and Tehmina Amjad. A survey on machine learning models for Natural Language Processing (NLP) A survey on the state-of-the-art machine learning models in the context of NLP. (October), 2016.
- [19] Prakash M. Nadkarni, Lucila Ohno-Machado, and Wendy W. Chapman. Natural language processing: An introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.
- [20] Paul Jasmin Rani, Jason Bakthakumar, Praveen Kumar B, Praveen Kumar U, and Santhosh Kumar. VOICE CONTROLLED HOME AUTOMATION (NLP) AND INTERNET OF THINGS (IoT). 2017.
- [21] Mamillapally Raghavender Sharma. A Short Communication on Computer Programming Languages in Modern Era. (September), 2020.
- [22] Roberto Navigli and Viale Regina Elena. Natural Language Understanding : Instructions for (Present and Future) Use. pages 5697–5702, 2003.
- [23] Judith Hurwitz, Daniel Kirsch, and John Wiley. *Machine Learning Machine Learning For Dummies*. Number june. 2018.
- [24] Jason Weston and Michael Karlen. Natural Language Processing (Almost) from Scratch. 12:2493–2537, 2011.
- [25] Diksha Khurana, Aditya Koli, Kiran Khatter, Sukhdev Singh, and Manav Rachna. Natural Language Processing : State of The Art , Current Trends and Challenges Department of Computer Science and Engineering Accendere Knowledge Management Services Pvt . Ltd ., India Abstract. (Figure 1).
- [26] Hrithik Sanyal, Rajneesh Agrawal, and Comp-tel Consultancy. STUDY OF PARSING TECHNIQUES FOR NATURAL LANGUAGE PROCESSING – A COMPARISON. 14(1), 2020.

- [27] Xiaoyi Zheng. Morphology of Chaucerian English: With reference to the portrait of the prioress. *2011 IEEE 3rd International Conference on Communication Software and Networks, ICCSN 2011*, pages 207–212, 2011.
- [28] Sai Kiranmai Gorla, Sriharshitha Velivelli, N. L. Bhanu Murthy, and Aruna Malapati. Named entity recognition for telugu news articles using Naïve bayes classifier. *CEUR Workshop Proceedings*, 2079:33–38, 2018.
- [29] Farayi Kambarami, Scott Mclachlan, Queen Mary, Bojan Bozic, and Kudakwashe Dube. Computational Modeling of Agglutinative Languages : The Challenge for Southern Bantu Languages , Arusha Working Papers in African Linguistics. (February), 2021.
- [30] Muhammed H. Muhammed, Bassim M. Salih, and Omer K. Jasim. An emerging standard miniaturization in Arabic morphological analysis. *Proceedings - 2018 1st Annual International Conference on Information and Sciences, AiCIS 2018*, pages 112–116, 2019.
- [31] Niladri Sekhar Dash. Context and Contextual Word Meaning. *SKSE Journal of Theoretical Linguistic*, pages 21–31, 2008.
- [32] Mdp Requejo. the Role of Context in Word Meaning Construction: a Case Study. *International Journal of English Studies*, 7(1):169–173, 2012.
- [33] Jamin Carson. A Problem With Problem Solving : Teaching Thinking Without Teaching Knowledge. 17(2):7–14, 2007.
- [34] Denis Creissels. Grammaticalization in Tswana. (August):1–30, 2017.
- [35] L Pretorius and B Viljoen. Towards a computational morphological analysis of Setswana compounds. *Literator*, 1(April 2008):1–20, 2008.
- [36] Naledi Kgolo and Sonja Eisenbeiss. The role of morphological structure in the processing of complex forms : evidence from Setswana deverbative nouns. *Language, Cognition and Neuroscience*, 0(0):1–18, 2015.
- [37] Moromi Gogoi and Shikhar Kumar Sarma. Document Classification of Assamese Text Using Naïve Bayes Approach. *International Journal of Computer Trends and Technology*, 30(4):182–186, 2016.
- [38] S O N A Taheri. Learning the naive bayes classifier with optimization models. 23(4):787–795, 2013.
- [39] Sebastian Raschka. Introduction and Theory. pages 1–20, 2014.
- [40] Naive Bayes and Sentiment Classification. Naive Bayes and Sentiment Classification. 2018.
