Natural Language Processing (2024), 1–20
doi:10.1017/nlp.2024.24

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Probing a pretrained RoBERTa on Khasi language for POS tagging

Aiom Minnette Mitri[1], Eusebius Lawai Lyngdoh[1], Sunita Warjri[2], Goutam Saha[1], Saralin A. Lyngdoh[3] and Arnab Kumar Maji[3]

[1]Department of Information Technology, North Eastern Hill University, Shillong, Meghalaya, India, [2]Faculty of Fisheries and Water Protection, University of South Bohemia in Ceské, Budejovicich, Czech Republic, and [3]Department of Linguistics, North Eastern Hill University, Shillong, Meghalaya, India
Corresponding author: Arnab Kumar Maji; Email: akmaji@nehu.ac.in

(Received 24 December 2022; revised 8 August 2023; accepted 25 October 2023)

Special Issue on '**Natural Language Processing Applications for Low-Resource Languages**'

Abstract

Part of speech (POS) tagging, though considered to be preliminary to any Natural Language Processing (NLP) task, is crucial to account for, especially in low resource language like Khasi that lacks any form of formal corpus. POS tagging is context sensitive. Therefore, the task is challenging. In this paper, we attempt to investigate a deep learning approach to the POS tagging problem in Khasi. A deep learning model called Robustly Optimized BERT Pretraining Approach (RoBERTa) is pretrained for language modelling task. We then create RoBERTa for POS (RoPOS) tagging, a model that performs POS tagging by fine-tuning the pretrained RoBERTa and leveraging its embeddings for downstream POS tagging. The existing tagset that has been designed, customarily, for the Khasi language is employed for this work, and the corresponding tagged dataset is taken as our base corpus. Further, we propose additional tags to this existing tagset to meet the requirements of the language and have increased the size of the existing Khasi POS corpus. Other machine learning and deep learning models have also been tried and tested for the same task, and a comparative analysis is made on the various models employed. Two different setups have been used for the RoPOS model, and the best testing accuracy achieved is 92 per cent. Comparative analysis of RoPOS with the other models indicates that RoPOS outperforms the others when used for inferencing on texts that are outside the domain of the POS tagged training dataset.

Keywords: Part of speech tagging; tagging; RoBERTa

1. Introduction

Khasi is an Austro-Asiatic language (Reddy *et al.* 2007), which is spoken by the people of Meghalaya, one of the North Eastern States of India. There are evidences of a good number of Khasi speakers in the neighbouring country Bangladesh. Khasi speakers are also found spreading across the neighbouring states like Assam and others. As far as Natural Language Processing (NLP) is concerned, the language is considered as low resource because of the paucity of electronic resources. Despite the fact that it is low resource, some works pertaining to Khasi language have been done in the area of part of speech (POS) tagging. The aim of the paper is to build a POS tagger using a deep learning approach where a variant of the Bidirectional Encoder Representations from Transformers (BERT) model (Devlin *et al.* 2019), Robustly Optimized BERT Pretraining Approach (RoBERTa) (Liu *et al.* 2019), is pretrained and tested on language understanding for

Khasi language. Then, by leveraging its embeddings, we fine-tune this pretrained model for the purpose of POS tagging, and we refer to the fine-tuned model as RoBERTa for POS (RoPOS) tagging. Other machine learning and deep learning based models have been tried and tested for the same task, and a performance comparison has been undertaken to evaluate the efficiency of RoPOS. Apart from this, the paper also proposes additional tags for inclusion into the existing Khasi POS tagset to bridge the gap of brevity. Additionally, tags for some words in the current tagged dataset have been modified so that they reflect their grammatical position more accurately. RoPOS is pretrained using different tokenisation schemes, and it is found that the best accuracy achieved is 92 per cent. In short, the paper's main contributions include updation of the existing Khasi POS tagset, modifying the tags assigned to certain words of the existing tagged corpus, increasing the size of the current POS tagged corpus and designing of a relatively more reliable POS tagger. In the subsequent subsection, a brief background of the RoBERTa model is discussed. We also include a brief description of the two tokenisation schemes that we have employed in this work. The paper has been organised into different sections which are given as follows: Section 2 discusses the related relevant works, Section 3 gives a detailed description of the methodology used, Section 4 discusses on the dataset acquisition and modification made in the proposed investigation, Section 5 elaborates on the experimental setup for implementation of RoPOS, Section 6 discusses on the experimental results obtained, Section 7 concludes with performance analysis of the proposed model and, finally, Section 8 highlights the limitations of the proposed model along with future research scope surfaced out.

1.1. Background

A brief description on the RoBERTa model and its architecture is given in the subsequent subsection. Following it is a discussion on the two tokenisation schemes that have been employed for the two setups of RoPOS.

1.1.1. Robustly Optimized BERT Pretraining Approach—RoBERTa

The RoBERTa model proposed in Liu *et al.* (2019) has the same architecture as BERT (Devlin *et al.* 2019) and may use a Byte-Pair Encoding (BPE) (Sennrich, Haddow, and Birch 2015) or a Wordpiece (Schuster and Nakajima 2012) tokeniser. It builds on BERT with some modifications, which include: eliminating the next sentence prediction objective, training on longer sequences, dynamically modifying the masking pattern used on the training data and training longer on larger batches and across more data. The pretraining strategy used in RoBERTa is masked language modelling (MLM), but unlike BERT, the masking operation is dynamic. According to the authors, with dynamic masking, there is an improvement on some NLP tasks. The basic architecture is shown in Fig. 1. The input sequence is first tokenised through a mapping to the vocabulary that gets constructed at the end of training of a compatible tokeniser on the corpus collected. In the sections that follow, we show how the model's overall performance is influenced by the tokenisation technique that is used. As indicated in the figure, the tokens obtained are masked, where 15 per cent of them are replaced by the <mask> token. Then, they are embedded by projecting them into some abstract feature space. Positional information is also embedded so that word ordering in a sequence is maintained because tokens are fed into the model in parallel. They, then, pass through the encoder block. There can be N such blocks. In this work we choose $N = 6$. Within the encoder block, they pass through the multi-head attention layer, which is the core of transformer models. The output goes through the Add and Norm layer, which adds the output to the original vector and then normalises it. Output from the last encoder block goes through a linear layer that generates the original sequence with <mask> tokens being replaced with words in the vocabulary that have the highest probability.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

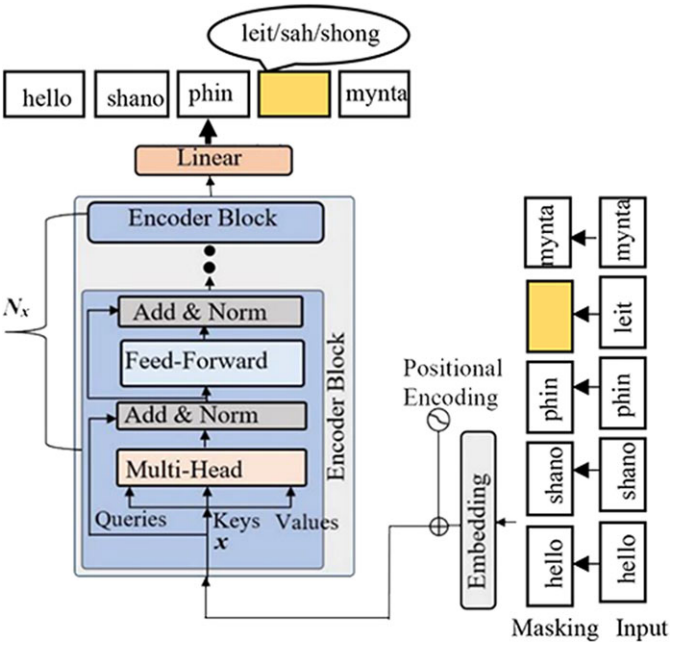

Figure 1. Bert Architecture (Image source: Thabah *et al.* 2022).

