Hierarchical Amharic Base Phrase Chunking Using HMM with Error Pruning

Abeba Ibrahim and Yaregal Assabie(✉)

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

Abstract. Segmentation of a text into non-overlapping syntactic units (chunks) has become an essential component of many applications of natural language processing. This paper presents Amharic base phrase chunker that groups syntactically correlated words at different levels using HMM. Rules are used to correct chunk phrases incorrectly chunked by the HMM. For the identification of the boundary of the phrases IOB2 chunk specification is selected and used in this work. To test the performance of the system, corpus was collected from Amharic news outlets and books. The training and testing datasets were prepared using the 10-fold cross validation technique. Test results on the corpus showed an average accuracy of 85.31 % before applying the rule for error correction and an average accuracy of 93.75 % after applying rules.

Keywords: Amharic language processing · Base phrase chunking · Partial parsing

1 Introduction

Chunking is a natural language processing (NLP) task that focuses on dividing a text into syntactically correlated non-overlapping and non-exhaustive groups of words, i.e., a word can only be a member of one chunk and not all words are in chunks (Tjong *et al.* 2000). Chunking is widely used as an intermediate step to parsing with the purpose of improving the performance of the parser. It also helps to identify non-overlapping phrases from a stream of data, which are further used for the development of different NLP applications such as information retrieval, information extraction, named entity recognition, question answering, text mining, text summarization, etc. These NLP tasks consist of recognizing some type of structure which represents linguistic elements of the analysis and their relations. In text chunking the main problem is to divide text into syntactically related non-overlapping groups of words (chunks).

The main goal of chunking is to divide a text into segments which correspond to certain syntactic units such as noun phrases, verb phrases, prepositional phrases, etc. Abney (1991) introduced the concept of chunk as an intermediate step providing input to further full parsing stages. Thus, chunking can be seen as the basic task in full parsing. Although the detailed information from a full parse is lost, chunking is a valuable process in its own right when the entire grammatical structure produced by a full parse is not required. For example, various studies indicate that the information obtained by

Z. Vetulani et al. (Eds.): LTC 2013, LNAI 9561, pp. 126–135, 2016.
DOI: 10.1007/978-3-319-43808-5_10

chunking or partial parsing is sufficient for information retrieval systems rather than full parsing (Yangarber and Grishman 1998). Alongside, partial syntactical information can help to solve many NLP tasks, such as text summarization, machine translation and spoken language understanding (Molina and Pla 2002). For example, Kutlu (2010) stated that finding noun phrases and verb phrases is enough for information retrieval systems. Phrases that give us information about agents, times, places, objects, etc. are more significant than the complete configurational syntactic analyses of a sentence for question-answering, information extraction, text mining and automatic summarization.

Chunkers do not necessarily assign every word in the sentence like full parses to a higher-level constituent. They identify simple phrases but do not require that the sentence be represented by a single structure. By contrast full parsers attempt to discover a single structure which incorporates every word in the sentence. Abney (1995) proposed to divide sentences into labeled, non-overlapping sequences of words based on superficial analysis and local information. In general, many of NLP applications often require syntactic analysis at various NLP levels including full parsing and chunking. The chunking level identifies all possible phrases and the full parsing analyzes the phrase structure of a sentence. The choice of which syntactic analysis level should be used depends on the specific speed or accuracy of an application. The chunking level is efficient and fast in terms of processing than full parsing (Thao et al. 2009). Chunkers can identify syntactic chunks at different levels of the parser, so a group of chunkers can build a complete parser (Abney 1995). Most of the parsers developed for languages like English and German use chunkers as components. Brants (1999) used a cascade of Markov model chunkers for obtaining parsing results for the German NEGRA corpus. Today, there are a lot of chunking systems developed for various languages such as Turkish (Kutlu 2010), Vietnamese (Thao et al. 2009), Chinese (Xu et al. 2006), Urdu (Ali and Hussain 2010), etc.

