3/3/2023


# Amharic Morphological Segmentation using bidirectional LSTM neural network and rule-based segmentation

Name: Sintayehu Zekarias

ID : GRS/8732/14

Submitted to

DR. MICHAEL MELESE

# Contents

## 1. Introduction

Morphological segmentation is the process of breaking down words into their component morphemes, which are the smallest units of meaning. This technique is particularly useful in morphologically rich languages where words can contain multiple morphemes. Morphological segmentation can be performed manually by linguists or automatically using computational methods. The primary objective of morphological segmentation is to facilitate the processing of natural language. By breaking down words into their component morphemes, it becomes easier to analyze and process text data. This can be particularly useful in applications such as information retrieval, machine translation, and speech recognition.

Morphological segmentation is especially important for languages with complex morphology, such as Amharic, Tigregna and Afan Oromo. In these languages, words can be composed of multiple morphemes, and the morphemes can have different forms depending on the context.

Amharic is the most widely spoken Semitic language in the world, primarily spoken in Ethiopia. Amharic is a highly inflected language, and its morphological segmentation is challenging. Unlike English, where words are usually separated by spaces, Amharic words are typically written as a continuous sequence of characters, which makes it difficult to determine word boundaries.

The proposed model development for Amharic morphological segmentation involves combining unsupervised deep learning and rule-based segmentation techniques to create a robust and accurate system for segmenting words into their constituent morphemes.

## 2. Challenges of Amharic Morphological segmentation

Amharic is a Semitic language spoken in Ethiopia, and it is known for its highly inflectional and agglutinative nature. These features make morphological segmentation in Amharic a challenging task. The inflectional nature of Amharic words means that they can have several morphemes that can change their meaning. This high level of inflection also means that words can take on different forms based on their context, adding to the complexity of segmentation.

Additionally, Amharic words can have multiple morphemes glued together, making it difficult to identify the boundaries between them. This agglutinative nature means that a single word can

contain several distinct morphemes, each with its own meaning. For example, the word "ገበያዊነት" (gebe-ya-win-ness) is composed of four morphemes that together mean "modernity."

Another challenge of Amharic morphological segmentation is the lack of standardization. Amharic has many dialects, and there is no one standard version of the language. This makes it difficult to create a standard set of rules for morphological segmentation that will work for all speakers and dialects. The lack of standardization also makes it challenging to create a comprehensive dictionary of Amharic words.

Finally, there is a shortage of labeled data for Amharic, making it challenging to train machine learning models for morphological segmentation. Without sufficient data, it is difficult to create accurate models that can effectively segment Amharic text. However, recent efforts have been made to address this challenge, such as the creation of the Amharic Treebank corpus, which provides annotated data for Amharic text.

To perform Amharic morphological segmentation, a series of preprocessing steps are required. These steps include text normalization, tokenization, stop word removal, stemming, and lemmatization. Additionally, web scraping can be used to collect Amharic text data for analysis.

## 3. Objective of Morphological Segmentation

The primary goal of this project is to achieve Amharic morphological segmentation through the integration of deep learning and rule-based segmentation techniques. The purpose is to break down words into their constituent morphemes, which will facilitate more precise natural language processing. Some advantages of Amharic morphological segmentation include:

- Improving the accuracy of natural language processing tasks such as part-of-speech tagging, machine translation, and speech recognition.
- Enabling the development of more effective language models by capturing the relationships between morphemes and their meanings.
- Facilitating the creation of language resources such as morphological dictionaries and machine-readable corpora.
- Supporting linguistic research by providing a more fine-grained analysis of word structure.

# 4. Methodology to Perform Amharic Morphological Segmentation

## 4.1. Data Collection

One of the primary challenges in performing Amharic morphological segmentation is the lack of high-quality annotated corpora. Therefore, web scraping is an effective way to collect Amharic text data. Web scraping involves extracting relevant text data from web pages, and it can be used to collect a large amount of text data in a short amount of time. For this project the data is collected from [www.fanabc.com](http://www.fanabc.com) and it has the following categories