1.1.2. Tokenisation

Tokenisation is an important prepocessing step in any NLP task. Transformer models typically have a vocabulary size of not more than 50,000, which is possible because of the tokenisation schemes designed. We discuss here two schemes that we employ in our model, namely, BPE and Wordpiece encoding tokenisation schemes. Both employ subword tokenisation, which is based on the notion that rare words should be split into smaller, meaningful subwords, while common words should remain unsplit.

- **BPE**: Sennrich *et al*. (2015) developed BPE, which learns merge rules to combine two symbols from the base vocabulary to create a new symbol. It develops a base vocabulary out of all the symbols that appear in the collection of unique words. It keeps doing this till the vocabulary reaches a certain size. This size is a hyperparameter set before the tokeniser is trained.
- **Wordpiece Encoding**: Wordpiece tokenisation, designed by Schuster and Nakajima (2012), is very similar to BPE, with the difference that BPE chooses the most frequent symbol pair, while the one that Wordpiece selects maximises the likelihood of the training data once it has been added to the vocabulary.

2. Related work

In general, there has been scarcity of research work done in the field of NLP on Khasi language. It is only as recent as 2018 that works on POS tagging and machine translation started. Tham (2018) proposed a POS tagger using the Hidden Markov Model (HMM) on a corpus that consisted of 86,087 tokens and achieved an accuracy of 95.68 per cent. Other tools available in NLTK for tagging have also been discussed in the paper, with reported accuracy rates of 86.76 per cent for BaselineTagger, 88.23 per cent for NLTK Bigram Tagger, 88.64 per cent for NLTK Trigram Tagger

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

and 89.7 per cent for Natural Language Toolkit (NLTK) Tagger. Warjri *et al.* (2019) developed a Khasi POS tagger based on the HMM model. For the task, a Khasi tagset consisting of fifty-four tags has been designed. A manually tagged Khasi lexicon of around 7,500 tokens has been used on the model, and a test accuracy of 76.70 per cent has been reported. Tham (2020) created a hybrid POS tagger by integrating Conditional Random Field (CRF) with a HMM Khasi POS tagger in order to reduce the HMM tagger's tagging errors and reported a tagging accuracy of 95.29 per cent, a 1.9 per cent improvement over the HMM tagger. Warjri *et al.* (2021) proposed a Khasi POS tagger using the CRF method. As part of their work, a tagset has been designed, customarily, for the Khasi language, and a corresponding POS tagged corpus of 71,000 tokens has been built. The CRF model as designed yielded a test accuracy of 92.12 per cent and an F1-score of 0.91. Thabah *et al.* (2022) used BERT, to get pretrained embeddings for Khasi, and they reported convergence after 500 epochs of training and a training accuracy of 0.08. Because the tokenisation technique used in this work creates a vocabulary that is comprised of all unique terms encountered in the corpus, one disadvantage is the potential for unknown tokens to appear if they are not already there in the vocabulary constructed.

Regarding POS tagging in other low resource Indian languages, a number of works have been reported in the literature. In their paper, Nongmeikapam and Bandyopadhyay (2016) have used genetic algorithm for selecting the best feature combination for use in the task of POS tagging with CRF model. According to them feature selection is crucial to improving the performance of the tagger. They reported an accuracy of 80.00 per cent Recall, 90.43 per cent Precision and 84.90 per cent F1-score. Nunsanga *et al.* (2021) employed CRF method in POS tagging of the Mizo Language. A Mizo tagset consisting of twenty-six tags has been presented, and the tagger was able to achieve an accuracy of 89.42 per cent on a corpus of 30,000 words. Odia is another Indian language that is low resource and an attempt to develop a POS tagger for it has been made by Dalai, Mishra and Sa (2022). They have experimented with statistical as well as deep learning based methods in their work and found that the deep learning Bidirectional Long Short-Term Memory (bi-LSTM) network with character sequence feature using a pretrained word vector yielded the best accuracy of 94.58 per cent. A comparative analysis of various POS taggers for the resource scarce Bengali language has been reported by Jahara *et al.* (2021). They have employed eight stochastic methods and eight transformation-based methods on a corpus of 7,390 lines. They compared the performance of these methods on two tagsets—one consisting of eleven tags, while the other consisting of thirty tags. An accuracy of 91.83 per cent has been reported using Brill with CRF method on the former tagset and an accuracy of 84.5 per cent on the later. A problem that has been highlighted in the paper is that the language has a complex morphology rendering the POS tagging task challenging. A deep learning approach, as described by Pathak, Nandi and Sarmah (2022) in their paper, has been utilised for the POS tagging task of the Assamese language. Again, in this work, the bi-LSTM with CRF model has been used to build the tagger and an accuracy of 86.52 per cent has been reported. Boro and Sharma (2020) discussed the POS tagging challenges encountered in Assamese language in their study on POS tagging for Assamese. The paper mentions that designing a tagger for Assamese is significantly more difficult because the language is regarded as having a free form word order and a rich morphology. Even though (subject-verb-object (SVO) is the most predominant form of word order, sentences can still be constructed using other word orders including SOV, VOS, VSO, OVS and OSV. Another issue that has been highlighted in their work is the tagging of ambiguous words where the same word may be associated with more than one tag.

In general, in most of the papers in the literature, where limitations have been discussed, the most pertinent issue in POS tagging that still needs to be addressed is the problem with ambiguous words. To resolve this, more information is required which may be morphological, syntactic, semantic, etc. However, such information is scarce for low resource languages. Hence, this issue may be considered a challenge till date.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

3. Methodology

We outline the methodology that we adopt for this work in the following steps :

1. **Pretrain a RoBERTa model for Khasi language understanding using MLM approach:** For this step, a generic Khasi corpus used in Thabah *et al.* (2022), built from resources available in the web such as newspaper articles, folklores, literary texts, the Khasi BiBle, etc., has been used. The corpus has been cleansed further by reducing its size from 64.2 MB to 53 MB. We train a RoBERTa on this corpus whose configuration and model set up are discussed in Section 5. Two instances of the model have been trained where:
   (a) For the first instance, a ByteLevelBPETokenizer (Sennrich *et al.* 2015), which is a BPE tokeniser, has been used
   (b) For the second instance, a BertWordpieceTokenizer (Schuster and Nakajima 2012), which is a Wordpiece tokeniser, has been used (refer to Section 1.1.2).

