MORPHOLOGICAL SEGMENTATION WITH LSTM NEURAL NETWORKS FOR TIGRINYA

Yemane Tedla and Kazuhide Yamamoto

Natural Language Processing Lab, Nagaoka University of Technology,
Nagaoka City, Japan

ABSTRACT

Morphological segmentation is a fundamental task in language processing. Some languages, such as Arabic and Tigrinya, have words packed with very rich morphological information. Therefore, unpacking this information becomes a necessary task for many downstream natural language processing tasks. This paper presents the first morphological segmentation research for Tigrinya. We constructed a new morphologically segmented corpus with about 45,127 manually segmented tokens. Conditional random fields (CRF) and window-based long short-term memory (LSTM) neural networks were employed separately to develop our boundary detection models. We applied language-independent character and substring features for the CRF and character embeddings for the LSTM networks. Experiments were performed with four variants of the Begin-Inside-Outside (BIO) chunk annotation scheme. We achieved 94.67% F1 score using bidirectional LSTMs with window approach to morpheme boundary detection.

KEYWORDS

Morphological segmentation, Tigrinya language, LSTM neural networks, CRF

1. INTRODUCTION

Morphological processing at the word level is usually the initial step in the different stages of natural language processing (NLP). Morphemes constitute the minimal meaning-bearing units in a language [1]. In this paper, we focus on the task of detecting morphological boundaries, which is also referred to as *morphological segmentation*. This task involves the breaking down of words into their component morphemes. For example, the English word “reads” can be segmented into “read” and “s”, where “read” is the stem and “-s” is an inflectional morpheme, marking third person singular verb.

Morphological segmentation is useful for several downstream NLP tasks, such as morphological analysis, POS tagging, stemming, and lemmatization. Segmentation is also applied as an important preprocessing phase in a number of systems including machine translation, information retrieval, and speech recognition. Segmentation is mainly performed using rule-based approaches or machine learning approaches. Rule-based approaches can be quite expensive and language-dependent because the morphemes and all the affixation rules need to be identified to disambiguate segmentation boundaries. Machine learning approach, on the other hand, is data-driven wherein the underlying structure is automatically extracted from the data. In this paper, we present supervised morphological segmentation based on CRFs [2] and LSTM neural networks [3]. Since morphemes are sequences of characters, we address the problem as a sequence tagging task and propose a fixed-size window approach for modeling contextual information of characters. CRFs are well-suited for this kind of sequence aware classification tasks. We also exploit the long-distance memory capabilities of LSTMs for modeling boundaries of morphemes.

Our main contributions are the following.

1. We constructed the first morphologically segmented corpus for Tigrinya. This corpus is annotated with boundaries that identify prefix, stem, and suffix morphemes.

2. We present the first supervised morphological segmentation research for Tigrinya. Extensive experiments were performed for different annotation schemes and learning approaches exploiting character embeddings as the sole features for LSTMs. We also compare our results with feature rich CRF-based segmentation employing language-agnostic character and substring features.

3. Tigrinya is an understudied language. Therefore, through this fundamental research, we hope to contribute in bettering the understanding of the properties and language processing challenges of the language and encourage further research.

This paper is organized as follows. In section 2, the Tigrinya morphology will be briefly introduced. In section 3, relevant previous works will be discussed. Section 4 describes the CRF and LSTM based methods employed in this research. In the sections that follow, the experimental settings will be explained and the results discussed. Finally, concluding remarks will be provided.

2. TIGRINYA LANGUAGE

2.1. Writing system

Tigrinya is one of the few African languages that still use an indigenous writing system for education and daily communication. The writing system, known as the Ge'ez script, is adopted from the ancient Ge'ez language, which is currently used as a liturgical language. The Ge'ez script is an abugida system in which each letter (alphabet) represents a consonant-vowel (CV) syllable. The Tigrinya alphabet chart, known as "Fidel", comprises of about 275 symbols. Gemination is not explicit in the Ge'ez script; however, this limitation does not seem to pose a problem for native speakers. In this paper, Tigrinya words are transliterated to Latin characters according to the SERA scheme with the addition of *"I"* for the explicit marking of the epenthetic vowel known as *"SadIsI"*. The SERA transliteration scheme is available at ftp://ftp.geez.org/pub/sera-docs/sera-faq.txt. We directly apply labeling to the Latin transliterations of Tigrinya words and not the Ge'ez script. The Ge'ez script is syllabic, and, in many cases, the boundary has fusional properties resulting in alterations of characters at the boundary. For example, the word *"sebere"* (He broke) would be segmented as *"seber"* + *"e"* because the morpheme *"e"* represents grammatical features. However, this morpheme cannot be isolated using the Ge'ez script because the last characters *"re"* forms a single symbol in the Ge'ez script, and segmenting *"sebere"* as *"sebe"* + *"re"* is not a correct analysis.

2.2. Tigrinya morphology

Tigrinya is a Semitic language spoken by over 7 million people in Eritrea and Ethiopia. The morphology of Semitic languages, known as "root-and-pattern" morphology, has distinct non-concatenative properties that intercalate consonantal roots and vowel patterns [4]. For example, in Tigrinya, the words *"sebere"* (he broke) and *"sebira"* (she broke) share a common tri-consonantal root or radical *"s-b-r"* but have different sequences of vowel patterns (*"e-e-e"* and *"e-i-a"*) that are inserted in-between the radicals. In addition to such a unique infixation, words are formed by affixing morphemes of prefix, suffix, as well as circumfix. These morphemes represent morphological features including gender, person, number, tense, aspect, mood, voice, and so on. Furthermore, there are clitics of mostly prepositions and conjunctions that can be affixed in other words. These components are arranged in the following manner;

(proclitics)(prefix/circumfix)(root-with-infix)(circumfix/suffixes)(enclitics)

Specifically, the order of morpheme slots is defined by [5] as follows.

