Protein Secondary Structure Prediction using Recurrent Neural Networks

R. Thendral, AN. Sigappi

Abstract: In bioinformatics the prediction of the secondary structure of the protein from its primary amino acid sequence is very difficult, which has a huge impact on the field of science and medicine. The hardest part is how to learn the most effective and correct protein features to improve prediction. Here, we carry out a deep learning model to enhance structure prediction. The core achievement of this paper is a group of recurrent neural networks (RNNs) that can manage high-level relational features from a pair of input protein sequence and target protein sequences. This paper contrasts the different type of recurrent network in recurrent neural networks (RNNs). In addition, the emphasis is on more advanced systems which incorporate a gating utility is called long short term memory (LSTM) unit and the newly added gated recurrent unit (GRU). This recurrent units has been calculated on the basis of predicting protein secondary structure using an amino acid sequence. The dataset has been taken from a publicly available database server (RCSB), and this study shows that advanced recurrent units LSTM is better than GRU for a long protein sequence.

Keywords : protein structure prediction, deep learning, LSTM,GRU.

I. INTRODUCTION

Recurrent neural networks give the remarkable outcome in machine methods projects particularly the input and output are long and various length [1]. More recently, [2] [3] Noted that RNN is capable of performing complicated task of machine translation in well developed system. The success has been accomplished with a recurrent neural vanilla network. The recurrent neural was advanced recurrent units called LSTM units [Hochreiter and Schmidhuber, 1997], which is highly applied in various model. By using this leading modules our project analyze two closely related module, They are LSTM and GRU units [4]. Generally, LSTM module is highly used for sequence-based activities with long-term dependencies, Based on the experiments, it is concluded that fixed amount of parameters is used for all models using for both GRU and LSTM datasets.

 Thendral R, Research Scholar, Computer Science and Engineering,AnnamalaiUniversity,India
 thendralrengarajanp@gmail.com
 Dr.Sigappi.AN, Professor, Computer Science Engineering,Annamalai University,India. aucse_sigappi@yahoo.com

II. BACKGROUND

The aim of RNN units are explained in detail and models illustrations in eventual sub-sections all take as examples the protein structure of prediction.

2.1 Recurrent neural network

A RNN is the outgrowth of a standard feed forward network which is capable of handling the variable-length input sequence. The Recurrent neural network uses a hidden recurrent state to manage varying-length sequence. Whose single activation time totally depends on the previous activation time. In Generally, the sequence $x = (x1, x2, \cdots, x_T)$, ht.
The network update the hidden state by the equation

$$h_t = \begin{cases} 0, & t = 0 \\ \emptyset(h_{t-1}, x_t), & \text{otherwise} \end{cases} \qquad (1)$$

Here φ is a non linear function. Alternatively, the Recurrent neural network gives output $y = (y1, y2, \ldots, y_T)$ which is again the length of the variable. Commonly, the upgrade of the RNN hidden state in Eq. (1) is applied as

$$h_t = g(Wx_t + Uh_{t-1}) \qquad (2)$$

here g is restricted function, Which is logistic sigmoid function and the sequence of probability can be divided into

$$p(x1, \ldots, xT) = p(x1)p(x2 \mid x1)p(x3 \mid x1, x2)\cdots p(xT \mid x1, \ldots, xT\text{-}1), \qquad (3)$$

If the last element is a end of sequence means the conditional probability distribution will be $p(xt \mid x1, \ldots, xt\text{-}1) = g(ht)$, refer ht on from Eq. (1)

During the process it was noted that[5],Recurrent neural networks tends difficult to catch long-term dependencies since the gradients tends to vanish frequently or breakout[12]. Due to that gradient-based optimization method is difficult not only for gradient magnitudes, Also for long-term dependency impact, because of this reason short-term dependence effect is hidden. There have been two powerful methods which have been used by many researchers to decrease the negative impacts of this problem. To minimize the adverse effect on this process strong strategy is to formulate a better learning method when compared to basic problematic gradient of descent [6],[7],[,8][12]

To overcome the negative effects, This process is the development of high complex activation function with normal activation mechanism, Which consists of an affine transformation accompanied with the basic element wise non linearity by using gate units, which is known as LSTM unit [Hochreiter and Schmidhuber, 1997].

Additionally, different type of recurrent unit called Gate recurrent unit proposed by[9]. RNN uses both the recurrent units to perform the task better by capturing the long-term dependencies, this functions is not restricted speech recognition[10] and machine translation[8],[12].
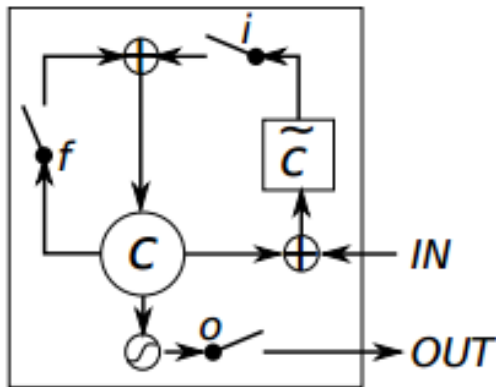
(a)LSTM units
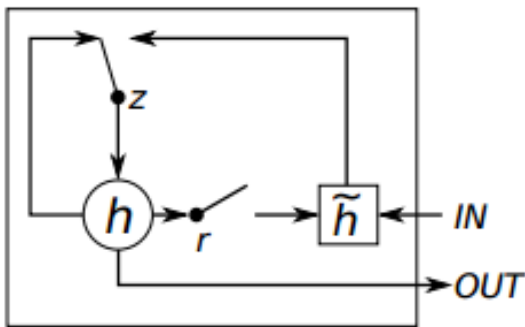
(b) GRU

