BIDIRECTIONAL GATED RECURRENT UNIT FOR SHALLOW PARSING

Medari Janai Tham
Research Scholar, Department of Computer Science and Engineering,
Assam Don Bosco University, Guwahati, Assam 781017, India
medaritham16@gmail.com

Abstract - Shallow parsing is one of the natural language processing tasks, where various syntactic phrases such as noun phrases, verb phrases, and others, are identified. Here shallow parsing is considered as a sequence tagging task, and this paper presents the application of deep learning approach using recurrent neural networks (RNN) for shallow parsing. Specifically, gated recurrent unit (GRU), and bidirectional gated recurrent unit (BiGRU) are applied in parsing Khasi, an Austro-Asiatic language. These variants of the RNN address the vanishing gradient problem and the dependency of the chunk tags on both preceding and subsequent information from the sequential input data. The results have shown an improved performance compared to an existing shallow parser for Khasi.

Keywords**:** Deep Learning; Khasi shallow parser; Recurrent neural networks; Gated recurrent unit; Bidirectional gated recurrent unit.

1. Introduction

Shallow parsing is a natural language processing task, where syntactic phrases such as noun phrases, verb phrases, prepositional phrases, and others are identified in a sentence. According to Abney [Abney (1991)] these phrases, also known as chunks, are non-recursive and do not contain each other. Natural language processing (NLP) tools and resources are still lacking for Khasi, a resource poor language, spoken by one of the tribes inhabiting the state of Meghalaya located in the northeastern part of India, which is one of the challenges faced in this work. Hence, when complete syntactic parsing for Khasi is not yet available, shallow parsing is a viable option as a means for providing sufficient information that can be used in other natural language processing tasks such as information extraction, Question Answering (QA) systems, and others. Deep learning approaches have shown very promising performances on sequence tagging [Yao et al. (2014)], and in this work, shallow parsing is also taken as a sequence tagging task. A deep learning mechanism with the use of gated recurrent unit (GRU), bidirectional gated recurrent unit (BiGRU) have been carried out in a recurrent network to perform shallow parsing for Khasi. The results have shown an improved performance when compared to an existing Khasi shallow parser. However, shallow parsing for Khasi is still in its infancy, where it has been possible to identify only noun and verb chunks. The data of what constitutes other types of chunks in Khasi are yet to be in place.

2. Related Work

Gated Recurrent Unit was introduced by Cho et al. [Cho et al. (2014)] to handle long distance data dependencies. It is an alternative to long short-term memory (LSTM) [Hochreiter and Schmidhuber (1997)] as suggested by Chung et al. [Chung et al. (2014)]. In their experiments, they concluded that the performance of a GRU versus an LSTM depended on the datasets and the task at hand, and neither is superior to the other. Neural networks have been applied to shallow parsing also known as chunking using convolutional neural network (CNN), but the problem was not approached sequentially [Collobert and Weston (2008)]. Here the chunks were part of a multi tasks learning which includes part-of-speech (POS) tags, named entity recognition (NER), and others. Another application of CNN with minimal pre-processing of input features gave encouraging results on chunking and other natural language processing tasks [Collobert et al. (2011)]. A variety of LSTM networks [Huang et al. (2015)] such as bidirectional LSTM (BI-LSTM) networks, LSTM with a Conditional Random Field (CRF) layer (LSTM-CRF), and bidirectional LSTM with a CRF layer (BI-LSTM-CRF) have been carried out on POS tagging, chunking, and NER, and the BI-LSTM-CRF model gave the best performance for chunking. Yang et al. [Yang et al. (2016)] carried out a multi-tasking and cross lingual training on English and Spanish. They employed gated recurrent units and CRFs and showed promising results for POS tagging, chunking, and NER tasks. State of the art performances based on task specific modeling on chunking have been reported by Shen and Sarkar [Shen and Sarkar (2005)] as well as Sun et al. [Sun et al. (2008)] with F1 scores of 94.01 and 94.34 respectively. Another work [Zhai et al. (2017)] which incorporates CNN for extracting features from chunks and integrating different neural models for chunking on the CoNLL shared task [Tjong Kim Sang and Buchholz (2000)] achieved F1 measure of 94.72. For Khasi, the only reported work on shallow parsing was the use of a specialized Hidden Markov Model (HMM) with an F1 measure of 95.51 [Tham (2018b)]. This accuracy is possibly due to the identification of only noun and verb chunks, which are the only available chunk representations for the language.