   The effect of choosing one tokenisation scheme over the other is discussed, in detail, in Section 6. This pretrained model is saved as its embeddings will be extracted later for use by RoPOS.
2. **Identify new tags:** An additional tag has been proposed for inclusion into the existing Khasi tagset which is elaborated in Section 4.1.
3. **Increase the size of the existing POS tagged dataset:** For this, an additional 35,109 number of tokens have been manually tagged, thereby expanding the tagged corpus size from 71,000 to 106,109 tokens.
4. **Build a POS tagger, RoPOS, by fine-tuning the pretrained model of step (1) above:** RoPOS's basic architecture is shown in Fig. 3. RoPOS is trained using the tagged dataset from Section 4. The problem is addressed as one of classification where RoPOS is trained to predict the tags corresponding to the input sequence. Again, two instances of RoPOS have been trained—(i) using ByteLevelBPETokenizer as a tokeniser of the input and (ii) using BertWordpieceTokenizer. Before feeding the input to the model, some preliminary steps have to be performed which are as follows:
   – Each line in the tagged dataset consists of a collection of word/tag pairs. From this set,we extract the words and tags separately to form our data and labels for training.
   – Tokenise the input texts using one of the tokenisers depending on the RoPOS instance being trained. In both BPE and Wordpiece tokenisers, it is possible that during the tokenisation process, a word may be broken up into more than one token (called subtokens). For example, the word *'jingngeit'* after tokenisation is broken up into subtokens *'jing'* and *'ngeit'*, with BPE tokenisation and assigned two separate identifications from the vocabulary. For the task of POS tagging where one whole word is assigned to one tag, such a kind of tokenisation poses a problem. This is addressed by extending the length of the labels to the length of the tokenised sentence as illustrated in Fig. 2. Whenever an input word is broken into more than one subtoken, each of the subtokens is labelled with the tag of the input word as in the case of the subtokens *'jing'* and *'ngeit'* for the input word *'jingngeit'*. A similar alignment is performed on data tokenised through Wordpiece tokenisation. Here, whenever a word gets decomposed into subtokens, all of the subtokens, except the first, have been prefixed with ## to indicate that they are all part of a bigger word as illustrated in Fig. 2.

   The (sub)tokens that result from the tokenisation process are fed as input to RoPOS for it to learn to predict the corresponding tags.
5. **Comparative analysis:** A comparison is made between the performance of the two instances of RoPOS in relation to each other as well as in comparison to other machine learning and deep learning models. The results and findings of this step are discussed in Section 6.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 1. List of borrowed words extracted from the corpus

pisa	miej	kolshor	politik	sorkar
ilekshon	trok	draibar	bos	kopi
laiñ	mahajon	myntri	dorbar	khulom
ophis	ophisar	doktor	nos	skhim
baje	tarik	eksam	jamin	pas
histori	sordar	lak	hajar	prokram

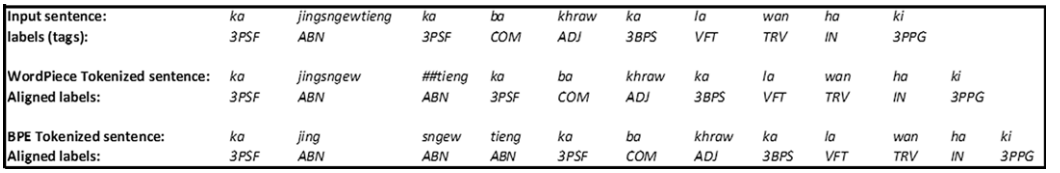

Figure 2. Aligning labels to tokens.

4. Dataset

The tagset designed by Warjri *et al.* (2018) and the corresponding POS tagged corpus of about 71,000 tokens have been used as the baseline. To this, an additional 35,109 number of tokens that are tagged manually has been added, thus expanding the POS tagged corpus size to 106,109. The additional set has been taken from articles from the local daily, *'U Rupang'*.

4.1. New and modified tags

In the original tagset defined by Warjri *et al.* (2018), foreign words that appear in the corpus, such as *'Sports'*, *'Association'*, *'Hills'* etc., have been tagged as *FR* (foreign word tag). Some foreign words, however, have evolved to become part and parcel of the Khasi language with their spellings changed to meet the Khasi manner of pronunciation and articulation. Words like *'sorkar'* and *'pisa'* are words that have been adopted and adapted into Khasi but are derived or borrowed directly or indirectly from other contact languages like Hindi or Assamese. In the same vein, words like *'kolshor'* and *'politik'* for *'culture'* and *'politics'*, respectively, come directly from English. In the original tagset, such words have been tagged as *FR*. In this paper we propose tagging them as *BR* (borrowed word tag), indicating that they have been borrowed and have become an intrinsic part of the Khasi language. Table 1 lists the borrowed words found in the corpus. Another modification that is proposed in this paper is in the way abbreviations have been tagged. Here, we propose tagging all abbreviations with *PPN*, the proper noun tag instead of *FR* as in the original dataset.

In line with the above, some of the *FR* tags in the base corpus have been manually changed to *BR* tags. Morphological break-up of certain words in the base corpus has been done to a certain extent. Therefore, as a part of preprocessing, this morphological break-up of certain words has to be done with the additional dataset to comply with the base corpus. At present there is no tool for morphology, so this task has to be done manually on a corpus of 53 MB and a lot of time has been spent in this painstaking task. For the experiments to follow, the dataset is split as in Table 2.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 2. Dataset for training, testing and validation

	Train set	Test set	Validation set	Total
Number of lines	1,807	452	399	2,658

Table 3. Hardware configuration

CPU	Intel(R)Xeon(R) CPU@2.30 GHz
Main memory	13 GB
GPU	Tesla P100-PCIE
GPU CUDA cores	3,584
GPU memory speed	732 GB/s
GPU Memory	16 GB

Table 4. Parameter and hyperparameter setting for pretraining RoBERTa

Parameter/hyperparameter	Values
Batch size	8
Learning rate	0.00001
Shuffle	True
Maximum sequence length	512
Embedding dimension	768

5. Experimental setup and implementation

The model has been implemented in the Pytorch framework developed by Paszke *et al.* (2019), with computational resources from Google Colaboratory. Implementation of RoPOS is outlined as a two-step process as follows:

- For pretraining of the language model, the generic Khasi corpus described in Section 3(1) is used. Since RoBERTa uses MLM, we mask out 15 per cent of the input sequence dynamically after tokenisation. Table 3 gives the hardware configuration on which the language model has been trained on, and Table 4 specifies parameters and hyperparameters that have been used. As described earlier, two instances of RoBERTa have been trained for ten epochs each and the best model is achieved at an average loss of 0.02 for both the instances and training accuracy of 98 per cent. As in Table 4, the maximum sequence length is set to 512, while Thabah *et al.* (2022) capped this length to forty. The models have been tested on the word prediction task using the transformers pipeline where words are masked out at random for the model to predict and it obtains comparable results as Thabah *et al.* (2022). The aim of this pretraining step is to extract the model's embeddings that it has learnt, in order to use them for the POS tagging task.
- Next is implementing the POS tagger, RoPOS, whose structure is shown in Fig. 3. A single linear layer is added that is used for prediction of the tag for each token in the input sequence. Within the model, the embeddings generated by the pretrained RoBERTa model are extracted and the embedding dimension, of size 768, is obtained from the pretrained

