Amharic Open Information Extraction with Syntactic Sentence Simplification

Seble Girma(✉) and Yaregal Assabie

Department of Computer Science, Addis Ababa University, Addis Ababa, Ethiopia
yaregal.assabie@aau.edu.et

Abstract. Open Information Extraction (OIE) is the process of discovering domain-independent relations from natural language text. It has recently received increased attention and been applied extensively to various downstream applications, such as text summarization, question answering, and informational retrieval. In this paper, we propose a method of OIE for Amharic language. To improve the performance of relation extraction, the proposed OIE method implements a sentence simplification technique that breaks down complex and compound sentences into simple sentences. Linguistic rules are utilized to extract domain-independent and unanticipated relation instances with their arguments from simple sentences. The proposed method and algorithms are implemented and evaluated with a dataset from different domains. Test results show that the system achieved an overall precision of 0.88.

Keywords: Open Information Extraction · Chunking · Sentence Simplification · Relation Extraction

1 Introduction

Information Extraction (IE) is the task of automatically extracting structured information from text. The core task of IE systems is to identify entities and relationships expressed using natural languages. However, the traditional paradigm of IE requires either hand-tagged training examples for each target relation or pre-specified relations along with hand-crafted extraction rules as input. As such inputs are specific to the target domain, shifting to a new domain requires extensive human involvement in creating new extraction patterns [1]. Thus, traditional IE systems are not portable across domains and do not scale to massive and heterogeneous corpora like the Web where the relations are unanticipated [2]. To overcome these limitations, Open Information Extraction (OIE) has become more strongly suggested. OIE has made possible to process massive text corpora without restriction to extract a certain type of relations and attributes, and without having to require much human effort [3].

The OIE paradigm was introduced by Banko *et al.* [1] aiming to develop domain-independent extractors of information by providing ways to extract unrestricted relational information from text. OIE has several advantages over traditional IE approaches [4, 5].

M. A. Delele et al. (Eds.): ICAST 2020, LNICST 384, pp. 457–469, 2021.
https://doi.org/10.1007/978-3-030-80621-7_33

It made easier to extract many kinds of relations without requiring manual labor to build extraction rules and hand-tagged training examples. Because of its ability to extract information for all relations at once without having them named explicitly, it also has a significant scalability advantage over previous IE architectures. Traditional IE systems usually search for entities that are associated with the type of relation which the system was configured to extract whereas an OIE system tries to find relations as well as the entities taking part in those relations which are not predefined. Traditional IE systems require a specific pattern for each relation. On the other hand, OIE systems need a set of patterns that are not related to any specific relation, and these features are useful to extract relations of any nature.

In recent years, a variety of OIE systems have been developed. However, most systems have been designed, implemented and evaluated predominantly for English language. Amharic language, in a variety of respects, has different linguistic structures from other languages like English. To the best of our knowledge, no research works have been done on Amharic OIE yet. This paper presents an OIE for Amharic text which extracts domain-independent relations from Amharic text.

The remaining part of this paper is organized as follows. Section 2 presents research and development works in OIE. The proposed solution is presented in Sect. 3 and experimental results are discussed in Sect. 4. Finally, we make our conclusion in Sect. 5.

2 Related Work

OIE is the task of extracting relations with their corresponding arguments from natural language text. Some of OIE systems which are developed for different natural language are reviewed below.

Etzioni *et al.* [1] introduced TextRunner, an OIE system that trained a Naïve Bayes classifier with POS and NP-chunk features to extract relationships between entities. The system has used a small set of handwritten rules to heuristically label training examples from the Penn Treebank. TextRunner was evaluated using a test corpus of 9 million Web documents and it obtained 7.8 million tuples. A set of 400 randomly selected tuples were evaluated by human reviewers and 80.4% were considered correct.

Wu and Weld [6] presented an OIE system called WOE which used Wikipedia as a source of training data. The WOE system generates relation-specific training examples by matching Wikipedia Infobox content with corresponding patterns. WOE can learn two kinds of extractor: *WOEparse* and *WOEpos*. WOEparse learns from dependency path patterns, and WOEpos is trained with shallow features like POS tags. Comparing with TextRunner, WOEpos runs at the same speed, but achieves an F-measure which is between 18% and 34% greater. WOEparse achieves an F-measure which is between 72% and 91% higher than that of TextRunner but runs about thirty times slower due to the time required for parsing.