(prep|conj)(rel)(neg)sbjSTEMsbj(obj)(neg)(conj)

The slots "prep" and "conj" are affixes of prepositions or conjunctions attached before or after the word. The "rel" indicates a relativizer (the prefix *"zI"*) corresponding in function to the English demonstratives like that, which, and who. The "sbj" on either side of the STEM are prefix and/or suffix of the four verb types namely; perfective, imperfective, imperative, and gerundive. As shown in examples 1-4, the perfective and gerundive verbs conjugate only on suffixes (examples 1 and 4) while imperfective verbs undergo both the prefix and suffix inflections (example 2). The imperatives show the suffix only conjugations or change prefix as well (example 3). In addition to the verb type, these fusional morphemes convey gender, person, and number information.

1. *seber* + *u* (they broke)

2. *yI* + *sebIr* + *u* (they break)

3. *yI* + *sIber* + *u* (break/let them break)

4. *sebir* + *omI* (they broke)

Moreover, Tigrinya independent pronouns have a pronominal suffix of gender, person, and number as shown in examples 5 and 6 (SUF).

5. *nIsu* (he) → *nIs* + *u/SUF*

6. *nIsa* (she) → *nIs* + *a/SUF*

The word order typology is normally subject-object-verb (SOV), though there are cases in which this sequence may not apply strictly [6, 7]. Changes in "sbj" verb affixes, along with pronoun inflections, enforce subject-verb agreements. One aspect of the non-concatenative morphology in Tigrinya is the circumfixing of negation morpheme in the structure *"ayI-STEM-nI"* [5]. Some conjunction enclitics such as *"do; ke; Ke"* can also be found in Tigrinya orthography as free or bound suffix morphemes. For example,

KeyIdudo? → *keyId* + *u* + *do* (did he go?)

nisuKe? → *nis* + *u* + *Ke* (what about him?)

The pronominal object marker "obj" is always suffixed to the verb as shown in the following examples. According to [6], Tigrinya suffixes of object pronoun can be categorized into two constructs. The first is described in relation to verbs (examples 7, 8 and 9) and another indicates the semantic role of applicative cases by inflecting for prepositional enclitic "*lI*" + a pronominal suffix as in example 10.

7. *beliOI* + *wo/*obj (he ate [something])

8. *hibu* + *wo /*obj (he gave [something] to him)

9. *hibu* + *ni /*obj (he gave me [something])

10. *beliOI* + *lu /*obj (he ate for him/he ate using [it])

Tigrinya words are also produced by derivational morphology. There are up to eight derivational categories that can be generated from a single verb [8]. For example, the passive form of perfective (example 11) and gerundive verbs (example 12) is constructed by prefixing "*te"* to the main verb. Furthermore, adverbs can be derived from nouns by prefixing "*bI-*" which has similar functionality as the English "-ly" suffix (example 13).

11. *zekere* (he remembered) → *te* + *zekere* (he was remembered)

12. *zekiru* (he remembered) → *te* + *zekiru* (he was remembered)

13. *bIHaqi* (truly) → *bI* + *Haqi*

In this work, we deal with boundaries of prefix, stem, and suffix morphemes. Inflections related to infixes (internal stem changes) are not feasible for this type of segmentation.

In conclusion, all the morphological and derivational processes generate a large number of complex word forms. According to [8], Tigrinya verb inflections, derivations, and combinations of the slot order can produce more than 100,000 word forms from a single verb root. The ambiguity and difficulty in relation to segmentation is briefly discussed in the following section.

2.3. Morphological ambiguity

In segmentation, ambiguity may occur at the word-level or due to sentential context. In Tigrinya, a major source of ambiguity is when certain character sequences of morphemes are natively present as part of words. In this case, the characters do not represent grammatical features and hence segmentation should not be applied. Consider the words *"bIrIhanI"* (light) and *"bIHayli"* (by force). The same prefix *"bI-"* which is an inseparable part of the noun *"bIrIhanI"*, represents an adverb of manner (by) in the second word. Moreover, morphemes may also appear as constituents of other morphemes. For instance, the noun suffix *"-netI"* (example 16) contains the sub-morph *"-etI"* that can have the role of a suffix for conjugation of third person, feminine, singular attributes as in example 15. Note that *"-netI"* in example 14 is not a morpheme.

14. *genetI* → *genetI*/NOUN_paradise
15. *wesenetI* → *wesen*/STEM_decided + *etI*/SUF_she
16. *naxInetI* → *naxI*/STEM_independence + *netI*/NOUN-SUF

Moreover, the lack of gemination marking may introduce segmentation ambiguity. For example, the word *"medere"* can be interpreted as the noun "speech" or the phrase "he gave a speech" if the *"de"* in *"medere"* is geminated. A computational system must discern both cases such that the phrase is segmented while the noun is left intact. Resolving such cases may require more context or additional linguistic information such as part-of-speech. In this work, we would like to avoid resorting to any language-specific knowledge. Therefore, in the LSTM approach, we use character embeddings to capture contextual dependencies of characters.

As explained earlier, Tigrinya words have multiple consecutive morpheme slots. This pattern causes under-segmentation confusion due to the numerous intermediate splits comprising atomic and composite morphemes. Table 1 lists some of the possible segmentations for the word token *"InItezeyIHatetIkayomI"* (if you did not ask them).

Table 1. Examples of intermediate splits caused by under-segmentation. The italicized second row is the expected segmentation.