Although Amharic is the working language of Ethiopia with a population of about 90 million at present, it is still one of less-resourced languages with few linguistic tools available for Amharic text processing. This work is aimed at developing Amharic base phrase chunker that generates base phrases. The remaining part of this paper is organized as follows. Section 2 presents Amharic language with emphasis to its phrase structure. Amharic base phrase chunking along with error pruning is discussed in Sect. 3. In Sect. 4, we present experimental results. Conclusion and future works are highlighted in Sect. 5. References are provided at the end.

2 Linguistic Structures of Amharic

2.1 Amharic Language

Amharic is the working language of Ethiopia. Although many languages are spoken in Ethiopia, Amharic is the lingua franca of the country and it is the most commonly learned second language throughout the country (Lewis *et al.* 2013). It is also the second most spoken Semitic language in the world next to Arabic. Amharic is written using Ethiopic script which has 33 consonants (basic characters) out of which six other characters representing combinations of vowels and consonants are derived for each character.

The base characters have the vowel ä(*ä*) and other derived characters have vowels in the order of ኡ(*u*), ኢ(*i*), ኣ(*a*), ኤ(*e*), እ(*ĭ*), and ኦ(*o*). For example, for the base character ከ(*kä*), the following six characters are derived from the base character: ኩ(*ku*), ኪ(*ki*), ካ(*ka*), ኬ(*ke*), ክ(*kĭ*), and ኮ(*ko*).

2.2 Phrasal Categories

Phrases are syntactic structures that consist of one or more words but lack the subject-predicate organization of a clause. These phrases are composed of either only head word or other words or phrases with the head combination. The other words or phrases that are combined with the head in phrase construction can be specifiers, modifiers and complements. Yimam (2000) classified Amharic word classes into five types, i.e. nouns, verbs, adverbs, adjectives and prepositions. In line with this classification, Yimam (2000) and Amare (2010) classified phrase structures of the Amharic language as: noun phrases, verb phrases, adjectival phrases, adverbial phrases and prepositional phrases.

Noun Phrase. An Amharic noun phrase (NP) is a phrase that has a noun as its head. In this phrase construction, the head of the phrase is always found at the end of the phrase. This type of phrase can be made from a single noun or combination of noun with either other word classes including noun word class. Examples are: ቀለበት (*qäläbät*/ring), የአልማዝ ቀለበት (y*ä'almaz qäläbät*/diamond ring), ትልቅ የአልማዝ ቀለበት (*tĭlĭq* y*ä'almaz qäläbät*/big diamond ring), ያ ትልቅ የአልማዝ ቀለበት (*ya tĭlĭq* y*ä'almaz qäläbät*/that big diamond ring), etc.

Verb Phrase. Amharic verb phrase (VP) is constructed with a verb as a head, which is found at the end of the phrase, and other constituents such as complements, modifiers and specifiers. But not all the verbs take the same category of complement. Based on this, verbs can be dividing into two. These are transitive and intransitive. Transitive verbs take transitive noun phrases as their complement and intransitive verbs do not. Examples are: ልኮላታል (*lĭkolatal*/[he] sent [her] [something]), ገንዘብ ልኮላታል (*gänzäb lĭkolatal*/[he] sent [her] money), ለሜሪ ገንዘብ ልኮላታል (*lämeri gänzäb lĭkolatal*/[he] sent money to Mary), በባንክ ለሜሪ ገንዘብ ልኮላታል (*bäbank lämeri gänzäb lĭkolatal*/[he] sent money to Mary via bank), etc.

Adjectival Phrase. An Amharic Adjectival phrase (AdjP) is constructed with an adjective as a head word and other constituents such as complements, modifiers and specifiers. The head word is placed at the end. Examples are: ጎበዝ (*gobäz*/clever), በጣም ጎበዝ (*bäţam gobäz*/very clever), እንደ ወንድሙ በጣም ጎበዝ (*ĭndä wändĭmu bäţam gobäz*/very clever like his brother), etc.

