# Generating Varied Training Corpora in Runyankore Using a Combined Semantic and Syntactic, Pattern-Grammar-based Approach

**Joan Byamugisha**
IBM Research Africa
45 Juta Street, Braamfontein
Johannesburg, South Africa
joan.byamugisha@ibm.com

## Abstract

Machine learning algorithms have been applied to achieve high levels of accuracy in tasks associated with the processing of natural language. However, these algorithms require large amounts of training data in order to perform efficiently. Since most Bantu languages lack the required training corpora because they are computationally under-resourced, we investigated how to generate a large varied training corpus in Runyankore, a Bantu language indigenous to Uganda. We found the use of a combined semantic and syntactic, pattern and grammar-based approach to be applicable to this purpose, and used it to generate one million sentences, both labelled and unlabelled, which can be applied as training data for machine learning algorithms. The generated text was evaluated in two ways: (1) assessing the semantics encoded in word embeddings obtained from the generated text, which showed correct word similarity; and (2) applying the labelled data to tasks such as sentiment analysis, which achieved satisfactory levels of accuracy.

## 1 Introduction

The application of machine learning algorithms to natural language processing, generation, and understanding has led to the development of highly accurate systems for information extraction, text classification, summarization, question answering, machine translation, image and video captioning (Otter et al., 2018), and language learning (assessment, support, and analytics) (Vajjala, 2018). However, large training sets are critical to achieving high levels of accuracy, and, for some applications, creating these training sets is the most time-consuming and expensive part of applying machine learning algorithms (Ratner et al., 2016). This has resulted in the absence, to a larger extent, of machine learning applications for the very under-resourced Bantu languages. A possible solution to this problem is to generate large datasets that can then be used as training data.

Artificially creating more training data has been applied to speech (Hannun et al., 2014), image (Taylor and Nitschke, 2017), and text (D'hondt et al., 2017; Ratner et al., 2016). Our interest lies in textual data, specifically, a method for how to generate a large training corpus in Runyankore, a Bantu language indigenous to Uganda. We posed the following questions:

1. What are the existing approaches for generating large training textual corpora?

2. Which one(s) can be applied to generate a varied, semantically coherent training corpus in Runyankore?

Our aim was to generate very large corpora, both labelled and unlabelled, which could be used for sentiment and morphological analysis, and to assess word similarity, respectively. We found the use of a combined semantic and syntactic, pattern and grammar-based approach sufficient to generate one million Runyankore sentences , both labelled and unlabelled, from a dictionary of terms categorized into their appropriate parts of speech. We used generation patterns to handle the phrasal structure that comprised: adjectives, adverbs, conjunctions, prepositions, nouns, and verbs. A Context-Free Grammar (CFG) was used for verb conjugation in the simple present, present continuous, near future, remote past, near past, participial present continuous, and participial near future tenses; both primary and secondary negation; as well as the applicative, causative, and passive extensions. The evaluation of the generated text showed that it was correctly semantically related, and applicable to supervised machine learning tasks.



*Proceedings of The 13th International Conference on Natural Language Generation*, pages 273–282,
Dublin, Ireland, 15-18 December, 2020. 

The rest of this paper is arranged as follows: Section 2 provides some basics on Runyankore and its complex grammatical structure; Section 3 discusses the existing approaches for generating large training corpora and their applicability to Runyankore; Section 4 details how we generated a large Runyankore corpus and evaluated its level of variation, applicability, and word similarity; and we discuss the implications of this work in Section 5 and conclude in Section 6.

## 2 Brief Background on Runyankore

Runyankore is a Bantu language spoken in the south-western part of Uganda (Asiimwe, 2014; Tayebwa, 2014; Turamyomwe, 2011). It has an agglutinating morphology, where words are formed by adding affixes to their bases, and each affix carries meaning such as tense and aspect (Nurse and Philippson, 2003; Turamyomwe, 2011) as shown in the example below.

**Runyankore:** *Ninkimumanya*.

**Morphemes:** ni-n-ki-mu-many-a

**English:** I still know him/her.

In the above example, the morpheme *ni* is the continuous marker; *n* is the pronoun 'I'; *ki* is the persistive aspect that translates to 'still'; *mu* is the third-person pronoun for 'him/her'; *many* is the verb-root for 'know'; and *a* is the indicative final vowel.

Like all Bantu languages, Runyankore assigns all nouns to a class, and it has 20 noun classes (Excluding class 19) (Asiimwe, 2014). The simple noun comprises a prefix and a stem; for example, *omuntu* 'person' comprises the class prefix *o-mu-* (where *o* is the initial vowel or augment), and the stem *-ntu*. Additionally, the noun class (NC) is at the heart of an extensive system of concordial agreement that governs agreement in verbs, adjectives, possessives, subject, object, etc. (Katamba, 2003; Maho, 1999; Tayebwa, 2014). Table 1 shows the noun class (NC) with its number and class prefix, as well as the subject concord (SC), possessive concord (PC), and adjective concord (AC).