InItezeyIHatetIkayomI
InIte-zeyI-HatetI-ka-yomI
InIte-zeyIHatetIkayomI
InIte-zeyI-HatetIkayomI
InIte-zeyI-HatetI-kayomI
InItezeyIHatetIka-yomI
InItezeyIHatetI-kayomI
InItezeyI-HatetIkayomI

Furthermore, in Tigrinya, compound words are often written attached. For example, *"betI* (house) *megIbi* (food)*"* collectively translates to "restaurant" in English. In the orthography, these words can be found either separate or attached. We queried for *"betI-"* starting words in a

text corpus containing about 5 million words extracted from the Haddas Ertra newspaper, which is published in Eritrea. The previous word, for instance, occurs delimited by a space in about 95% of the response. Another compound word *"betI SIHIfetI"* (office) was found attached in about 24% of the response, which is not a small portion. Although the words are not grammatical morphemes, we believe segmenting (normalizing) these words would be useful for practical reasons such as mitigating data sparseness.

3. RELATED WORKS

The work of [1] introduced the unsupervised discovery of morphemes based on minimum description length (MDL) and likelihood optimization. This method lacks the handling of representing contextual dependencies, such as stem and affix orders. Although several other unsupervised segmentation approaches have been proposed, it was shown by [9] that minimally supervised approaches provided better performance compared to solely unsupervised methods applied on large unlabeled datasets. For example, unsupervised experiments for Estonian, that achieved 73.3% F1 score with 3.9 million words, was outperformed by a supervised CRF that attained 82.1% with just 1000 word forms. This work also showed that semi-supervised approaches that use both annotated and unannotated data can be leveraged to improve upon simply using completely unsupervised methods. In related works for Semitic languages of Hebrew and Arabic, [10] uses a probabilistic model where segmentation and morpheme alignments are inferred from the shared structure between both languages using parallel corpus with and without annotation. Recent works on neural models rely on the use of some form of embedding for extracting relevant features. [11] demonstrated a generic window-based neural architecture that is applied to several NLP tasks while avoiding explicit use of feature engineering. Their system trains on large unlabeled data to learn internal representations. We have also adopted a similar window-based approach although with the input of character sequence window instead of words as in [11]. In [12], state-of-the-art results were achieved with a neural model that learns POS related features only from character-level and sub-word embeddings. Other research, however, argue that enriching embeddings with additional morphological information boosts performance. [13] demonstrates this by using the results of a morphological analyzer to further improve candidate ranking in a morphological disambiguation task for Arabic. In a research for Burmese word segmentation, [14] address the problem by employing binary classification with classifiers such as CRFs. The tagset restriction to the binary was mainly due to data size. Our data is similarly small size corpus; however, we report experiments with several schemes to investigate the effect of using simple to more expressive tags in the Tigrinya morpheme segmentation.

Amharic and Tigrinya are closely related languages. These languages share a number of grammatical features and vocabularies. [16] presented morphological rule learning and segmentation based on inductive logic programming where rules or affixes were learnt by exposing easy-to-complex examples incrementally to an intelligent teacher. Their system for affix segmentation achieved a performance of 0.94 precision and 0.97 recall measures. Tigrinya remains an under-studied and under-resourced language from the NLP perspective. However, as regards to morphological processing, [8] employed finite state transducers (FSTs) to develop, "*HornMorpho"*, a morphological analysis and generation system for Tigrinya, Amharic, and Oromo languages. The FST empowered by feature structures was effectively adopted to process the unusual non-concatenative root-and-pattern morphology. The Tigrinya module of *HornMorph 2.5* performs the full analysis of Tigrinya verb classes [5]. The system was tested on randomly selected 400 word forms with 329 of them being non-ambiguous. The analysis revealed remarkably accurate results with a few errors due to unhandled FSTs, which can be

integrated. Our approach is different from *HornMorph* in at least two aspects. First, our task is limited to identifying morphological boundaries. The results amount to partially analyzed segments although these segments are not explicitly annotated for grammatical feature. The annotation could be pursued with further processing of the output or training on morphologically annotated data, which is currently missing for Tigrinya. Second, the use of the FSTs relies heavily on the linguistic knowledge of the language in question. This would require time consuming manual construction of language-specific rules, which is more challenging for root-and-pattern morphology. In contrast, we follow a data-driven (machine learning) approach to automatically extract features of the language from a relatively small boundary annotated data. Moreover, on the limitations of *HornMorph*, [5] noted that analysis incurs a "considerable time" to exhaust all options before the system responds. Besides *HornMorph*, there was an attempt to use affix based shallow segmentation in the pre-processing phase for improving word alignment in the English-Tigrinya statistical machine translation [16].

4. METHOD

4.1. Morphologically segmented corpus

There is no publicly available morphologically segmented resource for Tigrinya. Therefore, we based our studies on a new morphologically segmented corpus developed in-house. The first version of this corpus comprises over 45,000 tokens derived from randomly selected 2774 sentences of the POS tagged Nagaoka Tigrinya Corpus (NTC) available at4 http://eng.jnlp.org/yemane/ntigcorpus.

For the purpose of boundary detection, we employed character-based BIO chunking scheme, which allows us to exploit character dependencies and alleviate out-of-vocabulary (OOV) problems by reducing the morpheme vocabulary to about 60 Latin characters that cover the transliteration mapping we adopted.

4.2. Tagging schemes

The popular IOB tagging scheme is used to annotate the Beginning (B), Inside (I), and outside (O) of chunks in tasks such as base phrase chunking and named entity recognition. Similarly, we address morpheme boundary detection by annotating every character in morpheme chunks with the appropriate IOB label. There are different variants of the IOB scheme including BIO, IOB, BIE, IOBES, and so on. There is no general consensus as to which variant performs best. [11] used the IOBES format as it encoded more information whereas extensive evaluations by [17] showed that the BIO has superior results compared to the IOB scheme. Furthermore, the BIES scheme gave better results for Japanese word segmentation [18].