Prepositional Phrase. Amharic prepositional phrase (PP) is made up of a preposition head and other constituents such as nouns, noun phrases, prepositional phrases, etc. Unlike other phrase constructions, prepositions cannot be taken as a phrase, instead they should be combined with other constituents and the constituents may come either previous to or subsequent to the preposition. If the complements are nouns or NPs, the position of prepositions is in front of the complements whereas if the complements are

PPs, the position will shift to the end of the phrase. Examples are: እንደ ልጅ (*ĭndä lĭj/*like a child), ከወንዙ አጠገብ (*käwänzu aṭägäb/*close to the river), etc.

Adverbial Phrases. Amharic adverbial phrases (AdvP) are made up of one adverb as head word and one or more other lexical categories including adverbs themselves as modifiers. The head of the AdvP is placed at the end. Unlike other phrases, AdvPs do not take complements. Most of the time, the modifiers of AdvPs are PPs that come always before adverbs. Examples are: ክፉኛ (*kĭfuña/*severely), በጣም ክፉኛ (*bäṭam kĭfuña/* very severely), እንደ ወንድሙ በጣም ክፉኛ (*ĭndä wändĭmu bäṭam kĭfuña/*very severely like his brother), etc.

2.3 Sentence Formation

Amharic language follows subject-object-verb grammatical pattern unlike, for example, English language which has subject-verb-object sequence of words (Yimam 2000; Amare 2010). For instance, the Amharic equivalent of the sentence "John killed the lion" is written as "ጆን (*jon/John*) አንበሳውን (*anbäsawn/*the lion) ገደለው (*gädäläw/*killed)". Amharic sentences can be constructed from simple or complex NP and simple or complex VP. Simple sentences are constructed from simple NP followed by simple VP which contains only a single verb. The following examples show the various structures of simple sentences.

- ጆን መኪና ገዛ።
  jon mäkina gäza።
  John bought a car.
- ማን መኪና ገዛልህ?
  man mäkina gäzalh?
  Who did buy a car for you?
- ጆን መጣ?
  jon mäṭa?
  Did John come?
- ዳቦ ጋግሪ!
  dabo gagri!
  Bake {feminine} bread!
- ሁለት ትልልቅ ልጆች በመኪና ወደ ጎጃም ሄዱ።
  hulät tĭlĭlĭq lĭjoc bämäkina wäda gojam hedu።
  Two big children went to Gojjam by car.

Complex sentences are sentences that contain at least one complex NP or complex VP or both complex NP and complex VP. Complex NPs are phrases that contain at least one embedded sentence in the phrase construction. The embedded sentence can be complements. The following examples show the various structures of complex Amharic sentences.

- [ጆን የገባበት የሳር ቤት] በጣም ትልቅ ነው።
  [*jon yägäbabät yäsar bet*] *bäṭam tĭlĭq näw*።
  [The thatched house that John entered in] is so big.
- ሜሪ [ጆን መኪና እንደገዛ] ሰማች።
  meri [*jon mäkina ĭndägäza*] *sämač*።
  Mary heard [that John has bought a car].
- ጆን [ከጎጃም እንደመጣ] [ሜሪ ወደ ናዝሬት እንደ ሄደች] ሰማ።
  jon [*kägojam ĭndämäṭa*] [*meri wädä nazret ĭndä hedäč*] *säma.*
  [When John came from Gojjam] he heard [that Mary went to Nazareth].
- [ከጎጃም የመጣችው ልጅ] [ጆን እንደወደዳት] አወቀች።
  [*kägojam yämäṭačĭw lĭj*] [*jon ĭndäwädädat*] *awäqäč.*
  [The girl who came from Gojjam] knew [that John loved her].

