Scientific Research Publishing

Journal of Computer and Communications, 2020, 8, 59-77
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Automatic Generation of Amharic Math Word Problem and Equation

Andinet Assefa Bekele

Department of Computer Science, School of Computing, Dire Dawa Institute of Technology, Dire Dawa, Ethiopia
Email: andbeyes@gmail.com

How to cite this paper: Bekele, A.A. (2020) Automatic Generation of Amharic Math Word Problem and Equation. *Journal of Computer and Communications*, **8**, 59-77.
https://doi.org/10.4236/jcc.2020.88006

Received: July 25, 2020
Accepted: August 28, 2020
Published: August 31, 2020

Abstract

Math word problem uses a real word story to present basic arithmetic operations using textual narration. It is used to develop student's comprehension skill in conjunction with the ability to generate a solution that agrees with the story given in the problem. To master math word problem solving, students need to be given fresh and enormous amount of problems, which normal textbooks as well as teachers fail to provide most of the time. To fill the gap, a few research works have been proposed on techniques to automatically generate math word problems and equations mainly for English speaking community. Amharic is a Semitic language spoken by more than hundred million Ethiopians and is a language of instruction in elementary schools in Ethiopia. And yet it belongs to one of a less resourced language in the field of linguistics and natural language processing (NLP). Hence, in this paper, a strategy for automatic generation of Amharic Math Word (AMW) problem and equation is proposed, which is a first attempt to introduce the use template based shallow NLP approach to generate math word problem for Amharic language as a step towards enabling comprehension and learning problem solving in mathematics for primary school students. The proposed novel technique accepts a sample AMW problem as user input to form a template. A template provides AMW problem with placeholders, type of problem and equation template. It is used as a pattern to generate semantically equivalent AMW problems with their equations. To validate the reality of the proposed approach, a prototype was developed and used as a testing platform. Experimental results have shown 93.84% overall efficiency on the core task of forming templates from a given corpus containing AMW problems collected from elementary school mathematics textbooks and other school worksheets. Human judges have also found generated AMW problem and equation as solvable as the textbook problems.

Keywords

AMW Problem, Automatic Problem Generation, Template Based Shallow NLP Approach, Question Generation, Math Word Problem

1. Introduction

Math word problem is a type of exercise in mathematics where major background information on the problem is given using textual sentences rather than in mathematical equations. As word problems often involve a narrative of real world scenario, they are sometimes referred to as story problems and may vary in the amount of technical language used [1]. With math word problems, learners gain basic linguistic knowledge together with skills of basic arithmetic operations such as addition, subtraction, multiplication and division. It is admitted that students can solve very well on the normal mathematical operations, but they find it challenging when these operations are behind word problems in text form.

Due to this challenge, the subject math word problem solving needs teacher's closer attention to student's activities so that students may get enormous amount of fresh math word problems to solve and proper problem-solving assistance. In spite of the fact that teacher's assistance cannot be ignored, yet teacher cannot give all types of questions and lacks to supply sufficient volume of exercises that considers individual differences among students. In addition, textbook math word problems lack freshness that students may still get board of solving the same kinds of problems again and again. On top this, in these days of pandemic where corona virus (COVID-19) is a global challenge, schools are closed and students are locked behind doors and forced to engage with their activities at home apart from their teacher's physical presence and assistance. This may still worsen the challenge for students to get sufficient math word problem as per their need. Though, the use of suitable teaching methodology can somehow help them to reduce this challenge, the power of technology needs to be devised to produce fresh and enormous amount of math word problems for student's unbounded practice to fix the aforementioned challenge from grassroot.

A study has shown that the use of NLP for automatic generation of math word problems with their equation can help students get a chance to practice on more problem-solving exercises as per their individual need and hence improve their problem-solving ability [2]. Accordingly, quit a few research works have been proposed on techniques to generate math word problems and equations automatically for English speaking community as well as for other few languages. In this paper, I have presented a strategy to automatically generate math word problem and equation for Amharic language.

Amharic is a Semitic language spoken by more than hundred million Ethiopians. Next to Arabic, it is the most spoken Semitic language. It is the official working language of the Federal Democratic Republic of Ethiopia and thus has official status nationwide. It is also the official or working language of several of the states/regions within the federal system, including Amhara and the multi-ethnic Southern Nations, Nationalities and Peoples region. Outside Ethiopia, Amharic is the language of millions of emigrants across the world and is also spoken in Eritrea [3]. It is one of the major languages used for instruction in

elementary schools in Ethiopia. It is written using a writing system called Fidel or abugida, adapted from the one used for the now-extinct Ge'ez language. Amharic being one of the morphologically rich languages presents a challenge to the area of natural language processing in general [4].

Even though Amharic belongs to one of a less resourced language in the field of linguistics and natural language processing (NLP), a number of baseline NLP research works on Amharic language has been done so far. For example, POS tagging and Morphological analyzer [5], Spell checking, and named entity recognition [6]. This NLP research works makes the way suitable to conduct further research works on Amharic language. Hence, this paper bases these research works to present proposed automatic generation of AMW problem and equation using template based shallow NLP approach as a step towards enabling comprehension and learning problem solving in mathematics for elementary level students.

The remainder of this paper is organized as follows. Section 2 discusses related works. Section 3 discusses preliminary concepts used in this research. Section 4 details my proposed approach. Section 5 presents developed prototype. Section 6 discusses experimental results and finally, Section 7 concludes this study and draws future research direction.

2. Related Works

So far, various kinds of approaches have been proposed for automatic generation of math word problem as well as equation generation for English language. And recently the area has gained interest with novel strategy that generates mathematical word problems from ontologies in unrestricted domains [7]. The approach builds on an existing ontology verbalizer that renders logical statements written in Web Ontology Language (OWL) as English sentences. On the other hand, an approach for problem generation for natural deduction is proposed by Umair z. *et al.* [8], a strategy for algebraic proof problems by Rohit *et al.* [9], a strategy for procedural problems by Erik *et al.* [10] and for embedded systems by Dorsa *et al.* [11]. Although their specific approach varies, all of them follows a similar fashion: they first form a template from a sample problem, and then use a template to generate a set of problems. Ahmed *et al.* proposed automatic template formation, Andersen *et al.* and Singh *et al.* has proposed a semi-automatic template to generate math word problems. However, Sadigh *et al.* do formation of templates manually.