Category 1 : ዜና
Category 2 : ርዕስ አንቀፅ
Category 3 : ባህል
Category 4 : ህብረተሰብ
Category 5 : ጥበብ
Category 6 : ኪነ-ጥበባዊ ዜና
Category 7 : ልብ ወለድ
Category 8 : የግጥም ጥግ
Category 9 : አግራሞት
Category 10 : ማራኪ አንቀፅ
Category 11 : ነፃ አስተያየት
Category 12 : የሰሞኑ አጀንዳ
Category 13 : ፖለቲካ በፈገግታ
Category 14 : ከአለም ዙሪያ
Category 15 : ጤና
Category 16 : ንግድና ኢኮኖሚ

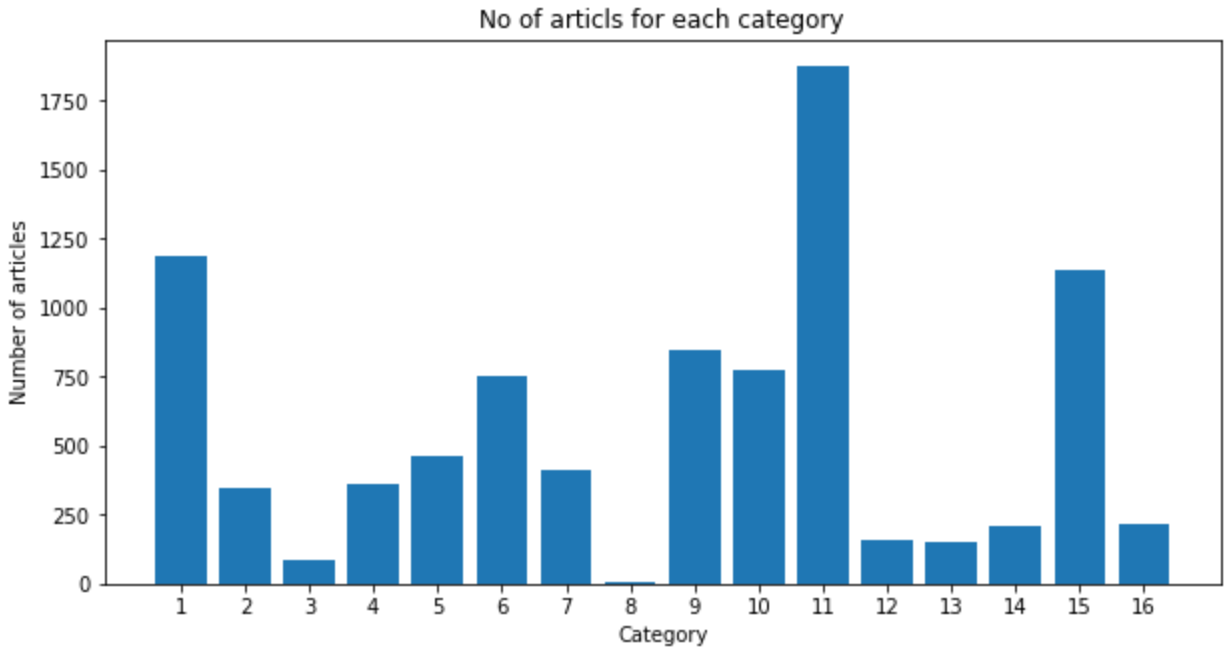


Fig 1. 1:ህብረተሰብ , 2:ልብ ወለድ , 3:ማራኪ አንቀፅ , 4:ርዕስ አንቀፅ , 5:ባህል , 6:ነፃ አስተያየት , 7:ንግድና ኢኮኖሚ , 8:አግራሞት , 9:ከአለም ዙሪያ , 10:ኪነ-ጥበባዊ ዜና , 11:ዜና , 12:የሰሞኑ አጀንዳ , 13:የግጥም ጥግ , 14:ጤና , 15:ጥበብ , እና 16:ፖለቲካ በፈገግታ ,

Fig 1. shows the categories appear to be related events in the www.fanabc.com.The numbers next to each category represent the approximate number of times the term was mentioned or appeared in the data that was web scraped. For example, the term "ዜና" (meaning news) appeared approximately 1800 times, while the term "ማራኪ አንቀፅ" (meaning market price) appeared approximately 80 times.

## 4.2.Data Preprocessing

After collecting the data the next is performing amharic preprocessing of text data and it involves several steps to ensure that the data is cleaned and ready for morphological segmentation. The first step is to preprocess the data sentence by sentence for each category. This is done using four sub-steps. The first sub-step involves removing all punctuation marks and special characters from the input text file using the Remove_punc_and_special_chars() function. The second sub-step involves removing all non-Amharic characters, ASCII characters, and numbers from the text using

the Remove_non_amharic_ascii_and_numbers() function. The third sub-step is the normalization of misspelled words and character encodings using the Normalize_char_level_missmatch() function.