3 Base Phrase Chunking

3.1 Chunk Representation

The tag of chunks can be noun phrases, verb phrases, adjectival phrases, etc. in line with the language construction rules. There are many decisions to be made about where the boundaries of a group should lie and, as a consequence, there are many different 'styles' of chunking. There are also different types of chunk tags and chunk boundary identifications. Nevertheless, in order to identify the boundaries of each chunk in sentences, the following boundary types are used (Ramshaw and Marcus 1995): IOB1, IOB2, IOE1, IOE2, IO, "[", and "]". The first four formats are complete chunk representations which can identify the beginning and ending of phrases while the last three are partial chunk representations. All boundary types use "I" tag for words that are inside a phrase and an "O" tag for words that are outside a phrase. They differ in their treatment of chunk-initial and chunk-final words.

IOB1: the first word inside a phrase immediately following another phrase receives a **B** tag.
IOB2: all phrase- initial words receive a **B** tag.
IOE1: the final word inside a phrase immediately preceding another same phrase receives an **E** tag.
IOE2: all phrase- final words receive an **E** tag.
IO: words inside a phrase receive an **I** tag, others receive an **O** tag.
"[": all phrase-initial words receive "[" tag, other words receive "." Tag.
"]": all phrase-final words receive "]" tag and other words receive "." Tag.

An example of chunk representation for the sentence ሁለቱ ልጆች በትልቅ መኪና ወደ ጎጃም ሄዱ (*hulätu lĭjoc bätĭlĭq mäkina wäda gojam hedu* / The two children went to Gojjam by a big car) is shown Table 1.

Table 1. Chunk representation for the sentence "ሁለቱ ልጆች በትልቅ መኪና ወደ ጎጃም ሄዱ".

Chunk	ሁለቱ	ልጆች	በትልቅ	መኪና	ወደ	ጎጃም	ሄዱ
IOB1	I-NP	I-NP	B-NP	I-NP	I-PP	I-PP	O
IOB2	B-NP	I-NP	B-NP	I-NP	B-PP	I-PP	O
IOE1	I-NP	E-NP	I-NP	I-NP	I-PP	I-PP	O
IOE2	I-NP	E-NP	I-NP	E-NP	I-PP	E-PP	O
IO	I-NP	I-NP	I-NP	I-NP	I-PP	I-PP	O
[	I-NP	.	[	.	[	.	.
]	.	]	.	]	.	]	.

In this work, we considered six different kinds of chunks, namely noun phrase (NP), verb phrase (VP), Adjective phrase (AdjP), Adverb phrase (AdvP), prepositional phrase (PP) and sentence (S). To identify the chunks, it is necessary to find the positions where a chunk can end and a new chunk can begin. The part-of-speech (POS) tag assigned to every token is used to discover these positions. We used the IOB2 tag set to identify the boundaries of each chunk in sentences extracted from chunk tagged text. Using the IOB2 tag set along with the chunk types considered, a total of 13 phrase tags were used in this work. These are: B-NP, I-NP, B-VP, I-VP, B-PP, I-PP, B-ADJP, I-ADJP, B-ADVP, I-ADVP, B-S, I-S and O. The followings are examples of chunk tagged sentences.

- ሁለቱ NUMCR O ትልልቅ ADJ B-NP ልጆች N I-NP በመኪና NPREP O ወደ Prep B-PP ጎጃም N I-NP ሄዱ V O
- የኢትዮጵያ ADJ B-NP ጠላቶች N B-NP ሀገሪቱ N O በድርጅቱ NPREP B-PP ውስጥ Prep I-PP የተሰጣትን V O ቦታ N O ተቃወሙ V O
- ንጉሱ N O ፋሺስቶች N B-S የተርበደበዱበትን V I-S ጀግና ADJ B-NP ሰው N I-NP ሰቀሉ V O
- ኢትዮጵያ N O ፈንጣጣ N B-S የተወገደበትን V I-S እለት N O ትላንት ADV B-VP አከበረች V I-VP

3.2 Architecture of the Chunker

To implement the chunker component, we used hidden Markov model (HHM) enhanced by a set of rules to prune errors. The HMM part has two phases: the training phase and the testing phase. In the training phase, the system first accepts words with POS tags and chunk tags. Then, the HMM is trained with this training set. Likewise in the test phase, the system accepts words with POS tags and outputs appropriate chunk tag sequences against each POS tag using HMM model. Figure 1 illustrates the workflow of the chunking process.

