Pluralising Nouns in isiZulu and Related Languages

Joan Byamugisha[1], C. Maria Keet[1], Langa Khumalo[2]

[1] Department of Computer Science, University of Cape Town, South Africa,
{jbyamugisha,mkeet}@cs.uct.ac.za
[2] Linguistics Program, School of Arts, University of KwaZulu-Natal, South Africa,
Khumalol@ukzn.ac.za

Abstract. There are compelling reasons for a Controlled Natural Language of isiZulu in software applications, which requires pluralising nouns. Only 'canonical' singular/plural pairs exist, however, which are insufficient for computational use of isiZulu. Starting from these rules, we take an experimental approach as virtuous spiral to refine the rules by repeatedly testing two test sets against successive versions of refined rules for pluralisation. This resulted in the elucidation of additional pluralisation rules not included in typical isiZulu textbooks and grammar resources and motivated design choices for algorithm development. We assessed the potential for reuse of the approach and the type of deviations with Runyankore, which demonstrated encouraging results.

1 Introduction

Although the imperative for Human Language Technologies in isiZulu—and, in fact, most Bantu languages—exists, the language is still under-resourced, especially for computational information and knowledge processing [19]. While some results have been obtained in natural language understanding, such as morphological analysers, there are scant results for *generating* isiZulu [12, 11]. We take here one aspect of the generation that, perhaps, seems rather basic: pluralising a noun. Automation of pluralisation is useful for software interfaces that need a controlled natural language (CNL) [11], technology-assisted learning, and other uses. A (very) simple example is one that in the calendar or weather forecast it should be '1 day' and '2 days', not '1 days' or '2 day', and it has been noted as a requirement for collecting and displaying information for South Africa's National Indigenous Knowledge Management System [1].

There is no computational approach for pluralising isiZulu nouns yet and, to the best of our knowledge, not for any language in the Bantu language family (of which isiZulu is a member). Looking at better resourced languages, the bulk of automation of pluralisation has been carried out in the 1990s. For instance, Conway's English pluralisation algorithm [7] is based on extensive linguistic resources that easily could be used for specification of the rules computationally. Likewise, rules for German pluralisation and their occurrences are well-known and have been experimentally assessed thanks to accessible databases [15]. Both

rely on regular expressions of the ending of the nouns. NLP research for Arabic, a more complex language, took up in the 2000s and, regarding pluralisation, now focuses more on comparing techniques [3] and refinements [2], noting that there are several usable computational resources for Arabic already.

The state-of-the-art for nouns from the linguistics viewpoint covers some theory and experiments on morphological analysers without online resources [5, 18], and a basic POS tagger and corpus exist [20]. Some recent advances in linguistics describe what goes in which noun class (NC) with which prefix [21, 17, 16], and all textbook resources have the 'standard table' of prefixes, as included in Table 1, but this does not state *when* or *why* which prefix is used in the plural, and whether this is the only thing to consider in pluralising nouns. Some of its limitations are known with anecdotical evidence: i) there are variations for some NCs even in the standard table, but no indication when to use which one; ii) some are known to be phonologically conditioned; iii) some do not have plurals; and iv) for some nouns the categorisation into the NC is not well-established and thus may generate exceptions.

This raises the following three questions:

Q1: How well does the 'standard' table of prefixes work for pluralising nouns?
Q2: What are the exceptions, and are there any rules among the exceptions?
Q3: Do the types of exceptions appear in Bantu languages other than isiZulu?

One option to answer this is to try to avail of corpora; however, the only available corpus consists of the bible and a few fiction novels [20], and, in taking a corpus-based approach, one needs the rules we aim to find. Practically, this is a chicken-and-egg problem. We propose here to take a combined approach to answer the questions, using rules with experimental evaluation to iteratively improve the rules and results of the pluraliser. We achieved over 90% correct pluralisation on the test data eventually, identified types of exceptions, and elucidated eight pluralisation rules beyond the 'standard table'. Question 3 was evaluated with Runyankore, a Bantu language spoken in Uganda and orthographically similar to isiZulu, which achieved higher initial correctness with the first test set, but it also had similar types of exceptions, suggesting that a bootstrapping approach—reusing the approach and similar rules presented here—for orthographically similar Bantu languages would be feasible and reduce development time.

In the remainder of the paper we first provide the basic background to isiZulu (and, by extension, Bantu) nouns and verbalisation of structured knowledge in Section 2. Section 3 discusses design considerations for developing the pluraliser and Section 4 presents the main results of the pluraliser. We discuss in Section 5 and conclude in Section 6.