The second step involves tokenizing the text into words from sentences. The resulting list contains 5802093 words before removing duplicates. To ensure that there is no redundancy in the data and to reduce the size of the data, all duplicate words are removed using the remove_duplicate() function. After removing duplicates, the resulting number of words in the list is reduced to 447137. These steps are crucial for preparing the data for further analysis and machine learning models.

Overall, these preprocessing steps are crucial for cleaning and standardizing the text data before performing any further analysis, such as morphological segmentation.

## 4.3.Model Development:

The process of model development for morphological segmentation involves leveraging unsupervised deep learning and rule-based segmentation techniques to create an effective system for segmenting words into their constituent morphemes.

The main objective of this methodology is to segment words into their morphemes or constituent parts, which is a fundamental task in natural language processing. The approach taken in this code is unsupervised, meaning that it does not rely on labeled data to learn the segmentation rules.

Unsupervised deep learning is used to build a multiclass classification model that can recognize and segment the morphemes in a given word. This is achieved by training a bidirectional LSTM neural network using the Keras and TensorFlow libraries. The model is trained using a dataset of Amharic words that have been preprocessed using the Tokenizer library. The Tokenizer library is used to convert the words into sequences of characters, which are then padded to a fixed length using the pad_sequences function. The padded sequences are then converted into a one-hot

encoding using the to_categorical function, which is suitable for multiclass classification problems.

The bidirectional LSTM neural network is used to learn the patterns and dependencies between the characters in a word to segment it into its constituent morphemes. The LSTM layer is designed to return sequences, which are then passed through a Dropout layer to prevent overfitting. Finally, a Dense layer is added to the model, which outputs a probability distribution over the classes of morphemes in the word.

In addition to the deep learning model, a rule-based segmentation algorithm is also utilized in this methodology. The algorithm is used to complement the deep learning model and to handle cases where the model may not be able to accurately segment a word. This is important because deep learning models can sometimes make mistakes or encounter unseen patterns, and the rule-based algorithm helps to ensure robustness and accuracy in the overall segmentation process.

## 5. How is the model working?

1. Algorithm to generate model using unsupervised deep learning techniques

```
from keras.utils.np_utils import to_categorical # Import to_categorical from keras.utils.np_utils
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM, Embedding, Bidirectional
from keras.callbacks import EarlyStopping
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from sklearn.model_selection import train_test_split
import re



# Load the Amharic dataset
infile = "WordTokenizedFile//forUnsupervised//words4"
```

```
with open(infile, 'r', encoding='utf-8') as f:
    data = f.readlines()

# Preprocess the data
tokenizer = Tokenizer(char_level=True)
tokenizer.fit_on_texts(data)
sequences = tokenizer.texts_to_sequences(data)
maxlen = max(len(seq) for seq in sequences)
X = pad_sequences(sequences, maxlen=maxlen, padding='post')
y = np.array([list(seq) for seq in X])
y = to_categorical(y, num_classes=len(tokenizer.word_index)+1)

# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# Define the model architecture
model = Sequential()
model.add(Embedding(input_dim=len(tokenizer.word_index)+1, output_dim=32,
input_length=maxlen))
model.add(Bidirectional(LSTM(units=32, return_sequences=True)))
model.add(Dropout(0.2))
model.add(Dense(units=len(tokenizer.word_index)+1, activation='softmax'))

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model
early_stop = EarlyStopping(monitor='val_loss', patience=3, verbose=1)
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_split=0.2,
callbacks=[early_stop])
```

```
# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Test loss: {loss}, Test accuracy: {accuracy}')

# Print the accuracy
acc = history.history['accuracy'][-1]
print(f'Training accuracy: {acc}')

# Predict sequences
y_pred = model.predict(X_test)
y_pred = np.argmax(y_pred, axis=-1)

# Convert predicted sequences to string format
pred_strings = []
for seq in y_pred:
    pred_string = "".join([tokenizer.index_word.get(i, '') for i in seq])
    pred_strings.append(pred_string)
```

This code is how to train a bidirectional LSTM neural network for a multiclass classification problem using Keras and TensorFlow. Specifically, it is trained on an Amharic language dataset.