The default phrasal structure in Runyankore, and across Bantu languages, is Subject-Verb-Object (SVO), and the noun precedes its modifiers within a noun phrase (Nurse and Philippson, 2003). Runyankore's verbal morphology comprises fourteen tenses, six aspects, and nine verbal extensions, and the general verbal structure is as below (Turamyomwe, 2011):

| NC | SC | PC | AC |
|---|---|---|---|
| 1. o-mu- | -a- | o-wa | o-mu- |
| 2. a-ba- | -ba- | a-ba | a-ba- |
| 3. o-mu- | -gu- | o-gwa | o-mu- |
| 4. e-mi- | -gi- | e-ya | e-mi- |
| 5. ei-/e-ri- | -ri- | e-rya | e-ri- |
| 6. a-ma- | -ga- | a-ga | a-ma- |
| 7. e-ki- | -ki- | e-kya | e-ki- |
| 8. e-bi- | -bi- | e-bya | e-bi- |
| 9. e-n-/e-m- | -e- | e-ya | e-n- |
| 10. e-n-/e-m- | -zi- | e-za | e-n- |
| 11. o-ru- | -ru- | o-rwa | o-ru- |
| 12. a-ka- | -ka- | a-ka - a-ka- | |
| 13. o-tu- | -tu- | o-twa | o-tu- |
| 14. o-bu- | -bu- | o-bwa | o-bu - |
| 15. o-ku- | -ku- | o-kwa | o-ku- |
| 16. a-ha- | -ha- | a-ha | a-ha- |
| 17. o-ku- | -ha- | - | a-ha- |
| 18. o-mu- | -ha- | - | a-ha- |
| 20. o-gu- | -gu- | o-gwa | o-gu- |
| 21. a-ga- | -ga- | a-ga | a-ga- |

Table 1: The Runyankore noun class system, showing the noun class (NC), subject concord (SC), possessive concord (PC), and adjective concord (AC) for each class (the dashes between the letters in the prefix and possessive concord show separation from the initial vowel (augment))

<PreInitial> <Initial> <PostInitial> <Formative> <Limitative> <Infix> <Root> <Extension> <Final>

Table 2 from Turamyomwe (2011) shows the different 'slots' in Runyankore's verbal morphology, as well as the morphemes which occupy these slots.

The 'PreInitial' contains the primary negation or continuous marker; the 'initial,', the NC-based subject concord; 'the 'PostInitial', the secondary negative; the 'Formative', all tenses except the near past tense; the 'Limitative', the persistive aspect; the 'Infix', the NC-based object concord; the 'Extensions', that specify valency-changing categories and include the causative, applicative, stative, reciprocal, reversive, repetitive, intensive, instrumental, and passive; and the 'Final' contains morphemes associated with mood (indicative or subjunctive), the near past tense, locatives, emphatic, or declarative (Turamyomwe, 2011).

| Slot | Grammatical Category | Morpheme |
|---|---|---|
| pre-initial | 1. primary negative<br>2. continuous marker | 1. *ti-*<br>2. *ni-* |
| initial | subject concord | depends on the NC as shown in Table 1 |
| post-initial | secondary negative | *-ta-* |
| formative | tense | all tenses except near past |
| limitative | persistive aspect | *-ki-* |
| infix | object concord | depends on NC |
| extentions | App; Cs; Ps; Rec; Rev; Stv; Itv; Red; Ism | *-er-, -erer-, -ir-; zi-, -is-; -w-; -n-; -ur-, -uur-; -gur- ; repeat the stem; -is + pre-initial* |
| final | 1. final vowel<br>(a) indicative<br>(b) subjunctive<br>2. near past tense | 1.<br>(a) *-a*<br>(b) *-e*<br>2. *-ire* |
| post-final | 1. locative<br>2. emphatic<br>3. declarative | 1. *-ho, -mu-yo*<br>2. -ga<br>3. *-nu* |

Table 2: Verbal morphology of Runyankore; App: applicative, Cs: causative, Ps: passive, Rec: reciprocal, Rev: reversive, Stv: stative, Itv: intensive, Red: reduplicative, Ism: instrumental

## 3 Approaches to Generating Textual Training Corpora

In this section, we only discuss the approaches used to produce large general-purpose corpora that are used in the applications stated in Section 1. We therefore do not include methods for task-oriented training data generation such as Gardent et al. (2017); Lebret et al. (2016); Wen et al. (2015). We instead focus on four approaches: thesaurus inflation, data counterfeiting, weak supervision, and a combined semantic and syntactic, rule-based and statistical approach.

Thesaurus inflation involves probabilistically replacing terms with their synonyms (Zhang and LeCun, 2015). Data counterfeiting is the process of delexicalizing the annotated values from existing training data, and then randomly replacing them with similar related values (Wen et al., 2016). In weak supervision, training documents are deliberately noisily annotated to produce weighted low quality training data, and the weights are used in a loss function to enable noise-aware training (Bach et al., 2017; Ratner et al., 2016, 2017). Weak supervision focuses on generating labelled training data, and its use was found to result in training on larger and more diverse corpora during OCR post-correction (D'hondt et al., 2017). The combined semantic and syntactic, rule-based and statistical approach has been applied by ForgeAI and comprises: (1) a grammatical model derived from a Probabilistic Context-Free Grammar (PCFG) and refined using human annotations, which learns the grammar that characterizes a particular event; (2) semantic planning, built with a probabilistic graphical model, which decides the semantically relevant roles and tokens to include in an expression; and (3) a surface realizer, which converts a semantic plan into a grammatically correct text (Neely, 2018).