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

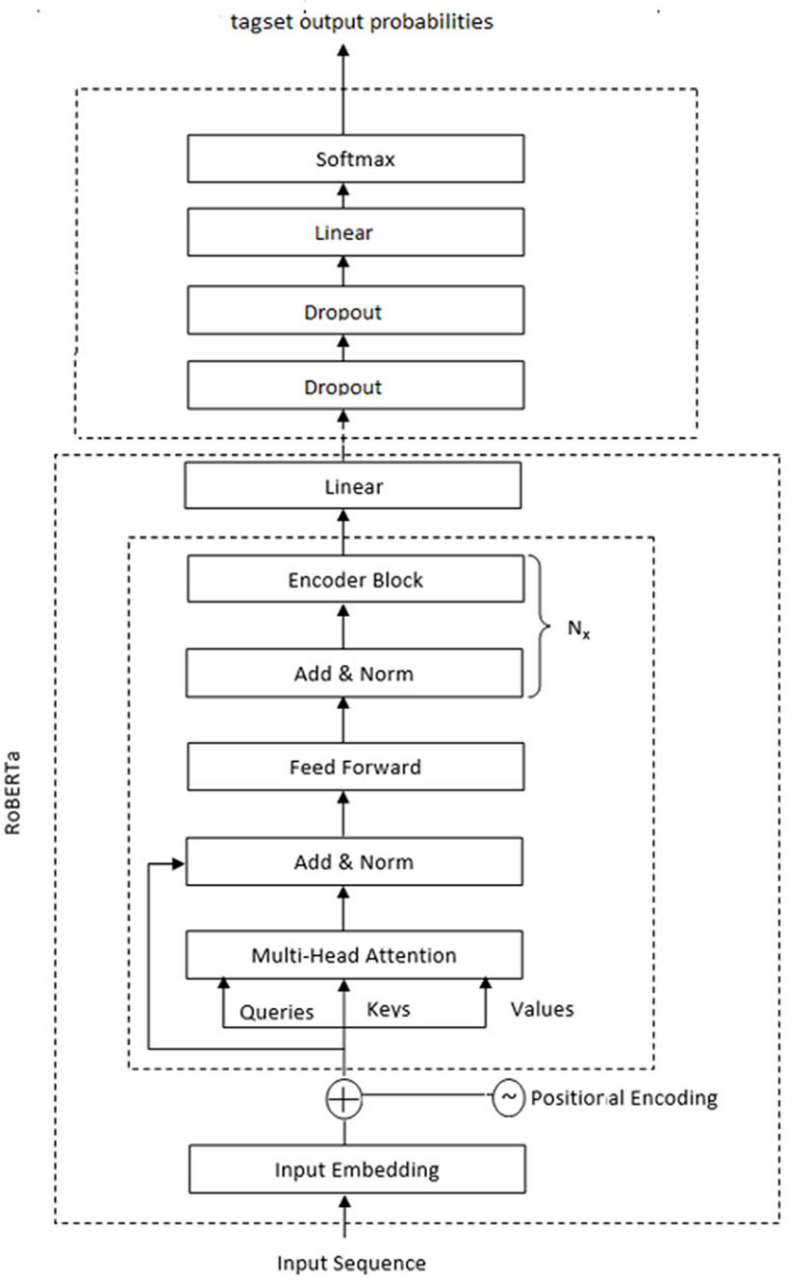

Figure 3. Basic structure of RoPOS model.

model's hidden_size attribute. The hyperparameter configuration of Table 5 is used for both instances of RoPOS. Before training RoPOS, the POS tagged dataset is prepared and preprocessed with label alignment as described in Section 3(4).

The accuracy and loss plots of the RoPOS over time for fifty epochs using BPE tokenisation and that using Wordpiece tokenisation for twenty epochs are shown in Fig. 4.

A training and test accuracy of 85 per cent and 83 per cent, respectively, is achieved for RoPOS with BPE while that of 95 per cent and 92 per cent for RoPOS with Wordpiece tokenisation.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 5. RoPOS model parameter and hyperparameter setting

Parameter/hyperparameter	Values
Batch size	8
Learning rate	5e-5
Dropout	0.5
Shuffle	True
Maximum sequence length	200
Number of classes	61

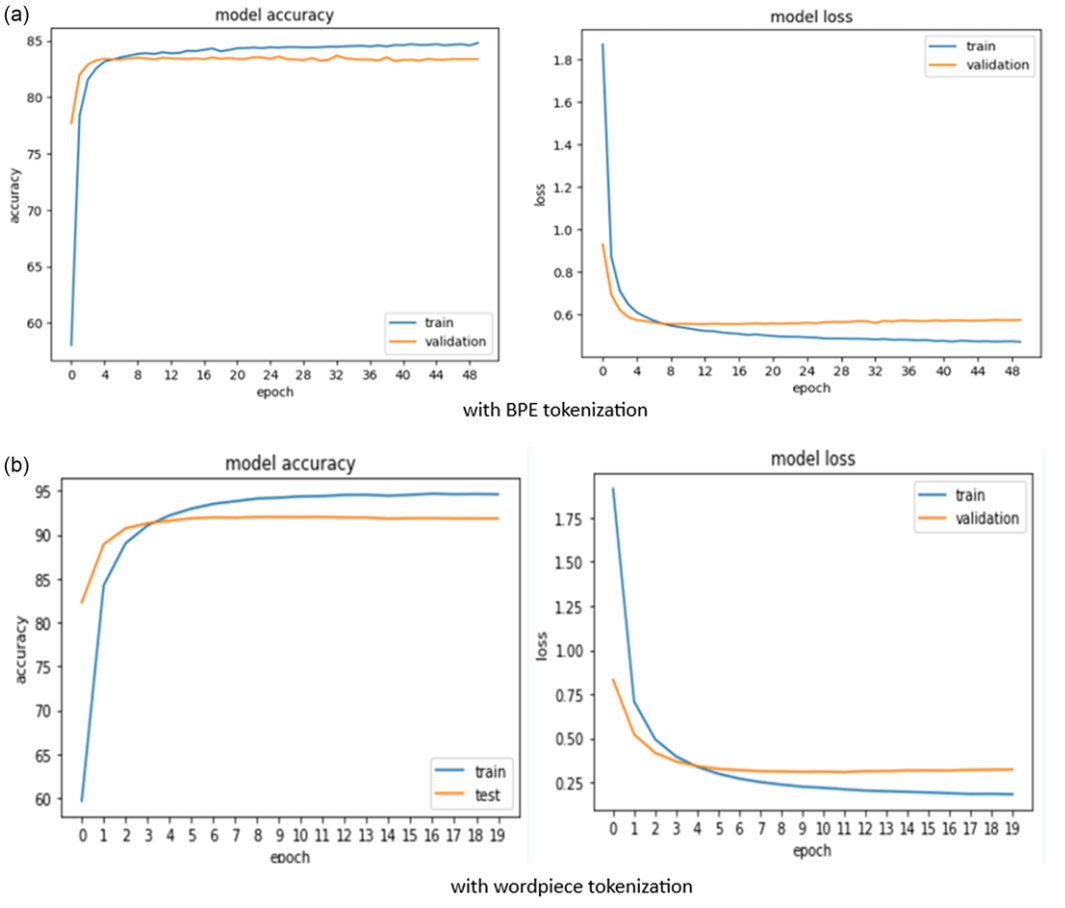

Figure 4. Accuracy and Loss of RoPOS.

5.1. Different machine and deep learning based models

Various machine learning and deep learning models have been experimented and implemented for the same task. The models that have been considered include Logistic Regression (Pranckevičius and Marcinkevicius 2016), Naive Bayes (Bulusu and Sucharita 2019), Random Forest (Jahara *et al.* 2021), Support Vector Machine (Ekbal and Bandyopadhyay 2008), simple

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 6. Deep learning models' configuration

Input dimension	111,425
Length of word vector	200
Batch size	10
Learning rate	0.01

