DISAMBIGUATION IN RECURRENT NEURAL NETWORKS

Thesis Report 2020-21-M

Submitted by : B Sai Avinash
Id- 11910160, M.Tech (EE)

IIT BHILAI

Outline	
I	Abstract
II	Introduction to Recurrent Neural Networks
III	Literature review
IV	Experiments and Results
V	Optimization
VI	Initialization of Recurrent Neural Networks
VII	Future work
VIII	References

I. Abstract

Recurrent neural networks are designed mainly for modelling of sequential data. Due to vanishing and exploding gradients, it make difficult for RNN to learn long term dependencies. Solution to this problem is the use of rectified linear units, Identity initialization of weight

matrix, Hessian free optimization method , Long short-term memory(LSTM), more recently Gated recurrent units(GRU). RNN has two potential bottlenecks. One is their ability to remember about input history in their units and other is their capacity to store information in their parameters. By careful training of RNN, LSTM, GRUs, we can observe same per parameter capacity for different depths and similar results for different tasks. Information(Bits) is linear to the number of parameters and found that on an average they can hold about 5 bits per parameter. Compared different optimizers and found Adam works better, as it has properties of both Adaptive Gradient (Adagrad) and Root Mean Square Propagation(RMSProp) algorithms.

II. Introduction to Recurrent Neural Networks

Feed forward neural networks consider only current input for the task and cannot memorize previous inputs. In order to predict next character in a sentence, network should remember information about its past history and RNNs are able to solve this problem of remembering input sequences with the help of hidden state. RNN has many structures as one-to-one, one-to-many, many-to-one, many-to-many. Thus RNN perform best in handling sequential data. Long Short Term Memory networks (LSTM) and Gated Recurrent Unit(GRU) are special kind of RNN. LSTMs are mainly designed to avoid the long-term dependency problem. LSTM consists of cell, input gate, output gate and forget gate as they help in the flow of information.

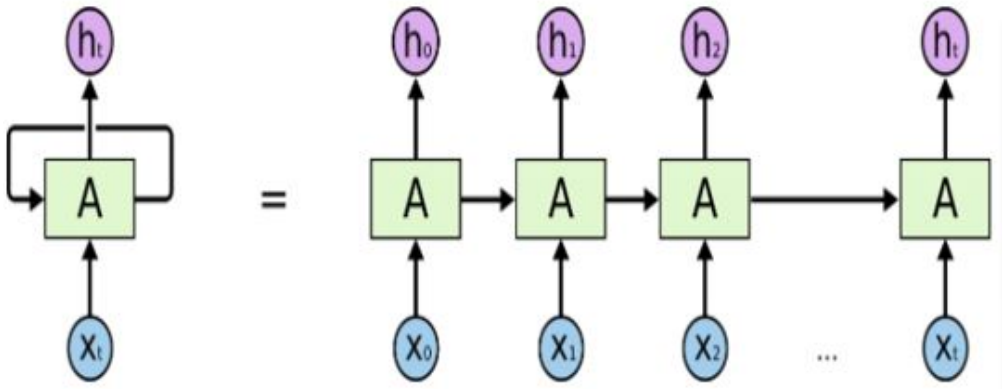

Fig. 1 Recurrent Neural Networks

III. Literature Review

Recurrent neural networks are an extension to feed forward forward neural networks which are capable of handling variable length input sequences. In Recurrent Neural Networks two main hindrances are their capacity to store information in their parameters, and their ability to remember input history in their units. We need to perform capacity

task [1], and found bits per parameter for RNN, LSTM, GRU architectures varying between 3-6 bits and average bits per parameter is nearly equals to 5 bits and observed very close performance of various RNN architectures. LSTM have slightly reduced capacity than RNN because of gating. When trained IRNN , a slight reduction in capacity is observed and found that the decrease in capacity is not because of identity initialization of weights but due to relu activation in IRNN. Two more tasks which shows similar results for all RNN architectures are step ahead character based prediction and Random continuous function(RCF). In step ahead character based prediction task, we can observe as number of parameters are increasing the cross-entropy loss being decreased , similarly in RCF, square error decreases as parameters are increasing.Thus we can resolve the ambiguity of how many bits an RNN can store in their parameters, as it was not known previously and also how different RNN architectures on careful training behave for different tasks. RNN suffer from vanishing and exploding gradients which makes it difficult for them to learn long term dependencies [3]. Solution to this is to use 1.Identity initialization of recurrent weight matrix, 2. Hessian-Free (HF) optimization, 3. LSTM. Also to find best optimizer which is suitable for non-convex optimization is another problem during training of RNN [2]. Stochastic gradient descent uses single learning rate for all weight updates and adaptive gradient algorithm(AdaGrad), Root Mean Square Propagation (RMSProp) maintains per-parameter learning rates. As we move towards optimal value we can observe decay in learning rate. AdaGrad will get struck when it is close to convergence but RMSProp overcome this problem by less aggressive on decay. Finally found that adam works best as it has features of both AdaGrad and RMSProp algorithms.