Fig. 1.(a) LSTM units and (b) GRU.

From Figure i is input gate, f is forget and o is output gate, c is memory cell and c~ new memory cell content and r is reset gate, z is update gate, h is the activation and h~ is for candidate activation.

2.2 Gated Recurrent Neural Networks

Here,the performance is evaluated using newly designed recurrent units LSTM and GRU on protein amino acid sequence dataset.

2.2.1 Long short-term memory unit

Since LSTM unit was proposed by Hochreiter and Schmidhuber [1997].The recurrent unit that simply calculates the weight total of the input uses a non linear function, each j and LSTM unit store a memory c_t^j at time t. The output h_t^j , or the activation, of the LSTM unit is then

$$h_t^j = o_t^j \tanh(c_t^j),$$

here o_t^j is output gate it adjusts the memory content . The output gate is calculated by

$$o_t^j = \sigma(W_o x_t + U_o h_{t-1} + V_o c_t)_j,$$ here σ is a sigmoid function and diagonal matrix is Vo.

here c_t^j is updated pervious memory and add the new memory content $c_t^{\sim j}$:

$$c_{t=}^j \ f_t^j c_{t-1}^j + i_t^j c_t^{\sim j} \tag{4}$$

The current memory content is

$$c_t^{\sim j} = \tanh(W_c x_t + U_c h_{t-1})_j$$

Where f_t^j is the forgot gate of existing memory, and added new memory cell to input gate i_t^j and gate are determined by

$$f_t^j = \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1})_j$$
$$i_t^j = \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1})_j$$

here Vf, Vi is diagonal matrices.

The classic recurrent unit override its content on every time-step it was shown in Eq. (2). This LSTM unit decides either to store the pervious memory through the gates it was introduced. If LSTM unit finds important feature on the input sequence from a previous stage, it will easily carry this data over a long sequence. The graphical illustration was shown in fig 1(a).

2.2.2 Gated recurrent unit

The GRU was initially preferred by Cho et al. [2014], to enable every recurrent unit to adaptively fetch the dependency of dissimilar time schedules.

The activation f_t^j of the GRU at time t is a linear interpolation between the previous activation h_{t-1}^j and the candidate activation h_t^j:

$$h_t^j = (1 - z_t^j) h_{t-1}^j + z_t^j h_t^{\sim j} \tag{5}$$

Here the update gate z_t^j decides total unit reset by its activation. The update gate is calculated by

$$z_t^j = \sigma(W_z x_t + U_z h_{t-1})_j,$$

This entire process takes linear sum values between pervious state and current calculated state. This is similar to LSTM unit. The GRU exposes the whole state every time but it does not have a system to control the degree, in which the state is exposed, See Fig. 1 (b) graphical representation of the GRU.

2.2.3 Discussion

Similarities between the LSTM system and the GRU were easy to notice. From Fig.1a and b the most noticeable feature is common between them, The additive portion update t, t+1 is absent in conventional recurrent system. This typical recurrent system is often substitutes to trigger or the new value output is defined from the current input and the previous secret condition. The LSTM unit and the GRU store pervious content and adding new content on top of it.

III. EXPERIMENT AND RESULTS

3.1 Datasets

To differentiate the LSTM and GRU units uses protein structure prediction from protein databases . The Protein Data Bank database RCSB (http:/www.rcsb.org) which provides access to biological macromolecules structures and is one of the world's leading biological databases.

Long Protein sequence is used in this work for experiments on protein structure prediction. The dataset consists of a 58673 sequence of amino acids. There is no redundancy in the list of protein sequences and the results obtained from experiments are evaluated.

We have divided the obtained protein sequences into 46863 sequences for train data and 5923 for testing and the rest5 887 for validation model.

3.2 Implementation of models

For two method focus two unlike recurrent neural networks, The model design either LSTM or GRU unit. The aim of this experiment is to compare and differentiate all two units equally. The size of each model was determined so that the model have approximately the equally number of variables. To escape the over fitting problem intentionally small models are selected. This can easily distract the comparison. This produce the comparing different types of hidden units in neural networks has been done before with accuracy, loss ,time and model size and is tabulated below.

