Bootstrapping a Runyankore CNL from an isiZulu CNL

Joan Byamugisha, C. Maria Keet, Brian DeRenzi

Department of Computer Science, University of Cape Town, South Africa,
{jbyamugisha,mkeet,bderenzi}@cs.uct.ac.za

Abstract. Runyankore is one of the top five languages spoken in Uganda. It is a Bantu language, thus it possesses the characteristic agglutinative structure, which is known to be challenging for the development of computational resources. It is also computationally under-resourced, which compounds the problem further. Given the recent progress in the verbalization (the process of writing the semantics expressed in axioms as a CNL) of most constructors in the Description Logic $\mathcal{ALC}$ in isiZulu, we take a bootstrapping approach to the verbalization of similar constructors in Runyankore. The key variables affecting verbalization in isiZulu indeed also hold for Runyankore, allowing us to build on existing background theory. We present verbalization patterns for most $\mathcal{ALC}$ constructors, also covering the 'hasX' role naming. An evaluation of the generated text with 18 non-linguists found a clear preference for verbalization in the singular for subsumption (as with isiZulu), existential quantification, and negation in the context of subsumption; but the plural form of verb negation.

1 Introduction

Runyankore is one of the top five most widely spoken languages in Uganda [2, 18, 20]. Wider internet access in Africa has led to the expansion of technology localization for indigenous languages. Large technology companies, such as Google and Mozilla, have made efforts to provide localized versions of their software. For Uganda, the Google search engine is available in Ekinyarwanda, Kiswahili, Luganda, Luo, and Runyakitara.

For any language, the development of computational components is resource-intensive [4]. However, when dealing with a very under-resourced language like Runyankore, it is justifiable to tailor what has already been done in a similar language, as a means of reducing development time and effort [4, 7]. One way of doing this is through the bootstrapping approach. Specifically, and similarly to [4], this means using the existing isiZulu verbalization patterns to generate sentences from axioms in a Description Logic (DL) as the starting point from which Runyankore ones can be tailored.

Bootstrapping has been applied to develop CNLs using templates [11] and the Grammatical Framework (GF) [1]. However, the template approach is inapplicable to languages with an agglutinating morphology [13, 14], and Runyankore is too under-resourced for GF, which requires a large resource grammar library; hence the need for bootstrapping for CNLs of agglutinated languages based on a grammar engine, which has not yet been done. The benefits of the bootstrapping approach for the development of language resources has been documented [4, 7, 12], most important of which is a reduction in development time and effort without sacrificing accuracy [4, 7].

We therefore seek to find out whether: (1) verbalization in Runyankore is affected by the same variables as isiZulu, namely the noun class of the name of the concept, the category of the concept, whether the concept is atomic or an expression, the quantifier used in the axiom, and the position of the concept in the axiom; and if so, (2) the existing isiZulu verbalization patterns can be tailored to Runyankore grammar to generate correct Runyankore text.

We define a Controlled Natural Language (CNL) as a constructed language based on a particular natural language, with a restricted lexicon, syntax, and semantics, but which still maintains its natural properties [15], and verbalization as the process of writing the semantics expressed in axioms as a CNL [11]. As a substantial amount of rules are necessary for the Runyankore CNL, it effectively blurs the line with NLG in the back-end. The Attributive Concept Language with Complements ($\mathcal{ALC}$) was used because of its expressiveness [3]. Here, we present how Runyankore verbalizations of subsumption ('is a'), conjunction ('and'), negation ('not'), existential quantification ('at least one'), and universal quantification ('for all'/'each') were tailored from isiZulu. The evaluation with 18 non-linguists showed preference for the singular form in the verbalisation for most constructors.

The paper is structured as follows: Section 2 introduces the basics of Runyankore; Section 3 presents related work in bootstrapping for language resource development; Section 4 illustrates how Runyankore verbalization patterns have been tailored from those of isiZulu; the evaluation is presented in Section 5; Section 6 discusses the implications of this work; and we conclude in Section 7.

