International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-1, October, 2019

A Novel Architecture for Predicting Pneumonia Patients by using LSTM, GRU and CNN

Nitin Arora, Ahatsham, Anupam Singh, Vivek Shahare

Abstract*: The Models based only on Long-Short Term Memory (LSTM), Gated Recurrent Unit (GRU) and Dynamic Recurrent Neural Network (Dynamic RNN) are not sufficient for prediction of a pneumonia patients using image processing. The proposed network uses the properties of LSTM, GRU and Convolutional Neural Network like capacity to remember long-term memory and handling the input parameters dynamically. Investigating these networks, it is found that the proposed network have a deep insight on the medical image before it can remember then forward the necessary information. In the proposed model, the LSTM and GRU directly connected with dual 1x1 convolutional network followed by the classification layers resulting with better performance. The proposed model provides the test accuracy of 94.20% and test loss of 0.04749 when tested on dataset of pneumonia patients consisting of 2 classes (normal chest, diseased chest with pneumonia) has provided better results than LSTM, GRU, Dynamic RNN and Convolutional with LSTM in all aspects like training loss, training accuracy, testing loss and testing accuracy.

Keywords: LSTM, GRU, DRNN, LRCN, 1DCRNN

I. INTRODUCTION

In the past few years, advancement in neural networks have given tremendous results in accuracy and precision. The perceptron networks where used in many applications like image classification, natural language processing tasks and many more but some drawbacks were identified in it. Thus a new network named RNN which can also be called as Recurrent Neural Network. Suppose there is a multilayer perceptron having 3 hidden layers and when all the weights and biases are equal, then it gets squashed into a single layer called recurrent layer. It mainly focuses in the domain of natural language processing. It can be also said that the RNN is used for sequence learning. Having loops in there network makes them more unique because of that the input information persists in the network. This method has given incredible success in speech recognition, language modeling, translation and image captioning. The information persisting property means that the RNN is capable to connect previous information to the present task, but was proved wrong due to wide gap between some information. This problem was referred by Hochreiter [1]. Then comes DRNN also called as Dynamic Recurrent Neural Network which was coined by Barak A. Pearlmutter [2]. Its response last longer than the input pulse but it does not assume non-linarites in the system and thus depends on the output only. Then came the Long-Short Term Memory which was coined by Hochreiter and Schmidhuber [3] usually called as LSTM network. It full filled the defaults faced by the RNN networks. It has the capacity to remember information for a long period of time. They are now widely used for language modeling and for speech recognition. After LSTM, Cho et al. [4] coined the word GRU also called as Gated Recurrent Unit. GRU is the recent development in the natural language processing task. It solved the vanishing gradient problem that persists in Recurrent Neural Network. It uses two gates that are Update Gate and Reset Gate. Through these two vectors only, the decision is made that which information should be passed to the output. Using these methods, the proposed method focuses only on the image processing domain. Image processing plays a vital role in health care domain. There is an increased need of an efficient and robust algorithm to correctly classify and generate the information regarding the disease of the patients. By applying image processing techniques, the diagnosis of a particular disease can be more accurate, clear and also the quality of the services can be improved. Classification in image processing is the most widely used technique in health care organization. But before classification process it is very important to extract correct features so that the prediction becomes more accurate. The classification process results in the prediction of the target classes. For chest related disease mainly neural network are used for the prediction of Lung Cancer, Asthma, and Pneumonia etc. Pneumonia is one of the chest related disease. When a person is suffering from pneumonia, then it can be said that his/her lungs suffered from inflammation. Inflammation is the word used when the lungs experience an attack of some external entity like bacteria, virus, fungus, etc. Due to this, the body's immune produces inflammation in the affected area. Thus, fluid and pus comes in contact with the lung's normal function and causes pneumonia. It mainly occurs in children and 28-34% deaths are caused by the pneumonia in developed countries and 95% deaths occurred in the developing countries. The architecture is built with the combinations of LSTM, GRU and also convolutional layer with pooling layer for feature extraction and for down sampling purpose followed by batch normalization function. Furthermore for classification purpose one fully connected layer is used followed by dropout function.