Thesaurus inflation, data counterfeiting, and weak supervision all rely on working on existing corpora (labelled data in the case of weak supervision), which Runyankore does not possess, thus creating a 'chicken and egg' problem. Also, thesaurus inflation and data counterfeiting introduce no new semantic variation in the generated text, and this is a key requirement for our preferred training corpus. The combined semantic and syntactic, rule-based and statistical approach is also limited because it requires statistical methods (PCFGs and probabilistic graphical models) which are obtained from large corpora, again, which Runyankore does not possess.

Despite this, and unlike the first three approaches, we found that the drawbacks of the combined semantic and syntactic, rule-based and statistical approach can be overcome, with some modifications, in order to generate a large corpus in Runyankore. For example, the PCFGs can be substituted with a Context-Free-Grammar-based generator that has already been shown to produce simple verbs in Runyankore (Byamugisha et al., 2016b) and more complex verbs in isiZulu[1] (Keet and Khumalo, 2017). The semantic planning can be built using generation patterns that have been used in surface realizers for Runyankore (Byamugisha et al., 2016a, 2017b) and isiZulu (Keet and Khumalo, 2014; Keet et al., 2017). However, the use of patterns requires a means of providing enough variation in the patterns so as to generate a varied training corpus. We therefore investigated the use of a combined semantic and syntactic, pattern-grammar-based approach to generate a varied training corpus in Runyankore.

[1] isiZulu is a Bantu language indigenous to South Africa

## 4 Generating Large Varied Training Corpora in Runyankore

From previous work on generating text in Runyankore, it has been shown that noun semantics play a crucial role in noun pluralization (Byamugisha et al., 2016c), verb conjugation (Byamugisha et al., 2016b), and the generation of other grammatical units such as quantifiers (Byamugisha et al., 2017a). On the other hand, the syntactical structure of Runyankore is also taken into account during noun pluralization (Byamugisha et al., 2016c) and phonological conditioning (Byamugisha et al., 2016b). This, together with evidence for the use of a grammar engine (Byamugisha et al., 2017a) and pattern-based generation (Byamugisha et al., 2016a) in Runyankore, are the basis for investigating the use of a combined semantic and syntactic, pattern-grammar-based approach to generate a Runyankore corpus that is large enough and has sufficient variation to be used as training data.. Given that there are supervised and unsupervised machine learning algorithms, we aimed to generate both labelled and unlabelled data, and focused on morphological analysis for the labels..

### 4.1 Materials And Methods

We first extracted different parts of speech from a Runyankore dictionary (Taylor, 2009). For both nouns and verbs, we only considered those that are applicable in multiple contexts (such as *omuntu* 'person' and *reeb-* 'see'), and avoided nouns like *egyora* 'a cloth measure' and verbs like *kusinsina* 'stop oneself from saying'. We also avoided proper nouns unless they referred to time or locations. The annotation process on nouns for their sentiment, category, and noun class, on verbs for their type, subject, object, sentiment, and category, and on other parts of speech for their concord, phonological conditioning, and sentiment, was done manually, following the definitions and examples provided in the dictionary.

**Nouns** From 2548 singular nouns extracted from the dictionary, we selected 385 nouns. We only considered singular nouns because an existing Runyankore pluralizer (Byamugisha et al., 2016c) is available. We annotated each noun with its noun class, category, and sentiment. We identified 34 noun categories, and also accounted for their taxonomic relationships. Table 3 shows the classifications for the different categories.

| Superclass | Noun Categories |
|---|---|
| abstract | abstract_give, abstract_have, abstract_rw, abstract_time, prop_time |
| time | abstract_time, prop_time |
| food | food_fruit, food_liquid, food_plant, food_solid |
| kins | kin, kin_f, kin_m |
| humans | human, human_f, human_m, human_med, human_y,kins |
| animal | animal_meat, animal_plant, animal_y |
| animals | animal, humans |
| loc | loc_in, loc_out, prop_loc |
| part | part_animal, part_plant |
| plants | plant, food_fruit, food_plant |
| non_living | food_cook, food_loc, thing_cloth, thing_move, thing_tool |
| living | animals, plants |
| all | living, non_living |
| <unclassified> | illness, thing_med |

Table 3: The taxonomic groupings for the different noun categories

From Table 3, it can be seen that a male kinship term (for example, grandfather) categorized as 'kin_m' belongs to the superclass 'kins', that in turn belongs to the superclass 'humans' that is a subclass of 'animals', and this is a subclass of 'living' for all living things. Similarly, a fruit belongs to the superclasses 'food' and 'plants', and the latter is a subclass of 'living'.

**Verbs** We selected 198 verbs from the 1330 extracted from the dictionary. As the verbs in the dictionary contain the infinitive *ku*, as well as the final vowel and verbal extensions, we further preprocessed the selected verbs to their roots, and annotated each with its subject category, sentiment, type, and object category. The subject categories correspond to the noun categories shown in Table 3, and we only considered seven verb types: action, catenative, copulative, dependent, performative, predicative, and stative. We also identified 28 object categories, which included whether the verb is intransitive, transitive, or ditransitive. Table 4 shows the object categories for the different verb categories.

| Verb Category | Object Categories |
|---|---|
| ditransitive | all, all; illness, med; all, loc |
| intransitive | |
| transitive | Nouns: abstract, all, animal, food, human, illness, living, non_living, part, plant |
| transitive | Verbs: action, all |

Table 4: Verb categories and their associated object categories

From the categories shown in Table 4, the subject and object of a verb can be obtained to produce a sentence. For example, the verb root *ih* for 'remove' is marked as having type 'dependent' and object category 'ditransitive_locative'. A dependent verb requires a preposition, and the indirect object is a location, resulting in a pattern where the direct object is *removed from somewhere*.