2 Basics of Runyankore

Runyankore is a Bantu language spoken in the south-western part of Uganda by over two million people and is one of the top five most widely spoken languages in Uganda [2, 18, 20]. Runyankore, like other Bantu languages, is a highly agglutinative language [2, 18]—a word can be composed of over five constituents [6]. As the following example demonstrates, word formation involves the addition of affixes to a base word, where each affix carries meaning such as tense and aspect [18] (AU: augment, PRE: prefix, NC: noun class, CONT: continuous tense marker, SC: subject concord, FV: final vowel):

Abaana nibazaana	A-ba-aana ni-ba-zaan-a
'The children are playing'	AU-PRE$_{NC2}$-child CONT-SC$_{NC2}$-play-FV

There are several similarities between the structure of isiZulu and Runyankore, which make it seem feasible to tailor isiZulu verbalization patterns to Runyankore grammar. In both languages, the verbal morphology is very complex, with five different tenses in isiZulu [13] and fourteen in Runyankore [20].

Each noun is associated with one of several noun classes [14, 6]. The noun class determines the affixes of the nouns belonging to it, and subsequently agreement markers on the associated lexical categories such as adjectives and verbs [13, 6]. Noun class prefixes are coupled as singular/plural pairs [14]. Nouns comprise of two formatives, the prefix and the stem [14], where prefixes express number and are used to determine the class to which a particular noun belongs [14]. Table 1 shows the noun class system for isiZulu and Runyankore, using Meinhof's 1948 noun class system, which is a standard for defining noun classes among linguists, and thus facilitates cross-language comparisons and use. There are, however, several differences that make the direct reuse of the isiZulu verbalization patterns impossible (e.g., vocabulary and grammatical differences).

Table 1. IsiZulu and Runyankore noun classes, with their respective standard list of prefixes; NC: Noun class, AU: augment, PRE: prefix, n/a: class is 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-

3 Related Work

Davel and Barnard [7] applied the bootstrapping approach to extend a German pronunciation dictionary, using automatically extracted rules to generate additional word/pronunciation pairs, which were then used to extract better rules [7]. They managed to reduce development time to less than a quarter of that required for manual development and still maintain a high level of accuracy [7].

Bootstrapping has also been used in the development of CNLs [11, 1]. Jarrar et al. [11] used templates to develop multilingual verbalizations of logical theories. The initial verbalization template file would be tailored to a grammatically related language by varying the text in the text tags and their position, to reflect the language structure of the target language [11]. Through the bootstrapping approach, e.g., a German verbalization template was tailored from the Dutch template [11]. Angelov and Ranta [1] instead used GF to develop their CNL first in English, and this was then ported to Finish, French, German, Italian, and Swedish. Bootstrapping reduced development time from four days for English to a matter of hours for each of the languages [1].

For Bantu languages, however, the biggest drawback to developing computational language resources is their complex agglutinating morphology [4, 13, 14]. Bosch et al. [4] considerably reduced development time by applying an experimental bootstrapping approach in developing morphological analyzers for isiXhosa, Swati, and Ndebele based on the existing one for isiZulu [4]. Starting with the isiZulu morphological analyzer, they made the following language-specific modifications for each language: word roots lexicon, grammatical morpheme lexicon, as well as the language appropriate morphophonological rules [4]. The reduction in development time was from over 3,000 hours for the initial isiZulu morphological analyzer to a total of about 300 hours for all three new ones, with a further improvement using language-specific resources and rules from an average accuracy of 71.3% to 95.6% [4].

In summary, it has been shown that the bootstrapping approach has so far been used to develop a pronunciation dictionary, morphological analyzer, and generate language using templates and GF. As explained in Section 1, templates cannot be applied to an agglutinative language like Runyankore, and GF requires a wide coverage grammar specification which currently does not exist in Runyankore, and will require a lot of time and effort to develop.

4 Comparing Verbalization in isiZulu and Runyankore

In the bootstrapping approach, machine-learning analyses are corrected by a human "trainer" and the corrections are then further used to update the system's rules, and the cycle continues until the system achieves a satisfactory level of accuracy [12] (as was the case in [4, 7]). We apply the same concept but perform a manual analysis instead of machine learning. Instead of developing Runyankore verbalization patterns from scratch, isiZulu verbalization patterns are manually analyzed and tailored to the Runyankore grammar and lexicon [19].

Here we describe how this was done for subsumption ($\sqsubseteq$), conjunction ($\sqcap$), negation ($\neg$), existential quantification ($\exists$), and universal quantification ($\forall$), i.e., most of the basic Description Logic language $\mathcal{ALC}$ [3], which is a proper fragment of the OWL 2 DL ontology language that is a relatively popular and standardised input for CNLs and NLG [5, 17].