Revised Manuscript Received on October 30, 2019.
* Correspondence Author
Nitin Arora*, School of Computer Science, University of Petroleum and Energy Studies, Dehradun, India. Email: narora@ddn.upes.ac.in
Ahatsham, School of Computer Science, University of Petroleum and Energy Studies, Dehradun, India. Email: ahatsham@ddn.upes.ac.in
Anupam Singh, School of Computer Science, University of Petroleum and Energy Studies, Dehradun, India. Email: anupam.singh@ddn.upes.ac.in
Vivek Shahare, School of Computer Science, University of Petroleum and Energy Studies, Dehradun, India. Email: vshahare@ddn.upes.ac.in

The proposed architecture also have the ability to train left out neurons through regression layer and uses Adam optimizer for optimization of the network.

II. LITERATURE REVIEW

Tara N. Sainath et al. [5] proposed a network called Convolutional, Long Short-Term Memory, fully Connected Deep Neural Networks where they have combined Convolutional Layer, LSTM and DNN layers into a single network and called it Long Short-Term Memory, Fully Connected Deep Neural Network. The dataset on which the proposed network was trained and tested are the noises which are taken from the YouTube and daily life noisy environment recording. The proposed network that is CLDNN provided 4-6% improvement in WER over an LSTM.

Zachary C. Lipton et al.[6] proposed a paper called A Critical Review of Recurrent Neural Networks for Sequence Learning where they presented a review on RNN, LSTM and Bidirectional RNN and based on these networks they have presented some application also which are Representation of natural language inputs and outputs, Evaluation methodology, Natural language translation and image captioning. Haşim Sak et al. [7] proposed a paper called Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling where they tested their proposed network on Google voice search task. LSTM RNN having 5 layers deep network with 440 cells and 13M parameters had gave the best performance of 10.8% WER when compared with other LSTM RNN networks having different number of layers. LSTMP RNN architecture of one layers deep network with 800 cells and 512 recurrent projection units has given 10.7% WER. The best DNN models with 85M parameters gave 11.3% WER at the same beam. Junyoung Chung et al. [4] proposed a paper called Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling where recurrent networks like tanh, LSTM and GRU are compared on two datasets which are Music dataset and Ubisoft dataset. The music dataset are further categorized into four parts that are Nottingham, JSB Chorales, Muse Data and Piano-midi where in Nottingham tanh has the least test average negative log-probability of 3.13, in JSB Chorales, GRU has the least test average negative log-probability of 8.54, in MuseData, GRU has the le6t test average negative log-probability of 5.99 and in Piano-midi, GRU has the least test average negative log-probability of 8.82. Ubisoft dataset is also divided into two parts that are Ubisoft dataset A and Ubisoft dataset B where in Ubisoft dataset A, LSTM has the least average negative log-probability of 2.70 and in Ubisoft dataset B, GRU has the least average negative log-probability of 0.88. However a concrete result was not found when compared between LSTM and GRU, which is much better. For music dataset GRU has performed well but for Ubisoft, it does not perform much. Thus the performance of GRU can differ for different dataset. Tristan Stérin et al.[8] presented a paper on An Intrinsic Difference Between Vanilla RNNs and GRU Models where the proposed that only specifying gradient training issue is not just sufficient to compare vanilla RNN with LSTM and GRU. Much better model can be constructed if actual problem is specified which can be done with the Reber's grammar rule. Wenling Shang et al. [9] presented a paper on Understanding and Improving Convolutional Neural Networks via Concatenated Rectified Linear Units where the applied CReLU activation function in different convolutional networks. When tested on CIFAR-10 dataset, VGG+CReLU has the least error rate of 5.09 and when tested on CIFAR-100 dataset, VGG+CReLU got the least error rate of 23.66 when compared with other models like Rippel, Snoek et al. [10][11], Liang & Hu[12], Wenling et al.[9], Srivastava et al.[13]. When tested on Imagenet dataset, CReLU applied on convolutional layer from 1 to 4 have the least top-1 error rate of 39.82 and least top-5 error rate of 18.28 and the parameters where 10.1M. CReLU applied in convolutional layer 1, 4 and 7 has least top-1 † of 35.70 and top-5 † of 15.32. Sign (†) means that error rate are obtained by averaging scores from 10 patches. Djork-Arné Clevert et al.[14] presented a paper on Exponential Linear Unit also called as ELU where they applied ELU activation function in the network having more than 5 hidden layers and output preformed ReLU activation function, LReLU and PReLU activation function. ELU network reaches the 20% top-5 error rate after 160K iteration where ReLU takes 200K iteration when tested on Imagenet dataset and when tested on CIFAR-10 dataset, it gives the test error of 6.55% and in CIFAR-100 dataset, it gives test error of 24.28%. Sergey Ioffe et al.[15] presented a paper on Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift where they proposed of appling batch normalization function to the network because the distribution of each layer input differ when it is passed on to the next layer, as the parameters of the previous layer changes. This leads to slow training process. Thus this problem can be addressed by batch normalization. When inception model having batch normalization is tested on Imagenet dataset gave the least top-1 error rate of 20.1% and top-5 error rate of 4.9%. Nitish Srivastava et al.[16] presented a paper on Dropout: A Simple Way to Prevent Neural Networks from Overfitting where they proposed dropout function on the neural networks to reduce over fitting. Due to this the error rate because less when tested on datasets like MNIST, TIMIT, CIFAR-10 and CIFAR-100, Street View House Numbers, ImageNet, Reuters-RCV1 and Alternative Splicing dataset. Using dropout function with DBM when tested on MNIST dataset gives the error rate of 0.79%. Dropout with Convnet and maxpooling when tested on Street View House Numbers gives the error rate of 2.47%. Dropout with Convnet and maxpooling when tested on CIFAR-10 gives error rate of 12.61% and on CIFAR-100 gives error rate of 37.20%. Dropout with Convolutional networks when tested on Imagenet dataset gives error rate of 16%. Dropout with DBM(8 layers and 4 layers) when tested on TIMIT dataset gives error rate of 19.7%. Dropout Neural Network is compared with Bayesian Neural Network where Bayesian Neural Network achieved 623 bits code quality more than dropout neural network. Furthermore comparison is also made on basis of regularization methods on MNIST where Dropout+Max-norm has the least test classification rate of 1.05%. Yann LeCun et al. [17] presented a paper on Convolutional Networks for Images, Speech and Time Series where he proposed that as convolutional network is inspired by the mammal's cortex, it removes the need for the feature extraction but normalizing the image for orientation is still required and fully invariant recognition is still beyond reach.