**Other Parts of Speech** For the other parts of speech, we extracted 21 adjectives, 6 adverbs, 7 conjunctions, and 8 prepositions. We annotated each with its concord (whether subject, adjective, relative, possessive, or pronomial), if phonological conditioning is required (and if so, what kind), and sentiment. The sentiment labels used here, as well as for nouns and verbs, are 'good', 'bad', 'none', and 'both'. The label 'both' is used where the sentiment of the part of speech can be either bad or good depending on the context in which it is used.

**Pattern Structures** When determining pattern structures, we referred to the sentence structure used in the Runyankore newspaper Orumuri[2]. We aimed to cover the past, present, and future tenses, and based on a manual analysis of the tenses, aspects, and extensions used in this newspaper, we considered the simple present, present continuous, near future, remote past, near past, participial present continuous, and participial near future tenses. We also considered the applicative, causative, and passive extensions; the indicative and subjunctive moods; as well as primary and secondary negation. Algorithm 4.1 below shows a simple sentence pattern.

The pattern shown in Algorithm 4.1 is the simplest possible pattern, with the object concord conjugated in the verb, instead of stating the object explicitly. It can be enhanced to include adjectives, adverbs, negation, tense and aspect, pluralization, and sentiment. Algorithm 4.2 shows a more complicated pattern.

In Algorithm 4.2, the sentiment is used when selecting the noun, vverb, and adjective. The sentence output pattern shows the placement of the different parts of speech, as well as the use of two verb types, 'copulative' and 'stative'.

**Verb Conjugation** Previous work shows that it is possible to use a Context-Free Grammar (CFG) to conjugate verbs in Runyankore (Byamugisha et al., 2016b). We extended the existing Runyankore CFGs to include the tenses and aspects observed in the sentences in the Orumuri newspaper. The slots in Table 2 formed the non-terminals in the CFG, while the morphemes formed the terminals. In the CFG shown below, *IG* is the non-terminal with the initial grouping, with a production rule for the *PN*, the 'PreInitial', *IT*, the 'Initial', and *SN*, the 'PostInitial'; *FM* is for the 'Formative'; *LM*, the 'Limitative'; *IF*, the 'Infix'; *VR*, the verb root; *EX*, the 'Extensions'; and *FN* the 'Final'.

**Algorithm 4.1** An example of a simple generation pattern

1: Variables: $n$ noun, $n_c$ noun class, $v_r$ verb root, $t$ tense, $o'$ object category, $o$ object, $o_c$ object category, $v$ conjugated verb, $s_c$ subject concord
2: Functions: $getNoun(nounCategory)$, $getNounClass(n)$, $getVerbRoot(type)$, $getTense(tenses)$, $getObjectCategory(v_r)$, $getObjectConcord(n_c)$, $conjugateVerb(t, s_c, o_c, v_r, f_v)$
3: $n \leftarrow getNoun(nounCategory)$ {Randomly obtain a noun based on one of the categories in Table 3}
4: $n_c \leftarrow getNounClass(n)$ {get the noun class of the noun}
5: $v_r \leftarrow getVerbRoot($ 'action'$)$ {Randomly get a verb root of type 'action'}
6: $t \leftarrow getTense(tenses)$ {Randomly select a tense from the available tenses}
7: $o' \leftarrow getObjectCategory(v_r)$ {get the appropriate object category for the verb}
8: $o \leftarrow getNoun(o')$ {Randomly obtain a noun based on the object category}
9: $o_c \leftarrow getObjectConcord(n_c)$ {Use the noun class to get the object concord}
10: $s_c \leftarrow getSubjectConcord(n_c)$ {Use the noun class to get the subject concord}
11: $v \leftarrow conjugateVerb(t, s_c, o_c, v_r, f_v)$ {Conjugate the verb for the tense $t$, object concord $o_c$, and final vowel $f_v$}
12: Result $\leftarrow$ " $n$ $v$ " {Generate the sentence}
13: **return** Result

$S \rightarrow IG\ FM\ LM\ IF\ VR\ EX\ FN$
$IG \rightarrow PN\ IT\ SN$
$PN \rightarrow$ ti | ni
$IT \rightarrow$ a | o | n | tu | mu | ba | gu | gi | ri | ga | ki | bi | e | zi | ru | tu | ka | bu | ku | gu | ga
$SN \rightarrow$ ta
$FM \rightarrow$ za | ka | riku | rikuza
$LM \rightarrow$ ki
$IF \rightarrow$ mu | ba | gu | gi | ri | ma | ki | bi | gi | zi | ru | tu | ka | bu | ha | gu | ga

[2]Orimuri is available from https://www.newvision.co.ug/new_vision/news/1044356/orumuri

$VR \rightarrow \ verbRoot$
$EX \rightarrow$ w | er | erer | ir | zi | is | n | ur | uur | gur | $VS$ | is$PN$
$FN \rightarrow$ a | e | ire

The above CFG accounts for rules stating which slots can and cannot co-occur. For example, the continuous marker *ni* cannot co-occur with the primary negative *ti* or the secondary negative *ta* (Turamyomwe, 2011).