Etzoni *et al.* [7] presented ReVerb, which aimed to prevent incoherent and uninformative extractions errors from TextRunner. To eliminate incoherent extractions and to reduce uninformative extractions, the system is designed to capture relation phrases expressed by a Verb-Noun combination that satisfies their pre-defined syntactic and lexical constraint. It is reported that ReVerb achieves an AUC (area under Precision-Recall curve) twice as big as TextRunner and WOEpos, and 38% greater than WOEparse.

Aiming to improve OIE by covering a larger number of relation expressions and expanding OIE representation to allow additional context information such as attribution and clause modifiers, Mausam *et al.* [8] presented the system OLLIE. OLLIE uses the output of OIE systems to bootstrap learning of the relation patterns and then additionally applies lexical and semantic patterns to extract relations that are not expressed through verb phrases. It is reported that OLLIE extracts up to 146 times as many extractions than ReVerb and it obtains 1.9 to 2.7 times more area under precision yield curves compared to ReVerb.

Del Corro and Gemulla [9] presented a clause-based approach implemented in ClausIE. For each input sentence, ClausIE first computes the dependency parsing of the sentence and then determines a set of clauses using the dependency parsing. Next, for each clause, it determines the set of coherent derived clauses based on the dependency parsing and finally it generates propositions from the coherent clauses. Handcrafted rules utilizing the dependency structure of a sentence are used. It is reported that ClausIE achieves better precision than Reverb. ClauseIE's accuracy relies on the dependency parser used for parsing.

OIE systems for languages other than English also have been implemented. Gamallo *et al.* [10] presented a multilingual system, named DepOE that uses the heuristic strategy to perform unsupervised extraction of triples using a rule-based analyzer and dependency parser to extract relations represented in English, Spanish, Portuguese, and Galician. The authors reported that accuracy of 68% is reached, while ReVerb reaches 52% accuracy for the same dataset.

Tseng *et al.* [11] presented a Chinese OIE called CORE that adopt existing Chinese text analyzing approaches to identify the main relation in a given sentence. It is reported that CORE yields relatively promising F1 scores than Reverb.

Nam *et al.* [3] presented a Korean OIE system called SRDF. The SRDF system is designed to extract triples from Korean natural language text based on the use of singleton property and other NLP techniques such as part-of-speech tagging and chunking. SRDF enables extracting multiple numbers of triples from a single sentence via reification. It is reported that the system achieves 81% precision, 86% recall, and 83% F-score for detecting relation and 66% precision, 65% recall and 65% F-score for generating triples.

In summary, according to the experiment results reported thus far, the rule-based systems achieve better accuracy than data-based systems. The results also show that systems that are based on dependency analysis achieved significantly higher precision and recall than those relying on shallow syntax. However, the shallow feature-based approach is very promising in terms of speed, ease of implementation, and portability to other languages. On the other hand, deep syntactic parsing methods are prone to slow performance and their implementation is not easily available for many languages [8].

Moreover, because of heavy reliance on linguistic tools such as part-of-speech taggers and dependency information as well as immediate lexical information to define patterns or constraints for relations, it is difficult to directly apply the aforementioned methods and techniques to low-resourced languages like Amharic. Moreover, the morphological complexity of Amharic poses unique challenges in the development of NLP applications in general and OIE in particular. Thus, in this work, we propose a design for Amharic that takes the characteristics of the language into consideration.

3 The Proposed Solution

The proposed method for Amharic OIE consists of two main tasks: *Sentence Simplification* and *Relation Extraction.* Sentence Simplification breaks complex and compound sentences down into simple sentences from which relation instances are extracted. Initially, the input sentence is divided into non-overlapping phrases using phrasal chunking which relies on POS and morphological tags of words. Then, the Sentence Simplification component segments the sentence into a number of self-contained simple sentences that are easier to process. Finally, relation instances are extracted in N-ary format from those simplified sentences.