Table 2. Annotating with different tagging schemes

Scheme	**tImali InItezeyIHatete (if he did not ask yesterday)**																			
Word	t	I	m	a	l	i	**I**	n	I	t	e	**z**	e	y	I	**H**	a	t	e	t	**e**
BIE	B	I	I	I	I	E	**B**	I	I	I	E	**B**	I	I	E	**B**	I	I	I	I	**E**
BIES	B	I	I	I	I	E	**B**	I	I	I	E	**B**	I	I	E	**B**	I	I	I	I	**S**
BIO	O	O	O	O	O	O	**B**	I	I	I	I	**B**	I	I	I	**B**	I	I	I	I	**I**
BIOES	O	O	O	O	O	O	**B**	I	I	I	E	**B**	I	I	E	**B**	I	I	I	E	**S**

We experimented with four schemes, namely, BIE, BIES, BIO, and BIOES to explore the modeling of morpheme segmentation in a low-resource setting with morphologically rich language Tigrinya. In our case, the B, I, and E tags marks Begin, Inside, and End of multi-character morphemes. The S tag annotates single character morpheme and the O tag assigns character sequences outside of morpheme chunks. An example of using these annotations is presented in Table 2.

4.3. Character embeddings

Morphological segmentation is primarily character-level analysis. For example, in Tigrinya the characters *"zI"* have a grammatical role when used as a relativizer that is prefixed only to the perfective (example 17) or imperfective verbs (example 18). Therefore, all other occurrences of *"zI"*, such as the noun *"zInabI"* (rain) should not be segmented. Consequently, recognizing the shape of relativized perfectives and imperfectives becomes crucial.

17. *zIsebere* 'that broke' → *zI* + *sebere*/PERFECTIVE
18. *zIsebIrI* 'that breaks' → *zI* + *sebIrI*/IMPERFECTIVE
19. **zI* + *yIsIberI*/IMPERATIVE – invalid
20. *zInabI* 'rain' → **zI* + *nabI* - incorrect segmentation

Character-level information must be extracted to learn such informative features useful for identifying boundaries. We generated past and future fixed-width character context for each character. The concatenated contextual characters form a single feature vector for the central target character. For example, the features of the word *"selamI"* (peace, hello) for each character are generated as depicted in Table 3. We also showed the corresponding label of the central character in the BIE scheme. The **Boldface** character represents the central character with its left (past) and right (future) context of width five characters. Characters are padded with underscores (_) to complete the remaining slots depending on the window size.

Table 3. An example of generated fixed window character sequences with assigned label

Window											Label
_	_	_	_	_	**s**	e	L	a	m	I	B
_	_	_	_	S	**e**	l	A	m	I	_	I
_	_	_	s	E	**l**	a	M	I	_	_	I
_	_	s	e	L	**a**	m	I	_	_	_	I
_	S	e	l	A	**m**	I	_	_	_	_	I
s	E	l	a	M	**I**	_	_	_	_	_	E

We settled for a window size of five as increasing it beyond five did not result in significant improvements. Moreover, the dimension of optimal character embedding was decided by hyper-parameter tuning experiments. We initialized the embedding layer from a lookup table for integer (index) representations of the fixed-width character vectors. The embedding is then fed to an LSTM after passing through a dropout layer.

4.4. CRF

CRFs are probabilistic approaches capable of modeling context-dependent sequence labeling [2]. In a morphological segmentation, the model is trained to predict a sequence of output tags (IOB labels) $\boldsymbol{y} = y_0, y_1, \ldots, y_T$ from a sequence of feature characters $\boldsymbol{x} = x_0, x_1, \ldots, x_T$. The training task is to maximize the log probability $\log(p(\boldsymbol{y}|\boldsymbol{x}))$ of the valid label sequence. The conditional probability is computed as:

$$\boldsymbol{p(y|x)} = \frac{\boldsymbol{e^{score(x,y)}}}{\boldsymbol{\sum_{y'} e^{score(x,y')}}} \quad (1)$$

The equation for the scoring function score is given as:

$$\boldsymbol{score(x,y)} = \sum_{\boldsymbol{i=0}}^{\boldsymbol{T}} \boldsymbol{A_{y_i,y_{i+1}}} + \sum_{\boldsymbol{i=1}}^{\boldsymbol{T}} \boldsymbol{P_{i,y_i}} \quad (2)$$

$A_{y_i,y_{i+1}}$ denotes the emission probability of the state change from label i to label j while $P_{i,j}$ is the transition probability denoting the score of the j^{th} label of the i^{th} word.

4.5. LSTM neural network

Recurrent Neural Networks (RNNs) are feed forward neural networks with feedback cycles to capture time dynamics using back-propagation through time. This recurrent connection allows the network to employ the current inputs as well as previously encountered states. Given the input vector X_t, (in our case, a window of five characters left and right of the target character), the hidden state h_t at each time step t, can be expressed by equation 4.

$$X_t = [\, x_{t-n}, x_{t-(n-1)}, \dots, x_t, \dots, x_{t+(n-)}, x_{t+n} \,] \quad (3)$$
$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (4)$$

where the W terms represent the weight matrices, b is a bias vector and σ is the activation function. However, due to the gradient vanishing (very small weight changes) or exploding problems (very large weight changes), long distance dependencies are not properly propagated in RNNs. [3] introduced LSTMs to overcome this gradient updating problem. The neurons of LSTMs called memory blocks are composed of three gates (forget, input and output) that control the flow of information and a memory cell with self-recurrent connection. The formulae of these four components are given in equations 5 to 9. In these equations, the input, forget, output and cell activation vectors are denoted by i, f, o and c respectively; σ and g represent sigmoid and tanh functions respectively; $\odot$ operation is the Hadamard product; W stands for the weight matrices and b is the bias.