**Algorithm 4.2** An example of a more complicated generation pattern

Variables: $n$ noun, $n_c$ noun class, $v_r$ verb root, $t$ tense, $o'$ object category, $o$ object, $o_c$ object category, $v$ conjugated verb, $s$ sentiment, $a_j$ adjective, $a_r$ adjectival root, $a_v$ adverb, $s_c$ subject concord
Functions: $getNoun(nounCategory, s)$, $getNounClass(n)$, $getVerbRoot(type, s)$, $getTense(tenses)$, $getObjectCategory(v_r)$, $getObjectConcord(n_c)$, $conjugateVerb(t, s_c, v_r, f_v)$, $getSentiment()$, $getAdjectivalRoot(s)$, $getAdjective(n_c, a_r)$, $getAdverb()$
$s \leftarrow getSentiment()$ {Randomly select from one of the four sentiments}
$n \leftarrow getNoun(nounCategory, s)$ {Randomly obtain a noun based on its sentiment and one of the categories in Table 3}
$n_c \leftarrow getNounClass(n)$ {get the noun class of the noun}
$v_r1 \leftarrow getVerbRoot($'copulative'$)$ {Randomly get a copulative verb root}
$v_r2 \leftarrow getVerbRoot($'stative'$, s)$ {Randomly get a stative verb root based on the sentiment}
$t \leftarrow getTense(tenses)$ {Randomly select a tense from the available tenses}
$a_c \leftarrow getAdjectivalConcord(n_c)$ {Use the noun class to get the adjectival concord}
$a_r \leftarrow getAdjectivalRoot(s)$ {Randomly get an adjectival root based on the tense}
$a_j \leftarrow getAdjective(n_c, a_r)$ {Obtain the full adjective using the adjectival root and concord}
$a_v \leftarrow getAdjective()$ {Randomly get an adverb}
$s_c \leftarrow getSubjectConcord(n_c)$ {Use the noun class to get the subject concord}
$v_1 \leftarrow conjugateVerb(s_c, v_r1)$ {Conjugate the copulative verb with the subject concord $s_c$}
$v_2 \leftarrow conjugateVerb(t, s_c, v_r, f_v)$ {Conjugate the stative verb for the tense $t$, subject concord $s_c$, and final vowel $f_v$}
Result $\leftarrow$ " $n\ a_j\ v_1\ v_2\ a_v$ " {Generate the sentence}
**return** Result

### 4.2 Training Data Generation

The surface realizer was implemented as a Java application, with the verb conjugation based on the CFG Java tool by Xu et al. (2011). From the annotated resources and the selected generation patterns, we generated text in seven tenses: the simple present tense, which has no tense morpheme; the present continuous tense that uses the continuous marker *ni-*; the near future tense, *-za-* that applies to the infinitive form of the verb; the remote past tense, *-ka-*; the near past tense, *-ire*; the participial present continuous tense, *-riku-*; and the participial near future tense, *-rikuza-* (Turamyomwe, 2011). All these tenses, except for the near past tense that is placed in the final slot, are placed in the formative slot in Table 2.

We also used the applicative (*-er-* and *-erer-*), causative (*-ir-*), and passive (*-w-*) extensions that are placed in the extensions slot in Table 2; the indicative (*-a-*) and subjunctive (*-e*) moods that are placed in the final slot; as well as primary negation (*ti-*) that is placed in the initial slot, and secondary negation (*-ta-*) that is placed in the post-initial slot in Table 2 (Turamyomwe, 2011).

Of the seven conjunctions, four (*haza*, *reero*, *kandi*, and *obwo*) are different variations of 'and', thus the proceeding phrase should maintain the same sentiment as the preceding phrase. On the other hand, three of the conjunctions (*kwonka*, *okwihaho*, and *baitu*) are different variations of 'but', and should therefore change the sentiment of the proceeding phrase. Given a type of verb, a sentiment, and a noun category, sentiment change was implemented in three ways: (1) using an adjective or adverb of the opposite sentiment; (2) negating the verb, which would make a positive verb negative, and vice versa; and (3) changing the sentiment itself, and then using it to obtain verbs and nouns of this new sentiment.

In order to vary the structure and content of each sentence, we randomly selected the sentence pattern to use, which specific part-of-speech to realize based on the different noun categories, verb types, and the sentiment of the adjectives, when to pluralize the nouns, as well as whether to change, negate, or keep the existing sentiment. We also performed phonological conditioning that is required during generation, where, due to the agglutinative structure of Runyankore, the generated text can contain letter combinations that do not exist in Runyankore phonology. When this occurs, phonological rules are used to make the required changes that reflect the sound change, and this is referred to as phonological conditioning (Maho, 1999). Phonological conditioning was performed during noun pluralization, verb conjugation, and pattern realization, and was achieved through vowel coalescence (adding an extra vowel), vowel elision (deleting a vowel),

vowel harmony (considering the presence of a nasal compound), vowel assimilation (replacing a vowel with an apostrophe), or by deleting or adding a consonant.