Quite a few other scholars have proposed various approaches for generating math word problems [12]. The use of neural network model to generate math word problems from the given equations and topics has also been proposed in by Qingyu *et al.* [13]. Tao Li focuses on generating questions with multiple variables for intelligent tutoring and further extends it with a complex algorithm that will generate more questions [14]. Andrenucci introduced a template-based approach to generate questions [15]. Other than research works on math word problem generation, very few others have proposed various strategies on solving

math word problem. Chiang *et al.* [16] have proposed a neural approach to automatically solve math word problems by operating symbols according to their semantic meanings in texts. Although question generation while keeping semantic meaning is challenging and yet interesting research area intelligent learning systems, there are few researches on math word problem question generation.

With regard to Amharic language, question generation for definitive type [17], and for factoid questions [18] has been proposed. But concerning math word problem generation, there is no any previous attempt for Amharic language. To sum up, strategies to generate questions can be generally classified into template-based [19], syntax-based [20] [21] and semantics-based [22]. Among these, template-based techniques are mostly used for special purpose applications such as math word problem generation. Syntax-based strategies are more effective for short and simple sentences. However, Semantic-based approach utilizes NLP concepts which is theoretically more interesting and practically challenging. Employing a strategy called template based shallow NLP approach, which is a hybrid approach combining template-based approach together with semantic based approach can greatly help to keep the freshness of problem generated by ensuring its semantic meaning and templates allows fast generation of problems and hence keeps the efficiency of the system. Hence, in this paper a strategy that utilizes template based shallow NLP technique to generate math word problem and equation for Amharic language is presented.

3. Preliminaries

In this section basic concepts relevant to the realm of this paper are presented.

3.1. AMW Problems

As mention above, math word problem presents a real word story using textual sentences for learning basic arithmetic operations. It integrates the normal mathematical skill to the daily life situation.

Math word problem using Amharic language is common across elementary schools in Ethiopia. **Example 3-1** provides a sample math word problem given in Amharic language with English translation alongside.

Example 3-1. [Sample AMW problem]

> "በፀጋው 7 ብርቱኳኖች አሉት። በእውነት 3 ተጨማሪ ብርቱኳኖችን ሰጠችው። በፀጋው በአጠቃላይ ስንት ብርቱኳኖች ይኖሩታል?" ["Betsegaw has 7 oranges. Bewunet gave him 3 more oranges. How many oranges does Betsegaw has now?".]

In this example, the arithmetic operation addition is presented behind a text. Solving such a kind of math word problems requires a comprehension skill of identifying the type of math word problem and the operator involved. There are basically three types of math word problems *i.e.* JOIN, SEPARATE and PART-PART-WHOLE, that characterize most of the addition and subtraction prob-

lems [23]. JOIN problem type is expressed by the equation START – CHANGE = RESULT, SEPARATE with START + CHANGE = RESULT and PART-PART-WHOLE type by PART + PART = WHOLE. JOIN and SEPARATE problem types can be given with START-UNKNOWN, CHANGE-UNKNOWN or RESULT-UNKNOWN. PART-PART-WHOLE problem type can be given as PART-UNKNOWN or WHOLE-UNKNOWN. **Table 1** gives more examples of AMW problems.

Identification of type of the problem and specifically determining the missing variable *i.e.* start, change or result involves challenging steps in NLP for automatic generation of math word problem and equation.

3.2. Automatic Generation of Math Word Problem and Equation

Math word problem generation is the task of producing meaningful and reasonable questions. Strategies to generate questions can be generally classified into template-based [19], syntax-based [20] [21] and semantics-based [22]. Among these, template-based techniques are mostly used for special purpose applications such as math word problem generation. Syntax-based strategies are more effective for short and simple sentences. However, Semantic-based approach utilizes NLP concepts which is theoretically more interesting and practically challenging.

3.3. Existing Amharic Resources

Language processing research works on Amharic language has grown in recent years because of the emergence of a good-sized Part of Speech (POS) tagged corpus and the development of morphological analyzer. The tagged corpus is news corpus from Walta Information Center (WIC) which is manually tagged by the staff member of the Ethiopian Languages Research Center (ELRC). The

Table 1. Math word Problems with their type.

6 ወንዶችና 7 ሴቶች በእጅ ኳስ ቡድን ውስጥ አሉ። በአጠቃላይ ቡድኑ ስንት ልጆችን ይዞአል? [There are 6 boys and 8 girls on the volleyball team. How many children are in the volleyball team?] [(Part + Part = ___) Result-Unknown]
ፈበን 5 እርሳሶች አላት። ስንት እርሳሶችን ብትጨምር በአጠቃላይ 7 እርሳሶች ይኖራታል? [Feben had 5 pencils. How many more pencils does she has to put with them so she has 7 pencils altogether?] [(Start + __ = Result) Change-Unknown]
ሩት 12 ጌጦች አሉአት። 5ቱን ለርብቃ ሰጠች። ሩት ስንት ጌጦች ይቀሩአታል? [Ruth has 12 dimes. She gave 5 dimes to Rebecca. How many dimes does Ruth has now?] [(Start - Change =__) Result-Unknown]
ከበደ ጥቂት ብስኩቶች አሉት። 5 ተጨማሪ ብስኩቶችን አገኘ። አሁን በአጠቃላይ 8 ብስኩቶች አሉት። ከበደ በመጀመሪያ ስንት ብስኩቶች ነበሩት? [Kebede got some cookies. He got 5 cookies more. Now he got 8 cookies. How many did Kebede has in the beginning?] [(__+ Change = Result) Start-Unknown]

corpus consists of 210,000 tokens collected from 1065 news documents [24]. The corpus is used to develop a stemmer [25] and Named Entity recognition [6] among the others. Another important resource is a morphological analyzer called HornMorpho [5]. The performance was tested on 200 randomly selected words and has been reported to have above 95% accuracy. It is described as "the most complete morphological processing tool for Amharic" [26]. The system can be used to analyze, segment and generate words. As a result, it provides POS, morphological and syntactic information related to function words that are attached to the word. The proposed strategy bases these resources for Automatic generation of AMW problem and equation.

4. Automatic Generation of AMW Problem and Equation

The proposed architecture of AMW problem and equation generator engine is shown in Figure 1. The engine involves two phases to generate AMW problem and equation: *template formation* and *generation of AMW problem and equation*. The first phase forms template using sample AMW problem provided from user. Template formation involves NLP tasks such as segmentation, tokenization and POS tagging together with equation formulation and template formation. The generator subcomponent in the engine uses formed template to generate AMW problem and equation in the second phase. Problem generation involves retrieval of semantically equivalent concepts from Amharic WordNet to generate new problems having a similar pattern given by generated template. After all, the *Equation Solver* subcomponent in the engine solves the equation and puts back the answer together with the necessary steps to a database. The next sub sections explain detail activities in the engine.