Dabal Pedamonti [18] presented a paper on Comparison of non-linear activation functions for deep neural network on MNIST classification task where they compared ReLU, LReLU, ELU and SELU on the basis of loss and accuracy. For SELU, 0.998 training accuracy was achieved when learning rate was 0.01 and 1.000 training accuracy when learning rate was 0.05, 0.1 and 0.2. For testing, 0.974 accuracy at 0.01 learning rate, 0.976 at 0.05 learning rate, 0.978 at 0.1 learning rate and 0.981 at 0.2 learning rate. However, in this ELU was preferred over SELU as in terms of loss and accuracy, ELU has achieved better performance.

III. METHODOLOGY

A. Dropout

Dropout is the method of dropping out a neuron randomly from a network. Suppose there is a network having 2 neurons in the hidden layer and one neuron from the outer layer. When the input feature is feed into one of the neuron of hidden layer, suppose 1st neuron outputs the prediction with a correct rate of 80% and will be passed on to the output layer and then the output layer also passes the same result. When the input is feed into the 2nd neuron of a hidden layer, then it gives some random prediction which goes to the output layer. There are chances that the prediction covers noisy feature point which causes over fitting. Thus while applying dropout, we can remove one neuron randomly. With an iteration, 2nd neuron can be removed and thus 1st neuron will be forced to learn the input features whose output will be passed on to the output layer. After this, 1st neuron will be removed and 2nd neuron will be forced to learn input features whose output will be passed on to the output layer. Now there will be 100% chances that output form the output layer gives the prediction rate, which will be more than 80%. Therefore, dropout solves the problem of over fitting. Thus, there are two key points regarding the process of dropout. First, one is dropping units while training and the second is scaling output to be matched between training and testing. Suppose we have a set of thinned layer as m and a network without dropout as $\mathcal{M}$. $\mathbf{y}^M$ as output of thinned network M where the weights of m and $\mathcal{M}$ are equal. Therefore, the expected output will be

$$E[y_j] = \sum_{M \epsilon\, m} \Pr(M) y_j^M \quad (1)$$

Let m* be the set of network in which unit j is active, then |m*|=p|m|. Assuming $\Pr(M) = \frac{1}{|m|}$ and $y^M = y^{\mathcal{M}}$. Therefore the output will be

$$E[y_j] = p|m| \frac{1}{|m|} y_j^{\mathcal{M}} = p y_j^{\mathcal{M}} \quad (2)$$

Now, there are two methods to fulfill the output to be matched between training and testing time. First is to scale down the weights that is $w_{ji}' = p w_{ji}$. Second is to scale the output at training time same as done for testing time that is $y_j' = \frac{1}{p} y_j$.