2 Background

It must be accentuated that isiZulu is a computationally under-resourced language. It is a Bantu language that belongs to the Nguni group of languages, which include isiXhosa, isiNdebele and siSwati. It is the most popular language

in South Africa spoken as a first (home) language by about 23% of South Africa's over 50 million people. Bantu languages have a characteristically agglutinating morphology, which makes it a challenge to develop computational technologies for them. One of the salient features of isiZulu, which is also true for other Bantu language like Runyankore, is the unique system of noun classes. This will be elaborated on in the next section, after which we introduce a motivating example use case.

2.1 IsiZulu nouns

Each noun is allocated a specific noun class. The canonical NC list is Table 1. The noun comprises of two formatives, the prefix and the stem. The prefix can be identified as a full prefix or an incomplete prefix. It is a full prefix when the augment (pre-prefix) is followed by a prefix proper. It is an incomplete prefix when it only has the augment. An example of a full prefix is *isihlalo* 'chair', *i-* (augment) *si-* (prefix proper) *-hlalo* (stem). An example of an incomplete prefix is *ubaba* 'father', *u-* (augment) *-baba* (stem). Because of the agglutinating nature of isiZulu coupled with a conjunctive writing system, which glues together elements of an isiZulu word, a number of NC prefixes in isiZulu are phonologically conditioned and yet others are homographs.

The NC has received considerable attention recently [17, 21, 22] in an effort to explicate the generation of, and semantic motivation for, the various NC assignments. Most NCs are paired such that there is a distinctive pattern of a singular form in one class and a plural form in another, and yet other classes are latent. It is notable in isiZulu that the NC prefix of class 1 and 3 is conditioned by the morphology of the stem it takes: *-mu-* before monosyllabic stems and *-m-* for other stems. Similarly, NC 5 and 11 are conditioned: the full prefixes only take monosyllabic stems [21], e.g., nc5 *ili-+-hlo* 'eye' nc11 *ulu-+-thi* 'stick'. Interestingly, the *n* of NCs 9 and 10 merges with the following consonant forming prenasalized consonants.

There are other known deviations with loanwords, vowel-commencing roots, and concordance in the second term of a compound noun, and anecdotal evidence suggests these are not catered for with the 'standard' table of singular-plural pairings. Pluralisation in Runyankore seems to have similar exceptions as isiZulu.

2.2 Use case: toward an isiZulu Controlled Natural Language

Controlled Natural Languages (CNLs) are a fragment of the full natural language that are used in applications with contextual text, such as automated generation of weather reports and medical apps, eLearning, and in prescription notes and instruction manuals. There are few results for an isiZulu CNL covering verbalisation patterns [12] and algorithms for subsumption, disjointness, conjunction, existential quantification and its negation [11]. This revealed that a template-based approach is not feasible due to isiZulu being a highly agglutinating language and the many NCs and the concordances it requires. To illustrate this, let us take the common Description Logic (DL) language $\mathcal{ALC}$ [4]

Table 1. Zulu noun classes, with the 'canonical' list of prefixes for isiZulu and Runyankore; NC: Noun class, AU: augment, PRE: prefix, n/a: class not used.

NC	isiZulu		Runyankore		NC	isiZulu		Runyankore	
	AU	PRE	AU	PRE		AU	PRE	AU	PRE
1	u-	m(u)-	o-	mu-	9	i(n)-	-	e-	n-, m-
2	a-	ba-	a	ba-	10	i-	zi(n)-	e-	n-
1a	u-	-	n/a		11	u-	(lu)-	o-	ru-
2a	o-	-	n/a		(10)	i-	zi(n)-	n/a	
3a	u-	-	n/a		12	n/a		a-	ka-
(2a)	o-	-	n/a		13	n/a		o-	tu-
3	u-	m(u)-	o-	mu-	14	u-	bu-	o-	bu-
4	i-	mi-	e-	mi-	15	u-	ku-	o-	ku-
5	i-	(li)-	e-	i-, ri-	6	n/a		a-	ma-
6	a-	ma-	a-	ma-	16	n/a		a-	ha-
7	i-	si-	e-	ki-	17	-	ku-	-	ku-
8	i-	zi-	e-	bi-	18	n/a		o-	mu-
9a	i-	-	n/a		20	n/a		o-	gu-
(6)	a-	ma-	n/a		21	n/a		a-	ga-

as knowledge representation language, which is becoming increasingly popular for CNLs [6], especially in its serialised form, OWL [14]. The straightforward 'all some' pattern ($C \sqsubseteq \exists R.D$ 'each C R at least one D') with for relationship R a present tense verb already generates interesting issues. A few corresponding pretty-printing examples are shown in Fig. 1. While the universal quantification