Runyankore verbalizations were derived by analyzing the similarities in factors. In both languages, the main variables that affect verbalization are the noun class of a concept's name, the concept name's category, whether the concept is atomic or an expression, the quantifier, and the position of the concept in the axiom [14]. The isiZulu verbalization patterns were then changed to reflect Runyankore grammar rules. Table 2 depicts the similarities and customizations. The enumerations in the isiZulu and Runyankore columns indicate that the use depends on the context, which may be the category or noun class it applies to, or other aspects in the axiom before or after the symbol [14].

Table 2. A few constructors, their typical verbalization in English, basic options in isiZulu and Runyankore; columns 1-3 were obtained from [14].

DL	English	Verbalization options isiZulu	Verbalization options Runyankore
$\sqsubseteq$	... is a ...	Depends on what is on the rhs of $\sqsubseteq$; requires either a semantic (living vs non-living thing) or syntactic (noun starts with either i or a, o, u) distinction	Depends on whether the noun on the RHS of $\sqsubseteq$ starts with a vowel
$\equiv$	1) ... is the same as ... 2) ... is equivalent to ...	I. Depends on what is on the rhs of $\equiv$: whether a person or not II. Depends on grammatical number on lhs of $\equiv$: whether singular or plural	I. Depends on what is on the LHS of $\equiv$ to obtain the subject prefix II. Depends on whether what is on the RHS of $\equiv$ starts with a vowel or not
$\sqcup$	... or ...	1) ... okanye ... 2) ... noma ...	... nainga ...
$\sqcap$	... and ...	Depends on whether $\sqcap$ is used to enumerate lists or connect clauses	Also depends on whether $\sqcap$ is used to enumerate lists or connect clauses
$\neg$	not ...	angi/akusiso/akusona/akubona /akulona/....	Depends on: I. both nouns, with the noun after $\neg$ dropping its initial vowel II. on the noun class of the concept in the relation
$\exists$	1) some ... 2) there exists ... 3) at least one ...	Depends on position in axiom: I. quantified over concept, depends on meaning of concept (living or non-living) II. includes relation (preposition issue omitted)	Depends on the noun class of the concept quantified over
$\forall$	1) for all ... 2) each ...	Depends on the semantic distinction of what is quantified over: I. (non)living thing II. noun class distinction	Depends on the noun class of the concept quantified over

We illustrate the grammar rules associated with the verbalization of the selected constructors using examples of verbalizations. Although these grammar rules are different for isiZulu and Runyankore, it is important to keep in mind that the resulting Runyankore verbalizations were obtained from analyzing how it was done in isiZulu. For some constructors, there were several possible alternative verbalizations. The ones presented below were selected as the best during the survey (described in Section 5).

Universal Quantification. In isiZulu, the 'all' or 'each' uses the same translation, *-onke* [13, 14]. Runyankore, however, has separate translations: *-ona* 'all' and *buri* 'each', with the latter only for nouns in the singular and the noun drops the initial vowel. In both isiZulu and Runyankore, the *-onke* and *-ona*, respectively, are prefixed with the appropriate prefixes of the noun class of the named concept (oral prefix for isiZulu [13] and genitive for Runyankore). However, in the isiZulu pattern, *-onke* is placed before the noun [13, 14], while Runyankore places *-ona* after the noun. The example illustrates verbalising $Girl \sqsubseteq ...$ with 'each' (1z, 1r) and 'for all' (2z, 2r), including vowel processing (e.g., *-a+o-* = *-oo-* in Runyankore):

1z:	isiZulu: <u>*Wonke*</u> *umfana ...*	(from *u-* + *-onke*)
1r:	Runyankore: <u>*Buri*</u> *mwishiki ...*	(always *buri*)
2z:	isiZulu: <u>*Bonke*</u> *abafana ...*	(from *ba-* + *-onke*)
2r:	Runyankore: *Abishiki* <u>*boona*</u>	(from *ba-* + *-ona*)