Finally, a boolean flag was used to decide whether to generate labelled or unlabelled data. Table 5 shows the different tags that were considered for labelling the morphology of the generated text. These tags were based on the labels used in a Runyankore dictionary (Taylor, 2009) for different parts of speech, as well as the tags used in the morphological analyzers by Eiselen and Puttkammer (2014) that covers nine Bantu languages.

| Tag | Meaning |
|---|---|
| <NC Number>ac | NC + Adjective concord |
| adj | Adjective |
| adv | Adverb |
| aug | Augment |
| conj | Conjunction |
| cont | Continuous marker |
| ext | Extension |
| fv | Final vowel |
| inf | Infinitive |
| n<NC number> | Noun + NC |
| <NC number>oc | NC + object concord |
| <NC Number>pc hline primNeg | NC + Possessive concord Primary negative |
| secNeg | Secondary negative |
| <NC number>sc | NC + subject concord |
| tn | Tense marker |
| v | Verb |

Table 5: List of tags used to label morphological units and parts of speech

### 4.3 Results and Evaluation

We generated a one million sentence general-purpose domain independent corpus. We also generated labelled data, with labels for sentiment, parts-of-speech (such as noun, adjective, preposition, etc.) as well as the morphological units of the conjugated verb. From the 28 object categories, 7 tenses, 3 extensions, 8 major patterns, and 4 sentiment adjustment options, we created 18,816 different ways of varying the sentence structure for a single subject, verb, and object. Further variation is introduced by performing noun pluralization, having 34 different noun categories and 7 different verb types, as well as 7 different conjunctions for the 8 major patterns.

We evaluated for the quality of the generated text using a task-based evaluation, where we applied the generated text to some supervised and unsupervised machine learning tasks. For the latter, we used FastText (Bojanowski et al., 2016) to obtain word vectors and assess the semantic relatedness from the generated text. We also trained and tested a sentiment analysis text classifier based on FastText (Joulin et al., 2016).

**Assessing Semantic Relatedness** We obtained word vectors and queried for nearest neighbors. The query word was selected based on its semantic category, that is, whether it is a noun for people, plants, or animals, or an adjective. The examples in Table 6 show the query word and the first five results according to highest confidence.

| Query Word | Results |
|---|---|
| *omuntu* (person) | *omugyesi* (reaper), *omutaahi* (companion), *omukoreesa* (overseer), *omushomesa* (teacher), *omukuru* (elder) |
| *omuti* (tree) | *omutumba* (banana tree), *omwani* (coffee tree), *omuzaabibu* (grape or grapevine), *omucungwa* (orange), *omugusha* (sorghum) |
| *omukono* (arm) | *omunwa* (mouth), *omutwe* (head), *eriino* (tooth), *enkokora* (elbow), *okuguru* (leg) |
| *embwa* (dog) | *embeba* (rat), *enkyende* (monkey), *empungu* (bird of prey), *enumi* (bull), *enyawaawa* (green ibis) |
| *rungi* (beautiful) | *rurungi* (beautiful), *rukuru* (important), *rirungi* (beautiful), *oruyonjo* (clean/tidy), *orurikutukura* (pure) |
| *rofa* (dirty) | *erirofa* (dirty), *eriruhire* (tired), *rigufu* (short), *erifiire* (stupid), *ribi* (ugly) |

Table 6: Results from word similarity evaluation

The results in Table 6 show that the semantics embedded in the generated text are correctly associated as similar.

**Performing Sentiment Analysis** In order to perform sentiment analysis on the generated text, we also stored the sentiment of each sentence (whether good, bad, none, or both) in a separate file; each sentence labelled according to the FastText default style of '__label__'. For example, a sentence with a 'bad' sentiment is labelled as: __label__bad *omunywi mugufu naaba naatomera obugaari kandi omurofa mugufu naaba naatomera ekyarani*, 'The short beer supplier spends time knocking over wheelbarrows and the short dirty one spends time knocking over sowing machines'.

We trained two models, one that accounts for all four sentiments, and another that only predicts 'good' or 'bad'. Each sentiment has over 200,000 examples in the dataset ('good'=270,720, 'bad'=271,031, 'none'=207,796, and 'both'=250,453). The four-sentiment model

was trained on 800,000 sentences and tested on 200,000 sentences, and achieved 64% accuracy. The binary sentiment model had a dataset with 541,751 examples, and it was trained on 500,000 sentences and tested on 41,751 sentences, and achieved 77.3% accuracy. These results show a good first attempt at sentiment analysis for Runyankore.

## 5 Discussion

We investigated how to solve the problem of the lack of training data in Runyankore, and found several ways in which training data can be generated. We found the use of a combined semantic and syntactic, pattern-grammar-based approach to be applicable to the grammatical complexity and under-resourced state of Runyankore. Using this approach, we were able to generate one million labelled and unlabelled sentences in seven of Runyankore's 14 tenses. This large dataset can be used in both supervised and unsupervised machine learning algorithms for various tasks as shown in our evaluation.