- [41] Amey K Shet Tilve. TEXT CLASSIFICATION USING NAÏVE BAYES , VSM AND POS TAGGER. 4(1), 2017.
- [42] Bo Tang, Student Member, Steven Kay, Haibo He, and Senior Member. Toward Optimal Feature Selection in Naive Bayes for Text Categorization. pages 1–14.

- [43] Computer Science. Part of Speech Tagging with Naïve Bayes Methods. pages 446–451, 2014.
- [44] Rund Mahafdah, Nazlia Omar, and Omaia Al-omari. ARABIC PART OF SPEECH TAGGING USING K-NEAREST NEIGHBOUR AND NAIVE BAYES CLASSIFIERS COMBINATION. 10(9):1865–1873, 2014.
- [45] Yishan Gong and Qiang Chen. Research of spam filtering based on Bayesian algorithm. *ICCAISM 2010 - 2010 International Conference on Computer Application and System Modeling, Proceedings*, 4(Iccasm):V4–678–V4–680, 2010.
- [46] Saiful Islam, Shah Mostafa Khaled, Khalid Farhan, Abdur Rahman, Joy Rahman, and A Naïve Bayesian Classifier. Modeling Spammer Behavior : Naïve Bayes vs . Artificial Neural Networks. pages 52–55, 2009.
- [47] Aniello Minutolo, Massimo Esposito, and Giuseppe De Pietro. Optimization of rule-based systems in mHealth applications. *Engineering Applications of Artificial Intelligence*, 59(July 2016):103–121, 2017.
- [48] Han Liu, Alexander Gegov, and Mihaela Cocea. *Rule Based Systems for Big Data*, volume 13. 2016.
- [49] Parisa Kordjamshidi, Dan Roth, and Kristian Kersting. Systems AI: A declarative learning based programming perspective. *IJCAI International Joint Conference on Artificial Intelligence*, 2018-July:5464–5471, 2018.
- [50] Marcel Bollmann, Petran F., and S. Dipper. Rule-Based Normalization of Historical Texts. *Proceedings of the International Workshop on Language Technologies for Digital Humanities and Cultural Heritage*, (September):34–42, 2011.
- [51] Shai Ben-David and Shai Shalev-Shwartz. *Understanding Machine Learning: From Theory to Algorithms*. 2014.
- [52] Mohssen M Z E Mohammed, Eihab Bashier, and Mohammed Bashier. *Algorithms and Applications*. Number July. 2016.
- [53] Jean-philippe Vert. Machine Learning in Computational Biology (the frequentist approach). *Encyclopedia of Database Systems*, (Gm 066387), 2008.
- [54] Marinka Zitnik, Francis Nguyen, Bo Wang, Jure Leskovec, Anna Goldenberg, and Michael M. Hoffman. Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. *Information Fusion*, 50(September 2018):71–91, 2019.
- [55] Dalya Baron. Machine Learning in Astronomy: a practical overview. 2019.
- [56] Nicholas M. Ball and Robert J. Brunner. Data mining and machine learning in astronomy. *International Journal of Modern Physics D*, 19(7):1049–1106, 2010.

- [57] George Tzanis, Ioannis Katakis, Ioannis Partalas, and Ioannis Vlahavas. Modern Applications of Machine Learning. *The 1st Annual SEERC Doctoral Student Conference*, pages 1–10, 2006.
- [58] Abhisek Chakrabarty, Akshay Chaturvedi, and Utpal Garain. A Neural Lemmatizer for Bengali. (2013):2558–2561, 2014.
- [59] Iskander Akhmetov, Alexander Krassovitskiy, Irina Maratovna Ualiyeva, Computational Technologies, Rustam Mussabayev, and Computational Technologies. An Open-Source Lemmatizer for Russian Language based on Tree Regression Models An Open-Source Lemmatizer for Russian Language based on Tree Regression Models. (October), 2020.
- [60] Iskander Akhmetov, Alexandr Pak, Irina Ualiyeva, and Alexander Gelbukh. Highly Language-Independent Word Lemmatization Using a Machine-Learning Highly Language-Independent Word Lemmatization Using a Machine-Learning Classifier. (October), 2020.
- [61] Jenna Kanerva, Filip Ginter, and Tapio Salakoski. Universal Lemmatizer: A sequence-to-sequence model for lemmatizing Universal Dependencies treebanks. *Natural Language Engineering*, (May):1–30, 2020.