3. A Brief Description of Deep Learning Approach to Shallow Parsing

3.1. *Recurrent neural networks (RNNs)*

RNNs ability to capture previously seen information in predicting their output is well suited with sequential data [Graves (2012)]. Fig. 1 shows an unrolled RNN where each hidden state h is evaluated not only from its input value x but from its previous state. For a given input sequence $x_1, x_2, \ldots x_n$, the RNN is represented by Eq. (1) and Eq. (2) to evaluate the output y_t. h_t denotes the hidden state at time step t, and W_{hh}, W_{hx}, and W_{yh} are the weights for the hidden-to-hidden layer connection, the input connection, and the hidden-to-output connection respectively. b_h denotes the bias for the hidden state and b_y denotes the bias for the output state. f is the activation function applied on the hidden and output nodes throughout the network.

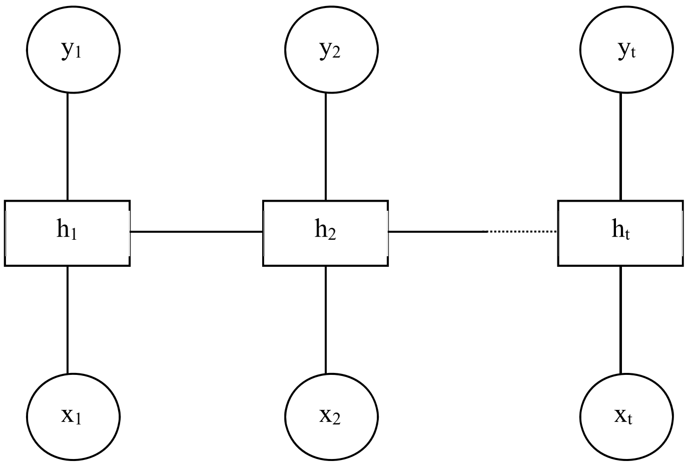

Fig. 1. An Unrolled RNN [Adapted from Graves (2012)]

$$h_t = f(W_{hh}h_{t-1} + W_{hx}x_t + b_h) \quad (1)$$
$$y_t = f(W_{yh}h_t + b_y) \quad (2)$$

3.2. *Gated Recurrent Unit (GRU)*

One variation of RNNs is a GRU [Cho et al. (2014)], which captures long term dependency and addresses the vanishing gradients problem often encountered when training RNNs. In this unit the hidden state h_t of the GRU is calculated as follows:

$$h_t = u \otimes \tilde{h}_t + (1 - u) \otimes h_{t-1} \quad (3)$$

where u is the update gate, which decides whether h_t is updated or not, and $\otimes$ is an element-wise multiplication. The update gate u is updated by applying a sigmoid logistic function σ in this way:

$$u_t = \sigma(W_{uh}h_{t-1} + W_{ux}x_t + b_u) \quad (4)$$

The candidate cell is updated by using the hyperbolic tangent as

$$\tilde{h}_t = tanh(W_{hx}x_t + W_{hh}(r \otimes h_{t-1}) + b_h) \quad (5)$$

where r is the reset gate to compute the relevance of h_{t-1} to h_t . The reset gate r is calculated in the following way,

$$r_t = \sigma(W_{rh}h_{t-1} + W_{rx}x_t + b_r) \quad (6)$$

Fig. 2 depicts a pictorial representation of the GRU given in Eq. (3), Eq. (4), Eq. (5), and Eq. (6)