(1) `Grandmother` $\sqsubseteq \exists$`eats.Apple`
bonke ogogo badla i-aphula elilodwa
('All grandmothers eat at least one apple')

(2) `Nurse` $\sqsubseteq \exists$`eats.Apple`
bonke abongi badla i-aphula elilodwa
('All nurses eat at least one apple')

(3) `Leopard` $\sqsubseteq \exists$`eats.Apple`
zonke izingwe zidla i-aphula elilodwa
('All leopards eat at least one apple')

Fig. 1. Pretty-printing of the verbalisation of 'all x eat at least one apple', with the plural underlined; their respective singulars are *ugogo* 'grandmother' (nc1a), *umongi* 'nurse' (nc1), and *ingwe* 'leopard' (nc9).

$\forall$ (silent on the left-hand side of the subsumption "$\sqsubseteq$") is just "each" or "for all" in English, in isiZulu it depends on the NC of the first noun, resulting in *bonke*, *zonke* etc., and, moreover, this first noun has to be in the plural according to [12]. However, although such knowledge is normally represented in the singular and

the algorithms in [11] do not describe the pluralisation step, yet three patterns rely on pluralisation to generate a grammatically correct sentence (disjointness and existential quantification). The success of these algorithms hinge on being able to find the correct plural for the given singular.

3 Design considerations

For the design of the pluralisation rules and algorithms, there are two distinct approaches from a linguistic viewpoint, and several representation options for each. Concerning the language, one can

1) base it on a purely syntactic/orthographic analysis, relying on the (patterns of the) characters of the string of text;
2) base it on a semantic analysis using the meaning of the nouns and, implicitly with that, their respective NC.

The consequence for a computational approach using the first option is that it will require some regular expressions with as formal foundation a Finite State Automaton for the first part of the process, being to analyse the word, or more advanced FSMs that also will handle the change in prefix. On cursory glance, it would seem similar to a pluraliser for English that processes a noun's final character(s), as with the design and Perl implementation by [7], but then applied to the beginning character(s) instead. This is not exactly the case, as we shall see below. The second one entails more investment upfront through either adding some encoding of the meaning or, in lieu of that, that at least the NC of the noun is stored with the noun.

We illustrate consequences of these first, basic, design choices through an example with aiming to pluralise a noun in the singular in nc1 into a noun in the plural in nc2; the same considerations hold likewise for other clusters of NCs with the same prefixes (nc1a, nc2a, nc3a, and nc11; and nc5, nc9a, and nc9). To abstract away from actual code, we use an automata-based programming notation, which is like the usual automata as used for morphological analysers, but the transitions/steps are code sections, like a condition, function, or some other routine, rather than consuming characters of the word inputted to the FSM. Three options are shown in Fig. 2, where, e.g., the "$noun[0:1] == \texttt{um}$" is to be understood as 'if the first and second character of the variable *noun* equals the character string `um`' [then transition state from state p to state q] and "$\frac{\texttt{um}}{\texttt{aba}}$" 'replace the first characters `um` with `aba`'. Returning the plural is the same for all and omitted for being trivial at this stage for the current argument.

The first option is a pure string-based approach, and systematically going through the NC pairs of Table 1, as depicted in the top-most automaton: check whether the word starts with `umu` or `um`, and replace the 2- or 3-character prefix with the prefix for the plural, `aba`; e.g., *umuntu* (singular 'human') $\mapsto$ *abantu* (plural). When we arrive at setting up the conditions for nc3, however, the conditions are the same, but the 2- or 3-character prefix is to be replaced by `imi` instead (e.g., *umumba* $\mapsto$ *imimba* 'stuff box containers'). In addition, a mass noun like *ummbila* 'maize', in nc3, does not have a plural. Thus is, using a string-