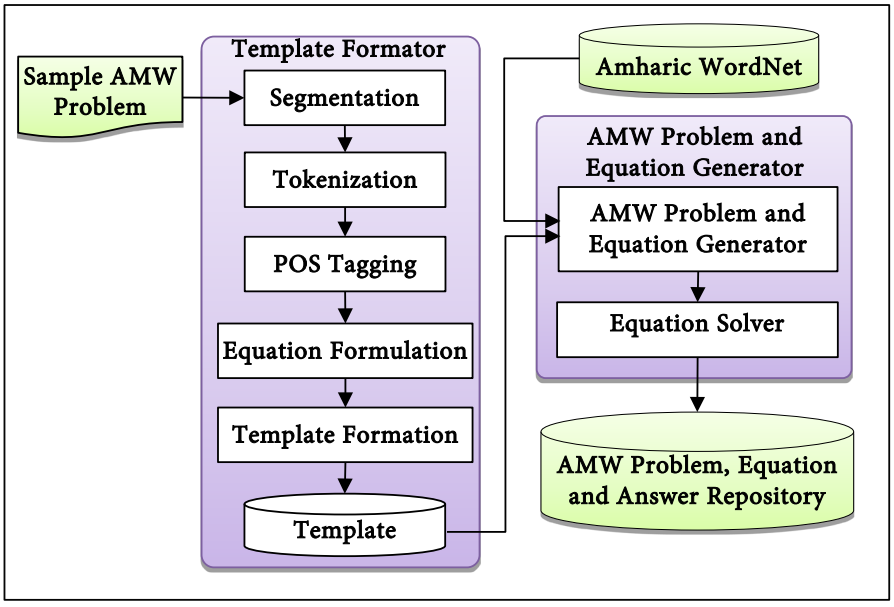

Figure 1. System architecture of the proposed AMW problem and equation generator engine.

4.1. Amharic WordNet

To generate AMW problem, the system needs knowledge base to extract semantically equivalent concepts and substitute in place of placeholders left in a given template. Accordingly, I constructed Amharic WordNet manually as a small knowledge base in which the basic relation between terms is "synonymy". Amharic WordNet is composed of 1090 single word terms (all are nouns) grouped into 357 synsets (synonym groups) and these synsets are representations of the concepts of terms in the group. Synsets are further related with each other by other three relations called "type-of", "part-of" and "antonym".

4.2. AMW Problem and Equation Template Formation

As mentioned above, the first stage starts when a sample AMW problem is provided as a user input to the engine. Then the template format or component involves a five-step process to form template. NLP tasks such as segmentation, tokenization, and POS tagging on given sample AMW problem are the first three steps to be carried out followed by equation formulation and finally template formation. Equation formulation task is vital and is in fact a core task for template formation as well as problem generation.

As part of equation formulation, a given sample AMW problem is identified for its problem type based on the types *i.e. join*, *separate and part-part-whole* problem as presented in Section 3.1 of this document. Each of these math word problem types has sub types. For *join* and *separate* problem category, the problem can be given as *Result-Unknown*, *Change-Unknown*, or *Start-Unknown*. And for *part-part-whole* problem, the problem can be given as *Result-Unknown* or *Part-Unknown*. Identifying the missing variable from a given sample problem is another challenging task in the engine for a successful template formation. The first stage is completed when a template is formed and put to a database. Next, the five steps in this phase are discussed.

STEP 1: Segmentation of AMW Problem

Segmentation is a process of extracting individual sentences in a given AMW problem. Assuming a given sample AMW problem has a valid Amharic sentence structure, the segmentation process produces a set $S = \{S_1, S_2, S_3, \cdots, S_n\}$, where S_i is individual sentence that make up the given AMW problem as explained by **Example 4-1**. This set is input to the next step of tokenization.

Example 4-1. [AMW problem segmentation]

Referring to AMW problem given in **Example 3-1**, the segmentation task produces a sentence set $S = \{S_1, S_2, S_3\}$ containing three sentences and is given below: -

S_1 = {በፀጋው 7 ብርቱኳኖች አሉት።} {Betsegaw has 7 oranges.},
S_2 = {በእውነት 3 ተጨማሪ ብርቱኳኖችን ሰጠችው።} {Bewenet gave him 3 more oranges.} , and
S_3 = {በፀጋው በአጠቃላይ ስንት ብርቱኳኖች ይኖሩታል?} {How many oranges does Betsegaw has now?}

STEP 2: Tokenization of AMW Problem

Tokenization is the process of breaking a given sentence into words or other meaningful elements called tokens. The list of tokens becomes input for the POS tagger in the template formator component. A segmented sentence set S produced as a result of segmentation process is input for tokenization process. The output of tokenization step is a set $S_T = \{S_{T1}, S_{T2}, \cdots, S_{TN}\}$, where every S_{Ti} contains tokens from S_i of segmented sentence set S. **Example 4-2** gives the explanation.

Example 4-2. [Tokenization]

Referring to a sentence set $S = \{S_1, S_2, S_3\}$ given in **Example 4-1**, the tokenizer produces a tokenized sentence set $S_t = \{S_{t1}, S_{t2}, S_{t3}\}$, where each element contains a collection of tokens from sentence elements S_1, S_2, *and* S_3 respectively of a sentence set S as given below.

S_{t1} = {በፀጋው, 7, ብርቱኳኖች, አሉት, ።} {Betsegaw, has, 7, oranges, .}
S_{t2} = {በእውነት,3, ተጨማሪ, ብርቱኳኖችን, ሰጠችው, ። } {Bewenet , gave, him, 3, more oranges, . }
S_{t3} = {በፀጋው, በአጠቃላይ, ስንት, ብርቱኳኖች, ይኖሩታል,?} {How, many, oranges, does, Betsgaw, has, now, ?}

STEP 3: POS Tagging of AMW Problem

The third step involves part of speech tagging, which is the process of marking up a word in a given sentence as corresponding to a particular part of speech based on both its definition as well as its context. HornMorpho [5], which is an Amharic POS tagger has been used for the tagging task.

A tokenized sentence set S_t produced as a result of tokenization step is input to the POS tagger. The output of the POS tagger is tagged sentence set $S_{TG} = \{S_{TG1}, S_{TG2}, \cdots, S_{TGn}\}$, where every S_{TGi} contains POS tagged tokens from S_{ti} of a tokenized sentence set S_t as explained by **Example 4-3**.