The effort required to generate this dataset is significant, as explained in Section 4.1. The grammatical aspects too are specific to Runyankore's morphology. Despite this, previous work has shown that the important text generation aspects–noun pluralization, verb conjugation, and pattern-based generation–can be generalized to other agglutinating Bantu languages. For noun pluralization, a generic noun pluralizer exists for agglutinating Bantu languages (Byamugisha et al., 2018). Verb conjugation using CFGs has also been shown to be possible for isiZulu (Keet and Khumalo, 2017), another agglutinating Bantu language. Finally, the ability to bootstrap text generation patterns from one agglutinating Bantu language to another was shown in (Byamugisha, 2019). We therefore hypothesize that, with some tailoring, this approach may be generalizable to other Bantu languages.

Interestingly, the results from word similarity evaluation in Table 6 hint on the possibility of using this approach to identify the noun class (NC) of a noun. Generally, the classes of nouns in Bantu languages are based on the semantics of the noun. Table 7 shows the semantic generalizations of the types of nouns in each class (Keet and Khumalo, 2014; Baertlein and Ssekitto, 2014; Kimenyi, 2004; Jeon et al., 2015; Zentz, 2016; Taraldsen, 2010; Mohlala, 2003; Katamba, 2003; Maho, 1999).

| Noun Class | Description of Associated Nouns |
|---|---|
| 1 and 2 | People and kinship |
| 3 and 4 | Plants, nature, and some parts of the body |
| 5 and 6 | Fruits, liquids, some parts of the body, and paired things |
| 7 and 8 | Inanimate objects |
| 9 and 10 | Tools and animals |
| 11 | Long thin stringy objects, languages, and inanimate objects |
| 12 and 13 | Diminutives |
| 14 | Abstract concepts |
| 15 | Infinitives and parts of the body |
| 16, 17, and 18 | Locative classes |
| 19 | Diminutives |
| 20, 21, and 22 | Augmentatives |
| 23 | Locative class |

Table 7: Classification of Bantu nouns into noun classes (the 'and' indicates that the two classes are a singular/plural pairing)

The inability to detect the noun class of nouns with the same prefix but belonging to different classes (such as ***omu**ntu* (person) in NC 1 and ***omu**ti* (tree) in NC 3) is a big problem in Bantu language computational linguistics. This is because, as explained in Section 2, the noun class (NC) is at the heart of an extensive system of concordial agreement, and getting the NC wrong can result in incorrect noun pluralization, verb conjugation, as well as other parts -of-speech such as adjectives and possessives.

Comparing the semantic categories of nouns in Table 7 with the examples in Table 6, it can be seen that *omuntu* and its related words, people terms, would belong to NC 1; the *omuti* group, plants, would fit in NC 3; the *omukono* group, parts of the body, can be split among NCs 3 and 5; and *embwa*, animals, can be placed in NC 9.

Existing approaches for surface realization in Runyankore (Byamugisha et al., 2016a, 2017b) and isiZulu (Keet and Khumalo, 2014; Keet et al., 2017) annotate nouns with their noun class (NC) in order to solve the problem of having the same class prefix in different classes (see classes 1, 3, and 18 in Table 1 in Section 2). However, our results from word similarity evaluation show that a semantic distinction is made between people nouns (that are found in NC 1; see the *omuntu* example in Table 6) and other nouns starting with the *omu-* prefix (see the *omuti* and *omukono* examples in Table 6).

Finally, while the results on sentiment analysis are not spectacular, our work is, to the best of our knowledge, the first sentiment analysis module for Runyankore. Additionally, the results from the

word similarity evaluation also show that different sentiments can be distinguished (see the *rungi* and *rofa* examples in Table 6).

## 6 Conclusion

In this paper, we investigated how to generate a large and varied corpus to act as training data for a grammatically complex and computationally under-resourced language, Runyankore. We found the use of a combined semantic and syntactic, pattern-grammar-based approach to be applicable to Runyankore. Using this approach, we were able to generate one million labelled and unlabelled sentences, that were evaluated as correctly encoding related word semantics, and performing well when applied to a supervised machine learning task, sentiment analysis. Future work will involve identifying a qualitative evaluation for the dataset; manually labelling sentences from Orumuri for sentiment, in order to have an independent dataset to evaluate sentiment analysis, and investigating how the labelled data can be used together with the word similarity results to determine the noun class of a noun.

## References


Allen Asiimwe. 2014. *Definiteness and Specificity in Runyankore-Rukiga*. Ph.D. thesis, Stallenbosch University, Cape Town, South Africa.

H. Stephen Bach, Bryan He, Alexander Ratner, and Christopher Ré. 2017. Learning the structure of generative models without labeled data. In *34th International Conference on Machine Learning (ICML 2017)*, Sidney, Austtralia. ArXiv.

Elizabeth Baertlein and Martin Ssekitto. 2014. Luganda nouns inflectional morphology and tests. *Linguistic Portfolios*, 3.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Joan Byamugisha. 2019. *Ontology Verbalization in Agglutinating Bantu Languages: A Study of Runyankore and its Generalizability*. Ph.D. thesis, University of Cape Town.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2016a. Bootstrapping a runyankore cnl from an isizulu cnl. In *5th Workshop on Controlled Natural Language (CNL 2016)*, volume 9767, pages 25–36, Aberdeen, Scotland. Springer LNAI.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2016b. Tense and aspect in runyankore using a context-free grammar. In *9th International Conference on Natural Language Generation (INLG 2016)*, Edinburgh, Scotland.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2017a. Evaluation of a runyankore grammar engine for healthcare messages. In *10th International Conference on Natural Language Generation (INLG 2017)*, Santiago de Compostela, Spain.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2017b. Toward an nlg system for bantu languages: first steps with runyankore (demo). In *10th International Conference on Natural Language Generation (INLG 2017)*, Santiago de Compostela, Spain.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2018. Pluralizing nouns in agglutinating bantu languages. In *27th International Conference on Computational Linguistics (COLING 2018)*, Santa Fe, New Mexico, USA.