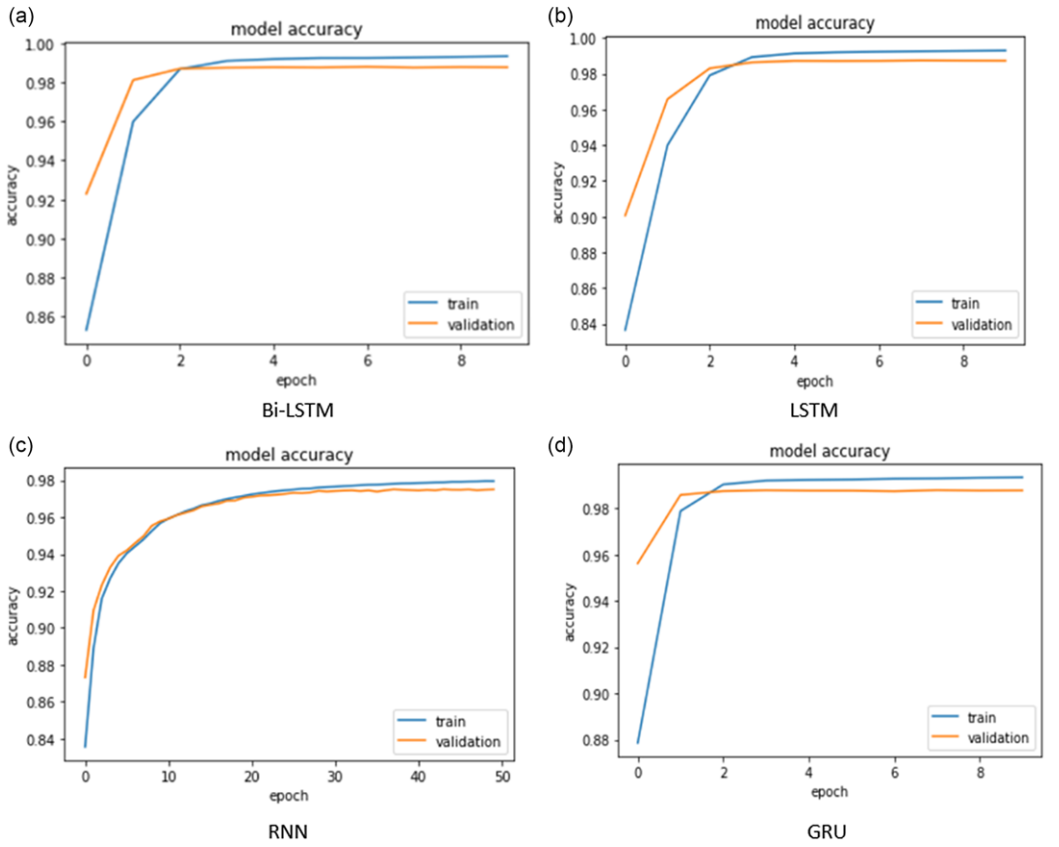

Figure 5. Accuracy plots of deep learning models.

Recurrent Neural Network (RNN) (Jordan 1997), LSTM (Hochreiter and Schmidhuber 1997), bi-LSTM (Fei and Tan 2018) and a Gated Recurrent Unit (GRU) (Cho *et al.* 2014). The approach to solve the problem of POS tagging in all of these models is one of classification with as many classes as there are tags in the tagset. The models have been implemented in Google Colab using the keras API (Chollet 2018) and Pytorch framework (Paszke *et al.* 2019).

The configuration shown in Table 6 is the one that has been used for the deep learning models, viz., RNN (with sixty-four RNN units), LSTM (sixty-four LSTM cells), bi-LSTM (128 LSTM Cells) and GRU (sixty-four recurrent units). The tokeniser from Keras API has been used for all of the deep learning models. The vocabulary has 111,425 distinct words because tokenisation is done word by word, making it significantly larger than the subword tokenisation outlined in Section 1.1.2. The accuracy plots of these models trained for a varied number of epochs are shown in Fig. 5.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 7. Machine learning models' configuration

Model	Parameters
Logistic Regression	C = 5
Naive Bayes	alpha = 0.1, fit prior = False
SVM	C = 10, kernel = sigmoid, coef0 = 0.1
Random Forest	$max_{features} = log2$

Table 8. Reported accuracy

	Training accuracy (%)	Test accuracy (%)
RoPOS with BPE Tokenisation	85	83
RoPOS with Wordpiece Tokenisation	95	92
Bi-LSTM	99	98
LSTM	99	98
RNN	97	97
GRU	99.3	98
Logistic Regression	97	93
Random Forest	97	93
SVM	95	93
Naive Bayes	95	91

Overfitting has been handled by adding a dropout layer to all the deep learning models. For RoPOS, we add two dropout layers before the embeddings are fed to the final linear layer. The model improve with this configuration which can be seen in the train and validation accuracy plots. Cross validation during training helps us decide on early stopping of the training.

Table 7 shows the parameter values that have been chosen for the machine learning models. For Logistic Regression we find that the regularisation parameter, C, gives better accuracy and convergence when set to value 5 while for SVM $C = 10$, gives a better accuracy value.

6. Results and discussion

Table 8 displays the accuracy values of the various models that have been implemented for POS tagging.

For evaluating each of the models' performance while inferencing, they have been tested on a random article that has not been included in the training, test or validation set but one from the same domain. The classification reports of RoPOS using BPE scheme of tokenisation and that using Wordpiece tokenisation are shown in Tables 9 and 10. It is clear from the reports that the model that makes use of Wordpiece tokenisation performs better. As mentioned earlier in Section 3, when BPE tokenisation is used, the word *'jingngeit'* is broken into the subtokens *'jing'* and *'ngeit'*. Although the term *'jing'* may not show up anywhere in the corpus as an independent word, the word *'ngeit'* will. The model learns one tag for *'ngeit'*, as an independent word and another different tag for *'ngeit'* as a subword. With Wordpiece tokenisation, on the other hand,

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 9. RoPOS with BPE: classification report on a random article

	Precision	Recall	F1-score	Support
3ppg	1.00	1.00	1.00	8
3psf	1.00	1.00	1.00	22
3psm	1.00	0.45	0.62	11
abn	0.71	1.00	0.83	5
ad	0.88	1.00	0.93	7
add	0.50	1.00	0.67	1
adj	0.50	1.00	0.67	1
adp	1.00	1.00	1.00	1
adt	1.00	1.00	1.00	2
br	1.00	1.00	1.00	1
cav	1.00	1.00	1.00	2
cmn	0.47	0.91	0.62	22
co	1.00	1.00	1.00	3
coc	1.00	1.00	1.00	6
com	1.00	1.00	1.00	16
dmp	1.00	0.67	0.80	3
dtv	1.00	1.00	1.00	1
fr	0.35	0.30	0.33	23
idp	0.00	0.00	0.00	0
in	1.00	1.00	1.00	24
itv	0.50	0.75	0.60	4
mod	1.00	1.00	1.00	1
pop	1.00	1.00	1.00	5
ppn	0.50	0.29	0.37	34
qnt	1.00	0.33	0.50	3
rfp	1.00	0.67	0.80	3
spa	1.00	1.00	1.00	1
suc	0.00	0.00	0.00	2
sym	0.91	0.94	0.93	33
trv	0.50	0.44	0.47	9
vft	1.00	0.91	0.95	11
Accuracy			0.76	265
Macro avg	0.80	0.80	0.78	265
Weighted avg	0.78	0.76	0.75	265

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 10. RoPOS with Wordpiece Encoding: classification report on a random article

	Precision	Recall	F1-score	Support
3ppg	1.00	1.00	1.00	8
3psf	1.00	1.00	1.00	22
3psm	1.00	0.45	0.62	11
abn	1.00	1.00	1.00	5
ad	0.88	1.00	0.93	7
add	0.50	1.00	0.67	1
adj	1.00	1.00	1.00	1
adp	1.00	1.00	1.00	1
adt	1.00	1.00	1.00	2
br	1.00	1.00	1.00	1
cav	1.00	1.00	1.00	2
cmn	0.81	0.95	0.88	22
co	1.00	1.00	1.00	3
coc	1.00	1.00	1.00	6
com	1.00	1.00	1.00	16
dmp	1.00	1.00	1.00	3
dtv	1.00	1.00	1.00	1
fr	0.91	0.91	0.91	23
idp	0.00	0.00	0.00	0
in	1.00	1.00	1.00	24
itv	0.60	0.75	0.67	4
mod	0.00	0.00	0.00	1
pop	1.00	1.00	1.00	5
ppn	0.93	0.79	0.86	34
qnt	1.00	0.33	0.50	3
rfp	1.00	0.67	0.80	3
spa	1.00	1.00	1.00	1
suc	1.00	1.00	1.00	2
sym	1.00	0.94	0.97	33
trv	0.88	0.78	0.82	9
vft	1.00	1.00	1.00	11
Accuracy			0.93	265
Macro avg	0.83	0.82	0.82	265
Weighted avg	0.95	0.93	0.93	265