Simple Taxonomic Subsumption Verbalizations of $\sqsubseteq$ in Runyankore and isiZulu both depend on the first letter of the superclass. In isiZulu, the right copulative was selected based on the first letter of the noun of the superclass (*ng* for nouns starting with a-, o-, or u-, else *y*) [13]. In Runyankore, *ni* is used if the superclass starts with a consonant, and *n'* otherwise. For example, the verbalization of $Giraffe \sqsubseteq Animal$ ('each giraffe <u>is an</u> animal'):

isiZulu: *indlulamithi* <u>*y*</u>*isilwane*

Runyankore: *entwiga* <u>*n'*</u>*enyamishwa*

If the subsumption is followed by negation, then the verbalization changes for both isiZulu and Runyankore. In isiZulu, the verbalization for subsumption and negation are combined into one term and the copulative is omitted, regardless the quantifiers in the verbalization [13]. Runyankore simply replaces *ni* for *ti* ('is not'); the noun after $\neg$ drops its initial vowel if it has one. This is illustrated for $Cup \sqsubseteq \neg Glass$ ('each cup <u>is not a</u> glass'):

isiZulu: *zonke izindebe* <u>*aziyona*</u> *ingilazi* (preferred verbalisation [14])

Runyankore: *Ekikopo* <u>*ti*</u> *girasi*

Conjunction isiZulu verbalizes $\sqcap$ depending on whether 'and' is used for a list of things or to connect clauses; in the former case, *na* is used and *kanye* or *futhi* for the latter [13]. Runyankore follows a similar pattern by using *na* when $\sqcap$ is used to enumerate lists, and *kandi* when used to connect clauses. Runyankore makes a further distinction for lists and uses *na* only when $\sqcap$ is between nouns, but *kandi* otherwise. If one of the concepts is an adjective, then the noun class is required to obtain the adjective prefix in order to form the full translation of the adjective. Algorithm 4.1 illustrates the verbalization of $\sqcap$, where the first concept is a noun and the second is either a noun or adjective.

Existential Quantification In both isiZulu and Runyankore, the noun class is crucial to the verbalization: in obtaining the relative and quantitative concords in isiZulu, and the subject prefix in Runyankore. Runyankore verbalizes $\exists$ as *hakiri* for 'at least' and *-mwe* with subject prefix of the concept quantified over in order to form the full word for 'one'. The latter is similar to isiZulu's *-dwa*, which also relies on the noun class to get the correct concords [13].

Algorithm 4.1 Verbalization of Conjunction ($\sqcap$)

```
1: $A$ axiom; Variables: $a_1$, $a_2$, $p_1$, $p_2$, $c_1$, $sp$, $ap$, $a'$; and Functions: $getLHS(A)$,
   $getRHS(A)$, $getPOS(a)$, $getNC(a)$, $getPreviousNoun(A)$
2: $a_1 \leftarrow getLHS(A)$                    {get element to the left of $\sqcap$}
3: $a_2 \leftarrow getRHS(A)$                    {get element to the right of $\sqcap$}
4: $p_2 \leftarrow getPOS(a_2)$                  {get the part of speech for $a_2$}
5: **if** $p_2 = noun$ **then**
6:     Result $\leftarrow$ '$a_1$ na $a_2$'      {Verbalize with 'na', with vowel assimilation}
7: **else**
8:     $c_1 \leftarrow getNC(a_1)$
9:     $ap \leftarrow getAdjectivePrefix(a_1)$
10:    Result $\leftarrow$ '$a_1$ kandi $apa_2$'  {Verbalize with 'kandi'}
11: **end if**
12: **return** Result
```

isiZulu: *wonke uSolwazi ufundisa isifundo* <u>*esisodwa*</u> (but pl. preferred [14])
Runyankore: *Buri mwegyesa nayegyesa* <u>*hakiri*</u> *eishomo* <u>*rimwe*</u>
Algorithm 4.2 includes verbalization of the verb in 3rd pers. sg. and 'hasX' named roles (e.g., $hasChild$).