3.1 Sentence Simplification

The structure of senetences affects the effectiveness of relation extraction in texts. Simple sentences have convenient structures for extracting relation instances. On the other hand, complex and compound sentences pose difficulties in the process of relation extraction. Thus, simplification of such sentences helps to improve the performance of OIE. In this work, We have developed a set of simplification rules that segment and paraphrase the input Amharic sentences and generate simpler sentences. Since the resulting sentences might need further simplification, the process of syntactic simplification is structured in a recursive loop. The syntactic simplification loop starts by checking if the input sentence is a simple sentence which is identified by counting the number of verb phrases. If the sentence contains exactly one verb phrase, the sentence will be classified as a simple sentence. If two or more verb phrases joined by coordinate conjunctions are detected, the sentence will be classified as a compound sentence. Otherwise, the sentence will be classified as a complex sentence. The overall process of sentence simplification involves *clause splitting* and *paraphrasing.*

Clause Splitting. Amharic clauses can be of two types: coordinate and subordinate. A compound sentence is composed of two or more independent clauses joined by a coordinating conjunction (i.e., እና/*ʔəna* 'and'/, ወይም/*wäyəm* 'or'/, ግን/g*ən* 'but'/, etc.,).. Semicolon ("፤") and comma ("፣") can also function as conjunctions. On the other hand, Amharic subordinate clauses contain both a subject and a verb, but do not express a complete thought. In Amharic, subordinate clauses are recognizable by affixes attached to the verb. Examples of Amharic subordinate clauses which are derived from the verb መጣ /*mäṭa* 'he came'/ are shown in Table 1. Thus, Amharic clause splitting comprises of *coordinate clause splitting* and *subordinate clause splitting*. A given sentence is iteratively split into clauses until all existing clauses are split.

Coordinate Clause Splitting. The coordinate conjunctions joining the clauses are required to be detected and then the sentence is split at the conjunctions. For instance, the following sentence contains three complete and independent clauses which are joined by semicolons.

> [በኢትዮጵያ እና ኤርትራ መካከል የአየር ትራንስፖርት ተጀምሯል]^{clause1}፤ [ሁለቱም አገራት ኤምባሲዎቻቸውን ከፍተዋል]^{clause2}፤ [በዚሁ ሳምንትም ሁለቱ ሀገራት ዝግ ሆነ የቆየውን ድንበራቸውን ክፍት በማድረግ በየብስ ትራንስፖረት መገናኘት ጀምረዋል]^{clause3}።

Table 1. Examples of Amharic subordinate clauses

Clause	Transliteration	Translation	Affixes
ቢመጣም	*Bimäţam*	although he came	ቢ-…-ም /*bi-…-m*/
እንደመጣ	*ʔəndämäţa*	as he came	እንደ- /*ʔəndä-*/
ካልመጣ	*Kalmäţa*	unless he comes	ካል- /kal-/
ስለመጣ	*Slämäţa*	because he came	ስለ- /*slä-*/
ከመጣ	*Kämäţa*	if he comes	ከ- /*kä-*/
እስኪመጣ	*ʔəskimäţa*	until he comes	እስኪ- /*ʔəski-*/
ሲመጣ	*Simäţa*	when he came	ሲ- /*si-*/
እየመጣ	*ʔəyämäţa*	while he came	እየ- /*ʔəyä -*/
የመጣ	*Yämäţa*	who/that/which came	የ- /*yä -*/

Before splitting the sentence, it is necessary to check if both parts of the sentence are independent clauses. To be considered as an independent clause, they should at least have one verb. Custom created POS tags and morphological information is used for splitting into coordinate clauses. Table 2 shows tags that are created by combining POS and Morphological information.

Table 2. Custom created POS tags and morphological information.

Tags	POS	Morphology
ND	N	Definite or Accusive noun
NPD	NP	Definite or Accusive noun with prefixes
GV	V	Gerundive Verb
IV	V	Imperfective Verb
AV	V	Auxiliary Verb
PV	V	Perfective Verb
PAV	V	Passive Verb
PPV	V	Perfective Passive Verb
IRV	V	Imperfective Relative Verb
PRV	V	Perfective Relative Verb
PPRV	V	Perfective Relative Passive Verb
IRPV	V	Imperfective Relative Passive Verb
PGAV	V	Passive gerundive Auxiliary verb
GAV	V	Gerundive Auxilary verb
INF	N	Infinitive