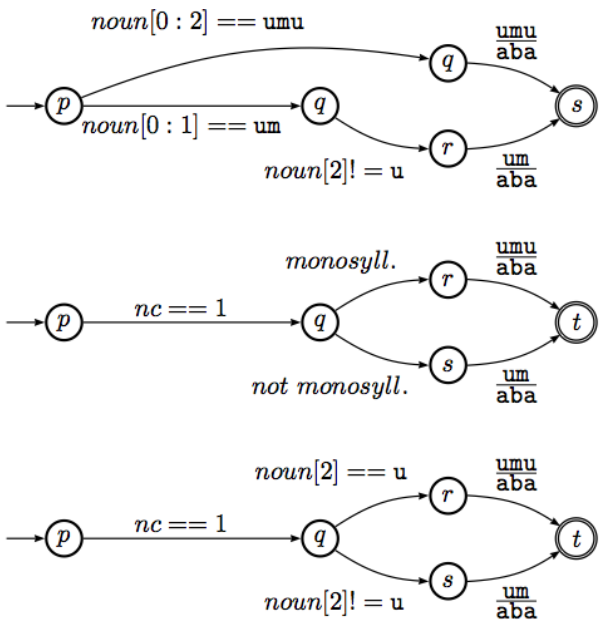

Fig. 2. Three variations in designing the algorithm illustrated for the case for noun classes 1 and 2, using an automata-based programming notation.

based approach only is expected to lead to a considerable error percentage in pluralisation.

To distinguish between when to swap `um` for `aba` and `umu` for `aba`, one either uses characters again, or checks whether the stem is monosyllabic. There are, however, no documented string-based rules for recognising monosyllabic stems, for it depends on the correct identification of the prefix, which is non-deterministic and thus always will have a certain error rate. That is, this is not an option at present, unless a list of monosyllabic stems is constructed upfront and consulted during pluralisation. The third strategy is to use both the NC and the prefix analysis, provided that it is annotated somewhere in the input (bottom figure in Fig. 2). This design in particular would cause a problem only if there were `u`-commencing stems in NC1. There are none, though there are a few that commence with an `a`, `e`, `i`, or `o`; e.g. *umongi* 'nurse'. This strategy, therefore, seems to have the most potential, hence probably results in the least amount of errors.

Combining the separate automata for nc1 and nc3 and with the additional rule for vowel-commencing stems (elaborated on Section 4), i.e., commencing with NCs, results in 15 states, as does the one that commences with the string processing and then NC checking; the latter is included in Fig. 3. For *umumba*, the states visited in the execution trace are: $\{q, x\} \rightarrow \{r, y\} \rightarrow \{a\} \rightarrow \{c\} \rightarrow \{d\}$.

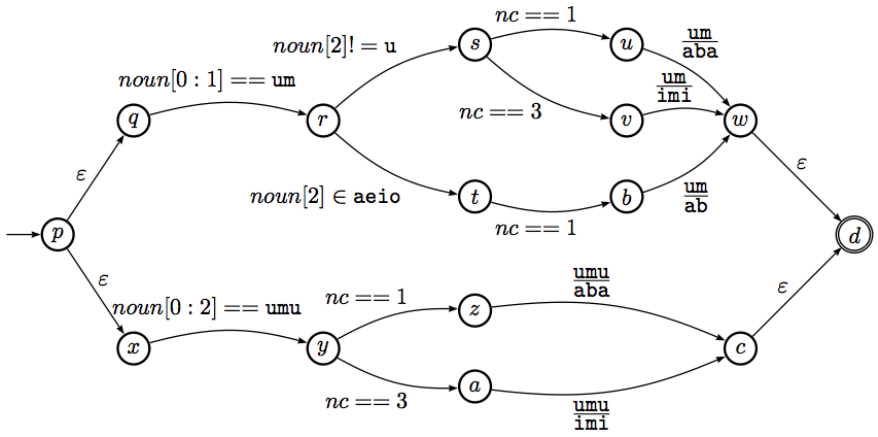

Fig. 3. The automata-based programming notation for words in nc1 and nc3 combined, using NC information of the noun.

4 Evaluation of the pluraliser

The aim of the evaluation is to examine the rules for pluralisation of isiZulu nouns, to iteratively improve on the encoded rules, and to consider an orthographically similar language, Runyankore, primarily used in Uganda. We first describe the materials & methods, followed by the results for the isiZulu pluraliser and then the results for Runyankore.

4.1 Materials and Methods

We encoded a basic pluraliser based on the standard isiZulu noun prefix table (Table 1), and tested it against the data set (see below) of whole words with(out) their respective NC. Informed by the errors, new rules were repeatedly added to the pluraliser and the set re-tested. The evaluation of the correctness of the computed plurals is carried out with an isiZulu speaker. Accuracy is calculated as $\frac{|\text{correct plural}|}{|\text{nouns}|} * 100$.