B. Batch Normalization

Batch Normalization is the normalization the outputs of the hidden layer. Through batch normalization, hidden layer gets the ability of learning some of the features of the inputs by itself. Batch normalization has two parameters that are gamma (γ) and beta (β). These two hyper parameters are learned by the network. They help in keeping the mean and variance fixed. Due to fixed mean and variance, the next hidden layers does not have to be dependent on the previous hidden layers and thus making the hidden layer capable of learning some features independently. Due to this also, the gradient descent reduces the oscillation while approaching towards global minimum. The formula for the batch normalization can be given as [15]

$$\mu_\beta \leftarrow \frac{1}{m} \sum_{i=1}^{m} x_i \quad \text{//mini-batch mean} \quad (3)$$

$$\sigma_\beta^2 \leftarrow \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_\beta)^2 \quad \text{//mini-batch variance} \quad (4)$$

$$\hat{x_i} \leftarrow \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}} \quad \text{//Normalize} \quad (5)$$

$$y_i \leftarrow \gamma \hat{x_i} + \beta \equiv BN_{\gamma,\beta}(x_i) \quad \text{//Scale and Shift} \quad (6)$$

Where, x_i represents the input feature and the output of the hidden layer, which has not passed from the activation function.

C. Exponential Linear Unit

ELU known as exponential linear unit, which can be given as

$$f(x) = \begin{cases} x, & x \geq 0 \\ \alpha(e^x - 1), & x < 0 \end{cases} \quad (7)$$

For the positive value of x, where x is the input, it acts like ReLU but for negative value, ELU activation function is bounded by fixed value -1 for α = 1. Unlike ReLU, ELU can take negative values which get saturated in the negative part of the domain from -1 to 0. Due to this, the mean activation comes closer to 0 and this makes the learning much better. If the input is positive for both ELU and ReLU, than in ELU mean activation value is much closer to 0 than in ReLU. As ReLU have more positive mean activation value, thus a bias is introduced in the next layer which can slow down the learning. Every positive mean value introduces a bias for the next layer but those which have more positive mean will result in slow learning.

D. Concatenated Rectified Linear Unit

CReLU can also be termed as Concatenated Rectified Linear Unit, which can be given as:

$$f_{CReL}(x) = [\max(0, x); \max(0, -x)] \quad (8)$$

As ReLU takes only positive inputs due to which the gradient becomes 1 and gets the ability to learn any data but if the inputs are negative, it becomes zero automatically due to which the gradient becomes 0 and results in dead neuron. However, CReLU takes both negative and positive inputs. Means CReLU have two outputs that is one positive ReLU and one negative ReLU. These two gets concatenated together and forms Concatenated ReLU. Suppose if x is our input, than for positive x, CReLU produces [x, 0] and for negative x, CReLU produces [0, x]. Due to these two outputs, CReLU doubles the output dimension.

It also have improved effect on regularization, invariance and reconstruction property as proposed in the paper by Wenling Shang et al. [9]. When CReLU applied on the proposed network, also improved he classification task.
But it was not applied in the convolutional network when talking about the proposed network, as it takes more time than ELU to execute.

E. Long Short-Term Memory

LSTM is termed as Long Short-Term Memory. As RNN also called as Recurrent Neural Network is used previously for natural language processing which has the tendency of remembering some information which comes from the input data. But the problem with RNN is that, it is good only when we are dealing with short term dependencies. It does not give accurate results when we are dealing with long term dependencies. The reason behind this is the vanishing gradient problem. This problem was solved by the LSTM network. The information flows through the network with the help of the mechanism called cell state. The input information at a particular cell state has three dependencies that are the previous cell state, previous hidden state and the input at the current time step. It has also another property, which is called conveyor belt. Through this, LSTM are able to slightly modify the information, which helps them in remembering it. The LSTM is executed in the following steps

$$i_t = \sigma(x_t U^i + h_{t-1} W^i) \quad (9)$$

Where σ specifies sigmoid activation function, x_t as an input, U^i and W^i are the weights of input i and h_{t-1} which is used to hold the information.
Input gate is responsible for adding the information to the cell state. This process requires the involvement of sigmoid function, which helps what information should be needed to add to the cell state. Then a vector space is created where all the possible values are stored which can be added. This involves Tanh function as it has range from -1 to 1. At last, through multiplication of sigmoid gate and Tanh gate, we finally gets the information, which is needed to be added to the cell state.