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \quad (5)$$
$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \quad (6)$$
$$c_n = g(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \quad (7)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot c_n \quad (8)$$
$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \quad (9)$$
$$h_t = o_t \odot g(c_t) \quad (10)$$

The LSTM architecture we propose to use is illustrated in Figure 1. Character-level features (embeddings) generated by the embedding layer are fed to the forward LSTM network. Dropout layers are adjusted before and after the LSTM layer to regularize the model and prevent overfitting. The output of the network is then processed by a fully connected (Dense) layer and finally tag probability distributions over all candidates are computed via the softmax classifier.

Due to the capacity to work well with long distance dependencies, LSTMs have achieved state-of-the-art performance in window based approaches as well as sequence labeling tasks including morphological segmentation [19], part-of-speech [20] and named entity recognition [21].

Figure 1: The LSTM network

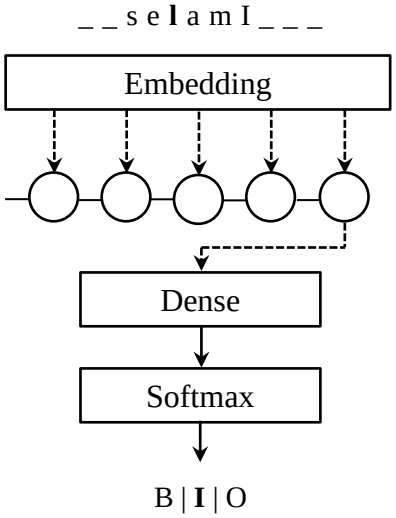
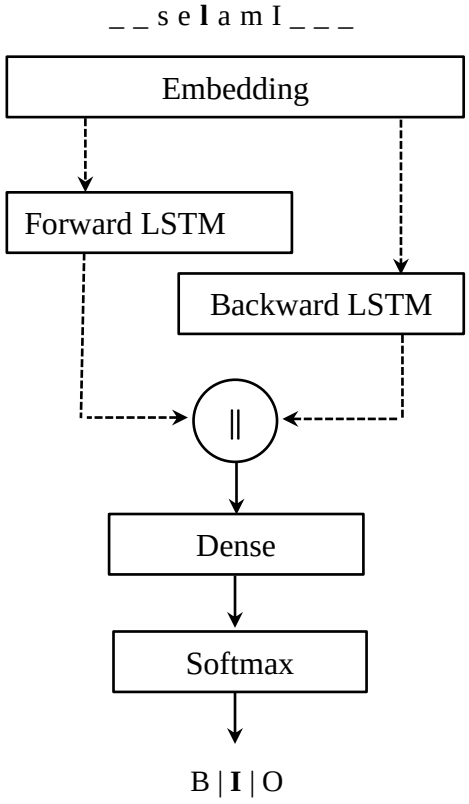


Figure 2: The BiLSTM neural network



4.6. Bidirectional LSTM

LSTMs encode the input representation in the forward pass. In Bidirectional LSTM network (BiLSTM), the input vector is processed in both forward and backward passes and presented separately to hidden states. The mechanism is useful in encoding past (forward) and future

(backward) information of the input vector. We illustrate the architecture of the BiLSTM we propose to use in Figure 2.

Both the forward and backward layer outputs are calculated by using the standard LSTM updating equations 5 to 9. Accordingly, for every time step t the forward and backward networks take the layer input X_t and then output h_t. The cell takes the input state c_n and the cell output c_t as well as the previous cell output o_t for training and updating parameters. Like in LSTMs, the output of each network in BiLSTM can be expressed as:

$$y_t = \sigma\left(W_{hy}h_t + b_y\right) \quad (11)$$

where W is the weight matrix, b is the bias vector of the output layer and σ is the activation function of the output layer. Both the forward and backward networks operate on input state of X_t and generate the forward output $\overrightarrow{h_t}$ and the backward output $\overleftarrow{h_t}$. Then the final output, y_t from both networks is combined using operations such as multiplication, summation or simple concatenation as given in equation 12. In our case, σ is a concatenating function.

$$y_t = \sigma(\overrightarrow{h_t}, \overleftarrow{h_t}) \quad (12)$$

This context-dependent representation from both networks is passed along to the fully-connected hidden layer (Dense) which then propagates it to the output layer.

5. EXPERIMENTS

In this section, the used data and the experimental settings are reported. Initially, we set training epochs to hundred and configured early stopping if training continued without loss improvement for 15 consecutive epochs. We used Keras (https://www.keras.io/) to develop and Hyperas (https://github.com/maxpumperla/hyperas) to tune our deep neural networks. Hyperas, in turn, applies hyperopt for optimization. The algorithm used is Tree-structured Parzen Estimator approach (TPE) over five evaluation trails (https://papers.nips.cc/paper/4443-algorithms-for-hyperparameter- optimization.pdf).

5.1. Settings

Datasets

The text used in this research is extracted from the NTC corpus. The NTC consists of around 72,000 POS tagged tokens collected from newspaper articles. Our corpus contains 45,127 tokens, of which 13,336 tokens are unique words. Training is performed with ten-fold cross validation, where about 10% of the data in every training iteration is used for development. We further split the development data into two equal halves allotting one set for validation during training and the other half for testing the model's skill.