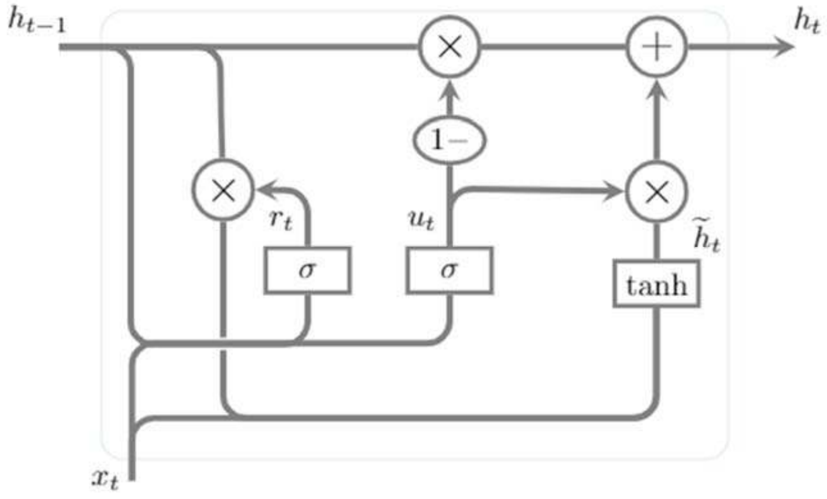

Fig. 2. Gated Recurrent Unit [Adapted from Cho et al. (2014); Chung et al. (2014)]

3.3. *Bidirectional Gated Recurrent Unit (BiGRU)*

In the current problem, the entire sequence of input is available to the parser. Hence, to take advantage of not only previously seen input, but also the next available input sequences, a BiGRU is incorporated. Here, instead of one GRU cell, two GRU cells are placed between the input and the output. One GRU cell's input (e.g h_{tf}) is the normal input sequence, while the other GRU cell's input (e.g h_{tb}) is the same input sequence in reverse order. Fig. 3 shows how these cells are placed in the network.

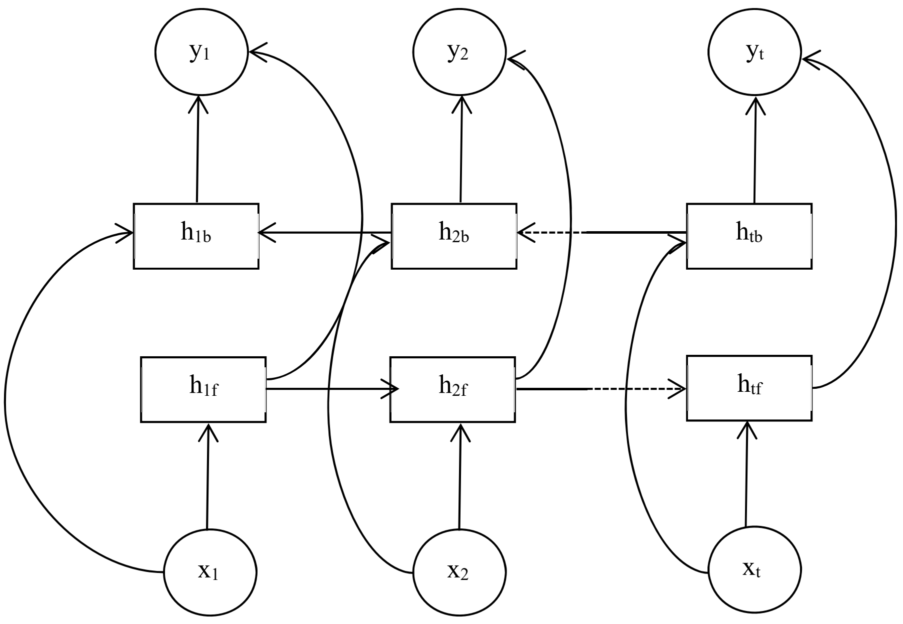

Fig. 3. Bidirectional Gated Recurrent Unit [Adapted from Graves (2012); Cho et al. (2014)]

4. Implementation

The annotated corpus for Khasi used in this work is specified in [Tham (2018b)]. The corpus contains 3,997 sentences, 86,087 tokens and 24,194 of noun and verb chunks. The present format of the training set contains only noun and verb chunks using the BIO labeling [Ramshaw and Marcus (1995)], since other chunk types are yet to be identified. Here ***B*** stands for beginning of a chunk, ***I*** means it is part of a preceding chunk, and ***O*** means it is outside of any chunk. A sample sentence from the annotated corpus is as follows:

Tiap/RB/O *tang*/RB/O *shu*/RB/O *poi*/V_VM/O *ha*/IN/O *bri*/N_NN/B-NP ,/RD_PUNC/O *u*/PR_PRP_M/B-NP *slap*/N_NN/I-NP *u*/PR_PRP/O *sdang*/V_VM/O *hap*/V_VM/O ./RD_PUNC/O

'Immediately when he reached the field the rain started falling.'

