Neural Sequence Labeling based Sentence Segmentation for Myanmar Language

No Author Given

No Institute Given

Abstract. In the informal Myanmar Language, for which most NLP applications are used, there is no predefined rule to mark the end of the sentence. Therefore, in this paper, we contributed the first Myanmar Sentence Segmentation corpus and systematically experimented with twelve *neural sequence labeling* architectures trained and tested on both sentence and sentence+paragraph data. The word LSTM + Softmax achieved the highest accuracy of 99.95% while trained and tested on sentence-only data and 97.40% while trained and tested on sentence + paragraph data.

Keywords: NCRF^{++} · Sentence Segmentation · CNN · Bi-LSTM

1 Introduction

Sentence Segmentation can be defined as the task of segmenting text into sentences that are independent units and grammatically linked words. In the formal Myanmar language, sentences are grammatically correct and typically end with a "။" pote-ma. Informal language is more frequently used in daily conversations with others due to its easy flow. There are no predefined rules to identify the ending of sentences in informal usages for the machine itself. Some of the applications based on conversations, e.g, Automatic Speech Recognition (ASR), Speech Synthesis or Text-to-Speech (TTS), and chatbots, need to identify the end of sentences. To address this problem, we used the sequence labeling approach in which each unit is labeled, and the pairs of unit and label are trained using a supervised learning algorithm. In this paper, we studied the neural sequence labeling models using character and word sequence representations with Convolutional Neural Networks (CNN) and Bi-directional Long Short-Term Memory (LSTM) Recurrent Neural Networks.

2 Related Works

With the sequence labeling approach, Win Pa Pa et al. [1] examined the effectiveness of CRFs for Myanmar word segmentation. Furthermore, there are additional text segmentation approaches for the Myanmar language. Ye Kyaw Thu et al. [2] proposed seven different word segmentation schemes for statistical machine translation systems. However, there were no methodological sequence labeling studies for Sentence Segmentation in the informal Myanmar Language.

Previous researchers have worked on sentence segmentation problem by using Rule-based approaches (e.g., Lingua::EN::Sentence [3], which is a Perl module for English Sentence Segmentation) and Machine Learning based sequence labeling approaches like Conditional Random Fields (CRFs) [4] and Hidden Markov Models (HMM) [5].

Sadvilkar et al. introduced a multilingual rule-based sentence segmentation tool called PySBD [6] in which Myanmar Sentence Segmentation is available but it is only useful for formal usages because sentence segmentation is based on the sentence delimiter "။" pote-ma, which is not used in informal communications. Deep learning-based sequence labeling, also known as the neural sequence labeling approach is the current state-of-the-art approach for sequence labeling. Yang et al. [7] investigated the design challenges of building effective as well as efficient neural sequence labeling systems. For Myanmar Sentence Segmentation, we examined the performances of state-of-the-art neural sequence labeling models.

3 Corpus Development

This section describes the information of *mySentence* tagged corpus, as well as an overview of word segmentation and tagged text data annotation.

3.1 Corpus Information

Myanmar NLP researchers are facing many difficulties arising from the lack of resources; in particular parallel corpora are scarce [8]. For this reason, we annotated text data manually with *mySentence* tag information. The myPOS corpus version 3.0 [9] consists of 43,196 meaningful word sequences written in formal and informal formats from various domain areas and the whole corpus has already been word-segmented manually. But not all sequences are used for the experiments as sequences with only one word are ignored except for interjections.

We also collected Myanmar sentences and paragraphs from different online resources such as Facebook and Wikipedia and from the short stories available on Facebook pages [10] [11].

Table I shows resources of data collected to use for building mySentence Corpus for Sentence Segmentation.

3.2 Word Segmentation

In the Myanmar language, spaces are used only to segment phrases for easier reading. There are no clear rules for using spaces in the Myanmar language. The myPOS version 3.0 corpus has been already word-segmented manually.

We used myWord word segmentation tool [12] to do word segmentation on our manually collected data and checked word segmentation results manually.