Table 4. Training data splits as per the count of the fixed-width character windows (vectors) used as the actual input sequences

Data size			
All	Training	Validation	Test
100%	90%	5%	5%
289,158	260,242	14,458	14,458

The evaluation reports in this paper are based on the final test set results. For every target character in a word, a fixed-width character window is generated as shown in Table 3. The data statistics according to the generated character windows is presented in Table 4.

Hyper-parameters

CRF: We performed preliminary experiments for deciding the regularization algorithm (L1 or L2). Similarly, we compared C-parameter values of [0.5, 1, 1.5]. Consequently, L2 regularization and C value of 1.5 were selected as these values gave better results. The character and substring features we considered spanned a window size of five. The full list of the character features is given as follows.

a. Left context: *-5 to 0, -4 to 0, -3 to 0, -2 to 0, -1 to 0*
b. Right context: *0 to 5, 0 to 4, 0 to 3, 0 to 2, 0 to 1*
c. Left + Right context: *a + b*
d. N-grams: *bi-grams*

LSTM: In order to search for the parameters that yield optimal performance, we explored hyper parameters that include embedding size, batch size, dropouts, hidden layer neurons, and optimizers. The complete list of the selected parameters for LSTM is summarized in Table 5. We achieved similar results for the BIE and BIES tunings as shown in the table. However, the BIO and BIOES schemes showed difference in the tuning results and, therefore, these were used separately.
Embeddings: We tested several embedding dimensions from the set {50, 60, 100, 150, 200, 250}. Separate runs were made for all types of tag sets.

Table 5. LSTM Hyperparameters selected by tuning

Parameter	**BIE/BIES**	**BIO**	**BIOES**
Window	5	5	5
Character embedding	150	150	250
hidden layer size	32	128	128
Batch size	32	256	256
optimizer	adam	adam	RMSProp

Dropouts: Randomly selecting and dropping-out nodes have proven effective at mitigating overfitting and regularizing the model [22]. We applied dropout on the character embedding as well as on the inputs and outputs of the LSTM/BiLSTM layer. For example, the selected dropouts for the BIO-based tunings are 0.07 and 0.5 for the embedding and LSTM output layers respectively. The dropout probabilities are selected from a uniform distribution over the interval [0, 0.6].
Batch size: We ran tuning for the batch size of the set {16, 32, 64, 128, 256}.
Hidden layer size: We searched for the hidden layers size from the set {64, 128, 256}.
Optimizers and learning rate: We investigated the stochastic gradient descent (SGD) with stepwise learning rate and other more sophisticated algorithms such as AdaDelta [23], Adam [24], and RMSProp [25]. The SGD learning rate was initialized to 0.1, momentum of 0.9 with rate updates for every 10 epochs at a drop rate of 0.5. However, the SGD setting did not result in significant gains compared to the automatic gradient update methods.

5.2. Baseline

The baseline considered in this study is the CRF model trained on only character features with a window size of five. The window size is kept the same with the neural LSTMs to compare the

models subject to the same information. The baseline experiments for the CRF achieved an F1 score of around 60% and 63% using the BIOES and BIO tags respectively. As for the BIE tags, the performance reached about 75.7%. All the subsequent CRF-based experiments employed two auxiliary features, which were the contextual and n-gram characters listed in section 5.1. As depicted in Table 6, significant performance enhancements were achieved with the auxiliary features. However, compared to the LSTM based experiments, the window five based CRF results were still suboptimal.

5.3. Evaluation

We report boundary precision, boundary recall and boundary F1 scores as given by equation 13 to 15. Precision evaluates the percentage of correctly predicted boundaries with respect to the predicted boundaries and recall measures the percentage of correctly predicted boundaries with respect to the true boundaries. F1 score is the harmonic mean of precision and recall and can be interpreted as their weighted average. The I or O classes are found more than double of the B classes. Therefore the results are presented using weighted macro average to account for any class imbalance. The correct predictions are the count of true positives while the actual (true) boundaries are these true positives added with the false negatives. Finally, the system proposed predictions can be found from the sum of the true positives and the false positives.

$$precision = \frac{correctly\ predicted\ boundaries}{predicted\ boundaries} \quad (13)$$

$$recall = \frac{correctly\ predicted\ boundaries}{true\ boundaries} \quad (14)$$

$$F1\ score = \frac{2 * (precision * recall)}{precision + recall} \quad (15)$$

6. RESULTS

We experimented and compared the performance of three models trained with four different tagging strategies as explained earlier. The results of ten-fold cross-validation are summarized in Table 6 with P, R, F1 representing precision, recall, and F1 score respectively.

Table 6. Results of CRF, LSTM and BiLSTM experiments with four BIO schemes.

Tagset	**CRF**			**LSTM**			**BiLSTM**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
BIE	92.62	92.62	92.62	94.38	94.37	94.37	94.68	94.68	**94.67**
BIES	92.44	92.44	92.44	94.22	94.20	94.20	94.60	94.59	**94.59**
BIO	84.88	84.88	84.88	90.91	89.96	89.96	90.16	90.11	**90.11**
BIOES	83.60	83.60	83.60	88.26	88.21	88.21	88.45	88.39	**88.39**

Generally, we observed the choice of segmentation schemes affecting the model performance. Overall, using the BIE scheme resulted in the best performance in all tests. Although the BIOES tagset is the most expressive, since the corpus is rather small, the simpler tagsets showed better generalization over the dataset. The BIE-CRF model achieved 92.62% in the F1 score while the performance of the BIO and BIOES-based CRF fell to about 84.88% and 83.6% respectively.