$$f_t = \sigma(x_t U^f + h_{t-1} W^f) \quad (10)$$

Forget gate helps in forgetting that information which are not necessary and are not needed any more. This removes the noisy data from the stack of information.

$$O_t = \sigma(x_t U^o + h_{t-1} W^o) \quad (11)$$

The output gate selects only the useful and sensible information from the cell state and put it forward means put it as an output.

$$\tilde{C}_t = \tanh(x_t U^g + h_{t-1} W^g) \quad (12)$$
$$C_t = \sigma(f_t * C_{t-1} + i_t * \tilde{C}_t) \quad (13)$$
$$h_t = \tanh(C_t) * O_t \quad (14)$$

Where, $\tilde{C}_t$ is the candidate hidden state, C_t is the internal memory and h_t is the output hidden state.

F. Gated Recurrent Unit

GRU also termed as Gated Recurrent Unit was proposed by Cho et al.[4] which aims to solve vanishing gradient problem. Unlike LSTM, GRU does not remove the irrelevant information into trash but keep that information with it. This keeping of information solves the vanishing gradient problem since model is not washing out the new input every single time steps. The GRU consist of two gated vectors which are Update Gate and Reset Gate. The Update Gate is given by:

$$z_t = \sigma(x_t U^z + h_{t-1} W^z) \quad (15)$$

Here x_t is the input which is multiplied by its weight W (z) and h_{t-1} which holds the information of the previous unit is multiplied by its weight U (z). This whole process is squashed between 0 and 1 with the help of sigmoid function. This helps the network to determine which information should be propagated forward. Now for the Reset Gate, it helps the network to decide which type of information is needed to be forget. The Reset Gate is given by:

$$r_t = \sigma(x_t U^r + h_{t-1} W^r) \quad (16)$$

Then comes the current memory content which is also called the storage area of GRU. The needed information which we get from the Reset Gate gets stored in the memory content which can be given as:

$$\sim h_t = \tanh(x_t U^h + (r_t * h_{t-1}) W^h) \quad (17)$$

The process is then squashed with the help of Tanh activation function. At last, comes the current memory at current time step. Through this, the required information which is stored is fetched by the Update Gate as this gate knows which information needs to be put forward. This is given as:

$$h_t = (1 - z_t) * h_{t-1} + z_t * \sim h_t \quad (18)$$

G. 1x1 Convolutional Layer

1x1 Convolutional layer is proposed in the paper by Christian Szegedy et al.[19] where they have applied 1x1 convolutional layer in GoogleNet Architecture. Suppose that we have a convolutional layer of shape (N x F x H x W) where N is the batch size, F is the number of convolutional filters and H and W are spatial dimensions. When we change F to F_1 and if $F_1 <$ F, then we get the reduced dimensionality in the filter space. Due to reduced dimensionality, chances are increased of getting better features because of which accuracy of the network is increased. It is mainly used in those network where the requirements of computation is high. Thus it's another advantage is that, it speeds up the network. Also in GoogleNet Architecture, first 1x1 convolutional layer is applied after the input layer and then expensive 3x3 convolutional layer and 5x5 convolutional layer is applied.

H. Network Architecture

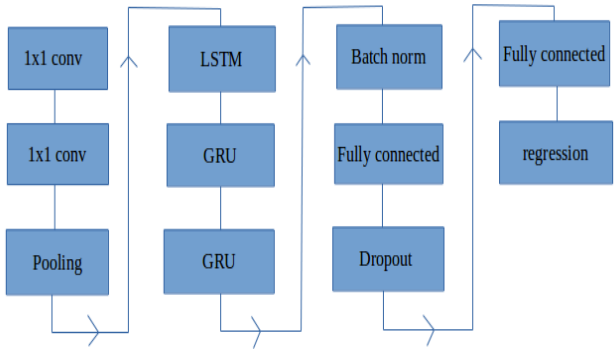

Fig. 1. The above figure shows the architecture of the 1D Convolutional Neural Network.