First, the Amharic dataset is loaded from a file and preprocessed using the Tokenizer class from Keras. The sequences of words are converted into sequences of integers using the tokenizer.texts_to_sequences() method. Then, these sequences are padded to have the same length using the pad_sequences() method. The output labels are converted into a one-hot encoded format using the to_categorical() method.

The preprocessed data is then split into training and testing sets using the train_test_split() method from scikit-learn.

Next, the model architecture is defined using the Sequential class from Keras. It consists of an embedding layer followed by a bidirectional LSTM layer with dropout regularization, and a dense layer with softmax activation. The model is compiled with the Adam optimizer and categorical cross-entropy loss.

The model is trained on the training set using the fit() method. Early stopping is implemented using the EarlyStopping() method from Keras to prevent overfitting. The training history is saved in the history variable.

Finally, the model is evaluated on the testing set using the evaluate() method. The accuracy of the model is printed along with the training accuracy from the training history.

2. Rule based segmentation

```
def morphological_segmention(inputList):
    for input1 in inputList:

        seq = tokenizer.texts_to_sequences([input1])
        padded_seq = pad_sequences(seq, maxlen=maxlen, padding='post')

        # Predict the next characters
        pred = model.predict(padded_seq)

        # Convert the predicted indices to characters
        pred_chars = []
        for i in range(pred.shape[1]):
```

```
    idx = np.argmax(pred[0][i])
    if idx in tokenizer.index_word:
        pred_chars.append(tokenizer.index_word[idx])



#predected data
input1=''.join(pred_chars)

# RULE 1 - Take input as it is
print(input1)
collection = [input1]



# RULE 2 - Take out the right most suffix - From input 1
input2 = re.match("(.+)(ነት|ቸው| ባት|ሮች|ዋ|ችኋል|ዎች|ለህ|ም|ለን|ለት|ዊ)",input1)
if input2:
    print(input2.group(1)+'-'+input2.group(2))
    input2 = input2.group(1);
    collection.append(input2)
else:
    input2 = input1

# RULE 3 - Take out the inner most suffix
input3 = re.match('(.+)(ቹ|ውያን|ዎች|ዋ|ኝ|ሮች|ያ|ችን|ቸው)',input2)
input3 = re.match('(.+)(ች|ቸው ዊ|በት|ችሁ|ዋ)',input2) if not input3 else input3
if input3:
    print(input3.group(1)+'-'+input3.group(2))
    input3 = input3.group(1)
    collection.append(input3)
else:
```

```
        input3 = input2

    # RULE 4 - Take out the most left prefix - From input 1
    input4 =
re.match('(የ|ለ|ይ|አል|በስተ|እየ|ሳይ|አት|አስ|እንደ|እስኪ|ያል|ባለ,እንዲ|እያስ|በስተ|ወደ|ያስ|ት
ል|ስለ|እስክ|ሲ|እንድ)(.+)',input1)
    if input4:
        print(input4.group(1)+'-'+input4.group(2))
        input4 = input4.group(2)
        collection.append(input4)
    else:
        input4 = input1

    # RULE 5 - Take out the right most suffix - From input 4
    input5 = re.match('(.+)(ው|ዉ|ወ|አይ|ና|ሚ|ማ|ሊ|ነ|ች)',input4)
    if input5:
        print(input5.group(1)+'-'+input5.group(2))
        input5 = input5.group(1)
        collection.append(input5)
    else:
        input5 = input4

    # RULE 6 - Take out the inner most suffix - From input 4
    input6 = re.match('(.+)(ኛ|አቸዋል|ቹ|ችሁ|ውያን|ቻቸው| ይ|ቸው|ህ|ኞች|ለ|ት)',input5)
    if input6:
        print(input6.group(1)+'-'+input6.group(2))
        input6 = input6.group(1)
        collection.append(input6)
    else:
        input6 = input5
```

```
    # RULE 7 - Take out the inner most prefix - From input 1
    input7 = re.match('(አስ|ምት|በስተ|ወደ|ያለ|ማይ|የ|ሳት)(.+)',input6)
    input7 = re.match('(ያስ|እንዲ| ት|ያ|አላ|እስከ|በ|ተ|ከ)(.+)',input6) if not input7 else input7
    input7 = re.match('(ት|ሚ|እን|በት|ከ|ተ|ወ|አይ|የ)(.+)',input6) if not input7 else input7

    if input7:
        print(input7.group(1)+'-'+input7.group(2))
        input7 = input7.group(2)
        collection.append(input7)
    else:
        input7 = input4


    print(collection)

  return collection
```