Example 4-3. [POS Tagging of AMW problem]

Referring to tokenized sentence set S_t in **Example 4-2**, the part of speech tagger produces a set of POS tagged sentences $S_{TG} = \{S_{TG1}, S_{TG2}, S_{TG3}\}$ as given below: -

S_{TG1} = {"(በፀጋው, NNP), (7, CD), (ብርቱኳኖች, NNP), (አሉት, VBZ), (::)} [{(Betsegaw, NNP),(has,VBZ), (7,CD), (oranges,NNP)}]
S_{TG2} = {(በእውነት, NNP), (3, CD), (ተጨማሪ, NNP), (ብርቱኳኖችን, NNP), (ሰጠችው, VBD), (::)} [{(Bewenet, NNP), (gave, VBD), (him, PRP), (3, CD), (more, NNP), (oranges, NNP)}]
S_{TG3} ={(በፀጋው, NNP), (በአጠቃላይ, VBZ), (ስንት, JJ), (ብርቱኳኖች, NNP), (ይኖሩታል, VB), ('?')} [{(How, WRB), (many, JJ), (oranges, NNS), (does, VBZ), (Betsegaw, NNP), (has, VB), (now, RB), (?,)}]

STEP 4: Equation Formulation

The prior task in equation formulation is problem type classification. Problem

type classification takes a POS tagged set of sentences S_{TG} from POS tagging step as input and classify it to one of the three problem categories *i.e. Join*, *Separate*, or *Part-Part-Whole* problem. Then identification of the UNKNOWN variable is made *i.e. Start-Unknown*, *Change-Unknown*, and *Result-Unknown* for *Separate* and *Join* types and *Part-Unknown* or *Whole Unknown* for *Part-Part-Whole* type of problem. A schematic representation of the *Equation Formulation* in AMW problem template formation is given in Figure 2.

Join and separate problem types can be given in a variety of forms as a result of varying discourse structure in phrasing constituent sentences in many ways. The constituent sentences can either be separate sentences separated with a dot or joined using a conjunction or could be formed as a complex sentence with the use of conditionals. First, the classified problem type is further identified for the missing variable (whether it is *Start-Unknown*, *Change-Unknown*, *or Result-Unknown*) by examining individual sentence if it holds value for the start, change or result. Then follows operator prediction (whether the problem calls for addition or subtraction). The last task is to combine the results of the first two stages and produce the corresponding AMW equation.

Once the UNKNOWN variable is identified the information in each sentence is extracted. For join and separate problem type, equation formulation involves constructing an equation of the form (*quantity in Start*) ± (*quantity in Change*) = *Result.* The quantity in the sentence containing change value holds the sign of the equation (depending on whether it is addition or subtraction). If a sentence with no numerical information is classified as Start, Change or Result, we assign an X to that sentence and the information is excluded from the equation.

The analogy also applies for the part-part-whole problem equation formulation. With its sentences classified as *Part* or *Whole*, we proceed to the equation formulation as follows. When the *Part* sentence has more than one numerical quantity, the first number is assigned as *Part*1 and the other number as *Part*2 (or

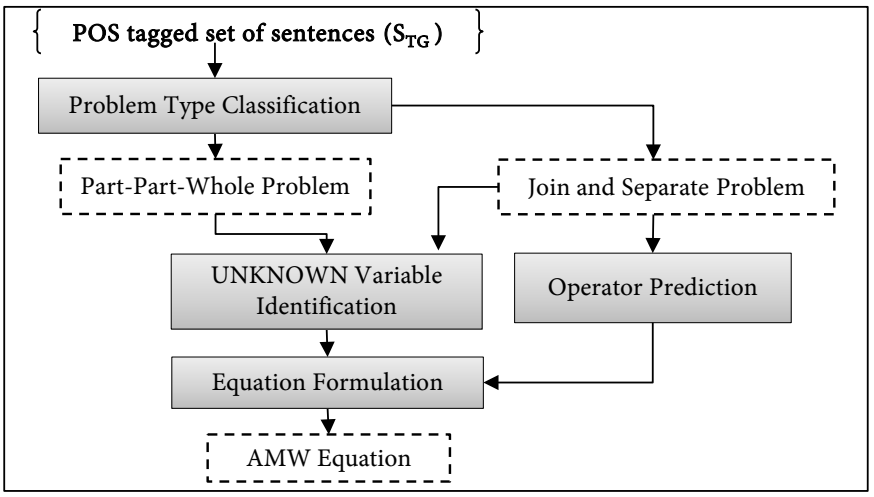

Figure 2. Schematic representation of AMW equation formulation.

into more buckets as the case may be). Then, include them into the corresponding equation as: *Part*1 + *Part*2 = *Whole*.

Example 4-4. [Equation formulation]

Referring to a POS tagged sentence set S_{TG} containing three sentences from **Example 4-3**, the problem type classification process categorizes it to JOIN type AMW problem with a *Start* and a *Change* numbers are included in S_{TG1} and S_{TG2} respectively and the third sentence, S_{TG3}, holds that the *Result* is UNKNOWN. Moreover, a dependency graph indicates that a second person with a verb "gave" has association with the first person mentioned in the first sentence referred here in the second sentence with a personal pronoun "him". Tracing this, the operator prediction identified that the operator involved is *addition*. And hence, the equation is formulated as $7+3=X$.

STEP 5: Template Formation

Here in template formation process, whose algorithm is given by Pseudo Code 1, those proper nouns and numerals in each sub element S_{TGi} of POS tagged sentence set S_{TG} are set with placeholder values for later substitution with real values in problem generation phase. In addition, the numbers in the formulated equation are replaced by placeholders for later generation AMW equation. The overall template provides three pieces of information: AMW problem, the type of problem,

Pseudo Code 1. Algorithm for template formation.

```
    Input:
1:  POSSentence: Array                  // POS tagged set of sentences
2:  FormulatedEQN: String               // Formulated Equation
3:  TypeOfProblem: String               // Type of problem
    Intermediete:
4:  Word: Array                         // Part of a sentence
5:  IndvSente: Array                    // Individual sentence template
6:  Template: String                    // Intermediate template sentence
    Output:
7:  AMWTemplate: String                 // AMW problem template
8:  Begin:
9:      For i = 0 to sizeOf (POSSentence)
10:         IndvSente = POSSentence[i]
11:             For j = 0 to sizeOF (IndvSente)
12:                 Word = SubElement[j]
13:                 If Word[1] = 'NNP' Then
14:                 Word[0] = append ('AMHARICNAME' + j)
15:                 Else if Word[1] = 'CD' Then
16:                 Word[0] = append ('NUMBER' + j)
17:                 Else if j > 0 AND Word[1] = 'NNP' Then
18:                 Word[0] = append ('ITEMNAME' + j)
19:                 Template = append (Template, Word)
20:             End For
21:         AMWTemplate = append (AMWTemplate, Template)
22:     End For
23: AMWTemplate = append (AMWTemplate, FormulatedEQN)
24: AMWTemplate = append (AMWTemplate, TypeOfProblem)
25: Return (AMWTemplate)
    End
```

and the equation generated as formally given by **Definition 4-1** and **Example 4-5** presents a working template example taken from the developed prototype.