In the above architecture, the first two layers are of 1x1 convolutional layers. Two 1x1 convolutional layer is used to increase the power of receptive field where first convolutional layer has 32 number of neurons of filter size 2 and the other convolutional layer has 64 number of neurons of filter size Both of them used the ELU activation function. The extracted feature through the receptive field is then down sampled by pooling layer. After this, LSTM layer is used having 128 unit cells and also having ELU activation function and sigmoid as inner activation function. Two GRU is used to make this process dense. The first GRU layer consists of 64 unit cells and the other GRU layer consists of 128 unit cells. Both uses ELU activation function and sigmoid as inner activation function. But in the proposed network, ELU activation function is used for the squashing of Update gate and the Current Memory Content because when we use Tanh activation function, there is a chance of over fitting in the most important area where the sensible information are stored. Over fitting can be caused due to the Tanh activation function because as the range of Tanh is [-1, 1], thus a large input space is squashed into a very low range. This results in high variance and thus can causes the problem of over fitting. High variance means a major change in the prediction result when the dataset in changed. After that, Batch Normalization function is used so that the layers gets the ability to learn some of the features by itself and should not much dependent on the previous layers. After that for classification purpose, fully connected layer is used having 32 neurons followed by the dropout function having 0.8 kernel size. After the output layer, regression layer is used to train the left out neurons. The complete network was tested up to 40 iterations.

IV. RESULTS

The proposed architecture when tested on the training dataset of pneumonia has resulted in exceptionally great performance and is presented in this section. All the networks are trained and tested up to 40 iterations for better optimization using Adam optimizer.

Table. 1. Comparative Results of the Proposed Network

Network	Train Loss	Test Loss	Train Accuracy	Test Accuracy
LSTM	0.09610	0.08758	0.8643	0.8800
GRU	0.09095	0.08964	0.8789	0.8880
DRNN	0.09719	0.09875	0.8637	0.8640
LRCN	0.08837	0.09084	0.8789	0.8740
1DCRNN	**0.04119**	**0.04749**	**0.9492**	**0.9420**

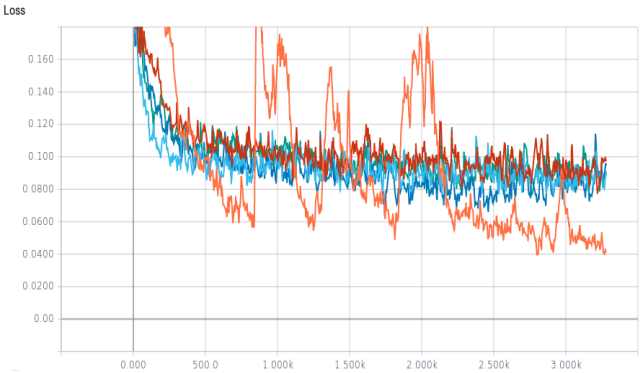

Fig. 2. Comparison of Train Loss

The top graph shows the train loss and the bottom graph shows the test loss when compared with proposed network that is 1DCRNN with LSTM, GRU, DRNN and LRCN. The orange line signifies 1DCRNN, blue line signifies LSTM, red line signifies DRNN, light blue line signifies GRU and green line signifies LRCN. The x-axis signifies time step and y-axis signifies loss in both the graphs.

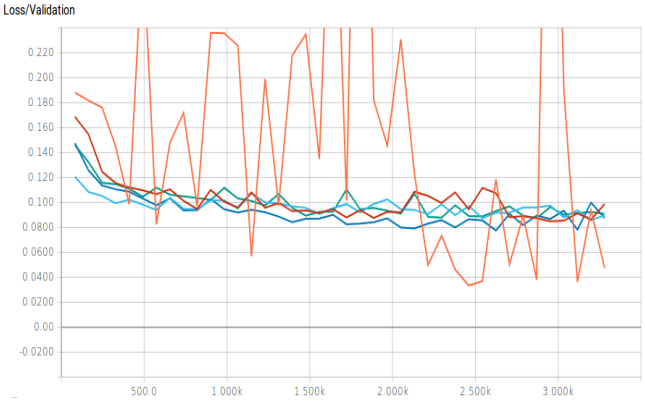

Fig. 3. Comparison of Test Loss

The top graph shows the train loss and the bottom graph shows the test loss when compared with proposed network that is 1DCRNN with LSTM, GRU, DRNN and LRCN. The orange line signifies 1DCRNN, blue line signifies LSTM, red line signifies DRNN, light blue line signifies GRU and green line signifies LRCN. The x-axis signifies time step and y-axis signifies loss in both the graphs.