For instance the word *slap* 'rain' is tagged with the Khasi BIS POS tag N_NN [Tham (2018a)], which stands for common noun. It is also tagged with I-NP using the BIO labeling which means it is part of the preceding noun phrase [Tham (2018b)]. Since, only noun and verb chunks are annotated in the corpus, the total number of chunk tags utilized is 5. However, the total number of chunk tags used for training is 83. This is due to the transformation of the training and test data as carried out in [Tham (2018b)]. The transformed input data comprises only of POS tags, and the transformed output comprises of a combination of POS tags along with the chunk tags. The output tag is a concatenation of the POS tag along with the chunk tag using a period (.). The sample sentence given above is therefore transformed in the following way:

RB/RB.O RB/RB.O RB/RB.O V_VM/V_VM.B-VP IN/IN.O N_NN/N_NN.B-NP RD_PUNC/RD_PUNC.O PR_PRP_M/PR_PRP_M.B-NP N_NN/N_NN.I-NP PR_PRP/PR_PRP.O V_VM/V_VM.B-VP V_VM/V_VM.B-VP RD_PUNC/RD_PUNC.O

The above sentence is a sample of the training data format given to the neural network. The training is carried out using the Keras library in Google Colab a Jupyter notebook environment. As per Keras specification, the input and output data are uniquely indexed into integers i.e., a unique integer is assigned to each input token and each output tag. Another requirement of Keras is that the input sequence be of fixed length. To estimate the maximum length possible, the maximum length of the input sentence present in the training corpus is taken to be the standard length for all input sentences during training. Any sentences that are shorter are zero padded to the right during training and likewise during testing.

The neural network is initially trained on a GRU with various numbers of neurons. The test data consist of unseen data during training with 402 sentences and 2,210 noun and verb chunks. The above input sentence comprises of the Khasi BIS tags explicated in [Tham (2018a)]. The sample indicates that given any Khasi sentence, the first step required before parsing is for the sentence to be tagged with POS tags. This can be carried out using an existing Khasi POS tagger. The next step is to extract only the POS tags from the sentence, which then becomes the input sequence for the parser. The results of the network comprising GRUs are shown in Table 1. The network performed best with 256 neurons, but its performance is not at par with the existing Khasi HMM shallow parser.

Table 1. Parsing with GRU

Parser	No. of Neurons	Precision	Recall	F1 score
HMM Parser [Tham (2018b)]	-	94.39	96.65	**95.51**
GRU	64	91.9	93.92	92.9
GRU	128	91.71	93.77	92.73
GRU	256	91.17	94.99	**93.04**

Therefore, as explained earlier, to incorporate information from the future input along with previously seen input, a network comprising of BiGRU is also carried out. The results of the BiGRU parser are shown in Table 2. In both cases (GRU as well as BiGRU), a softmax function is used in the output layer for multi-classification along with the default embedding provided by Keras. In the case of the BiGRU, best performance is obtained with 256 neurons with an F1 score of 98.91. This shows that the ability to extract information from the future improves the performance of the parser when compared to the HMM parser. Table 3 shows the noun (NP) and verb (VP) chunks precision and recall of the BiGRU parser with 256 neurons.

Table 2. Parsing with BiGRU

Parser	No. of neurons	Precision	Recall	F1 score
HMM Parser [Tham (2018b)]	-	94.39	96.65	**95.51**
BiGRU	64	97.11	98.06	97.58
BiGRU	128	98.45	98.79	98.62
BiGRU	256	98.79	99.03	**98.91**

Table 3. Noun and verb chunk results

BiGRU parser (256 neurons)	Precision	Recall	F1 score
NP	97.87	98.24	98.05
VP	99.8	99.9	99.85

The results shown in Table 2 and Table 3 are very optimistic. One reason is that the HMM parser and the BiGRU parser are tested on an input sequence where the POS tags are extracted from gold data which is deemed to be a correct input sequence. The input sequence is not influenced by the performance of the Khasi POS tagger which is reported to have 95.68% accuracy [Tham (2018a)]. To gain insights to the extent of the influence of the Khasi POS tagger on the performance of the BiGRU parser, the output of the Khasi POS tagger is given as input to the BiGRU parser. In other words, firstly, the same test data is tagged with the existing POS tagger, and secondly, it is parsed by the BiGRU parser. Table 4 and Table 5 indicate the results of the BiGRU parser on the POS tagged data. Here, we can see that the performance of the parser is greatly affected by the performance of the tagger. This is a natural occurrence when one phase depends on another phase.