The materials consist of the program code with its related input files (in txt). Due to lack of shared resources in isiZulu, we manually compiled two wordlists. The first set of nouns, *Set1*, is a 'random' list of 101 words informed by content of multiple ontologies (african wildlife, building, tourism, wine, and other domain ontologies) and general domain words covering categories of entities as in the DOLCE foundational ontology [13]. This list was created in English so as to ensure that the words were not cherry-picked for their linguistic features in isiZulu, but only on domain coverage. This list was manually translated into isiZulu and the NC manually added, which appeared to cover all isiZulu NCs except 9a and the locative nc17. The second set of 117 words, *Set2*, was compiled by taking the first-listed noun on every left-hand page of the Shuter &

Shooter isiZulu Scholar's dictionary's isiZulu-English section [8]; hence, this set is an alphabetically balanced random sample, also with a near-full NC coverage (missing nouns in nc15 and nc17). For both sets, about 2/3 of the nouns are in either nc5, nc7, or nc9.

The same set of 101 English words of *Set1* were manually translated to Runyankore. 13 had no direct translations and four were added to cover NCs 20 and 21, as well as special orthographical cases; this resulted in a set of 92 words, *Set1r*. A second set was obtained from an attempt to extract every singular noun from the Runyankore dictionary [23]; 2542 words were extracted, *Set2r*, which represented all NCs except 20 and 21, mainly because they are considered derogatory and as such are common in speech but not writing.

The materials—code, output, and analysis—are online available at `http://www.meteck.org/files/geni/`.

4.2 Results

IsiZulu pluraliser The accuracy of the pluraliser versions is included in Table 2. The iterations could have been done in any order, and are listed in the order in which they have been carried out.

Table 2. Accuracy of the different versions of the pluraliser on the test sets.

Pluraliser version	**Set1**	**Set2**
Whole words	53	45
Words + nc	92	78
Words + nc + compounds	96	78
Words + nc + compounds + mass	99	87
Words + nc + compounds + mass + pl. exceptions	99	87
Words + nc + compounds + mass + pl. exceptions + prefix exceptions	100	91
Words + nc + compounds + mass + pl. exceptions + prefix exceptions + pl. only	100	92

Test "0": just words, and words with NC The main contributor to the errors is sameness in prefixes for some NCs, as described in Section 3, so it pluralises on the first if-statement that evaluates to true, rather than the NC later in the code. Adding the NC explicitly in the input had the greatest reduction in error rate for both sets, as can be seen in Table 2.

First iteration: addressing compound nouns Compound nouns in isiZulu typically have a nominal lexeme plus an adjective (e.g., *ukhilimu oyiqhwa* 'ice cream'). The initial morpheme of the adjective must be in agreement with the prefix of the first nominal lexeme, which means that the NC prefix must be in agreement with the possessive concord (PC) (e.g., *u-* must agree with *o-*). The first iteration included devising rules that correctly adds the applicable

Table 3. Additional pluralisation rules, denoted in easily readable pseudo-code (e.g. secondWordMinus1stLetter is in the code `second[1:]`).

No.	Rule
	Compound nouns
1.	**if** nc = 1 **and** 1stLetterOfTheSecondWord != vowel, **then** return 'b' + secondWordMinus1stLetter
2.	**if** nc = 3a, **then** return 'aba' + secondWordMinus1stLetter
3.	**if** (nc = 9 **or** nc = 7) **and** 1stLetterOfSecondWord != vowel, **then** return 'z' + secondWordMinus1stLetter
4.	**if** (nc = 9 **or** nc = 7) **and** 1stLetterOfSecondWord = vowel), **then** return secondWord1stLetter+'zi'+secondWordFrom4thLetterOnwards
	Mass nouns
5.	**if** 'm' in nc, **then** return word
	Prefix exceptions / vowel-commencing stems
6.	**if** nc = 1 **and** thirdLetterOfWord in 'aeio', **then** return 'ab' + wordMinusFirstTwoLetters
7.	**if** nc = 7 **and** thirdLetterOfWord in 'aeou', **then** return 'iz' + wordMinusFirstTwoLetters
	Noun in plural only
8.	**if** nc in '24682a' **or** nc = 10 , **then** return word

concord; e.g., *indawo yokubhukuda* (nc9) ↦ *izindawo zokubhukuda* 'swimming pools', where *in-* agrees with *yo-* (from the nc9 PC *ya-*+*u-* of *ukubhukuda* 'to swim'/'swimming') and *izin-* with *zo-* (from the nc10 PC *za-*+*u-*). There are some variations to this, such as the extra *e-* in *isilwane esifuyiweyo* (nc7) ↦ *izilwane ezifuyiweyo* 'pets', which is addressed by rules 1-4 in Table 3.