Joan Byamugisha, C. Maria Keet, and Langa Khumalo. 2016c. Pluralizing nouns in isizulu and related languages. In *17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2016)*, volume 9626, Konya, Turkey. Springer LNCS.

Eva D'hondt, Cyril Grouin, and Brigitte Grau. 2017. Generating a training corpus for ocr post-correction using encoder-decoder model. In *8th International Joint Conference on Natural Language Processing*, pages 1006–1014, Taipei, Taiwan.

Roald Eiselen and Martin J Puttkammer. 2014. Developing text resources for ten south african languages. In *LREC*, pages 3698–3703.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for NLG micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Bryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Y. Andrew Ng. 2014. Deep speech: Scaling up end-to-end speech recognition. *Computational Research Repository (CoRR)*, abs/1412.5567.

Lisa Jeon, Jessica Li, Samantha Mauney, Anaí Navarro, and Jonas Wittke. 2015. A basic sketch grammar of gíkúyú. *Rice Working Papers in Linguistics*, 6.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Francis Katamba. 2003. Bantu nominal morphology. In *The Bantu Languages: Routledge Language Family Series 4*, chapter 7, pages 103–120. Taylor and Francis Routledge, London.

C. M. Keet, M. Xakaza, and L. Khumalo. 2017. Verbalising owl ontologies in isizulu with python. In *14th Extended Semantic Web Conference (ESWC'17)*, Portoroz, Slovenia. Springer LNCS.

C. Maria Keet and Langa Khumalo. 2014. Towards verbalizing ontologies in isizulu. In *4th Workshop on Controlled Natural Languages (CNL'14)*, pages 78–89, Galway, Ireland.

C. Maria Keet and Langa Khumalo. 2017. Grammar rules for the isizulu complex verb. *Southern African Linguistics and Applied Language Studies*, 35:183–200.

Alex Kimenyi. 2004. Kinyarwanda morphology. In Geert Booij, Christian Lehmann, Joachim Mudgan, and Stavros Skopeteas, editors, *Morphology: An International Handbook for Inflection and Word Formation*, volume 17.2. De Gruyter.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

Jouni Maho. 1999. *A Comparative Study of Bantu Noun Classes*. Ph.D. thesis, Goteborg University, Goteborg, Sweden.

Linkie Mohlala. 2003. The bantu attribute noun class prefixes and their suffixal counterparts, with special reference to zulu. Master's thesis, University of Pretoria, Pretoria, South Africa.

Jake Neely. 2018. How we are using natural language generation to scale forge.ai. Webpage.

Derek Nurse and Gerard Philippson. 2003. Introduction. In *The Bantu Languages: Routledge Language Family Series 4*, chapter 1, pages 1–9. Taylor and Francis Routledge, London.

Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2018. A survey of the usages of deep learning in natural language processing. *Computing Research Repository (CoRR)*, abs/1807.10854.

Alexander Ratner, H. Stephen Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christophher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *VLDB Endowment (PVLDB)*, 11(3).

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, pages 3567–3575. Curran Associates, Inc., Barcelona, Spain.

Knut Tarald Taraldsen. 2010. The nanosyntax of nguni noun class prefixes and concords. *Lingua*, 120(6):1522–1548.

Doreen Daphine Tayebwa. 2014. Demonstrative determiners in runyankore-rukiga. Master's thesis, Norwegian University of Science and Technology, Norway.

C. Taylor. 2009. *A Simplified Runyankore-Rukiga-English Dictionary*. Fountain Publishers, Kampala, Uganda.

Luke Taylor and Geoff Nitschke. 2017. Improving deep learning using generic data augmentation networks. *Computing Research Repository (CoRR)*, abs/1708.06020.

Justus Turamyomwe. 2011. Tense and aspect in runyankore-rukiga: Linguistic resources and analysis. Master's thesis, Norwegian University of Science and Technology, Norway.

Sowmya Vajjala. 2018. Machine learning and applied linguistics. *The Encyclopedia of Applied Linguistics*.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina Maria Rojas-barahona, Pei-hao Su, David Vandyke, and J. Steve Young. 2016. Multi-domain neural network language generation for spoken dialogue systems. In *15th Annual Conference of the North American Chapter of the Association for Computational Linguistics-Human Language Technologies (NAACL-HLT)*, pages 120–129, San Diego, California, USA. Association for Computational Linguistics (ACL), Association for Computational Linguistics (ACL).

Tsung-Hsien Wen, Milica Gasíc, Nikola Mrksíc, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal. Association for Computational Linguistics.

Zhiwu Xu, Lixiao Zheng, and Haiming Zhen. 2011. A toolkit for generating sentences from context-free grammars. *International Journal of Software and Informatics*, 5:659–676.

Jason Zentz. 2016. *Forming Wh-Questions in Shona: A Comparative Bantu Perspective*. Ph.D. thesis, Yale University.

Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *Computing Research Repository (CoRR)*, abs/1502.01710.