In this work, chunking is treated as a tagging problem. We use POS tagged sentence as input from which we observe sequences of POS tags represented as T. However, we also hypothesize that the corresponding sequences of chunk tags form hidden Markovian properties. Thus, we used a hidden Markov model (HMM) with POS tags serving as

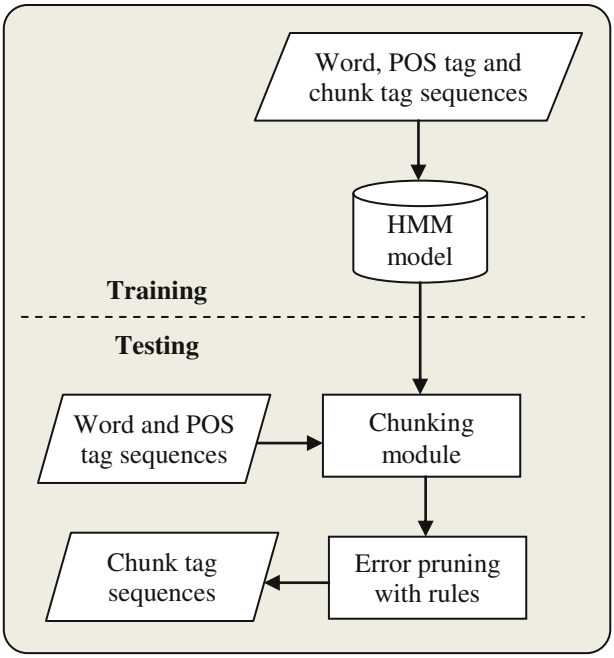

Fig. 1. Workflow of the chunking process.

states. The HMM model is trained with sequences of POS tags and chunk tags extracted from the training corpus. The HMM model is then used to predict the sequence of chunk tags C for a given sequence of POS tag T. This problem corresponds to finding C that maximizes the probability *P(C|T)*, which is formulated as:

$$C' = \arg\max_{C} P(C|T) \tag{1}$$

where C′ is the optimal chunk sequence. By applying Baye's rule can derive, Eq. (1) yields:

$$C' = \arg\max_{C} P(T|C) * P(C) \tag{2}$$

which is in fact a decoding problem that is solved by making use of the Viterbi algorithm. The output of the decoder is the sequence of chunk tags which groups words based on syntactical correlations. The output chunk sequence is then analyzed to improve the result by applying linguistic rules derived from the grammar of Amharic. For a given Amharic word *w*, linguistic rules (from which sample rules are shown in Algorithm 1) were used to correct wrongly chunked words ("*w*−1" and "*w*+1" are used to mean the previous and next word, respectively).

```
1. If POS(w)=ADJ and POS(w+1)=NPREP, NUMCR, then chunk
   tag for w is O.
2. If POS(w)=ADJ and POS(w-1)!= ADJ and POS(w+1)=
   AUX,V, then chunk tag for w is B-VP.
3. If POS(w)=NPREP and POS(w+1)=N ,then chunk tag for w
   is B-NP.
4. If POS(w)=NUMCR and POS(w+1)=NPREP, then chunk tag
   for w is O.
5. If POS(w)=N and POS(w+1)=VPREP and POS(w-1) =N, ADJ,
   PRON, NPREP, then chunk tag for w is    B-VP.
6. If POS(w)=ADJ and POS(w+1)=ADJ, then chunk tag for w
   is B-ADJP.
```

Algorithm 1. Sample rules used to prune chunk errors.

4 Experiment

4.1 The Corpus

The major source of the dataset we used for training and testing the system was Walta Information Center (WIC) news corpus which is at present widely used for research on Amharic natural language processing. The corpus contains 8067 sentences where words are annotated with POS tags. Furthermore, we also collected additional text from an Amharic grammar book authored by Yimam (2000). The sentences in the corpus are classified as training data set and testing data set using 10 fold cross validation technique.