Algorithm 4.2 Verbalization of Existential Quantification ($\exists$)

```
1: $A$ axiom; Variables: $a_1$, $a_2$, $sp_1$, $r$, $r'_1$, $r'_2$, $sp_r$, $c_r$, $c_1$; and Functions: $getConcept(A)$,
   $getRole(A)$, $getRoleElement(r)$, $getNC(a)$, $dropInitialVowel(a)$, $splitRole(r)$
2: $a_1 \leftarrow getConcept(A)$               {get the concept quantified over in the axiom}
3: $r \leftarrow getRole(A)$
4: $a_2 \leftarrow getRoleElement(r)$           {get the element of the role}
5: $c_1 \leftarrow getNC(a_1)$
6: $sp_1 \leftarrow getSubjectPrefix(c_1)$
7: $a'_1 \leftarrow dropInitialVowel(a_1)$
8: **if** $r.hasForm(hasX) = true$ **then**
9:     $splitRole(r)$                           {split the role into its constituent terms}
10:    $r'_1 \leftarrow r[0]$                   {the role}
11:    $r'_2 \leftarrow r[1]$                   {the named concept in the role}
12:    $c_r \leftarrow getNC(r'_2)$
13:    $sp_r \leftarrow getSubjectPrefix(c_r)$
14:    Result $\leftarrow$ 'Buri $a'_1$ hakiri $sp_1r'_1$e $r'_2$ $sp_r$mwe $sp_r$ri $c_2$'
15: **else**
16:    Result $\leftarrow$ 'Buri $a'_1$ ni$sp_1r$a hakiri $a_2$ $sp_2$mwe'  {Verbalize with 'hakiri ... -*mwe*'}
17: **end if**
18: **return** Result
```

Negation with Roles Negation of verbs also uses *ti* (as is the case for subsumption), but additionally requires the subject prefix of the concept and the infinitive *ku*. In the case of 'hasX'-named roles, the subject prefix of X—the concept contained within the role name—is used. Algorithm 4.3 shows how this has been achieved for both cases.

Algorithm 4.3 Verbalization of Negation ($\neg$) for ‘hasX’

```
1: $A$ axiom; Variables: $a_1$, $a_2$, $r$, $r_1$, $r_2$, $c_1$, $c_r$, $sp_1$, $sp_r$; and Functions: $getRole(A)$, $getConcept(A)$, $getNC(a)$
2: $a_1 \leftarrow getConcept(A)$    {get the concept at the start of the axiom}
3: $r \leftarrow getRole(A)$    {get the role after $\neg$}
4: $c \leftarrow getNC(a_1)$
5: $sp_1 \leftarrow getSubjectPrefix(c)$
6: if $r.hasForm(hasX) = true$ then
7:     $splitRole(r)$    {split the role into its constituent terms}
8:     $r'_1 \leftarrow r[0]$    {the role}
9:     $r'_2 \leftarrow r[1]$    {the named concept in the role}
10:    $c_r \leftarrow getNC(r'_2)$
11:    $sp_r \leftarrow getSubjectPrefix(c_r)$
12:    Result ← ‘ ... ti$sp_1 r'_1$e $r'_2$ $sp_r$ri ... ’    {Verbalize with ‘ti’ to negate ‘has’ and the subject prefix of the named concept in the role}
13: else
14:    Result ← ‘ ... ti$sp$rikua$_2$a ... ’    {Verbalize with ‘ti’ which negates verbs}
15: end if
16: return Result
```

5 Evaluation

The verbalizations of $\forall$, $\sqsubseteq$, $\neg$, and $\exists$ produced several alternative texts during the initial analysis, as they did for isiZulu. In order to decide which verbalization to implement, an evaluation similar to that of [14, 9] was carried out to ascertain which pattern was preferred by survey participants.

Survey Design The participants in the evaluation were obtained from Runyankore speakers in Kampala, Uganda. WhatsApp was used to conduct the survey because it is more familiar and widely used than email or online surveys. 18 participants completed the survey, who were middle-class Banyankore and spoke both English and Runyankore; 78.8% were female and their age ranged from 24 to 59. Participants were recruited using snowball sampling, starting with a family WhatsApp group. They were instructed to answer five questions by subjectively selecting the best verbalization for each question in the form 1c, 2d, 3a, etc. They were also encouraged to explain the reasons for their choices, though only 3 participants did so. The answers were either directly delivered to us via WhatsApp, or were sent through an intermediary. Nineteen participants replied, though one was excluded because it was incomplete.

The five questions in the survey tested verbalisations of the following axioms: (1) $Teacher \sqsubseteq \exists teaches.Subject$; (2) $Cup \sqsubseteq \neg Thing$; (3) $Cat \sqsubseteq Animal$;

(4) $Man \sqsubseteq \exists hasChild.Doctor$; and (5) $Giraffe \sqsubseteq \neg\exists eats.Meat$. The English verbalization was included before the Runyankore alternatives in order to enable the participants to translate to Runyankore, and then compare with the alternatives generated from the verbalization patterns.