Table I: Data Resources of the corpus

Data Resources	sentence	paragraph
myPOS ver3.0 [9]	40,191	2,917
Covid-19 Q&A [13]	1,000	1,350
Shared By Louis Augustine Page [10]	547	1,885
Maung Zi's Tales Page [11]	2,516	581
Wikipedia	2,780	1,060
Others	93	672
Total	47,127	8,465

We applied the word segmentation rules proposed by Ye Kyaw Thu et al. in myPOS [14] corpus. The segmented example for the Myanmar sentence (How are you, Sayar?) is shown as follows:

Unsegmented sentence : ဆရာနေကောင်းလား
Word segmented sentence : ဆရာ|နေကောင်း|လား

3.3 Corpus Annotation

After the word segmentation, we annotated the word sequences in the corpus into a tagged sequence of words. Each token within the sentence is tagged with one of the four tags: B (Begin), O (Other), N (Next), and E (End).

The beginning word which is on the left of the sentence in the Myanmar language is tagged B and the ending word of each sentence is tagged E. The three words left to the ending words are tagged N while other words in the sentence are tagged O. Tagging process was done manually for both sentences and paragraphs in the dataset.

Table II: Statistics of Tags in the corpus

Tag	Frequency	Proportion
B	47,264	7.24%
E	48,690	7.33%
N	137,592	20.46%
O	436,942	64.97%

Table II shows the statistics of mySentence tags in the corpus. If there are more than two /E tags in a sequence, it is considered to be a paragraph. The tagged example Burmese sentence, (I get bored.) is shown as follows:

Untagged sentence : ကျွန်တော် ပျင်း လာ ပြီ
Tagged sentence : ကျွန်တော်/B ပျင်း/N လာ/N ပြီ/E

The tagged example Burmese paragraph, (I am sorry. I like drama films more.) is shown as follows:

Untagged paragraph : တောင်းပန် ပါ တယ် ကျွန်တော် က အချစ် ကား ပို ကြိုက် တယ်
Tagged paragraph : တောင်းပန်/B ပါ/N တယ်/E ကျွန်တော်/B က/O အချစ်/O ကား/N ပို/N ကြိုက်/N တယ်/E

4 Methodology

For neural sequence labeling, Pytorch-based framework NCRF^{++} [15], a toolkit with flexible running time and a customizable configuration file, was used. It is designed for the rapid implementation of different neural sequence labeling architectures with a CRF or softmax inference layer. NCRF^{++} can be regarded as a neural version of a famous statistical CRF framework, CRF^{++}.

As shown in Figure 1, NCRF^{++} framework supports three layers, i.e, character and word sequence representation layers with CNN and LSTM for feature extractions, and a CRF or Softmax inference layer for predicting.

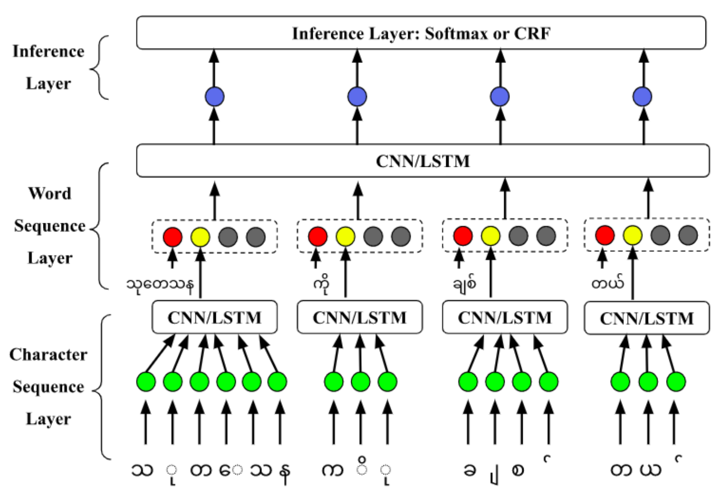

Fig. 1: NCRF^{++} for Burmese sentence "သုတေသန ကို ချစ် တယ်" (I love research.) Green, red, yellow, and blue circles represent character embeddings, word embeddings, character representations, and word representations. The sparse feature embeddings are represented as grey circles.

4.1 Character Sequence Layer

Character features can be represented with character embeddings through neural network models without human-defined hand-engineering. NCRF^{++} supports different character sequence representation approaches such as character CNN, character LSTM, character GRU, and handcrafted word features. In our experiments, we used character CNN and character LSTM, which are state-of-the-art models for sequence representation.

- **Character CNN** used a CNN structure to learn character-level representations. The idea was first introduced by Santos et al. [16] to learn character representations of words for Part-of-Speech Tagging.
- **Character LSTM** used a Bi-directional LSTM structure to capture the global feature of the character sequence information. The forward LSTM captures the character sequence information from left to right then right to left and concatenates the final hidden states of two RNNs as the encoder of the input character sequence, i.e, word.