Second iteration: addressing the mass nouns Mass nouns refer to those entities that are not countable on their own but only in certain quantities—in ontology also called amount of matter or stuff—such as 'water' and 'wine'. The term does not change in isiZulu, like in other languages, and they can be in any NC; e.g., *amanzi* 'water' is in nc6 and *iwayini* 'wine' is in nc5. No known marker exist to determine a noun is a mass noun, and a cursory evaluation on types of mass nouns (such as pure vs mixed stuffs [10]) did not reveal a pattern either. To be able to handle it computationally, we therefore append the NC with an 'm' and add rule 5 (Table 3), i.e., the "word" that will be returned is the same as the original noun.

Third iteration: addressing exceptions to plural classes Exceptions to the plural classes given a NC for the singular do exist, but are very rare; e.g., *indoda* ↦ *amadoda* 'men'. They are true exceptions without rules and therefore put in a separate look-up file of exceptions for reusability.

Fourth iteration: addressing exceptions to the prefixes There are (at least) two types of exceptions to the 'standard' prefixes. One is due to the stem beginning

with a vowel rather than a consonant, such as *-akhiwo* as stem of *isakhiwo* ↦ *izakhiwo* 'buildings', and other phonologically conditioned plurals, such as *ucingo* ↦ *izingcingo* 'wires'/'telephones' (an additional *-g-*). The former is detectable by checking the third character of the noun for the vowel; e.g., it is *is-/iz-* rather than *isi-/izi-* for nc7, and *ab-* for nc2. This is captured in rules 6 and 7 (Table 3).

Fifth iteration: nouns in plural only This issue came afore only with *Set2*, although it essentially also existed with *Set1*. Some nouns in both sets exist in the plural only, but were also mass nouns (*imicikilisho* 'slow work' and *amanzi* 'water'), so were addressed by that rule already. However, *amanqamu* 'final act' is not a mass noun, yet also in a plural NC (nc6). To cover any such instances, rule 8 is added (see Table 3).

Remaining errors Generally, loanwords that have not been assimilated yet are assigned to nc5, such as *iradio* 'radio' and *i-okhestra* 'orchestra'. *Set2* contains the loanword *iaphula* 'apple', which also can be written as *i-aphula* so as to agree with the rule that isiZulu does not allow vowel sequencing. The latter writing would also simplify the rules for pluralisation, for then it is straightforwardly *i-aphula* ↦ *ama-aphula* rather than *iaphula* ↦ *ama-aphula*, and therewith increasing the accuracy of the pluraliser to 93%. The other errors have no obvious possible solution: the name of a disease remains in the singular, e.g., *isichenene* 'bilharzia', and when to use *izin-* or *izim-* and not *izi-* for nc9 and nc11, i.e., the plural class nc10. It has been suggested that *im-/izim-* applies to noun stems commencing with a labial sound (b, f, p, v) [9] (p43), which is yet to be evaluated.

4.3 Generalisability: Runyankore

The pluraliser had 88% accuracy on *Set1r* without making the NC explicit. This is because classes 1, 3, and 18, all have the prefix *omu* and nc15 and nc17 both have the prefix *oku*. Making the NC explicit increased the accuracy to 92%, with the remaining errors mainly resulting from loanwords and compound nouns.

With further clarification that loan words, such as *univasite*, *guriini*, and *kompuyuta*, belong to nc9/10, and the inclusion of this in the pluraliser, the accuracy further improved to 97%. The mass nouns in *Set1r* belong to the plural class nc6, so are not pluralised. Compound nouns in Runyankore have the genitive of the main noun associated with the second noun: for example *ekyokurya ky'enjangu* 'cat food'. This is pluralised by pluralising the main noun, getting the genitive of this plural, and associating that with the second noun. Thus, *ekyokurya* ↦ *ebyokurya*, its genitive *ebya* is obtained, and this drops its initial vowel, and if the second noun starts with a vowel too, the genitive's ending vowel is replaced with an apostrophe and associated with the second noun. This results in *ebyokurya by'enjangu*. This final step resulted in 100% accuracy.