Algorithm 1 shows how compound sentences are split into coordinate clauses. The algorithm accepts chunked compound sentences tagged with POS and morphological information and returns a list of clauses. The algorithm first looks for coordinate conjunction. If conjunction is detected, the sentence will be split there and the first part will be checked if it contains a verb. If it contains a verb, the first part will be added to the detected clause list and the remaining part of the sentence will be processed iteratively using a similar procedure.

```
Input:Chunked sentences tagged with POS and morphological
information(S)
Output: list of clauses (CLAUSE_LIST)
Begin
   Initialize INIT to Zero.
   For each token T in S which is tagged as "CONJ",
      Substring INIT to index of T and assign into a variable CLAUSE.
      Define a variable VERB_COUNT to the number of chunks in CLAUSE
       which contain a verb type of (IV, AV, PV PAV, PPV, PGAV, GAV).
      Set INIT to the index of T.
      If VERB_COUNT >= 1
          Add CLAUSE to CLAUSE_LIST
      End If
   End For
   Output CLAUSE_LIST
End
```

Algorithm 1. Coordinate clause splitting

Subordinate Clause Splitting. In order to simplify complex sentences, the main clause should be separated from the subordinate clause. Then, the main and the subordinate clauses are transformed into independent sentences. To this effect, we have developed an algorithm that takes chunked complex sentences tagged with POS and morphological information as an input and returns a list of clauses. Since the algorithm takes chunked sentences as input, the boundaries of chunks are used to identify subordinate clauses. Algorithm 2 shows how complex sentences are split into clauses. The algorithm iterates through all chunks of the sentence to look for a chunk containing a verb. If found, the phrase will be marked as a subordinate clause and it will be removed from the main clause. Both resulting clauses, i.e. main clause and subordinate clauses will be paraphrased to generate independent sentences.

```
Input:Chunked sentences tagged with POS and morphological
information(S)
Output: list of clauses (CLAUSE_LIST)
Begin
   For each Chunk C in S
     If C contain a verb of type of (PRV, IRV, PPRV, IPRV)
        Add C to CLAUSE_LIST
        Remove C from S
     End If
   End For
   Add S to CLAUSE_LIST
   Output CLAUSE_LIST
End
```

Algorithm 2. Subordinate clause splitting

Paraphrasing. In order to produce well-formed sentences from the individual clauses, they should be paraphrased. Sentence paraphrasing is a process of rewriting a sentence generated by clause splitting while preserving its meaning. The process includes rearranging the order of words in a sentence and changing the verb form by removing affixes of the verb. After splitting the subordinate clause from the main clause, affixes of the verb of the subordinate clause will be removed and then the order of the phrases will be rearranged based on the voice of the verb. For instance, to generate a simple sentence from a relative clause, the algorithm first checks if the voice of the verb is passive or active. If it is passive, the noun phrase found after the verb is a subject and a noun phrase found before the verb is an object. If the verb has an active voice, the noun phrase found before the verb will be a subject and a noun phrase found after the verb is an object. Finally, a sentence will be formed by removing affixes of the verb and combining the subject, object, and verb. Subordinate clauses might share the subject or object of the main clause. Thus, the main clause needs to be paraphrased. For example, the following sentence contains a relative clause where paraphrasing is made by taking the subject of the main clause.

Sentence:	[ለዓመታት በውጣ ውረዶች የተፈተነው የኢትዮጵያ ትራንስፖርት አሠሪዎች ፌዴሬሽን] [ምሥረታ] [ተከናወነ] ።
Relative clause:	ለዓመታት በውጣ ውረዶች የተፈተነው የኢትዮጵያ ትራንስፖርት አሠሪዎች ፌዴሬሽን
Main clause:	የኢትዮጵያ ትራንስፖርት አሠሪዎች ፌዴሬሽን ምሥረታ ተከናወነ።
Paraphrased relative clause:	የኢትዮጵያ ትራንስፖርት አሠሪዎች ለዓመታት በውጣ ውረዶች ተፈተነ።
Paraphrased main clause:	የኢትዮጵያ ትራንስፖርት አሠሪዎች ፌዴሬሽን ምሥረታ ተከናወነ ።

Algorithm 3 shows the implementation of the paraphrasing algorithm. The algorithm takes a list of clauses as input and generates a list of well-formed sentences.

```
Input: List of POS and morphological tagged clauses (Cs )
Output: list of simple sentences (SENTENCE_LIST)
Begin
   For each Clause C in Cs
      If C is relative clause
          Replace C in main clause by a noun phrase found after the verb
          If C contain  active verb
             Set SUBJECT by a noun phrase found before the verb
             Set OBJECT by a noun phrase found after the verb
          End If
          If C contain passive verb
             Set SUBJECT by a noun phrase found before the verb
             Set OBJECT by a noun phrase found after the verb
          End If
          Remove affixes from the verb
          Concatenate SUBJECT, OBJECT and verb and add it to
          SENTENCE_LIST
      End If
      If C is not relative clause
          Remove affixes from the verb
          Concatenate C with all chunks found before C and add it to
          SENTENCE_LIST
      End If
   End For
      Add main clause to SENTENCE_LIST
      Output SENTENCE_LIST
End
```