The CRF model using the BIO scheme outperformed the CRF model using the BIOES by 1.28% absolute change. A majority of the errors are associated with under-segmentation and confusion between I (inside) and O (outside) tags. Masking these differences by dropping the O tag has proved useful in our experiments. As a result, the CRF model using the BIE scheme outperformed the BIO-based CRF by around 7.7 percentage point. We compared the regular LSTM with its bidirectional extension. The overall results showed that the BiLSTMs performed slightly better than the regular LSTMs sharing the same scheme. In the window-based approach, the past and future context is partly encoded in the feature vector. This window was fed to the regular LSTM at training time making the information available for both networks quite similar. This may be the reason for not seeing much variation between the two models. Nevertheless, the additional design in BiLSTMs to reverse-process the input sequence allows the network to learn more detailed structures. Therefore, we saw slightly improved results with the BiLSTM network over the regular LSTM. As with CRFs, a significant increase in the F1 score was achieved when changing from the BIOES to the BIE scheme. In both LSTM models, we observed a gain in performance of over 6 percentage point. Overall, in these low-resource experiments, the model generalized better when the data was tagged with the BIE scheme as this reduced the model complexity introduced by the O tags. The BiLSTM model using the BIE scheme performed superior to all others scoring 94.67% in the F1. This result was achieved by employing only character embeddings to extract morphological features. On the other hand, the CRF model which used a rich set of character + bi-gram + substring features scored around 92.62%. For a fair comparison, we used features of the CRF model that spanned the same window of characters as the LSTM models. In other words, we avoided heavy linguistic engineering on the CRF side. Although the features were not linguistically motivated, achieving an optimal result this way still requires many trials and better design of features. It is to be noted that additional hand-crafted features and wider windows for the CRF model may produce better results than what has been reported. However, from the LSTM results, we saw that forgoing feature engineering and extracting features using character embeddings could sufficiently encode the morphological information desired for high performance boundary detection.

6.1. Error analysis

The models using the BIE/BIES or the BIO/BIOES tagsets achieved comparable results. We analyzed the effect of the tagset choice by comparing the BiLSTM models of the BIE and BIO tagging schemes. For easier comparison, the confusion matrices (in percentage) of both experiments are presented jointly in Table 7. The values are paired in X/Y format where X is a value from the set {B, I, E} in the BIE scheme and Y is one of {B, I, O} in the BIO scheme. The number of I or O classes is about two-fold greater than the B class. Therefore, there is more confusion stemming from those tags. The B tag represents the morpheme segmentation boundaries; therefore, the under-segmentation and over-segmentation errors can be explained in relation to the B tags. Looking at the BIE figures, around 91% of the predictions for the B tags are correct. In other words, 9% of the true B tags were confused for the I and O tags, which together amounts to the under-segmentation errors. These types of errors are almost doubled with the use of the BIO scheme. On the other hand, the over-segmentation errors are about 5.9% and 7.4% in the BIE and BIO scheme respectively. Over-segmentation occurs when true I, O, or E tags are confused with the B tag. In this case, morpheme boundaries are predicted while the characters actually belong to the inside, outside, or end of the morpheme. These results show that the under-segmentation errors contribute more compared to the over-segmentation errors.

In the BIE scheme results, we observe that the I and B tags are correctly predicted with relatively higher accuracies than with the BIO scheme. Specifically, the prediction accuracy of the B tag (boundaries) increased from 82.6% to 91.2% (8.6% absolute change). The main reason is the considerable reduction of errors introduced due to the confusions of tag I for O or vice-versa in the BIO scheme. This comparison suggests that improved results can be achieved by

denoising the data from the O tag. Therefore, for a low resource scenario of boundary detection, starting with the BIE tags may be a better choice for improving model generalization compared to more complex schemes that are feasible for larger datasets.

Table 7. BiLSTM Confusion matrix comparison for **BIE** (**boldfaced**) and BIO schemes in %. Rows and columns represent predicted and true values respectively.

		Predicted		
		B/B	**I**/I	**E**/O
True	**B**/B	**91.20**/82.60	**4.54**/9.94	**4.26**/7.46
	I/I	**2.83**/3.21	**93.84**/90.74	**3.34**/6.05
	E/O	**3.09**/4.19	**9.65**/10.83	**87.26**/84.98

The following sample sentence extracted from the Bible is segmented with all four models. Note that the training corpus does not include text from the Bible.

Tigrinya: *"amIlaKI kea mIrIayu zEbIhIgI mIbIlaOu zITIOumI kWlu omI abI mIdIri abIqWele"*
English: "And out of the earth the Lord made every tree to come, delighting the eye and good for food".

The expected segmentation is given under the column labeled "reference". We observe that the BIE model segmentation is the nearest to the reference segmentation. The other models showed the under-segmentation errors for the word *"ZEbIhIgI"* and *"kWlu"*. The verbal noun prefix *"mI-"* is correctly segmented by all models whereas the models failed to recognize the boundary of the causative prefix *"a-"* in the last word *"abIqWele"*.

Table 8. Sample of segmentation result; the underscore character marks segmentation boundaries and the **boldfaced** characters denote segmentation errors.