Definition 4-1. A template T is a tuple (S, P, E) where:

- S is a problem statement with placeholders,
- P is problem type *i.e.* SEPARATE, JOIN or PART-PART-WHOLE with a specific UNKNOWN variable, and
- E is formulated equation.

Example 4-5. [Template formation]

Taking a POS tagged set of sentence $S_{TG} = \{S_{TG1}, S_{TG2}, S_{TG3}\}$, problem type as JOIN-RESULT-UNKNOWN and formulated equation START + CHANGE = *X*, a final template is given as: -

> ["[AmharicName1(በፀጋው)] [Number1] [ItemName1(ብርቱካን)] አሉት። [AmharicName2(በእውነት)] [Number2] ተጨማሪ [ItemName1(ብርቱካን)] ሰጠችው ። [AmharicName1(በፀጋው)] በአጠቃላይ ስንት [ItemName1(ብርቱካን)] ይኖሩታል?], [JOIN-RESULT-UNKNOWN], [Number1+Number2=X]
>
> [[AmharicName1(Betsegaw)] has [Number1] [ItemName1(Oranges)]]. [AmharicName2(Bewenet)] gave him [Number2] more [ItemName1(Oranges)]]. How many [ItemName1(Oranges)] does [AmharicName1(Betsegaw)] has now?], [JOIN-RESULT-UNKNOWN] [Number1+Number2=X]

4.3. AMW Problem and Equation Generation and Solving

AMW problem and equation generation is made using template formed in the first phase. The generator subcomponent reads template from database and starts AMW problem and equation generation. This process of generating AMW problem and equation followed by the solver sub component which solves the equation. Finally, the engine writes back generated word problem together with the answer back to a database. The coming sections gives detail explanation of this phase as a two-steps process.

STEP 1. AMW Problem and Equation Generation

The AMW problem generator subcomponent reads a template from database and starts to produce AMW problem and equation using the algorithm given in Pseudo Code 2. Once more, a template has got placeholders where real values can be set by the problem generator sub component. **Example 4-6** explains problem and equation generation process.

Example 4-6. [AMW Problem and Equation Generation]

Referring to a working template example given in **Example 4-5**, the problem generator sub component retrieves semantically related values from Amharic wordnet to substitute for placeholders left in the template *i.e. AmharicName*1, *ItemName*1, *AmharicName*2, and *ItemName*1. And give random number values for *Number*1 and *Number*2. For *AmharicName*1 and *AmharicName*2, the system retrieved other Amharic names "አማኑኤል (Amanuel)" and "ሩት (Ruth)" from Amharic WordNet. Similarly, semantically related word "bananas" is used in

place of ItemName1 = "oranges". A complete AMW problem generated taken from developed prototype is given below:

["አማኑኤል 9 ሙዞች አሉት። ሩት 4 ተጨማሪ ሙዞችን ሰጠችው። አማኑኤል በአጠቃላይ ስንት ሙዞች ይኖሩታል? [9+4=X]] [Amanuel has 9 bananas. Ruth gave him 4 more bananas. How many bananas does Amanuel has now? [9+4=X]]

Pseudo Code 2. Algorithm for AMW problem and equation generation.

```
    Input:
1:  AMWTemplate: String              // AMW Problem and Equation template
    Intermediete:
2:  AMName: Array                    // part of a sentence
3:  SegAMWTemp: Array                // Segmented template
4:  SubGenProb: String               // part of a problem
5:  repArray: Array                  // real values to replace placeholders in a template
    Output:
6:  AMWProblemEQN: Array             // generated AMW Problem and Equation
    Begin:
7:  SegAMWTemp=segment (AMWTemplate, '::') //segment template at '::'
8:  For i = 0 to sizeOf (SegAMWTemp)
9:      AmharicName = getName() // get a Amharic name from knowledge base
10:     randomNumber = generateRandNumber() // get random number
11:     ItemName = getItmeName() //refer Amharic Wordnet for similar concept
12:     repArray[i].add (AmharicName, randomNumber, ItemName)
13:     SubGenProb = replace (segAMWTemp[i], repArray[i]) // replace all
                  those texts in segAMWTemp with elements of repArray
14:     AMWProblemEQN = append (AMWProblemEQN, SubGenProb)
15: End For
16: Return (AMWProblemEQN)
    End
```

STEP 2: Equation Solving

The equation solver subcomponent is responsible to solve a given equation whose template is formed in template formation phase with real values substituted later in problem generation phase. Numpy [27] is used to solve the equation.

Finally, after the equation is solved, the engine writes back generated AMW problem together with the equation with the answer to database which holds AMW problem and equation with answer. This operation can be called repeatedly to generate multiple AMW problems and equation. The next section presents developed prototype used for experimentation platform.

5. AMW Problem and Equation Generator Prototype

To demonstrate the validity of the proposed automatic AMW Problem and equation generation, I have developed a desktop-based prototype using java programming language. The prototype is used as a testing platform to test the performance of the proposed strategy. The next sub sections present architecture of the prototype and user interfaces consecutively.

5.1. Architecture of AMW Problem and Equation Generator Prototype

The architecture of the proposed prototype for AMW problem and equation generator engine is shown in Figure 3. It contains interface component that allows to define sample AMW problem other than the main engine components as discussed in Section 4 of this document.

5.2. Graphical User Interfaces

I have developed a set of Graphical User Interfaces to facilitate the interaction between a user and the proposed system. The interfaces include sample AMW problem definition interface and an interface to view list of generated AMW problems with their equation formulated as presented next.

1) Sample AMW Problem Definition and Management Interface

The sample AMW problem definition interface shown in Figure 4 is used to register sample AMW problems to the system. The system uses user provided sample AMW problems to form templates as well as generate AMW problems and equations.

2) Generated AMW Problem and Equation Displaying Interface

After the formation of AMW problem and equation templates from user provided sample AMW problems, the interface shown in Figure 5 is used to display generated AMW problems with a corresponding equation.