When this pluraliser was used on the larger *Set2r*, there was an initial accuracy of 67%. Two reasons for this have so far been identified. First, according to the NC table, nc12 should be pluralised as nc13. However, among Runyankore speakers, nc12 is instead pluralised as nc14. This also seems to be in agreement

with the Runyankore dictionary, and when applied improved the accuracy to 71%. Second, there are two types of exceptions: (1) those words which do not get pluralised, even if they belong to a singular NC; such words include *eihangwe* 'day light', *eiriho* 'thirst', *orunyankore* (the language Runyankore), *orwakabiri* 'Tuesday', *omururu* 'greed', etc.; and (2) those words whose pluralisation does not follow the standard NC table such as *eka*, which should be in nc9/10 but is pluralised as *amaka* (nc6), and *orutunguru* (nc11) but is pluralized as *obutunguru* (nc14 instead of nc10). When these nouns were placed in a separate look-up file of exceptions, as was proposed for isiZulu, the accuracy improved to 74%.

Finally, some words for different nouns are written the same but can be differentiated when spoken; e.g., *omubazi* in writing can be either 'medicine' (nc3) or 'accountant' (nc1), but their respective plurals are *emibazi* 'medicines' and *ababazi* 'accountants'. These issues appear in isiZulu as well, because the tone is not indicated in the orthography (e.g., *umfundisi* (nc1/- or nc1/2)), but such instances were coincidentally not in the set of test words.

5 Discussion

For both isiZulu and Runyankore, adding the NC was useful, though for isiZulu that effect was more pronounced due to more sameness of prefixes compared to the Runyankore prefixes. Thus, the results show that pluraliser algorithms for Bantu languages, with their emblematic noun class system, will need to take a mixed syntax (word string combinations) and semantics (word meaning; NC at least) approach in order to achieve acceptable results in automated pluralisation. Answering research question Q1 stated in Section 1, it depended on the test set how well the 'standard' table works for pluralising nouns (78% and 92% accuracy without the extra rules), thereby deserving further investigation to discover and record additional pluralisation rules. The canonical structures of the nominal class system and the agreement morphology as well as those underlying patterns of the additional rules for isiZulu and Runyankore that were described are remarkably similar, making a compelling case for a bootstrapping approach, answering question Q3 in the affirmative.

The main outcome from a linguistic viewpoint, is that the 'standard' prefix table for each NC is definitely *not* comprehensive for isiZulu plurals. While we knew upfront that just words would return a substantial number of errors, the other assumption was that using the NC as well would be most effective and also suffice for all cases but a few exceptions to the rule. There are more 'exceptions' than anticipated, both true exceptions and regular ones. The combined rules-based and (limited) experiment with manual word analysis made it possible to make explicit, and thus start documenting, those rules that apply in pluralisation of isiZulu nouns beyond the 'standard' ones, contributing to answering question Q2. They may not yet cover all regular exceptions, but is expected to suffice within the CNL use case.

While this set did not show any issues with phonologically conditioning other than concerning vowel-commencing stems, they do exist (e.g., *ucingo* $\mapsto$ *iz-*

ingcingo). It is not known how they are conditioned, so this is a point of further investigation for linguists.

6 Conclusions

Automation and experimentally evaluating pluralising nouns from their singular to their respective plural showed that including the noun class of the noun is essential to obtaining reasonable success rates for isiZulu. This shows the need for a combination of syntax and semantics in the pluralisation algorithm. Resolving the remaining issues revealed several regular exceptions, which resulted in new pluralisation rules. They concern compound nouns, mass nouns, exceptions to the prefix due to the stem commencing with a vowel, and true exceptions. Some of them were also compared with another Bantu language, Runyankore, suggesting a generalisability of the types of exceptions across the Bantu language family.

Acknowledgements This work is based on the research supported in part by the National Research Foundation of South Africa (CMK: Grant Number 93397), and by the Hasso Plattner Institute (HPI) Research School in CS4A at the University of Cape Town (JB).