4.2 Word Sequence Layer

Words from word sequences can be represented similarly to character sequences in words. Word sequence information can be captured with word embeddings through CNN and LSTM models. NCRF^{++} supports different word sequence representation approaches such as word CNN, word LSTM and word GRU. In our experiments, we used word CNN and word LSTM, which are state-of-the-art architectures for sequence representation.

- **Word CNN** used a multi-layer CNN on the word sequences to learn word-level representations. If the character sequence layer is used, the character sequence representations and word embeddings are concatenated for word representations.
- **Word LSTM** used a Bi-directional LSTM structure to capture the contexted information of each word, i.e, the global feature of the word sequence information. The forward LSTM captures the word sequence information from left to right and the backward LSTM in a reversed direction. And calculate the global information of the whole word sequence.

4.3 Inference Layer

The inference layer accepts the word sequence representations from the CNN and LSTM based feature extractors and learns with the assigned label (tag) to predict the correct *mySentence* tags. NCRF^{++} provides two inference functions - Softmax and CRF. In this paper, we examined both of them in order to compare the performances of the different approaches.

- **Softmax** maps the input sequence representations to the label scores, which are used to model the label probabilities of each word and support parallel decoding. In the training process, for classification, NCRF^{++} supported cross-entropy loss.

- **CRF** considers the label dependencies among the predicted segmentation tags that are inherent in the state transitions of finite state sequence models, on which exact inference over sequences can be efficiently performed. The decoding process is done with the Viterbi algorithm by searching the label sequence with the highest probability.

5 Experimental Setup

In this section, we describe data preparation, hyperparameters, and evaluation for the experiments. As shown in Figure 1, NCRF^{++} framework provides different structure combinations on three levels: character sequence representation, word sequence representation, and inference layer with Softmax or CRF function. We trained and tested all of the combinations on both sentence-level and sentence+paragraph-level data.

5.1 Data Preparation

mySentence corpus was used to prepare two types of data - one containing sentence-only data and the other with sentence+paragraph data. And we split both types of data into training, validation, and test data as shown in TABLE III. Here, sent is the abbreviation for sentence-level data and para for paragraph-level data.

Table III: Dataset Split for Experiments

	sent	**sent+para**
train	40,000	47,000
validation	2,414	3,079
test	4,712	5,512

NCRF^{++} does not work to train with another dataset format. Therefore, after splitting the corpus, the format of train, validation, and test *mySentence*-tagged corpora were converted into word and tag parallel columns. Both types of data, i.e, sentence-only and sentence+paragraph data, were used for training, validation, and testing.

5.2 Hyperparameters

We used character embedding size 30 and word embedding size 50 with 50 hidden layers for character representations and 200 hidden layers for word representations. For both character CNN and word CNN, 4 layers of CNN with kernel size 3 were used.

In order to prevent overfitting and underfitting the models, we used L2 regularization λ. Learning rate $\eta = 0.015$ was used for word LSTM-based models but $\eta = 0.010$ was used for word CNN-based models because a large learning rate could cause the convergence problem. The learning rate η was reduced to 0.008 for word CNN with character CNN and 0.005 for word CNN with character LSTM models respectively. Although a variety of optimizers are available in NCRF^{++}, we only used the mini-batch SGD with the batch size of 10 and learning rate decay of 0.5.

Table IV: Hyperparameters used in experiments

Parameter	**Value**	**Parameter**	**Value**
char emb size	30	word emb size	50
char hidden	50	word hidden	200
CNN layer	4	CNN kernel size	3
dropout rate	0.5	batch size	10
L2 regularization λ	1e-8	learning rate decay	0.05
Epochs	100	Optimizer	SGD

5.3 Evaluation

For evaluation, we conducted several experiments with various models on both sentence-only level and sentence+paragraph level test data. The automatic tagging performance was measured using accuracy.

$$Accuracy = \frac{No.\ of\ Correct\ mySentence\text{-}tags}{No.\ of\ predicted\ tokens\ in\ the\ test\ corpus} \tag{1}$$

In our experiments, the accuracy score is used as an evaluation metric, which measures the number of tokens tagged correctly by the model in relation to the number of tagged tokens. It can be calculated by dividing the number of correct mySentence-tags the model predicted by the total number of predictions.