Algorithm 3. Paraphrasing algorithm

3.2 Relation Extraction

The main goal of an OIE system is to extract arbitrary relations with their corresponding arguments from natural language text. In this work, we identify and extract four relation types from a give sentence: *verb-based relation*, *has-relation*, *is-a relation* and *noun-mediated relation.*

Verb-based Relation Extraction. Verb-based relations can be represented in predicate-argument structure as *Rel* (*Arg1*, *Arg2*) where *Rel* is the verb of the sentence, *Arg1* is the subject of the sentence and *Arg2* is indirect object, direct object, complement, or adverb. Since Amharic sentence has a subject-object-verb structure, verb-based relations can be detected by the presence of a verb at the end of the sentence. Algorithm 4 shows the verb-based relation extraction algorithm. Consider the following example.

[አበበ] [ትላንትና] [ከአዲስ አበባ] [መጣ]::

The first chunk is the first argument አበበ and the verb መጣ is the relation phrase and the other phrases are the second argument of the relation. Accordingly, the extracted relations are:

መጣ (አበበ , ከአዲስ አበባ).
መጣ (አበበ , ትላንትና).

```
INPUT:Chunked sentences tagged with POS and morphological
information(S)
OUTPUT: relation tuples in predicate argument structure
BEGIN
    For each chunk C in S.
        IF C contain a verb
            Predicate = C
        End IF
        IF C is the first chunk and it is a noun phrase
            Argument1 = C
        END IF
        IF C does not contain a verb
            Add C to ArgumentList
        END IF
    END FOR
        output the relation in form of predicate (Argument1,
        ArgumentList))
END
```

Algorithm 4. Verb-based relation extraction

HAS Relation Extraction. HAS relation expresses possession or ownership in a sentencce. In Amharic sentence, HAS relation is implicitly expressed between two consecutive nouns when the first noun is in genitive case. The following examples show HAS relations in Amharic.

የአበበ ልጅ ፡ HAS (አበበ ,ልጅ).
የአለም ጓደኛ: HAS (አለም, ጓደኛ).

Algorithm 5 shows the implementation of the HAS relation extraction. The algorithm takes noun phrases tagged with POS and morphological information. The algorithm checks if a noun in genitive case followed by another noun is found in the input noun phrase. If found, the word itself will be the first argument and words found after it is the second argument.

```
INPUT:Noun phrases tagged with POS and morphological information (NP)
OUTPUT: HAS relation tuple in predicate argument structure
BEGIN
    IF a noun in genitive case, followed by other noun is found.
          Argument1 = the noun itself
          Argument2 = the next noun
     END IF
     output the predicate in form of HAS (Argument1, Argument2))
END
```

Algorithm 5. HAS relation extraction

IS-A Relation Extraction. IS-A relation is an implicitly expressed relation between a proper noun and a common noun. In Amharic, this kind of relation is found when a proper noun comes after a common noun. The following example shows IS-A relation.

የኢትዮጲያ ሯጭ ሃይሌ ገብረስላሴ: IS-A (ሃይሌ ገብረስላሴ , ሯጭ).

The implementation of IS-A relation extraction is shown in Algorithm 6. The algorithm takes noun phrases tagged with POS and morphological information as an input

and returns a list of IS-A relations. By iterating to each word in the text, it looks for an agent noun which is followed by another noun. If found, the words found after the agent noun is extracted as the first argument and the agent noun will be extracted as the second argument.

```
INPUT:Noun phrases tagged with POS and morphological information (NP)
OUTPUT: IS-A relation tuple in predicate argument structure
BEGIN
    If an agent noun followed by a noun is found
        Argument1 = noun phrase found after the agent noun
        Argument2 = the agent noun
    END If
    output the predicate in form of IS (Argument1, Argument2))
END
```