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 11. F1-score, Precision and Recall while inferencing on texts from the same domain (Newspaper Articles)

	Precision		Recall		F1-score	
	Macro average (%)	Weighted average (%)	Macro average (%)	Weighted average (%)	Macro average (%)	Weighted average (%)
RoPOS with BPE Tokenisation	80	78	80	76	78	75
RoPOS with Wordpiece Tokenisation	83	95	82	93	82	93
Bi-LSTM	92	96	88	94	89	94
LSTM	93	98	91	97	92	97
RNN	78	91	73	89	73	89
GRU	92	98	90	96	90	96
Logistic Regression	89	95	89	94	87	94
Random Forest	87	96	87	95	86	95
SVM	85	96	86	93	84	94
Naive Bayes	83	97	84	94	82	95

because the symbols ## prefix each of the subtokens (except the first), the model is able to learn that *'ngeit'* and *'##ngeit"* are two different tokens with two different tags without any ambiguity. As a result, while inferencing, the performance improvement shown in the RoPOS Classification report of Table 10 could be attributable to this fact. Further, the F1-score, Precision and Recall for the other models implemented are as listed in Table 11. These scores are obtained while inferencing, using a random article that is not part of the training, validation or test but from the same domain. Tables 12 and 13, on the other hand, provide the set of scores produced by the models while inferencing on text from two other domains, the Bible and Khasi literary texts, that are different from the domain on which the models have been trained on.

In general, it is seen that the deep learning models perform better than the other models. Table 11 reports the scores of the various models for one random article from the same domain as the training set. Considering the fact that the article has imbalanced data, as indicated by the support counts, and that each class has equal importance, the LSTM model has a higher overall macro average F1-score.

However, when the models are used for inferencing on a random text that is not from the domain on which they have been trained which is the newspaper domain, we observe that RoPOS (with Wordpiece tokens) performs better than the other models. This is shown in Table 12, where the text has been taken, randomly, from the Bible domain and in Table 13, from the domain of Khasi literary texts which is indicated by the higher macro average F1-scores of the model in each of these cases. In this work, the macro average F1-score is a measure that is considered because the corpus has imbalanced data and each of the tags are of equal importance. This, relatively, high macro average F1-score may be attributed to the fact that RoPOS' input vectors are pretrained embeddings that have captured the semantic as well as syntactic features of the language to some degree. Therefore, its performance is more consistent and better on generic data that is outside of the training domain or in other words it generalises better.

Another benefit of RoPOS is the vocabulary size. The vocabulary size selected for the experiments in the paper is 30+k for RoPOS and 111,425 for the other deep learning models. Due to the way the vocabulary has been constructed, the likelihood of unknown words even with a fresh

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 12. F1-score, Precision and Recall while inferencing *on out of domain* texts taken from the Bible

	Precision		Recall		F1-score	
	Macro average (%)	Weighted average (%)	Macro average (%)	Weighted average (%)	Macro average (%)	Weighted average (%)
RoPOS with BPE tokenisation	83	93	84	91	83	91
RoPOS with Wordpiece tokenisation	87	97	87	94	86	95
Bi-LSTM	79	95	78	94	78	94
LSTM	79	94	78	93	78	93
RNN	73	89	71	87	71	88
GRU	85	95	85	94	84	94
Logistic Regression	80	94	80	95	80	95
Random Forest	79	95	79	98	79	94
SVM	78	94	78	92	78	93
Naive Bayes	69	96	69	91	69	93

Table 13. F1-score, Precision and Recall while inferencing on *out of domain*—literary Khasi texts

	Precision		Recall		F1-score	
	Macro average (%)	Weighted average (%)	Macro average (%)	Weighted average (%)	Macro average (%)	Weighted average (%)
RoPOS with BPE tokenisation	79	85	77	83	78	84
RoPOS with Wordpiece tokenisation	88	94	92	89	88	91
Bi-LSTM	83	92	82	92	82	92
LSTM	88	93	87	91	86	92
RNN	52	73	52	66	51	68
GRU	84	92	81	90	81	90
Logistic Regression	81	90	83	89	81	89
Random Forest	80	89	82	85	79	86
SVM	86	94	86	92	84	92
Naive Bayes	75	90	78	84	76	85

set of data is almost 0, which may not be the case with the others. This is a performance issue because there is no limit to new words particularly in the domain of newspaper articles, where foreign words, especially English, appear randomly in the text. Increasing the size of the vocabulary, therefore, entails an increase in the vectors' dimensions thereby increasing the size of the parameters of the model.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Table 14. Observed performance of RoPOS with Wordpiece

Percentage of noise (%)	Observed accuracy (%)
2	92.02
5	91.9
10	91.8
15	91.3
20	91.3

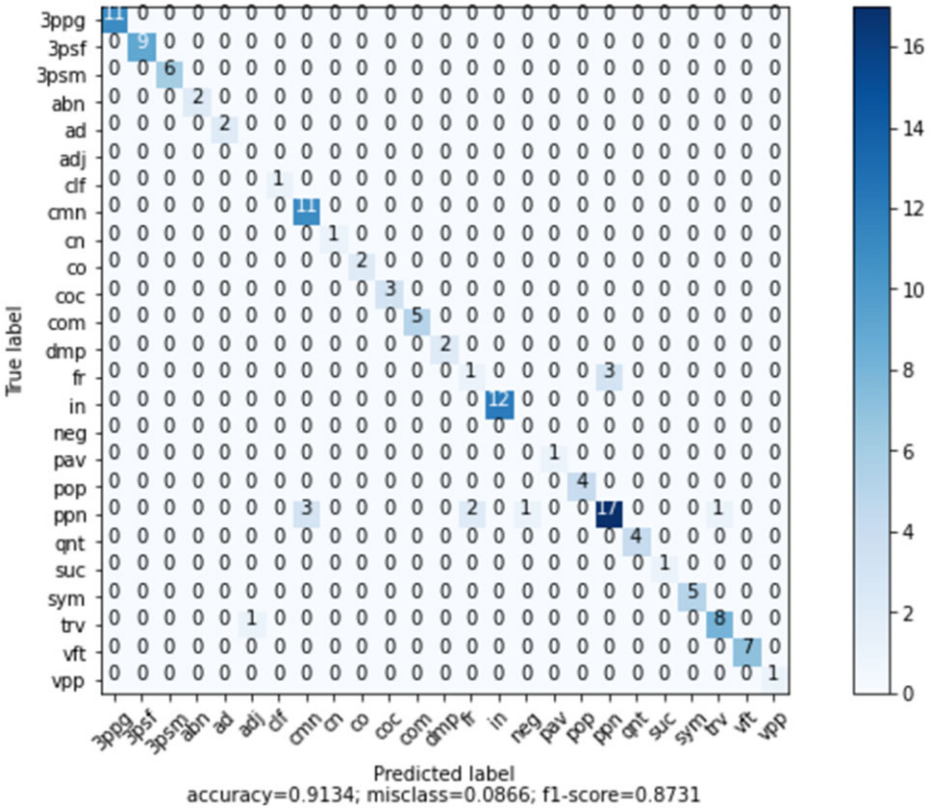

Figure 6. Confusion Matrix on inferencing with RoPOS (with Wordpiece tokens) on a random article from the newspaper domain.