- [62] Vaishali Gupta, Nisheeth Joshi, and Banasthali Vidyapith. Design & Development of a Rule Based Urdu Lemmatizer Design & Development of a Rule Based Urdu Lemmatizer. (August), 2015.
- [63] R J Prathibha. Design of Rule based Lemmatizer for Kannada Inflectional Words. 2015.
- [64] Joël Plisson, Nada Lavrac, and Dr. Dunja Mladenić. A rule based approach to word lemmatization. *Proceedings of the 7th International Multiconference Information Society (IS'04)*, (November):83–86, 2004.
- [65] Derwin Suhartono, David Christiandy, and Rolando Rolando. Lemmatization Technique in Bahasa: Indonesian Language. *Journal of Software*, 9(5):1202–1209, 2014.
- [66] Snigdha Paul, Nisheeth Joshi, and Iti Mathur. Development of a Hindi Lemmatizer. (May 2013), 2015.
- [67] Bertus van Rooy and Rigardt Pretorius. A word-class tagset for Setswana. *Southern African Linguistics and Applied Language Studies*, 21(4):203–222, 2003.
- [68] George A. Miller and Florentina Hristea. WordNet nouns: Classes and instances. *Computational Linguistics*, 32(1):1–3, 2006.
- [69] Adam Kilgarrieff and Christiane Fellbaum. WordNet: An Electronic Lexical Database. *Language*, 76(3):706, 2006.
- [70] Seid Muhie Yimam, Iryna Gurevych, Richard Eckart, and De Castilho Chris. WebAnno : A Flexible , Web-based and Visually Supported System for Distributed Annotations. 1(1):2–7.

- [71] Seid Muhie and Chris Biemann. Automatic Annotation Suggestions and Custom Annotation Layers in Automatic Annotation Suggestions and Custom Annotation Layers in WebAnno. (January 2014), 2016.
- [72] Seid Muhie Yimam, Chris Biemann, Ljiljana Majnarić, Šefket Šabanović, and Andreas Holzinger. An adaptive annotation approach for biomedical entity and relation recognition. *Brain Informatics*, 3(3):157–168, 2016.
- [73] Richard Eckart, De Castilho Chris, Biemann Iryna, and Seid Muhie Yimam. WebAnno : a flexible , web-based annotation tool for CLARIN. 2(1):3–5, 2014.
- [74] The Webanno Team. WebAnno User Guide.
- [75] Morulaganyi Kgasa and Joseph Tsonope. *Thanodi ya Setswana*. Longman Botswana, 1995.
- [76] Rund Mahafdah, Nazlia Omar, and Omaia Al-omari. ARABIC PART OF SPEECH TAGGING USING K-NEAREST NEIGHBOUR AND NAIVE BAYES CLASSIFIERS COMBINATION. 10(9):1865–1873, 2014.
- [77] Michal Konkol and Miloslav Konopík. Named entity recognition for highly inflectional languages: Effects of various lemmatization and stemming approaches. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8655 LNAI(December 2014):267–274, 2014.
- [78] Yuji Roh, Geon Heo, Steven Euijong Whang, and Senior Member. A Survey on Data Collection for Machine Learning. pages 1–20.
- [79] Muhammad Firman, Aji Saputra, Triyanna Widiyaningtyas, and Aji Prasetya Wibawa. Illiteracy Classification Using K Means-Naïve Bayes Algorithm. 2:153–158.
- [80] Steven Bird and Edward Loper. NLTK : The natural language toolkit NLTK : The Natural Language Toolkit. (March), 2014.
- [81] Abinash Tripathy, Ankit Agrawal, and Santanu Kumar Rath. Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications*, 57:117–126, 2016.
- [82] Myroslava O Dzikovska, Peter Bell, Amy Isard, and Johanna D Moore. Evaluating language understanding accuracy with respect to objective outcomes in a dialogue system. pages 471–481, 2012.
- [83] Mohammad Hossin and Nasir Sulaiman. A Review on Evaluation Metrics For Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):1–11, 2015.
- [84] Dkk Visa Sofia. Confusion Matrix-based Feature Selection Sofia Visa. *ConfusionMatrix-based Feature Selection Sofia*, 710(May 2014):8, 2011.
- [85] Steven T Piantadosi and Cognitive Sciences. *and future directions*, volume 21. 2015.