This function seems to be implementing a morphological segmentation algorithm that splits a given word into its morphemes based on a set of predefined rules. Here's what the function does:

1. For each word in the input list, tokenize it and pad it to a maximum length using a pre-trained Keras model.
2. Predict the next characters in the word using the same Keras model.
3. Apply seven different rules to segment the word into morphemes:
    - Rule 1: Do nothing and take the input word as is.
    - Rule 2: Take out the rightmost suffix from the input word.

- Rule 3: Take out the innermost suffix from the word obtained in Rule 2.
- Rule 4: Take out the leftmost prefix from the input word.
- Rule 5: Take out the rightmost suffix from the word obtained in Rule 4.
- Rule 6: Take out the innermost suffix from the word obtained in Rule 5.
- Rule 7: Take out the innermost prefix from the input word.

4. Add all the segments obtained from the above rules to a list called "collection".
5. Print each segment obtained from the rules for the input word.

## 6. Experimental Result and Discussion

The model development for morphological segmentation used an 80-20 split of the data for training and testing purposes. Specifically, 80% of the clear subset's data was arbitrarily selected for the training set, while the remaining 20% was randomly selected for the testing set. The model was then trained using the training set and achieved a Test loss of 0.005 and Test accuracy of 0.99906. The Training accuracy achieved during the training process was 0.9985. These results suggest that the model performed exceptionally well on the testing set and has a high level of accuracy in classifying the Amharic language words.

Figure 2 depicts two subplots: one for accuracy and one for loss. It illustrates the model's accuracy and loss during training and validation, presented on the same graph. The accuracy graph demonstrates a gradual increase in accuracy over each training step, ultimately reaching a peak of 99.89% accuracy. Conversely, the loss graph demonstrates a steady decrease in the loss over the course of training, ultimately reaching a minimum of 0.05%.

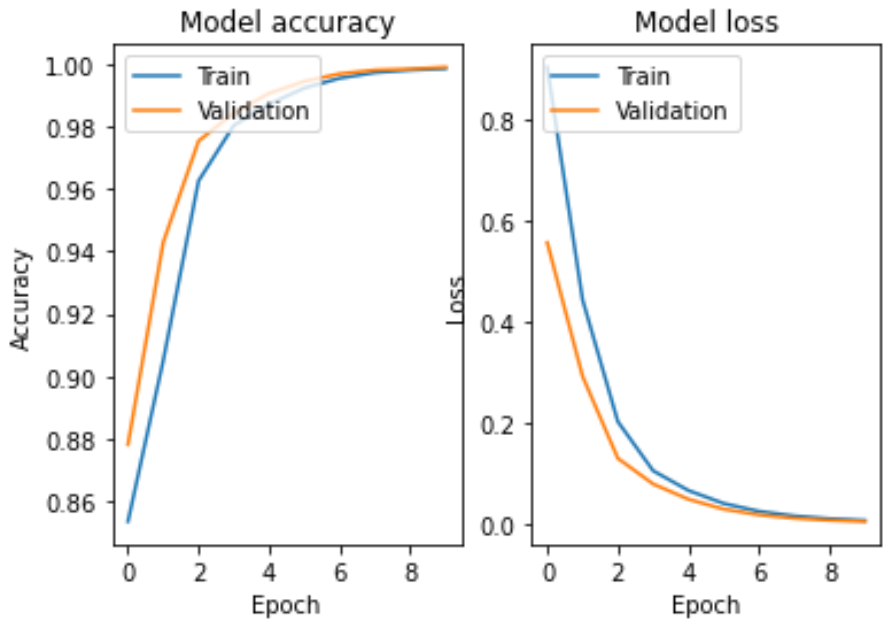


Fig 2. Model accuracy and loss

## 7. Conclusion

In conclusion, the developed model for Amharic morphological segmentation integrates unsupervised deep learning and rule-based segmentation techniques to build a reliable and precise system for segmenting words into their constituent morphemes. The model achieved an accuracy of 99.89% using unsupervised deep learning techniques, and this was combined with rule-based segmentation to predict the segmentation of Amharic words accurately.