Table 4. Parsing on data that is already tagged by the Khasi POS tagger having 95.68% accuracy

	Precision	**Recall**	**F1 score**
BiGRU parser (256 neurons)	89.59	90.24	89.91

Table 5. Noun and verb chunk results on data that is tagged by the Khasi POS tagger

BiGRU parser (256 neurons)	**Precision**	**Recall**	**F1 score**
NP	85.0	85.24	85.12
VP	94.57	95.72	95.14

5. Conclusion

The performance of a deep learning approach with an F1 score of 98.91 has shown considerable improvement when compared to an existing HMM Parser for Khasi. BiGRUs have proven again that they work well with sequence labeling task because of the accessibility of the preceding and subsequent information in the input sequence. In reality, both the HMM parser [Tham (2018b)] and the BiGRU parser depend on the output of the POS tagger to perform shallow parsing. The effect of the dependency on the POS tagger's performance is tested with the BiGRU parser which has a better performance on test data than the available HMM parser (Table 2). Table 4 clearly indicates that even with the POS tagger's accuracy of 95.68% [Tham (2018a)], it drastically affects the BiGRU parser where naturally it's F1 score is 89.91. Because of this dependency, hopefully a POS tagger with a higher accuracy for Khasi will be available.

References

[1] Abney, S.(1991). Parsing by Chunks. In R. Berwick, S. Abney and C. Tenny (Eds), Principle–based Parsing. Kluwer Academic Publishers. MA, pp. 257-278.
[2] Cho, K., et al. (2014). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. arXiv:1409.1259v2.
[3] Chung, J., et al. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv:1412.3555v1.
[4] Collobert, R.; Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning, pp. 160–167. ACM.
[5] Collobert, J., et al. (2011). Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research (JMLR), pp. 2493-2537.
[6] Graves, A. (2012). Supervised Sequence Labelling with Recurrent Neural networks. Springer, Heidelberg.
[7] Hochreiter, S.; Schmidhuber, L. (1997). Long short-term memory. Neural Computation. 9(8), 1735–1780.
[8] Huang, Z., Xu, W.; Yu, K. (2015).Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991.
[9] Ramshaw, L. A.; Marcus, M. P. (1995). Text Chunking using Transformation-Based Learning. arXiv:cmp-lg/9505040
[10] Tjong Kim Sang, E. F.; Buchholz, S. (2000). Introduction to the conll-2000 shared task chunking. In Proceedings of CoNLL, pp. 127–132, Lisbon, Portugal.
[11] Shen, H.; Sarkar, A. (2005). Voting Between Multiple Data Representations for Text Chunking. In: Kégl B., Lapalme G. (eds) Advances in Artificial Intelligence. Canadian AI 2005. Lecture Notes in Computer Science, vol 3501. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11424918_40
[12] X. Sun et al. (2008). Modeling latent-dynamic in shallow parsing: a latent conditional model with improved inference. In COLING, pp. 841–848.
[13] Tham, M. J. (2018a). Challenges and Issues in Developing an Annotated Corpus and HMM POS Tagger for Khasi In: Proceedings of the 15th International Conference on Natural Language Processing (ICON 2018). Patiala, Punjab, India, pp 15-18.
[14] Tham, M. J. (2018b). Khasi Shallow Parser. In Proceedings of the 15th International Conference on Natural Language Processing (ICON 2018). Patiala, Punjab, India pp 43-49
[15] Yang, Z., Salakhutdinov, R. ; Cohen, W. (2016). Multi-Task Cross-Lingual Sequence Tagging from Scratch. arXiv:1603.06270v2.
[16] Yao K.. et al. (2014). Spoken Language Understanding Using Long Short-Term Memory Neural Networks. In 2014 IEEE Spoken Language Technology Workshop (SLT), pp. 189-194, IEEE.
[17] Zhai F. et al. (2017). Neural Models for Sequence Chunking. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17) pp. 3365-3371.