Table 1: Shows the result of LSTM and GRU units

Model	Trainable parameter	LSTM units	Epoch	Loss	Accuracy %	time
LSTM	118,904	100	40	.21	86	560 min
GRU	118,904	100	40	.23	85	480min

3.3 results and analysis

The result of our experiment shown in Table1. It is on the basis of protein sequence datasets. Also it is clear that LSTM is better than GRU a unit which is more accuracy. Figure 2 shows the learning curve of the more accuracy LSTM model which is well fit for training system not over fit or under fit. Additional the result shows that using of gating unit is more advantage than the traditional recurrent units. Also, computational time requirements for GRU units are much smaller than LSTM units.

However LSTM talks more computational time but to handle lengthy protein amino acid sequence. The LSTM recurrent networks produce high accuracy and also handle large sequence effectively when compared to GRU recurrent units.
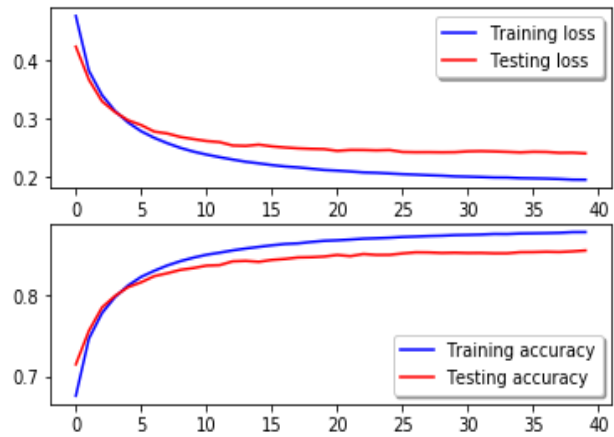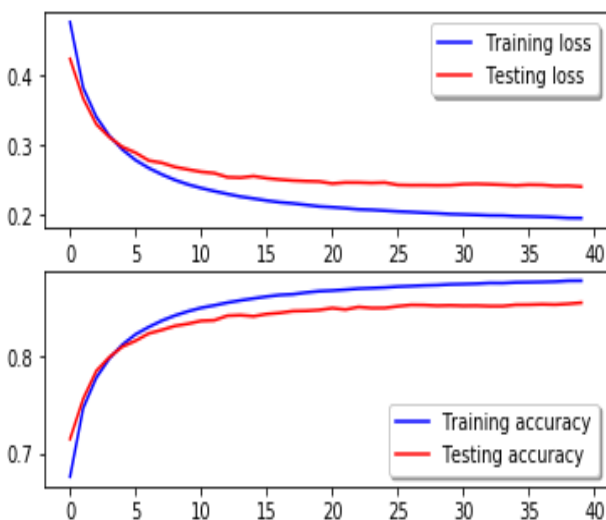
Fig. 2 1)LSTM recurrent units 2) GRU recurrent units.

The Learning curve for training accuracy and testing accuracy and it shows well fit learning models.

IV. CONCLUSIONS & FUTURE WORK

In this study, we evaluate recurrent neural networks (RNN) with two broadly used recurrent units (1) LSTM unit and (2) Gated recurrent unit (GRU). Our results focused on the application of secondary structure prediction from protein sequence with large dataset. The secondary structure prediction experiments revealed that LSTM is better than GRU for long protein sequence. In future we can computational time reduce by using GPU.

REFERENCES

1. Graves. Supervised Sequence Labelling with Recurrent Neural Networks. Studies in Computational Intelligence. Springer, 2012
2. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. Technical report, arXiv preprint arXiv:1409.3215, 2014.
3. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. Technical report, arXiv preprint arXiv:1409.0473, 2014
4. K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259, 2014.
5. Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2):157–166, 1994.
6. Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu. Advances in optimizing recurrent networks. In Proc. ICASSP 38, 2013.
7. R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In Proceedings of the 30th International Conference on Machine Learning (ICML'13). ACM, 2013. URL http://icml.cc/2013/.
8. J. Martens and I. Sutskever. Learning recurrent neural networks with Hessian-free optimization. In Proc. ICML'2011. ACM, 2011.
9. K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259, 2014.
10. A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In ICASSP'2013, pages 6645–6649. IEEE, 2013.
11. Gulcehre, K. Cho, R. Pascanu, and Y. Bengio. Learned-norm pooling for deep feedforward and recurrent neural networks. In Machine Learning and Knowledge Discovery in Databases, pages 530–546. Springer, 2014
12. Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

AUTHORS PROFILE

Thendral R, Research Scholar, Computer Science and Engineering, Annamalai University, India. She finished Master Engineering (CSE) in sathiyabama University. She has teaching experience 15 years. Her interested areas are Artificial Intelligence, Computer Graphics, Data Mining

Dr. AN. SIGAPPI, Received her Ph.D in Computer Science and Engineering from Annamalai University in 2013. She did her Master Degree in Computer science and engineering from Anna University. Currently she is serving as a Professor in the Department of Computer Science and Engineering, Annamalai University, India. Her areas of interest include Image Processing, Machine Learning, and Data Analytics. She has published more than 25 research articles in international journals and conferences.