6. Result and Discussion

The overall performance of the proposed approach is dependent on the performance of problem type classifier used as part of equation formulation. Hence,

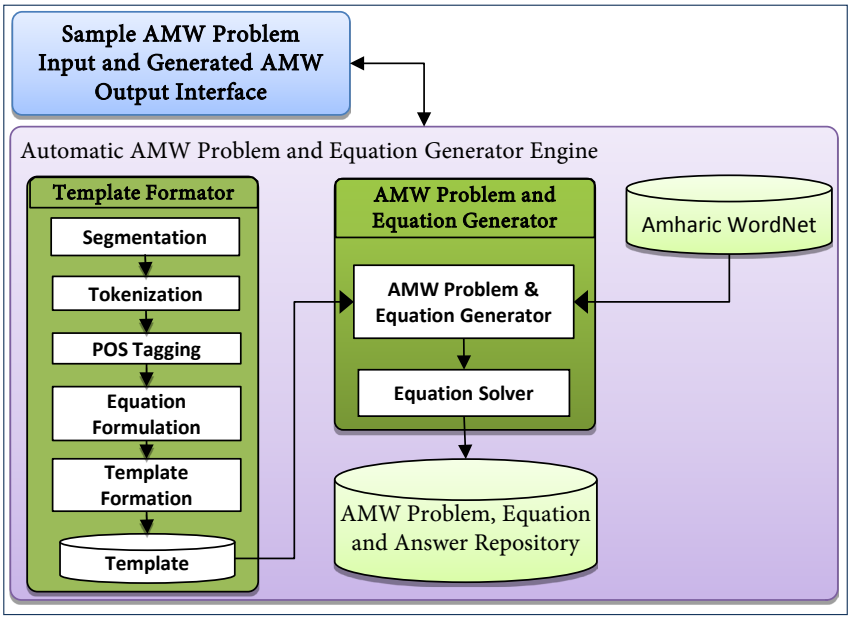

Figure 3. System architecture of the proposed AMW problem and equation generator prototype.

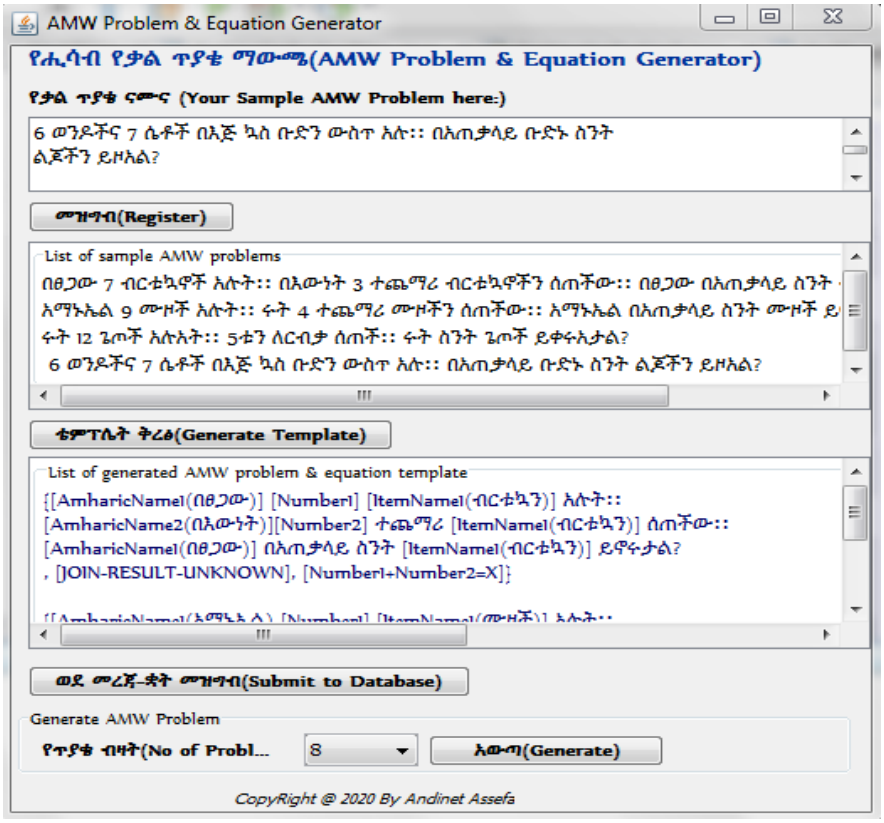

Figure 4. Sample AMW problem definition and management interface.

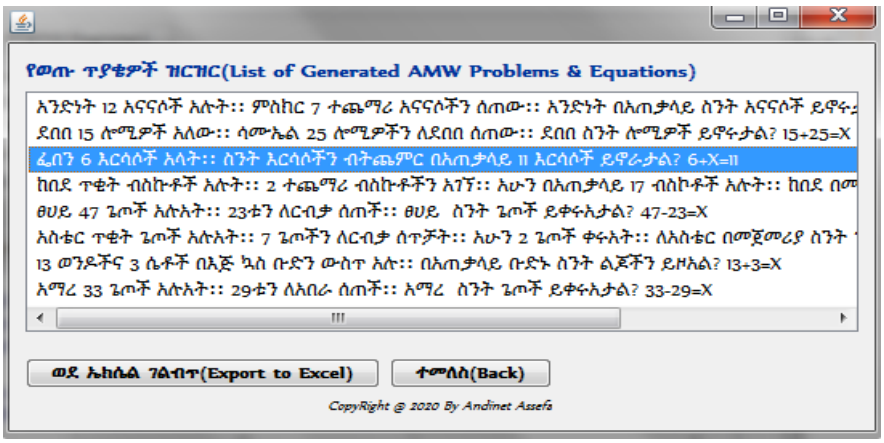

Figure 5. Generated AMW Problem and Equation displaying interface.

the first experiment focus on evaluating the performance of problem type classifier. For this experiment a corpus containing 154 AMW problems collected from elementary school mathematics textbooks and other collected school worksheets is used. In addition to evaluating a classifier's performance, an experiment was conducted to assess the relatedness of auto generated AMW problem with human prepared questions. The third experiment was conducted to check how solvable are auto generated AMW problems compared with human solver and finally the fourth experiments was conducted to check whether generated

AMW problems are grammatically correct, if there is a sound subject verb agreement and flow of idea in generated sentences subjected to human judges.

6.1. Problem Type Classification

The problem type classifier as part of equation formulation categorizes a given POS tagged AMW problem into JOIN, SEPARTE or PART-PART-WHOLE types. It plays crucial role for the accuracy of equation formulation as well as AMW problem generation. The accuracy of problem type classifier is tested on a corpus containing 154 AMW problems. Among the total problems from the corpus, JOIN types are 64, SEPARATE types are 57, PART-PART-WHOLE types are 25 and 8 of them were neither of these. From the corpus, the classifier collected 61 JOIN type, 52 SEPARATE type, 24 PART-PART-WHOLE type problems and reported 17 of them as invalid types of problem as summarized by Table 2. Accordingly, the classifier yielded an accuracy of 95.32% on identification of JOIN types, 91.23% on SEPARATE types, and 96% on PART-PART-WHOLE types and an overall performance of 93.84%.