Question (1) had four alternatives, with the difference being either the singular or plural form, as well as the placement of *hakiri* before or after the noun. Question (2) had four alternatives: singular with *ti*, plural with *ti*, singular with *ti... ri*, and plural with *ti... ri*. Question (3) had two alternatives: either singular or plural. The placement of *hakiri* before or after the verb, the choice of whether to include *-mwe*, and either singular or plural resulted in six alternatives for question (4). Question (5) also had six alternatives, due to singular/plural, and the difficulty of translating $\neg\exists eats$ as 'never eats', 'does not eat', or 'is not eating'.

Results The singular form was generally preferred by the majority of survey participants. 72.2% chose the singular form for $\exists$ with *hakiri* after the verb; 55.6% preferred the singular with *ti* for $\sqsubseteq \neg$, and the singular with $\sqsubseteq$ was preferred by 72.2% of the participants. The plural form was only preferred in the case of negating a verb ($\neg\exists eats$) by 33.3%. Three verbalizations were not selected by any participants: question 4, the plural form with *hakiri* after the verb, as well as the plural with no *-mwe*' and question 5, the singular form with the translation as 'does not eat.' As no explanations were offered by the participants, we cannot speculate as to the reasons for this.

The results were not as clear for individual choices for $\exists hasChild.Doctor$ as the singular with *hakiri* before or after the verb, and the plural with *hakiri* before the verb were all chosen by 27.8% of participants. However, when generalized along singular/plural lines, then 72.2% selected the singular; when based on the placement of *hakiri* and the inclusion of *-mwe*, then the most preferred by 44.4% was the one which had *hakiri* before the verb and included *-mwe*.

The evaluation of $\neg\exists eats$ also produced interesting results. The plural form for the translation as 'never eat' was chosen by the majority of participants (33.3%). However, because 'never' verbalizes the axiom as if it has a temporal dimension, which OWL does not have, we considered the second best alternative, which was the plural form of the translation as 'are not eating' (27.8%). Despite this, the evaluation still showed that the plural is preferred for negation of roles.

6 Discussion

Kuhn [15] stated that one of the applications of CNLs is to provide natural and intuitive representations of formal languages. Verbalization is one of the ways of doing this, and our work here further solidifies what was done in [13, 14] to show that it is possible for Bantu languages, despite their complex linguistic structure. From our evaluation, we are able to control for certain factors during verbalization, such as the grammatical form as well as the placement of *hakiri* and the inclusion of *-mwe* in the resulting text, thus reducing the number of construction and interpretation rules. This is important because it ensures a deterministic outcome of the resulting CNL. Additionally, the subset of Runyankore applied during verbalization is restricted by the types of axioms that can be represented in $\mathcal{ALC}$. This further ensures that a predictable interpretation is obtained, and this can only be done by having a strict syntactic subset of natural language [9, 8].

The application of the bootstrapping approach in the development of language resources, especially for similar languages, has been mainly associated with reduction in development time and effort. This paper further highlights its importance to under-resourced languages like Runyankore. Tailoring Runyankore verbalizations from those of isiZulu made it a lot easier, because the underlying theory, like the factors affecting verbalization and the role of the noun class, had already been identified for isiZulu. Table 2 showed that verbalization in both languages is affected by similar factors, and Section 4 showed how the tailoring was done to obtain Runyankore verbalizations. This jumpstarting also facilitated extending the verbalisations with 'hasX' named roles. Our Java implementation of the above algorithms also creates the possibility to verbalize longer axioms.

Further, with the development of an ontology based on the Bantu noun class system—for annotating an ontology with noun class information [6]—there is now the possibility of applying such verbalizations to real-world use cases. Our research shows that the customization of these patterns is not only possible, but can obtain good results, as is the case in our evaluation. The same approach can thus be applied to languages like Rukiga, Rutooro, and Runyoro that are regarded to be between 78% and 99% similar to Runyankore [2, 18, 20]. Finally, given that a bootstrapping approach was still possible between Runyankore and isiZulu, which are in different zones according to Guthrie's classification of Bantu languages [10, 16], the same approach could be feasible even for those Bantu languages which are not classified under the same category.