IV. Experiments and Results

A. Capacity of RNN

I have drawn a data set of binary inputs X and binary target labels Y. Inputs X has shape (n_{in}Xb) and labels Y has shape (1Xb), where b is number of samples which is treated as Hyperparameter(HP). Now I build different RNN architectures using keras. Number of input neurons in my network is equal to dimensionality of input. As b is HP, by varying b along with all HPs, to obtain the maximum mutual information(Bits), given by I (y;$\hat{y}$) = b + b (p log_2 p+(1-p) log_2(1-p)). Now i have performed similar tasks by varying number of parameters and finally plotted Bits vs parameters plot and found that they have linear relationship. Also calculated bits-per-parameter, which on an average equals to 5 bits. Also observed LSTM and GRU show slightly reduced capacity because of gating.

B. ReLU Reduce Capacity

In our capacity tasks, when we train network using IRNN which has identity initialization of weight matrix and relu as activation function, we can observe decrease in capacity. To find whether this reduction in capacity is because of relu or identity initialization, conducted an experiment on comparing RNN-tanh and RNN-relu and found RNN-relu is giving reduced capacity. So we can conclude that IRNN is giving reduced capacity because of Relu.

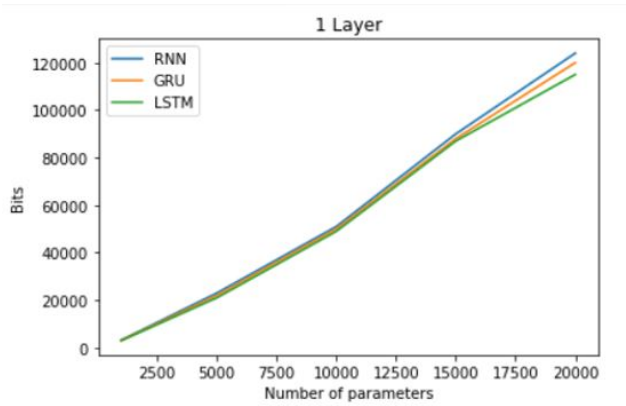

Fig. 2 Capacity for 1 hidden layer

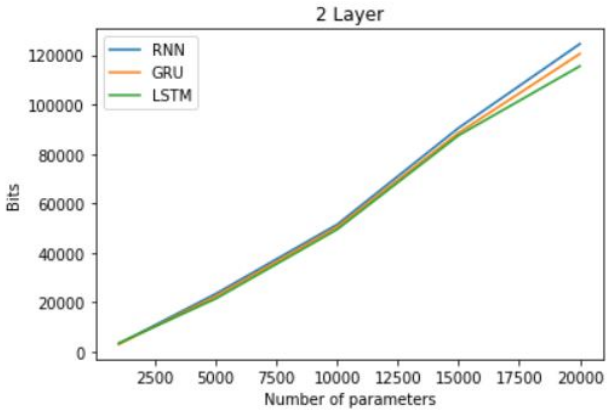

Fig. 3 Capacity for 2 hidden layers

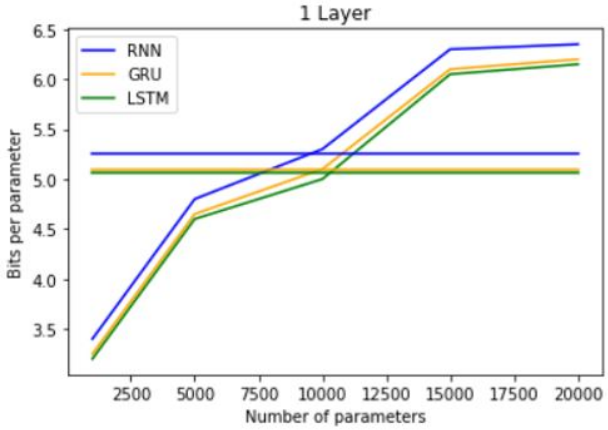

Fig. 4 Bits per parameter for 1 hidden layer

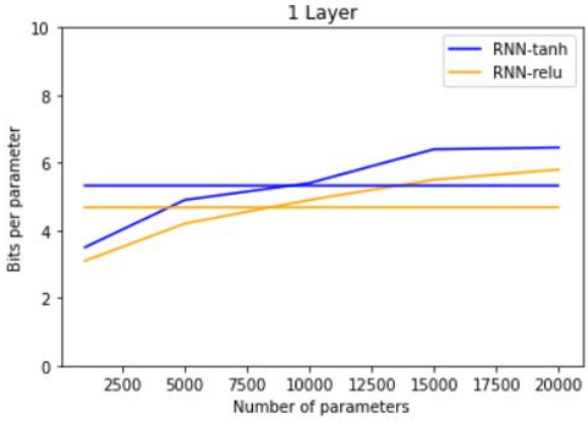

Fig. 5 ReLU vs Tanh

C. Per-Unit Capacity of RNN to remember inputs

Consider an RNN with some fixed number of inputs dimensions say 64. So we have 64 neurons in the input layer of RNN. Now we need to change the number of units per layer from 10 to 100, with careful training of RNN we can

observe as number of units per layer is greater than or equal to 64, the evaluation loss is getting closer to zero.

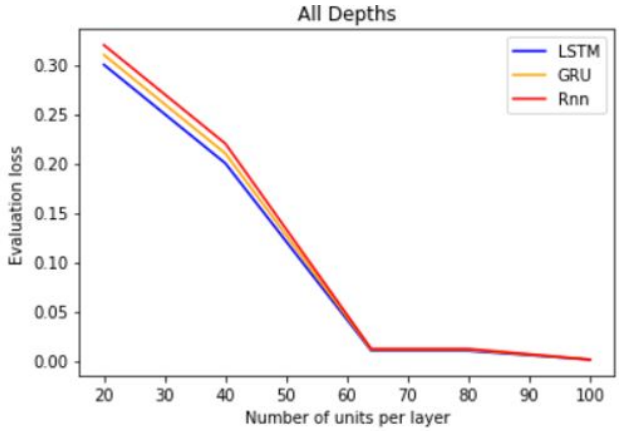

Fig. 6 Per unit capacity

D. Step ahead character based prediction

The task was to predict one character ahead in the text data set. Input and output are one-hot encoded sequences. The loss was cross-entropy on a softmax output layer and we will assign loss obtained as bits per character. Also went through various techniques for text to vector conversion, they are Bag Of Words, Term-Frequency and Inverse Document Frequency and WORD2VEC.