6.2. Relatedness of Auto Generated AMW Problems to Human Prepared Problems

The second experiment focus on checking how realistic are auto generated AMW problems. Therefore, in this experiment, 40 sample elementary school students were chosen and given a mixed type of auto generated and human prepared AMW problems on the same sheet. 30 AMW problems; 15 auto generated and 15 human prepared problems from a given same sample questions were given to the students. After the conducting the exam, the exam sheets were evaluated out of 15 for both auto generated as well as human prepared. And the average score that students attain for human prepared problems were 12.5 and 11.28 for auto generated AMW problems by the proposed system as depicted by Figure 6. Accordingly, the students have shown an average performance of 83.34% for human prepared questions and 75.2% for auto generated questions. This shows the proposed approach has 90.24% relatedness efficiency to generate AMW problems compared to human.

6.3. Comparison of Auto Solved AMW Problems with Human Answers

The third experiment was conducted to check the validity of answers generated by the proposed approach and how solvable is it. For this experiment 30 auto

Table 2. Classifier performnace on a given AMW problem corpus.

	Join	Separate	Part-Part-Whole
Number of problems	64	57	25
Accurately Classified	61	52	24
	95.32%	91.23%	96%

generated AMW problem were given to human solver and the result is compared to auto generated answer. There was a variation on three questions out of the total 30 questions, which is a 10% variation due to classifying it to invalid problem type as a result of inappropriate punctuation of sentences. This shows that auto generated AMW problems are 90% solvable. Figure 7 shows a graphical presentation of the result of comparison.

6.4. Human Rating of Generated Amharic Math Word Problem/Answer

The fourth experiment was conducted to check generated Amharic math word problems for grammatical correctness, subject verb agreement, coherence in the sentences and the validity of the answer generated. For this experiment, 50 auto generated Amharic math word problem with answers were given to 30 randomly selected mathematics teachers for their rating on each problem and answer. The rating scale was with a minimum value of 1 and a maximum value of 5. Figure 8 shows the result of the rating. The average rating was found to be 4.52 for the word problems and 4.23 for the answers, which indicates that the proposed approach has exhibited 90.4% efficiency for auto generation of Amharic math word problems and 84.6% efficiency on generating equations and solving problems, which results with a combined efficiency of 87.5% subjected to human judges.

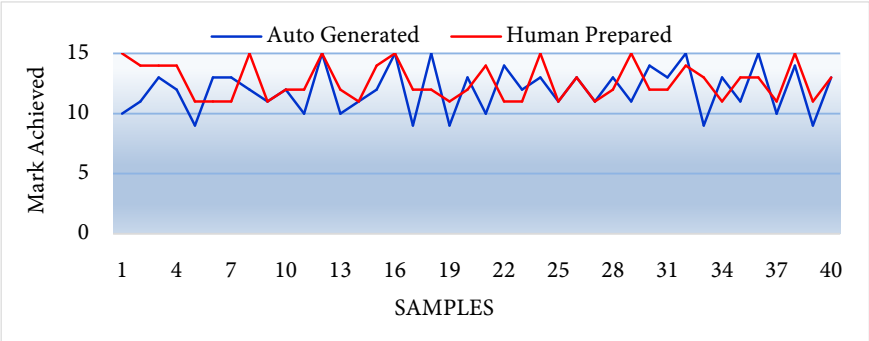

Figure 6. Comparison of auto generated AMW problems with human prepared problems.

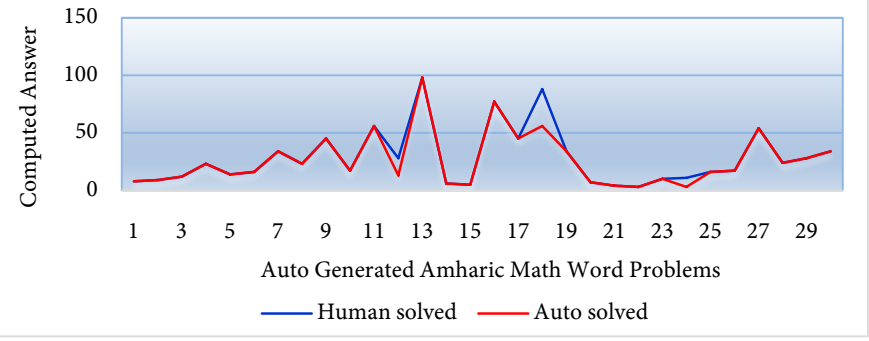

Figure 7. Comparison of human solved and auto solved AMW problems.

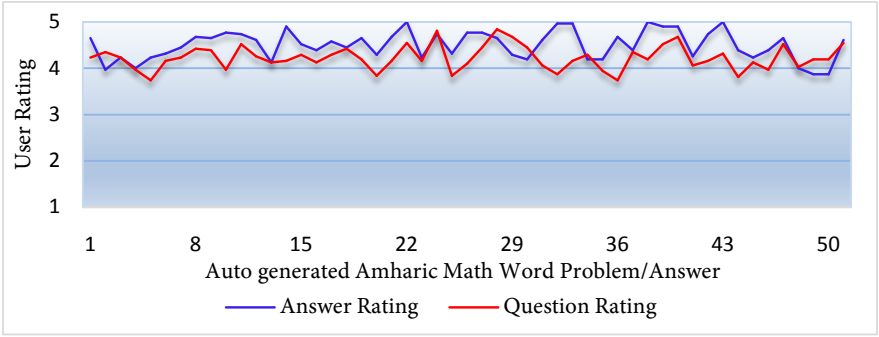

Figure 8. Human Rating of Auto generated AMW problem/Answer.