6 Results and Discussion

This paper contributes the first corpus with a total size of around 55K sentences and paragraphs for Myanmar Sentence Segmentation. NCRF^{++} architectures were trained and tested on sentence and sentence + paragraph data.

Table V and VI show the accuracy comparison between each neural sequence labeling model on different levels of test data. The NCRF^{++} architectures trained on sentence-only data were considered sent models and those trained on

Table V: Accuracy % comparison of sentence-level models
(c = Character, w = Word, sent = sentence and para = paragraph)

	Test Data	wCNN + Softmax	wCNN + CRF	wLSTM + Softmax	wLSTM + CRF
NoChar	sent	99.92	**99.95**	**99.95**	**99.95**
	sent+para	93.58	**93.63**	**93.63**	**93.63**
cCNN	sent	**99.95**	**99.95**	**99.95**	92.02
	sent+para	**93.63**	**93.63**	**93.63**	87.65
cLSTM	sent	**99.95**	**99.95**	**99.95**	99.91
	sent+para	**93.63**	**93.63**	**93.63**	93.59

sent+para data are sent+para models. Both types of models were not only tested on the sentence data but also on the sent+para test data.

The bold results show the highest accuracies achieved in each test data. According to Table V and VI, the best sentence models achieved **99.95%** accuracy, and the wLSTM+Softmax model with no character representation has the highest value with **97.40%** accuracy. We also run cross-testing with the trained models. According to the cross-test results, the best sentence models achieved **93.63%** accuracy on sentence+paragraph data, and the wCNN+Softmax model with LSTM-based character representation has the highest value with **99.66%** accuracy on sentence-only test data respectively.

Table VI: Accuracy % comparison of sentence+paragraph-level models
(c = Character, w = Word, sent = sentence and para = paragraph)

	Test Data	wCNN + Softmax	wCNN + CRF	wLSTM + Softmax	wLSTM + CRF
NoChar	sent	99.41	99.49	99.44	86.44
	sent+para	96.82	96.25	**97.40**	96.61
cCNN	sent	99.26	99.27	74.81	86.44
	sent+para	96.87	96.17	74.69	83.13
cLSTM	sent	**99.66**	99.49	99.49	99.56
	sent+para	96.36	96.04	97.29	96.61

7 Error Analysis

We also did the error analysis using the SCLITE (score speech recognition system output) program from the NIST scoring toolkit (Version 2.4.11). It is used to align the hypothesis tags with error-free reference tags and calculate the word error rate (WER) [17]. This program shows the recognition rate at the sequence level and word level and also gives the confusion pairs.

For WER calculation, the SCLITE scoring method first aligns the hypothesis and reference sequences and then calculates a minimum Levenshtein distance which weights the cost of correct words (C), insertions (I), deletions (D), substitutions (S), and the number of words in the reference (N).

To know the counts of I, D, C, and S for the tag sequence "B O N N N E", at first, the output (hypothesis) sequence is compared to the reference sequence. Then, WER is calculated based on the counts.

Scores: (#C #S #D #I) 4 2 0 0
REF: B O N N N E
HYP: B N N N N N
Eval: S S

For this example, there are no deletions (D=0) or insertions (I=0) and only two substitutions (N=>O) and (O=>N) are happened so the number of correct words C is 4. Using the WER equation, the SCLITE program calculated the WER value for the given example as 16.67%.

Table VII: The Top 5 confusion pairs of sent cCNN+wLSTM+CRF tested on sent+para test data (87.65% accuracy)

Freq	**Confusion Pair (REF==>HYP)**
7078	n ==> o
1951	o ==> n
1229	e ==> o
1224	b ==> o
48	b ==> n

After analysis of confusion pairs, we found out that some of the confusion pairs are related to "O" Tags. Here, from TABLE VII and VII, the confusion pairs of "n ==> o", "b ==> o" and "o ==> n" happened because of false recognition. According to TABLE II, 64.97% of the tags in the mySentence corpus are O" tags. Therefore, the models might predict most of the uncommon words as "O".

8 Conclusion

According to the comparison of twelve NCRF^{++} architectures trained and tested on both sentence and sent+para data, we can see that the word LSTM with soft-

Table VIII: The Top 5 confusion pairs of sent+para cCNN+wLSTM+Softmax tested on sent test data (74.81% accuracy)

Freq	**Confusion Pair (REF==>HYP)**
9790	n ==> o
4193	b ==> o
541	n ==> e
321	e ==> o
64	b ==> n

max inference layer and no character representation layer had the best accuracy with sent-level (**99.95%**) as well as sent+para-level (**97.40%**) data. According to the error analysis, most of the errors occurred because the models falsely recognized "O" tags, which have the highest proportion in the dataset. In the near future, we plan to release our tagging data and hold new experiments with other approaches. We also plan to release the corpus publicly that was used in this experiment.