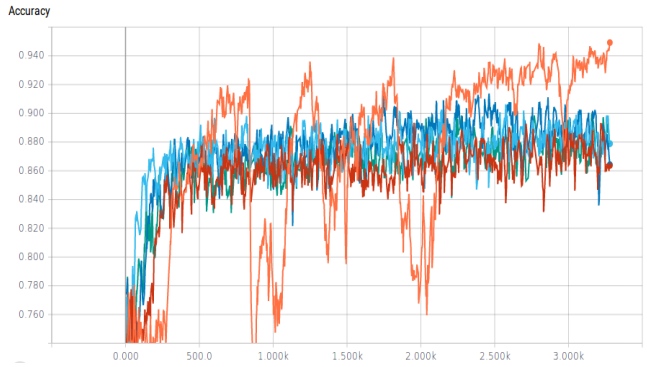

Fig. 4. Comparison of Train Accuracy

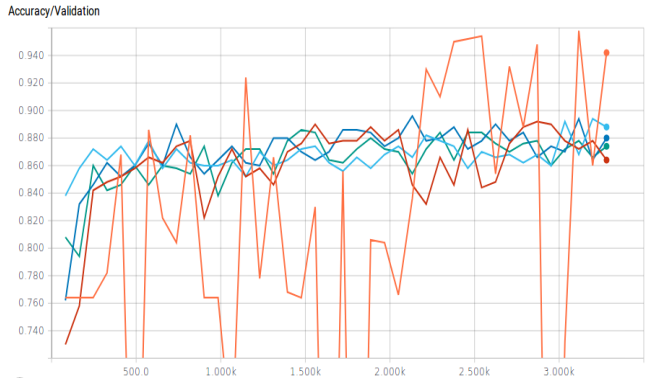

Fig. 5. Comparison of Test Accuracy

The top graph shows the train accuracy and the bottom graph shows the test accuracy when compared with proposed network that is 1DCRNN with LSTM, GRU, DRNN and LRCN The orange line signifies 1DCRNN, blue line signifies LSTM, red line signifies DRNN, light blue line signifies GRU and green line signifies LRCN. The x-axis signifies time step and y-axis signifies accuracy in both the graphs. When noticing the loss parameters that are train loss and test loss, it can be seen that the proposed architecture achieved the least loss when compared to other four networks. Similar to that, when noticing the accuracy parameters that are train accuracy and test accuracy, it can be seen that the network has achieved the highest accuracy. These results are acquired using Tensorflow (Google deep learning framework) and Tensorboard.

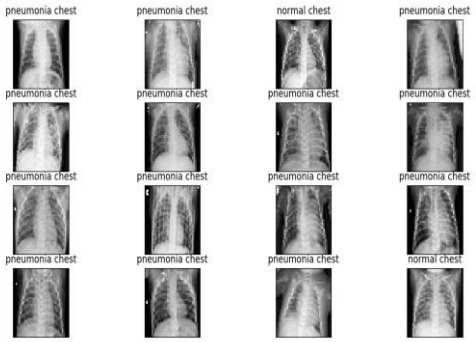

Fig. 6. The above figure is the predicted output of the 1DCRNN network. The network is tested on the pneumonia dataset [20] where there are two classes, which are normal chest and the diseased chest, means the image where pneumonia is caused.

V. CONCLUSION

In this paper, the proposed network is a combined network of two 1x1 CNN, LSTM, two GRU having ELU activation function that are followed by the Batch Normalization, dense classification layer having two fully connected layer with dropout function one after the another, the output layer which uses the softmax function and regression layer together called as 1D Convolutional Recurrent Neural Network (1DCRNN) and has resulted in the State-Of-Art performance. The proposed network has been tested upto 40 iteration and the results when compared showed that the 1DCRNN out-performed LSTM, GRU, DRNN, LRCN in very aspects that is training loss, testing loss, training accuracy and testing accuracy where training loss is 0.04119, testing loss is 0.04749, training accuracy is 0.9492 and testing accuracy is 0.9420