4.2 Test Results

In 10-fold cross-validation, the original sample is randomly partitioned into 10 equal size subsamples. Of the 10 subsamples, a single subsample is used as the validation data for testing the model, and the remaining 9 subsamples are used as training data. The cross-validation process is then repeated 10 times, with each of the 10 subsamples used exactly once as the validation data. Accordingly, we obtain 10 results from the folds which can be averaged to produce a single estimation of the model's predictive potential. By taking the average of all the ten results the overall chunking accuracy of the system is presented in Table 2.

Table 2. Test result for Amharic base phrase chunker.

Chunking model	Accuracy
HMM	85.31 %
HMM pruned with rules	93.75 %

5 Conclusion and Future Works

Amharic is one of the most morphologically complex and less-resourced languages. This complexity poses difficulty in the development of natural language processing applications for the language. Despite the efforts being undertaken to develop various Amharic NLP applications, only few usable tools are publicly available at present. One of the main reasons frequently cited by researchers is the morphological complexity of the language. Amharic text parsing also suffers from this problem. However, not all Amharic natural language processing applications require full parsing. In this work, we tried to overcome this problem by employing chunker. It appears that chunking is more manageable problem than parsing because the chunker does not require deeper analysis of texts which will be less affected by the morphological complexity of the language. Thus, future work is recommended to be directed at improving the chunker and use this component to develop Amharic natural language processing applications that do not rely on deeper analysis of linguistic structures.

References

Abney, S.: Parsing by chunks. In: Berwick, R., Abney, S., Tenny, C. (eds.) Principle-Based Parsing. Kluwer Academic Publishers, Dordrecht (1991)

Abney, S.: Chunks and dependencies: bringing processing evidence to bear on syntax. In: Computational Linguistics and the Foundations of Linguistic Theory. CSLI (1995)

Ali, W., Hussain, S.: A hybrid approach to Urdu verb phrase chunking. In: Proceedings of the 8th Workshop on Asian Language Resources (ALR-8), COLING-2010, Beijing, China (2010)

Amare, G.: ዘመናዊ የአማርኛ ሰዋስው· በቀላል አቀራረብ (Modern Amharic Grammar in a Simple Approach). Addis Ababa, Ethiopia (2010)

Brants, T.: Cascaded markov models. In: Proceedings of the Ninth Conference on European Chapter of the Association for Computational Linguistics, EACL 1999, Bergen, Norway (1999)

Kutlu, M.: Noun phrase chunker for Turkish using dependency parser. Doctoral dissertation, Bilkent University (2010)

Lewis, P., Simons, F., Fennig, D.: Ethnologue: Languages of the World, 17th edn. SIL International, Dallas (2013)

Molina, A., Pla, F.: Shallow parsing using specialized HMMs. J. Mach. Learn. Res. **2**, 595–613 (2002)

Ramshaw, A., Marcus, P.: Text chunking using transformation-based learning. In: Proceedings of the Third ACL Workshop on Very Large Corpora, pp. 82–94 (1995)

Thao, H., Thai, P., Minh N., Thuy, Q.: Vietnamese noun phrase chunking based on conditional random fields. In: International Conference on Knowledge and Systems Engineering (KSE 2009), pp. 172–178 (2009)

Tjong, E.F., Sang, K., Buchholz, S.: Introduction to the CoNLL-2000 shared task: chunking. In: Proceedings of the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning, vol. 7, pp. 127–132 (2000)

Xu, F., Zong, C., Zhao, J.: A hybrid approach to Chinese base noun phrase chunking. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney (2006)

Yangarber, R., Grishman, R.: NYU: description of the Proteus/PET system as used for MUC-7. In: Proceedings of the Seventh Message Understanding Conference, MUC-7, Washington, DC (1998)

Yimam, B.: የአማርኛ ሰዋስው (Amharic Grammar). Addis Ababa, Ethiopia (2000)