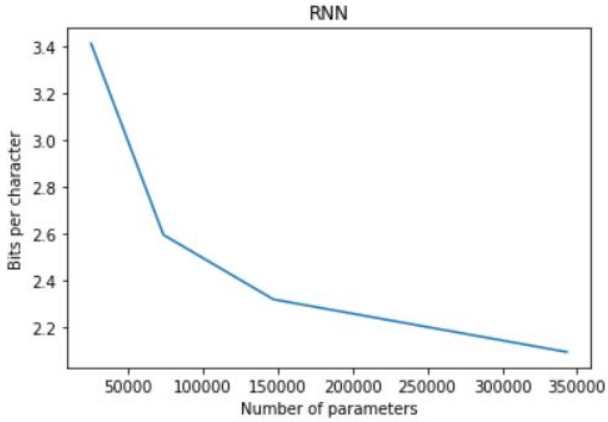

Fig. 7 RNN

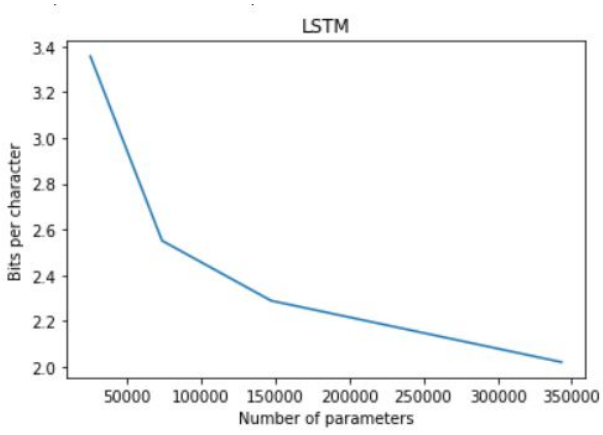

Fig. 8 LSTM

E. Random Continuous Functions (RCF)

Consider Gaussian input vector X which has N =10^6 samples and target scalar output y. The dimensionality(d) of input vector is 50 and number of neurons in input layer of RNN is 1 as i am considering my input to be of 50 time steps. As this is regression problem i have taken activation function to be linear and calculated loss function after 50 time

steps. We can observe, as the number of parameters are increasing, square error is being decreased and almost similar results for RNN and LSTM.

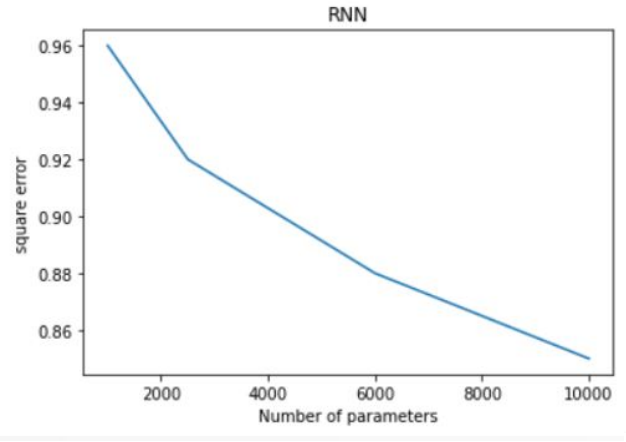

Fig. 9 RNN

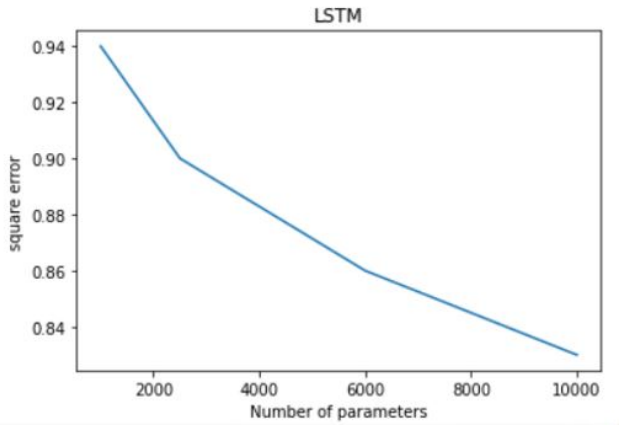

Fig. 10 LSTM

V. Optimization

To find optimizer which performs well for non-convex optimization problems. Compared Adam , SGDNestrov, Adagrad optimizers and found Adam optimizer gives best accuracy as it has properties of Adaptive Gradient and Root Mean Square Propagation algorithms.

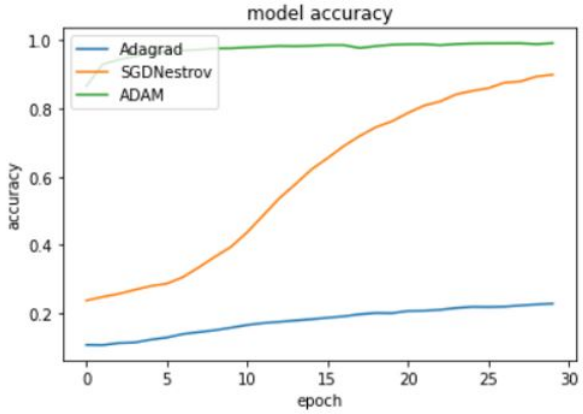

Fig. 11 ACCURACY VS EPOCH

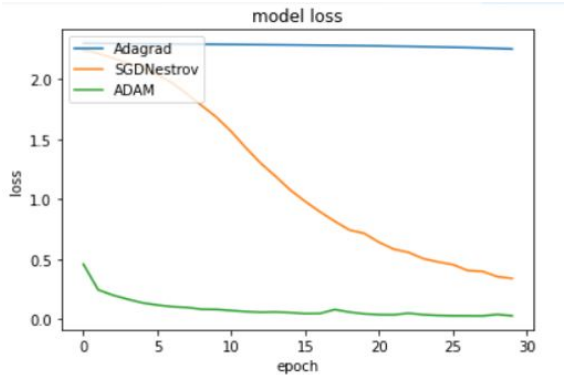

Fig. 12 LOSS VS EPOCH

VI. Initialization of RNN

Due to vanishing and exploding gradients it make difficult for RNN to learn long term dependencies. Solution to this is 1. Use Identity initialization of recurrent weight matrix, 2. Hessian-Free (HF) optimization, 3. LSTM. Also