Impact of noise In POS tagging, the relevant noise in the corpus is mistagged words where words appearing in the same syntactic or semantic context are annotated differently, Marquez (1999). Problems arise when the test corpus itself has mistagged words. In such a case, the real and the observed accuracy will not be the same. Here, the real accuracy is the performance of the tagger on a noise-free test set while the observed accuracy is the performance of the tagger on a noisy test set. To investigate how a noisy test set affects the performance of RoPOS, we create six sets of test data, starting from one extreme where the test set is noise-free to the other extreme where the test set contains approximately, 20 per cent noise. In between these two extremes, are test sets with, approximately, 2 per cent, 5 per cent, 10 per cent and 15 per cent noise. These noisy test

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

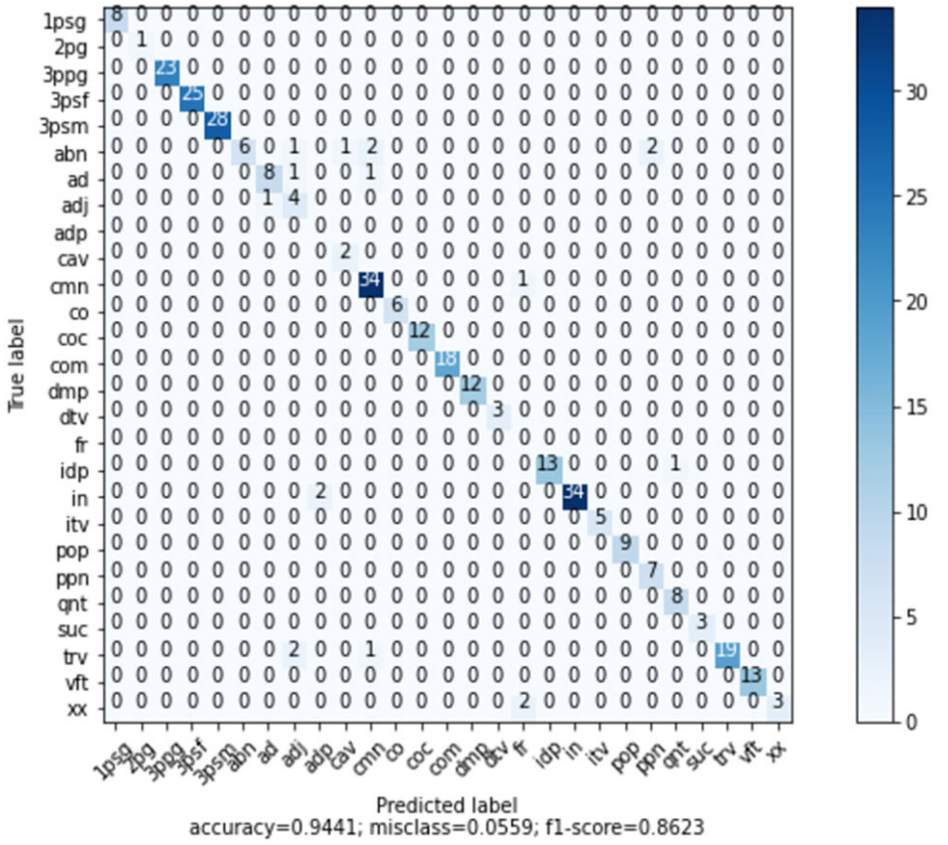

Figure 7. Confusion Matrix on inferencing with RoPOS (with Wordpiece tokens) on a random text from the Bible domain.

sets have been created by, deliberately, assigning wrong annotations to ambiguous words. The observed performance of RoPOS (with Wordpiece encoding) on these sets is shown in Table 14. Here, we considered 20 per cent (approximately), as the maximum noise. From the table, it is seen that, for test sets containing up to, approximately, 20 per cent noise, the observed performance of the model ranges between 91.3 per cent to 92.02 per cent with respect to accuracy, while the real performance is 92.07 per cent.

7. Conclusion

From the results obtained and discussed in Section 6, it is clear that Wordpiece encoding is a better tokenisation scheme than BPE for this work. It is also observed that all the other models exhibit excellent performance on training and testing data compared to RoPOS (with Wordpiece tokens) but when they are used to infer on texts that are not from within the training domain, their performance degrade. This could be because there is less randomness or unpredictability in the train and test datasets, which is mostly true for the newspaper articles utilised in this work. RoPOS (with Wordpiece tokens), on the other hand, displays a more consistent performance when we utilise it for inferencing both inside and outside of domain texts. The pretrained embeddings are contextualised and are more robust for use as inputs to the model than the raw token ids. RoPOS' performance will improve with the increase in the size of the POS tagged dataset. Therefore, because of its, relatively, consistent performance, it may be concluded that

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

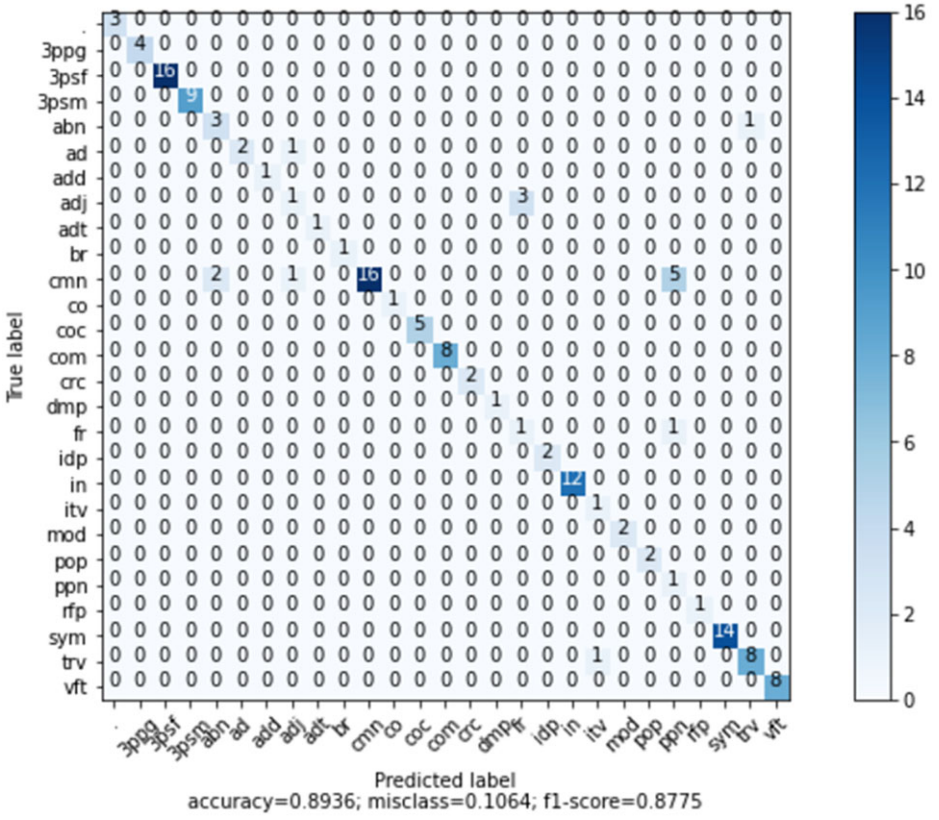

Figure 8. Confusion Matrix on inferencing with RoPOS (with Wordpiece tokens) on a random article from the domain of Khasi Literary texts.

RoPOS (with Wordpiece tokens) is a reliable model to consider for POS tagging as it exhibits generalisation.