REFERENCES

1. Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *6*(02), 107-116.
2. Pearlmutter, Barak. (1991). Dynamic Recurrent Neural Networks.
3. Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.
4. Junyoung Chung, Caglar Gulcehre, KyungHyun Cho and Yoshua Bengio. “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.” *NIPS Deep Learning and Representation Learning Workshop.* 2014.
5. Tara N. Sainath, Oriol Vinyals, Andrew Senior and Has¸im Sak. “Convolutional, Long Short-Term Memory, Fully Connected Deep Neural Networks.” *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 2015.
6. Berkowitz, Zachary C. Lipton and John. “A Critical Review of Recurrent Neural Networks for Sequence Learning.” *Machine Learning (cs.LG); Neural and Evolutionary Computing (cs.NE).* 2015.
7. Haşim Sak, Andrew Senior and Françoise Beaufays. “Long Short-Term Memory Recurrent Neural Network Architecture for Large Scale A.” *Neural and Evolutionary Computing (cs.NE); Computation and Language (cs.CL); Machine Learning (cs.LG); Machine Learning (stat.ML).* 2014.
8. Tristan Stérin, Nicolas Farrugia, Vincent Gripon. “An Intrinsic Difference between Vanilla RNNs and GRU Models.” *The Ninth International Conference on Advanced Cognitive Technologies and Applications.* Athenes, Greece, 2017. 76-81.
9. Wenling Shang, Kihyuk Sohn, Diogo Almeida and Honglak Lee. “Understanding and Improving Convolutional Neural Networks via Concatenated Rectified Linear Units.” *ICML*. 2018
10. Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Md. Mostofa Ali Patwary, Prabhat and Ryan P. Adams. “Scalable Bayesian Optimization Using Deep Neural Networks.” *Machine learning (stat.ML).* 2015.
11. Oren Rippel, Jasper Snoek and Ryan P. Adams. “Spectral Representation for Convolutional Neural Networks.” *Neural Information Processing Systems (NIPS).* 2015.
12. Hu, Ming Liang and Xiaolin. “Recurrent Convolutional Neural Netowork for Object Recognition.” *CVPR.* 2015.
13. Rupesh Kumar Srivastava, Klaus Greff and Jurgen Schmidhuber. “Training Very Deep Networks.” *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems.* 2015. 2377-2385.
14. Djork-Arné Clevert, Thomas Unterthiner and Sepp Hochreiter. “Fast and Accurate Deep Network Learning Exponential Linear Units (ELUS).” *ICLR.* 2016.
15. Szegedy, Sergey Ioffe and Christian. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” *Machine Learning (cs.LG).* 2015

16. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." *Journal of Machine Learning Research*, 2014: 1929-1958.
17. Bengio, Yann LeCun and Yoshua. "Convolutional Networks for Images, Speech and Time Series." In *The handbook of brain theory and neural networks*, 255-258. 1998.
18. Pedamonti, Dabal. "Comprasion of non-linear activation functions for deep neural network on MNIST classification task." *Computer Vision and Pattern Recognition (cs.CV).* 2014.
19. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich, "Going deeper with convolutions", Computer Vision and Pattern Recognition (cs.CV), 2014, Vol. 1.
20. Kemmany, Daniel, Khang Zhang, and Michael Goldnaum. "Labelled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification." 2018.

AUTHORS PROFILE

Nitin Arora received B. Tech. (Comp. Sci. Engg.) from NIT ALLAHABAD in 2008 and M. Tech. (Comp. Sci. & Engg.) in 2012 with GOLD MEDAL from Uttarakhand Technical University, Dehradun, the member of IANEG (USA), ISOC (USA) has published over Fifteen plus research papers in National and International Journals/Conferences and IEEE proceeding publication in field of Data Structures and Algorithms, Mobile Ad-hoc networks & Digital Image Processing. He is working as an Assistant Professor (Senior Scale) at University of Petroleum and Energy Studies (UPES), Dehradun.

Ahatsham was born in Roorkee, India in 1993. He received his B.tech degree from Uttrakhand Technical University in 2014 and M.tech degree from NIT Nagpur in 2016.He is currently working as Assistant Professor in University of Petroleum and Energy Studies, Dehradun, India. His research interests include data mining, machine learning and data streaming.

Anupam Singh is Assistant Professor in Department of Informatics, School of Computer Science, University of Petroleum & Energy Studies, Dehradun. He is pursuing Ph.D. from Dr. A P J Abdul Kalam Technical University Lucknow. Formerly UPTU Lucknow). He has done B. Tech. in 2004, M. Tech. in 2011 from Uttar Pradesh Technical University Lucknow. His area of interest are Formal Methods, Distributed System and Database System. He is Reviewer of many referred journals. He has evaluated many presentations in International Conference as Session Chair.

Vivek Shahare is Assistant Professor in School of Computer Science and Engineering, UPES, Dehradun. He has completed his M.Tech from Visvesvaraya National Institute of Technology (VNIT), Nagpur in 2016 and B.Tech from Government College of Engineering, Amravati in 2012. He has 5 years of experience in Teaching, Research and Industry. His areas of interest are Bioinformatics and Theoretical Computer Science.