found He-initialization works better for ReLU.

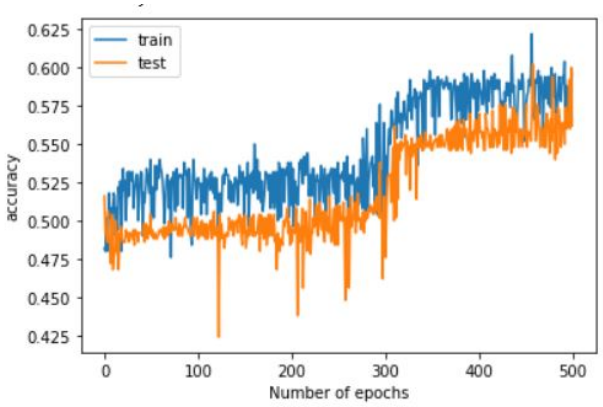

Fig. 13 RNN-Tanh

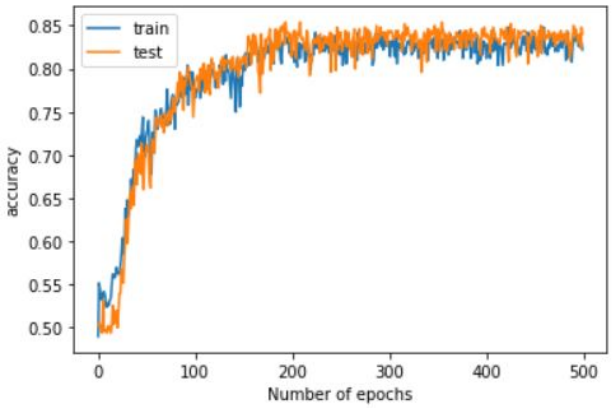

Fig. 14 RNN-ReLU

VII. Future Work

I need to look into various network architectures and optimization techniques to overcome difficulty due to vanishing and exploding gradient problems for achieving long term dependencies. Also understanding RNN is a difficult task, because of complex architectures of LSTM and GRU. So i need to look into minimal design of RNN, which can achieve accuracy comparable to LSTM and GRU.

VIII. References

1) Collins, Jasmine, Jascha Sohl-Dickstein, and David Sussillo. "Capacity and trainability in recurrent neural networks." arXiv preprint arXiv:1611.09913 (2016).
2) Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).
3) Le, Quoc V., Navdeep Jaitly, and Geoffrey E. Hinton. "A simple way to initialize recurrent networks of rectified linear units." arXiv preprint arXiv:1504.00941 (2015).
4) Talathi, Sachin S., and Aniket Vartak. "Improving performance of recurrent neural network with relu nonlinearity." arXiv preprint arXiv:1511.03771 (2015).
5) Doya, Kenji. "Universality of fully connected recurrent neural networks." Dept. of Biology, UCSD, Tech. Rep (1993).
6) Gers, Felix A., Jürgen Schmidhuber, and Fred Cummins. "Learning to forget: Continual prediction with LSTM." (1999): 850-855.

7) Short Term Memory Capacity in Networks via the Restricted Isometry Property.
8) Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation
9) Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems. 2014.
10) Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).
11) DRAW: A Recurrent Neural Network For Image Generation
12) Greff, Klaus, et al. "LSTM: A search space odyssey." IEEE transactions on neural networks and learning systems 28.10 (2016): 2222-2232.
13) Karpathy, Andrej, Justin Johnson, and Li Fei-Fei. "Visualizing and understanding recurrent networks." arXiv preprint arXiv:1506.02078 (2015).
14) Zhou, Guo-Bing, et al. "Minimal gated unit for recurrent neural networks." International Journal of Automation and Computing 13.3 (2016): 226-234.