7 Conclusions

The verbalization of most $\mathcal{ALC}$ constructors in Runyankore was made easier by applying the bootstrapping approach to what was done for isiZulu. We have identified that verbalization in Runyankore is affected by the same factors as in isiZulu, namely: the noun class of the name of the concept, the category of the concept, whether the concept is atomic or an expression, the quantifier used in the axiom, and the position of the concept in the axiom. A few differences were identified, so the Runyankore ones were tailored to this. The evaluation by non-linguists provided clear favorites among alternative verbalization options. We plan to complete an evaluation among linguists concerning grammatical correctness, as well as a more inclusive sample of participants along age, education, and socio-economic lines.

Acknowledgements This work is based on the research supported by the Hasso Plattner Institute (HPI) Research School in CS4A at UCT and the National Research Foundation of South Africa (Grant Number 93397).

References

1. Angelov, K., Ranta, A.: Implementing controlled languages in GF. In: Proc. of CNL'09. pp. 82–101. Springer, Marettimo Island, Italy (2009)
2. Asiimwe, A.: Definiteness and Specificity in Runyankore-Rukiga. Ph.D. thesis, Stallenbosch University, Cape Town, South Africa (2014)
3. 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)
4. Bosch, S., Pretorius, L., Fleisch, A.: Experimental bootstrapping of morphological analyzers for nguni languages. Nordic J. of African Studies 17(2), 66–88 (2008)
5. 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)
6. Chavula, C., Keet, C.M.: Is lemon sufficient for building multilingual ontologies for Bantu languages? In: Proc. of OWLED'14. CEUR-WS, vol. 1265, pp. 61–72 (2014), Riva del Garda, Italy
7. Davel, M., Barnard, E.: Bootstrapping in language resource generation. In: Proc. of PRASA'13. Langebaan, South Africa (2003)
8. Gruzitis, N., Barzdins, G.: Towards a more natural multilingual controlled language interface to owl. In: 9th International Conference on Computational Semantics (IWCS). pp. 335–339 (2011)
9. Gruzitis, N., Nespore, G., Saulite, B.: Verbalizing ontologies in controlled baltic languages. In: Proc. of Int. Conf. on HLT–The Baltic Perspective. FAIA, vol. 219, pp. 187–194. IOS Press (2010), Riga, Latvia
10. Guthrie, M.: The Classification of the Bantu Languages. Oxford University Press, London (1948)
11. Jarrar, M., Keet, C.M., Dongilli, P.: Multilingual verbalization of ORM conceptual models and axiomatized ontologies. Tech. rep., Vrije Universiteit, Brussels, Belgium (2006)
12. Joubert, L., Zimu, V., Davel, M., Barnard, E.: A framework for bootstrapping morphological decomposition. In: Proc. of PRASA'04. Grabouw, South Africa (2004)
13. Keet, C.M., Khumalo, L.: Basics for a grammar engine to verbalize logical theories in isiZulu. In: Proc. of RuleML'14. LNCS, vol. 8620, pp. 216–225. Springer (2014)
14. Keet, C.M., Khumalo, L.: Towards verbalizing ontologies in isiZulu. In: Proc. of CNL'14. LNAI, vol. 8625, pp. 78–89. Springer, Galway, Ireland (2014)
15. Kuhn, T.: A survey and classification of controlled natural languages. Computational Linguistics 40(1), 121–170 (2014)
16. Maho, J.F.: Nugl online: The online version of the updated guthrie list, a referential classification of the bantu languages (2009), http://goto.glocalnet.net/mahopapers/nuglonline.pdf
17. Safwat, H., Davis, B.: CNLs for the semantic web: a state of the art. Language Resources & Evaluation in print, DOI: 10.1007/s10579–016–9351–x (2016)
18. Tayebwa, D.D.: Demonstrative Determiners in Runyankore-Rukiga. Master's thesis, Norwegian University of Science and Technology, Norway (2014)
19. Taylor, C.: A Simplified Runyankore-Rukiga-English Dictionary. Fountain Publishers, Kampala, Uganda (2009)
20. Turamyomwe, J.: Tense and Aspect in Runyankore-Rukiga: Linguistic Resources and Analysis. Master's thesis, Norwegian University of Science and Technology, Norway (2011)