Algorithm 6. IS-A relation extraction

Noun-Mediated Relation Extraction. Noun-mediated relation expresses a binary relation between two nouns. In Amharic sentence, a common noun found between two proper nouns indicates the presence of a noun-mediated relation between two the proper nouns. For example, from a noun phrase የኢትዮጵያ ሯጭ ሃይሌ ገብረስላሴ, a common noun ሯጭ is a relation between ኢትዮጵያ and ሃይሌ ገብረስላሴ. It can be represented in predicate-argument structure as: ሯጭ (ሃይሌ ገብረስላሴ, ኢትዮጵያ). Agent nouns are used to detect noun-mediated relations. The implementation of noun-mediated relation extraction is demonstrated in Algorithm 7. The algorithm takes noun phrases tagged with POS and morphological information as input. The algorithm first looks for an agent noun that is found between two nouns. If found, the noun found after the agent noun is extracted as the first argument, the noun found before the agent noun is extracted as the second argument, and the agent noun will be extracted as the predicate.

```
INPUT: Noun phrases tagged with POS and morphological information (NP)
OUTPUT: noun-medicated relation tuple in Predicate-Argument structure
BEGIN
    IF an agent noun is found between two nouns
        Predicate = agent noun
        Argument1 = noun phrase found after the agent noun
        Argument2 = noun phrase found before the agent noun
    END IF
    output the predicate in form of predicate (Argument1, Argument2)
END
```

Algorithm 7. Noun-mediated relation extraction algorithm

4 Experiment

4.1 Dataset Collection

To test the performance of the system, text corpus was collected from online Amharic news sources such as The Reporter Ethiopia[1] and Walta Media and Communication

[1] https://www.ethiopianreporter.com.

Corporate[2]. The corpus was collected randomly from different domain areas to study the sensitivity of the proposed system to variation in domain. In order to create an annotated dataset for extraction, each sentence is tagged with POS, morphological information and relation. A total of 215 tagged sentences are used for testing the system. From these sentences, we annotated 768 relations that consists of 414 verb-based relations, 207 HAS relations, 78 noun-mediated relations and 69 IS relations.

4.2 Test Result

A system implementing the algorithms was developed and used to evaluate the proposed approach, The collected corpus was used to test the performance of the system. Test results are shown in Table 3.

Table 3. Test result

Relation	Ground truth	Extraction result		
		Total extracted	Correctly extracted	Precision
Verb-Based	414	310	286	0.92
HAS	207	140	128	0.91
IS	69	50	39	0.78
Noun-Mediated	78	56	41	0.73
TOTAL	768	556	494	0.88

4.3 Discussion

Test results show that the proposed system has extracted 556 instances of relations from 215 sentences with a precision of 88%. Considering the errors made by the system, in general, we found that 56.5% of errors are due to complex and malformed sentences. Although the performance of AOIE can be significantly improved by simplifying complex sentences, the system achieves only 73% accuracy in sentence simplification. After a thorough analysis of each error returned by the sentence simplification component on the test dataset, most of the errors are due to failures in simplifying highly complex sentences. The sentence simplification algorithm has shortcomings in handling sentences that contain one or more clauses that share the subject or objects with the main clause, and/or contain other embedded clauses. This limitation often leads to incorrect and over-specified predicates and arguments, missed relation instances, and it also produces relation instances that are inconsistent with the information contained in the original sentence. For instance, consider the following sentence.

[2] http://www.waltainfo.com.

በሻኪሶ ከተማ አካባቢ ነዋሪዎች በተነሳ ተቃውሞ ምክንያት ሥራውን እንዲያቋርጥ በተደረገው የሚድሮክ ወርቅ ኩባንያ ንብረት በሆነው የለገደንቢ ወርቅ ማምረቻ ላይ የካናዳ ከፍተኛ ባለሙያዎች ጥናት ሊያካሂዱ እንደሆነ ታወቀ።።