8. Limitations and future scope

One aspect of RoPOS that may be considered as a drawback is the fact that the length of the input to the model is not only limited by the maximum sequence (sentence) length that we cap RoPOS with, but it is also further limited by the length of the tokens and/or subtokens that results from the process of tokenisation. This has been discussed in Section 3(4) where the tokenised input sequence length may be equal to or greater than the length of the original input sentence. Another limitation is that, more often, the model misclassifies certain types of nouns such as the common nouns, <CMN>, and proper nouns, <PPN>, and the foreign word tag, <FR>. The tagging of proper nouns is not always accurate, for example, it is common in Khasi to have *Wonderful*, *Preacher*, and English words of the like, as the name of a person. Such nouns have been tagged as <FR>, the foreign word tag and vice versa. The confusion matrices obtained after inferencing on articles taken from three different domains are shown in Figs. 6, 7 and 8. In all of the three confusion matrices, most of the misclassification is on the nouns and foreign words. Such ambiguity can be minimised if the embeddings could be pretrained on a larger corpus. Another possible way of minimising this is to incorporate Named Entity Recognition (NER) tags which may be a future work as there is no standard NER corpus built for Khasi. Since pretrained embeddings are the

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

core behind RoPOS, it is beneficial to improve upon them by pretraining upon a larger corpus. Thus, corpus collection and cleaning is an important future work. Next is, increasing the size of the POS tagged dataset. This task will no longer be tedious as RoPOS can be used for preliminary tagging and then manually validating them for correctness. With the availability of a significant volume of POS tagged dataset, a future work extension would be designing a shallow parser on this specific dataset. As not much work has been done in this area other than POS tagging, such extension work is a requirement. Additionally, this will support subsequent NLP tasks such as text summarisation, NER, Question Answering and others. Another area that has room for improvement is the tagging of borrowed words with the
 tag. Based on the usage context, this tag may be further classified to specify the precise POS of the borrowed word.

References

Boro K.K. and Sharma D. (2020). An in-depth study on POS tagging for Assamese language. *ADBU-Journal of Engineering Technology* **9**, 1–8.

Bulusu A. and Sucharita V. (2019). Research on machine learning techniques for POS tagging in NLP. *International Journal of Recent Technology and Engineering* **8**, 897–900.

Cho K., Merrienboer B., Bahdanau D. and Bengio Y. (2014). On the properties of neural machine translation: encoder-decoder approaches. In *SSST@EMNLP*, pp. 103–111.

Chollet F. (2018). Keras: The Python Deep Learning library.

Dalai T., Mishra T.K. and Sa P. (2022). Part-of-speech tagging of Odia language using statistical and deep learning based approaches. *ACM Transactions on Asian and Low-Resource Language Information Processing* **22**, 1–24.

Devlin J., Chang M.-W. and Toutanova K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *2019 Proceedings of NAACL-HLT*, pp. 4171–4186.

Ekbal A. and Bandyopadhyay S. (2008). Part of speech tagging in Bengali using support vector machine. In *2008 International Conference on Information Technology*, pp. 106–111.

Fei H. and Tan F. (2018). Bidirectional grid long short-term memory (BiGridLSTM): a method to address context-sensitivity and vanishing gradient. *Algorithms* **11**(11), 172.

Hochreiter S. and Schmidhuber J. (1997). Long short-term memory. *Neural Computation* **9**(8), 1735–1780.

Jahara F., Barua A., Iqbal A., Das A., Sharif O., Hoque M.M. and Sarker H. (2021). Towards POS tagging methods for Bengali language: a comparative analysis. *Intelligent Computing and Optimization* **1134**, 1111–1123.

Jordan I. (1997). Serial order: a parallel distributed processing approach. *Advances in Psychology* **121**, 471–495.

Liu Y., Ott M., Goyal N., Joshi M., Chen D., Levy O., Lewis M., Zettlemoyer L. and Stoyanov V. (2019). A Robustly Optimized BERT Pretraining Approach. ArXiv, abs/1907.11692.

Marquez L. (1999). Part-of-speech Tagging: A Machine Learning Approach based on Decision Trees, pp. 137–150.

Nongmeikapam K. and Bandyopadhyay S. (2016). Genetic Algorithm (GA) implementation for feature selection in Manipuri POS tagging. In *13th International Conference on Natural Language Processing, ICON*, pp. 267–274.

Nunsanga M.V.L., Pakray P., Lallawmsanga C. and Singh L.L.K. (2021). Part-of-speech tagging for Mizo language using conditional random field. *Computación y Sistemas* **25**(4), 803–812.

Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L., Desmaison A., Köpf A., Yang E., DeVito Z.Raison M., Tejani A., Chilamkurthy S., Steiner B., Fang L., Bai J. and Chintala S. (2019). PyTorch: an imperative style, high-performance deep learning library. In *Neural Information Processing Systems*, pp. 8026–8037.

Pathak D., Nandi S. and Sarmah P. (2022). AsPOS: Assamese part of speech tagger using deep learning approach. In *2022 IEEE/ACS 19th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1–8.

Pranckevičius T. and Marcinkevicius V. (2016). Application of logistic regression with part-of-the-speech tagging for multi-class text classification. In *2016 IEEE 4th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE)*, pp. 1–5.

Reddy B.M., Langstieh B.T., Kumar V., Reddy A.N.S., Meka A., Reddy A.G. and Thangaraj K. (2007). Austro-Asiatic tribes of Northeast India provide hitherto missing genetic link between South and Southeast Asia. *PLoS One* **2**(11), 1–12.

Schuster M. and Nakajima K. (2012). Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5149–5152.

Sennrich R., Haddow B. and Birch A. (2015). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* **1**, 1715–1725.

Thabah N.D.J., Mitri A.M., Saha G., Maji A.K. and Purkayastha B.S. (2022). A deep connection to Khasi language through pre-trained embedding. *Innovations in Systems and Software Engineering*.

https://doi.org/10.1017/nlp.2024.24 Published online by Cambridge University Press

Tham M. (2018). A hybrid POS tagger for Khasi, an under resourced language. Challenges and issues in developing an annotated corpus and HMM POS tagger for Khasi. In *Proceedings of ICON2018: 15th International Conference on Natural Language Processing*, pp. 10–19.

Tham M. (2020). A hybrid POS tagger for Khasi, an under resourced language. *International Journal of Advanced Computer Science and Applications* **11**, 333–342.

Wagner W., **Bird S.**, **Klein E. and Loper E.** (2010). Natural language processing with Python, analyzing text with the natural language toolkit. *Language Resources and Evaluation* **44**(4), 421–424.

Warjri S., **Pakray P.**, **Lyngdoh S. and Maji A.** (2018). Khasi language as dominant Part of Speech(POS) Ascendant in NLP. *International Journal of Computational Intelligence & IoT (IJCIIoT): Proceedings* **1**, 109–115.

Warjri S., **Pakray P.**, **Lyngdoh S. and Maji A.** (2019). Identification of POS tag for Khasi language based on hidden Markov model POS tagger. *Computación y Sistemas* **23**(3), 795–802.

Warjri S., **Pakray P.**,**Lyngdoh S.A. and Maji A.** (2021). Part-of-speech (POS) tagging using conditional random field (CRF) model for Khasi corpora. *International Journal of Speech Technology* **24**(4), 853–864.

Cite this article: Mitri AM, Lyngdoh EL, Warjri S, Saha G, Lyngdoh SA and Maji AK. Probing a pretrained RoBERTa on Khasi language for POS tagging. *Natural Language Processing* https://doi.org/10.1017/nlp.2024.24