References

1. Win Pa Pa, Ye Kyaw Thu, Finch,A. , Sumita, E.: Word Boundary Identification for Myanmar Text Using Conditional Random Fields, In Zin, T., Lin, JW., Pan, JS., Tin, P., Yokota, M., (eds) Genetic and Evolutionary Computing. GEC 2015. Advances in Intelligent Systems and Computing, vol 388. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-23207-2_46
2. Ye Kyaw Thu, Finch, A., Sagisaka, Y., Sumita, E.: A Study of Myanmar Word Segmentation Schemes for Statistical Machine Translation, In Proceedings of the 11th International Conference on Computer Applications, pp. 167-179, Yangon, Myanmar (2013). http://onlineresource.ucsy.edu.mm/handle/123456789/2335
3. Lingua::EN::Sentence:https://metacpan.org/release/KIMRYAN/Lingua-EN-Sentence-0.29/view/lib/Lingua/EN/Sentence.pm
4. Tomanek, K., Wermter, J., Hahn, U.: Sentence and token splitting based on conditional random fields, In Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics, vol. 49, pp. 57 (2017). https://api.semanticscholar.org/CorpusID:15539970
5. Jurish, B., Würzner, K.-M.: Word and Sentence Tokenization with Hidden Markov Models, JLCL, vol. 28, no. 2, pp. 6183. (2013) https://api.semanticscholar.org/CorpusID:6659476
6. Sadvilkar, N., Neumann, M.: PySBD: Pragmatic Sentence Boundary Disambiguation, In Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS), Association for Computational Linguistics, pp. 110-114. (2020) https://doi.org/10.18653/v1/2020.nlposs-1.15
7. Yang, J., Liang, S. and Zhang, Y.: Design Challenges and Misconceptions in Neural Sequence Labeling, In Proceedings of the 27th International Conference on Computational Linguistics, pp. 38793889, Association for Computational Linguistics, Santa Fe, New Mexico, USA. (2018) https://aclanthology.org/C18-1327

8. Ye Kyaw Thu, Chea, V., Finch, A., Utiyama, M., Sumita, E.: A Large-scale Study of Statistical Machine Translation Methods for Khmer Language, In Proceedings of 29th Pacific Asia Conference on Language, Information and Computation, Shanghai, China, pp. 259-269. (2015) https://aclanthology.org/Y15-1030.pdf
9. Zar Zar Hlaing, Ye Kyaw Thu, Supnithi, T., Netisopakul, P.: Improving Neural Machine Translation with POS-tag features for low-resource language pairs, Heliyon, vol. 8. (2022) https://doi.org/10.1016/j.heliyon.2022.e10375
10. Shared By Louis Augustine: www.facebook.com/sharedbylouisaugustine
11. Maung Zis Tales: www.facebook.com/MaungZiTales
12. Ye Kyaw Thu, myWord: Syllable, Word and Phrase Segmenter for Burmese, GitHub Link: https://github.com/ye-kyaw-thu/myWord,September2021
13. NHK World-Japan, Corona Virus Questions and Answers in Burmese, December 2022, Link: https://www3.nhk.or.jp/nhkworld/my/news/qa/coronavirus/
14. Ye Kyaw Thu, myPOS: Myanmar Part-of-Speech Corpus, GitHub Link: https://github.com/ye-kyaw-thu/myPOS
15. Yang, J., Zhang, Y.: NCRF++: An Open-source Neural Sequence Labeling Toolkit, In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 74-79, Association for Computational Linguistics, Melbourne, Australia. (2018) https://doi.org/10.18653/v1/P18-4013
16. Santos, C.D, Zadrozny, B.: Learning Character-level Representations for Part-of-Speech Tagging, In Proceedings of the 31st International Conference on Machine Learning (ICML). vol.5. (2014) https://proceedings.mlr.press/v32/santos14.html
17. Ali, A., Renals, S.: Word Error Rate Estimation for Speech Recognition: e-WER, In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, vol.2: Short Papers, pp. 20-24, Melbourne, Australia. (2018) https://doi.org/10.18653/v1/P18-2004