The relation generated by the system is: ሥራውን እንዲያቋርጥ ጥናት ሊያካሂዱ እንደሆነ ታወቀ (Arg1: "በሻኪሶ ከተማ አካባቢ ነዋሪዎች", Arg2: "በተነሳ ተቃውሞ ምክንያት ")". The first argument of this extraction is incorrect, it should be "የካናዳ ከፍተኛ ባለሙያዎች". It also contains an over-specified predicate. There is also one missed relation: ነው (Arg1: "የለገደንቢ ወርቅ ማምረቻ", Arg2: "የሚድሮክ ወርቅ ኩባንያ", Arg3: "ንብረት"). Moreover, 32.7% of the errors are due to errors in morphological analysis and POS tagging. For instance, from the sentence "በዓድዋ ጦርነት የተሰው ጀግኖችን አፅም በክብር ለማሳረፍ ታሰበ:", two relation instances were expected from the system: ታሰበ (Arg1: "ጀግኖችን", Arg2: "አፅም", Arg3: "በክብር ለማሳረፍ") and ለማሳረፍ") and ተሰው ("ጀግኖች", "በዓድዋ ጦርነት"). However, since the morphological analyzer did not label the "የተሰው" as a relative verb, the relative clause is not detected and the sentence couldn't be simplified. As a result of this, only one instance of relation which contains an over-specified argument is extracted by the system: "ታሰበ (Arg1:"በዓድዋ ጦርነት የተሰው ጀግኖችን ", Arg2: "አፅም", Arg3: "በክብር ለማሳረፍ ").

The remaining types of errors made by the system are due to erroneously chunked phrases. Incorrectly chunking the sentence often leads to incorrect predicate or arguments. For instance, the sentence: "የኢትዮጵያ አየር መንገድ በረራዎች ለሁለት ሰዓታት ተቋረጡ" is chunked erroneously as: [የኢትዮጵያ አየር] [መንገድ በረራዎች] [ለሁለት ሰዓታት] [ተቋረጡ]. Thus, incorrect relation ተቋረጡ (Arg1:" የኢትዮጵያ አየር" Arg2: "መንገድ በረራዎች", Arg3:-"ለሁለት ሰዓታት") is extracted instead of: ተቋረጡ (Arg1: "የኢትዮጵያ አየር መንገድ በረራዎች", Arg2:-"ለሁለት ሰዓታት").

5 Conclusion

In this paper, we present Amharic OIE system. It is understood that rule-based approaches operating on deep parsed sentences yield the most promising results for OIE systems, as they enable higher precision. However, Amharic has limited tools and resources making it difficult to apply deep dependency parser. As a result, we have implemented a rule-based Amharic OIE system that operates on shallow parsed sentences. To minimize the difficulty of relation extraction from shallow parsed sentences, we introduced a sentence simplification component that converts complex and compound sentences into a list of simple sentences without losing the overall semantics of the original sentence. Our experimental results have shown that sentence simplification minimizes the reliance of relation extraction on deep parsed sentences. This indicates that the performance of system can be further improved by enhancing the capability of sentence simplification component.

References

1. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. IJCAI **2007**, 2670–2676 (2007)

2. Fader, A., Soderland, S., Etzioni, O.: Identifying relations for open information extraction. In: Proceedings of the Conference on EMNLP, pp. 1535–1545. Association for Computational Linguistics, Stroudsburg (2011)
3. Sangha, N., Younggyun, N., Sejin, N., Key-Sun, C.: SRDF: korean open information extraction using singleton property. In: Proceedings of the 14th International Semantic Web Conference (2015)
4. Mausam M.: Open information extraction systems and downstream applications. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016. AAAI Press: 2, pp. 4074–4077 (2016)
5. Abreu, S.C., Bonamigo, T.L., Vieira, R.: A review on relation extraction with an eye on portuguese. J. Braz. Comput. Soc. **19**, 553–571 (2013)
6. Wu, F., Weld, D.S.: Open Information Extraction using Wikipedia. In: The 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden (2010)
7. Etzioni, O., Fader, A., Christensen, J., Soderland, S.: 2011. open information extraction: the second generation. In: Proceedings of the 22nd international joint conference on Artificial Intelligence IJCAI 2011, pp. 3–10, Barcelona, Catalonia, Spain, 16–22 July 2011
8. Mausam, S.M., Soderland, S., Bart, R., Etzioni, O.: Open language learning for information extraction. In: EMNLP-CoNLL, pp. 523–534 (2012)
9. Del Corro, L., Gemulla, R.: ClausIE: clause-based open information extraction. In: Proceedings of the 22nd International Conference on World Wide Web, WWW 2013, pp. 355–366. ACM, New York (2013)
10. Gamallo, P., Garcia, M., Fernández-Lanza, S.: Dependency-based open information extraction. In: ROBUS-UNSUP Workshop at EACL-2012, Avignon, France (2012)
11. Tseng, Y.-H., et al.: Chinese open relation extraction for knowledge acquisition. In: EACL, pp 12–16 (2014)