References

1. Alberts, R., Fogwill, T., Keet, C.M.: Several required OWL features for indigenous knowledge management systems. In: Proc. of OWLED'12. CEUR-WS, vol. 849 (2012), 27-28 May, Crete, Greece
2. Alkuhlani, S., Habash, N.: Identifying broken plurals, irregular gender, and rationality in arabic text. In: Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics. pp. 675–685. ACL (2012), 23-27 April, 2012, Avignon, France.
3. Altantawy, M., Habash, N., Rambow, O., Saleh, I.: Morphological analysis and generation of arabic nouns: A morphemic functional approach. In: Chair), N.C.C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., Tapias, D. (eds.) Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10). European Language Resources Association (ELRA), Valletta, Malta (May 2010)
4. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logics Handbook – Theory and Applications. Cambridge University Press, 2 edn. (2008)
5. Baumann, P., Pierrehumbert, J.: Using resource-rich languages to improve morphological analysis of under-resourced languages. In: Maegaard, B., Mariani, J., Moreno, A., Odijk, J., Piperidis, S. (eds.) In Proceedings of 9th International Conference on Language Resources and Evaluation (LREC'14). European Language Resources Association (ELRA) (2014), reykjavik, Iceland, May 26-31, 2014
6. Bouayad-Agha, N., Casamayor, G., Wanner, L.: Natural language generation in the context of the semantic web. Semantic Web Journal 5(6), 493–513 (2014)
7. Conway, D.M.: An algorithmic approach to English pluralization. In: Salzenberg, C. (ed.) Proceedings of the Second Annual Perl Conference. O'Reilly (1998), san Jose, USA, 17-20 August, 1998

8. Dent, G.R., Nyembezi, C.L.S.: Scholar's Zulu Dictionary. Shuter & Shooter Publishers, 4 edn. (2009)
9. Groenewald, H.C., Turner, N.S.: Siyasiqonda! Basic Zulu for the Health Sciences, University of KwaZulu-Natal, South Africa. 144p.
10. Keet, C.M.: A core ontology of macroscopic stuff. In: Janowicz, K., Schlobach, S. (eds.) 19th International Conference on Knowledge Engineering and Knowledge Management (EKAW'14). LNAI, vol. 8876, pp. 209–224. Springer (2014), 24-28 Nov, 2014, Linkoping, Sweden
11. Keet, C.M., Khumalo, L.: Basics for a grammar engine to verbalize logical theories in isiZulu. In: Bikakis, A., et al. (eds.) Proceedings of the 8th International Web Rule Symposium (RuleML'14). LNCS, vol. 8620, pp. 216–225. Springer (2014), august 18-20, 2014, Prague, Czech Republic
12. Keet, C.M., Khumalo, L.: Toward verbalizing logical theories in isiZulu. In: Davis, B., Kuhn, T., Kaljurand, K. (eds.) Proceedings of the 4th Workshop on Controlled Natural Language (CNL'14). LNAI, vol. 8625, pp. 78–89. Springer (2014), 20-22 August 2014, Galway, Ireland
13. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A.: Ontology library. WonderWeb Deliverable D18 (ver. 1.0, 31-12-2003). (2003), http://wonderweb.semanticweb.org
14. Motik, B., Patel-Schneider, P.F., Parsia, B.: OWL 2 web ontology language structural specification and functional-style syntax. W3c recommendation, W3C (27 Oct 2009), http://www.w3.org/TR/owl2-syntax/
15. Nakisa, R.C., Hahn, U.: Where defaults don't help: the case of the German plural system. In: Proceedings of the 18th Annual Conference of the Cognitive Science Society. pp. 177–182 (1996), San Diego, USA, July 12-15, 1996
16. Ngcobo, M.: Loan words classification in isiZulu: The need for a sociolinguistic approach. Language Matters: Studies in the Languages of Africa 44(1), 21–38 (2013)
17. Ngcobo, M.N.: Zulu noun classes revisited: A spoken corpus-based approach. South African Journal of African Languages 1, 11–21 (2010)
18. Pretorius, L., Bosch, S.E.: Computational aids for zulu natural language processing. Southern African Linguistics and Applied Language Studies 21(4), 267–282 (2003)
19. Sharma Grover, A., Van Huyssteen, G., Pretorius, M.: The South African human language technology audit. Language Resources & Evaluation 45, 271–288 (2011)
20. Spiegler, S., van der Spuy, A., Flach, P.A.: Ukwabelana – an open-source morphological Zulu corpus. In: Proc. of COLING'10. pp. 1020–1028. ACL (2010)
21. van der Spuy, A.: Zulu noun affixes: a generative account. South African Journal of African Languages 29(2), 195–215 (2009)
22. van der Spuy, A.: Generating zulu noun class morphology. Language Matters 41(2), 294–314 (2010)
23. Taylor, C.: A Simplified Runyankore-Rukiga-English Dictionary. Fountain Publishers, Kampala, Uganda (2009)