Sentence	**True**	**BIE**	**BIES**	**BIO**	**BIEOS**
amIlaKI	amIlaKI	amIlaKI	amIlaKI	amIlaKI	amIlaKI
Kea	kea	kea	kea	kea	Kea
mIrIayu	mI_rIay_u	mI_rIay_u	mI_rIay_u	mI_rIay_u	mI_rIay_u
zEbIhIgI	zE_bIhIgI	zE_bIhIgI	z**Eb**IhIgI	z**Eb**IhIgI	z**Eb**IhIgI
mIbIlaOu	mI_bIlaO_u	mI_bIlaO_u	mI_bIlaO_u	mI_bIlaO_u	mI_bIlaO_u
zITIOumI	zI_TIOumI	zI_TIOumI	zI_TIOumI	zI_TIOumI	zI_TIOumI
kWlu	kWl_u	kWl_u	kW**lu**	kW**lu**	kW**lu**
omI	omI	omI	omI	omI	omI
abI	abI	abI	abI	abI	abI
mIdIri	mIdIri	mIdIri	mIdIri	mIdIri	mIdIri
abIqWele	a_bIqWel_e	**ab**IqWel_e	**ab**IqWel_e	**ab**IqWel_e	**ab**IqWel_e

7. CONCLUSION AND FUTURE WORK

In this work, we presented the first morphological segmentation research for the morphologically rich Tigrinya language. The research was performed based on a new manually segmented corpus comprising over 45,000 words. Four variants of the BIO chunk annotation scheme were employed to train three different morphological segmentation models. The first model was based on CRFs with language-independent features of characters, n-grams, and substrings. The other two were based on LSTM and BiLSTM deep neural architectures leveraging character embeddings of a fixed-size window for extracting the morpheme boundary

features. We evaluated the BIE, BIES, BIO, and BIOES tagging schemes for morphological boundary detection. Although the size of the corpus is relatively small, we achieved 94.67% F1 score in boundary detection using the BIE chunking scheme.

In the future, we plan to improve and enlarge the corpus by including vocabularies from different domains. This will allow the inclusion of potentially unseen patterns as the current orthographic style is limited to news domain. We also plan to extend our experiments to use embeddings with character and substring concatenated features for the BIO and BIOES schemes, which currently have lower performance. We are also interested in integrating minimally supervised approaches to make use of large unlabeled datasets. Segmented words have proved useful in mitigating the adverse effects of data sparseness in Semitic language processing [26]. We would like to explore the use of our segmentation output in machine translation, full-fledged morphological analysis, stemming, part of speech tagging, and other downstream NLP tasks.

ACKNOWLEDGEMENT

We would like to thank Chenchen Ding, from NICT Japan, for his constructive comments.

REFERENCES

[1] M. Creutz and K. Lagus, "Unsupervised discovery of morphemes," in *Proceedings of the ACL02 workshop on morphological and phonological learning-Volume 6*. Association for Computational Linguistics, 2002, pp. 21–30.

[2] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML '01, San Francisco, USA, 2001, pp. 282–289.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," in *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[4] S. Wintner, *Morphological processing of Semitic languages*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 43–66.

[5] M. Gasser, "Hornmorpho 2.5 user's guide," *Indiana University, Indiana*, 2012.

[6] N. A. Kifle, "Differential object marking and topicality in Tigrinya," pp. 4–25, 2007.

[7] A. Sahle, *A comprehensive Tigrinya grammar*, Lawrenceville NJ: Red Sea Press, Inc., 1998, pp. 71–72.

[8] M. Gasser, "Semitic morphological analysis and generation using finite state transducers with feature structures," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2009, pp. 309–317.

[9] T. Ruokolainen, O. Kohonen, K. Sirts, S.-A. Grnroos, M. Kurimo, and S. Virpioja, "A comparative study of minimally supervised morphological segmentation," *Computational Linguistics*, vol. 42, no. 1, pp. 91–120, 2016.

[10] B. Snyder and R. Barzilay, "Unsupervised multilingual learning for morphological segmentation," *Proceedings of ACL-08: HLT*, pp. 737–745, 2008.

[11] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural Language Processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[12] C. D. Santos and B. Zadrozny, "Learning character-level representations for part-of-speech tagging," in *Proceedings of the 31st International Conference on Machine Learning*, vol. 32, no. 2, Bejing, China, 22–24 Jun 2014, pp. 1818–1826.

[13] N. Zalmout and N. Habash, "Don't throw those morphological analyzers away just yet: Neural morphological disambiguation for Arabic," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 704–713.

[14] C. Ding, Y. K. Thu, M. Utiyama, and E. Sumita, "Word segmentation for Burmese (Myanmar)," *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, vol. 15, no. 4, p. 22, 2016

[15] W. Mulugeta, M. Gasser, and B. Yimam, "Incremental learning of affix segmentation," *Proceedings of COLING 2012*, pp. 1901–1914, 2012.

[16] Y. Tedla and K. Yamamoto, "The effect of shallow segmentation on English-Tigrinya statistical machine translation," in *2016 International Conference on Asian Language Processing (IALP)*, Nov 2016, pp. 79–82.

[17] N. Reimers and I. Gurevych, "Optimal hyperparameters for deep LSTM networks for sequence labeling tasks," *arXiv preprint arXiv:1707.06799*, 2017.

[18] Y. Kitagawa and M. Komachi, "Long short-term memory for Japanese word segmentation," *arXiv preprint arXiv:1709.08011*, 2017.

[19] L. Wang, Z. Cao, Y. Xia, and G. de Melo, "Morphological segmentation with window LSTM neural networks." in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16) Morphological*. Association for the Advancement of Artificial Intelligence, 2016.

[20] B. Plank, A. Søgaard, and Y. Goldberg, "Multilingual part-of-speech tagging with bidirectional long short term memory models and auxiliary loss," *arXiv preprint arXiv:1604.05529*, 2016.

[21] X. Ma and E. H. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," *CoRR*, vol. abs/1603.01354, 2016.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[23] M. D. Zeiler, "Adadelta: An adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[25] Y. Dauphin, H. de Vries, J. Chung, and Y. Bengio, "RMSProp and equilibrated adaptive learning rates for non-convex optimization," *CoRR*, vol. abs/1502.04390, 2015.

[26] H. Al-Haj and A. Lavie, "The impact of Arabic morphological segmentation on broad-coverage English-to Arabic statistical machine translation," *Machine Translation*, vol. 26, no. 1, pp. 3–24, Mar 2012.