7. Conclusion

Math word problem is an essential component of learning in mathematics. Students need to get fresh and more math word problems for a successful development of problem-solving skill as well as improve their reading comprehension. Automatic generation of math word problems is proofed to be successful to provide fresh and volume of exercises as per each individual student's needs. Hence, quite a few research works are proposed on automatic generation of math word problem and equation for English language and for other very few languages. Amharic is a language spoken by more than hundred million of Ethiopians and is a medium of instruction for primary schools in Ethiopia. Taking this opportunity to address such a vast amount of population, in this paper I have proposed an automatic generation of AMW problem and equation using template based shallow NLP approach. The proposed approach accepts sample AMW problem to form template which is used for AMW Problem generation. To validate the proposed approach, three experiments were conducted. Experimental results show that the proposed approach has 93.84% overall efficiency on formulating AMW problem and equation template. Experiment has also shown a 90.24% relatedness to human generated AMW problems and has also proofed to generate solvable problems as it has shown a 90% accuracy of giving correct answer compared to answer by human solver. Finally, though the proposed strategy has exhibited a promising performance utilizing template based shallow NLP technique, it has limitations on generating AMW problems with deeper semantic meanings. And hence, this research work opens a venue for further research works on the use of deeper NLP techniques to explore more efficient strategies.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Word Problem. https://en.wikipedia.org/wiki/Word_problem_(mathematics_education)

[2] Nandhini, K. and Balasundaram, S.R. (2011) Techniques for Generating Math Word

Questions for Learning Disabilities. *IJCA Proceedings on International Conference and Workshop on Emerging Trends in Technology* (*ICWET*), No.4, 45-50.

[3] Amharic Language. http://www.lonweb.org/link-amharic.htm

[4] Dehadari, J., Tounsi L. and Genabith, J. (2011) Morphological Features for Parsing Morphologically-Rich Languages: A Case of Arabic. *Proceedings of the* 2*nd Workshop on Statistical Parsing of Morphologically Rich Language* (*SPMRL* 2011), Dublin, Ireland, October 2011, 12-21,

[5] Gasser, M. (2011) HornMorpho: A System for Morphological Processing of Amharic, Oromo, and Tigrinya. *Conference on Human Language Technology for Development*, Alexandria, 2-5 May 2011, 94-99.

[6] Alemu, B. (2013) A Named Entity Recognition for Amharic. MA Thesis, Addis Ababa University, Addis Ababa.

[7] Williams, S. (2011) Generating Mathematical Word Problems. *Association for the Advancement of Artificial Intelligence* (AAAI) *Fall Symposium on Question Generation*, Arlington, Virginia, 4-6 November 2011, 61-64.

[8] Ahmed, U.Z., Gulwani, S. and Karkare, A. (2013) Automatically Generating Problems and Solutions for Natural Deduction. *Proceedings of the* 23*rd International Joint Conference on Artificial Intelligence*, Beijing, 3-9 August 2013, 1968-1975.

[9] Singh, R., Gulwani, S. and Rajamani, S. (2012) Automatically Generating Algebra Problems. Proceedings of 26*th AAAI Conference on Artificial Intelligence*, Toronto, 22-26 July 2012.

[10] Andersen, E., Gulwani, S. and Popovic, Z. (2013) A Trace-Based Framework for Analyzing and Synthesizing Educational Progressions. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, April 2013, Paris, 773-782. https://doi.org/10.1145/2470654.2470764

[11] Sadigh, D., Seshia, S.A and Gupta, M. (2012) Automating Exercise Generation: A Step towards Meeting the MOOC Challenge for Embedded Systems. *Proceedings of the Workshop on Embedded and Cyber-Physical Systems Education*, October 2012, Tampere, 1-8. https://doi.org/10.1145/2530544.2530546

[12] Fuchs, L. and Fuchs, D. (2007) Mathematical Problem Solving: Instructional Interventions. In: Berch, D.B. and Mazzocco, M.M., Eds., *Why Is Math So Hard for Some Children*? *The Nature and Origins of Mathematical Learning Difficulties and Disabilities*, Paul H. Brookes Publishing, Baltimore, 397-414.

[13] Zhou, Q.Y. and Huang, D.Q. (2019) Towards Generating Math Word Problems from Equations and Topics. *Proceedings of the* 12*th International Conference on Natural Language Generation*, Tokyo, 28 October-1 November 2019, 494-503. https://doi.org/10.18653/v1/W19-8661

[14] Li, T. and Sambasivam, S. (2020) Automatically Generating Questions in Multiple Variables for Intelligent Tutoring. *Issues in Information Sciences and Information Technology*, 2, 471.

[15] Andrenucci, A. and Sneiders, E. (2005) Automated Question Answering: Review of Main Approaches. *Proceedings of* 3*rd International Conference on Information Technology and Applications*, Sydney, 4-7 July 2005, 514-519.

[16] Chiang, T. and Chen, Y. (2019) Semantically-Aligned Equation Generation for Solving and Reasoning Math Word Problems. *Proceedings of the* 2019 *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, June 2019, 2656-2668. https://doi.org/10.18653/v1/N19-1272

[17] Teshome, W. (2013) Designing Amharic Definitive Question Answering, MA Thesis, Addis Ababa University.

[18] Yimam, S.M. (2013) TETEYEQ: Amharic Question Answering for Factoid Questions, MA Thesis, Addis Ababa University, Addis Ababa.

[19] Mostow, J.and Chen, W. (2009) Generating Instruction Automatically for the Reading Strategy of Self Questioning. *Proceedings of the* 14*th International Conference on Artificial Intelligence in Education*, Brighton, 6-10 July 2009, 465-472.

[20] Wyse, B. and Piwek, P. (2009) Generating Questions from Open Learn Study Units. *Proceedings of the* 2*nd Workshop on Question Generation*, Brighton, 6-10 July, 66-73.

[21] Sag, I.A. and Flickinger, D. (2008) Generating Questions with Deep Reversible Grammars. *Proceedings of the* 1*st Workshop on the Question Generation Shared Task and Evaluation Challenge, Arlington*.

[22] Heilman, M. and Smith, N.A. (2009) Question Generation via over Generating Transformations and Ranking. Technical Report, Language Technologies Institute, Carnegie Mellon University, Pittsburgh.

[23] Types of Math Word Problem. http://www.homeofbob.com/math/numVluOp/wholeNum/addSub/adSubTypsChrt.html

[24] Demeke, G.A. and Getachew, M. (2006) Manual Annotation of Amharic News Items with Part-of-Speech Tags and Its Challenges. Ethiopian Languages Research Center Working Papers, 1-16.

[25] Alemu, A.A. and Lars, A. (2007) An Amharic Stemmer: Reducing Words to their Citation Forms. *Proceedings of the* 2007 *Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, Prague, June 2007, 104-110. https://doi.org/10.3115/1654576.1654594

[26] Gamback, B. (2012) Tagging and Verifying an Amharic News Corpus. 8*th International Conference on Language Resources and Evaluation* (*ELRA*), *Workshop on Language Technology for Normalization of Less Resourced Languages*, Istanbul.

[27] Oliphant, T.E. (2006) Guide to NumPy. Trelgol Publishing.